UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
COORDENADORIA DO CURSO DE ENGENHARIA DE SOFTWARE

YURI REIS CORREA

IDENTIFICAÇÃO DE SISTEMAS DINÂMICOS NÃO LINEARES PARA O CONTROLE PREDITIVO DE CLIMATIZAÇÃO, USANDO MODELOS *HAMMERSTEIN*.

TRABALHO DE CONCLUSÃO DE CURSO

DOIS VIZINHOS

2019

YURI REIS CORREA

IDENTIFICAÇÃO DE SISTEMAS DINÂMICOS NÃO LINEARES PARA O CONTROLE PREDITIVO DE CLIMATIZAÇÃO, USANDO MODELOS *HAMMERSTEIN*.

Trabalho de Conclusão de Curso apresentado como requisito parcial à obtenção do título de Bacharel em Engenharia de Software, da Universidade Tecnológica Federal do Paraná.

Orientador: Prof. Dr. Yuri Kaszubowski Lopes

DOIS VIZINHOS

2019

TERMO DE APROVAÇÃO

Identificação de sistemas dinâmicos não lineares para o controle preditivo de climatização, usando modelos Hammerstein.

por

Yuri Reis Correa

Este Trabalho de Conclusão de Curso foi apresentado em 25 de Novembro de 2019 como requisito parcial para a obtenção do título de Bacharel em Engenharia de Software. O(a) candidato(a) foi arguido(a) pela Banca Examinadora composta pelos professores abaixo assinados. Após deliberação, a Banca Examinadora considerou o trabalho aprovado.

Yuri Kaszubowski Lopes
Presidente da Banca

Dionatan Augusto Guimaraes Cieslak
Membro Titular

Francisco Carlos Monteiro Souza
Membro Titular

* A Folha de Aprovação assinada encontra-se na Coordenação do Curso

Dedico este trabalho à minha mãe, que enfrentou diversas dificuldades, com todas as forças que possui, para proporcionar-me a melhor formação pessoal e acadêmica possível, ao meu irmão, que sempre entregou-me seu amor e admiração e a meus avós, que participaram ativamente de minha formação pessoal e humana, provendo-me sabedoria e dedicação.

AGRADECIMENTOS

Agradeço ao meu estimado orientador, que me proporcionou esplêndido aconselhamento, determinante para a produção deste trabalho. Estendo os agradecimentos a todos os professores, que me proveram conhecimento técnico científico essencial para minha formação, aos colegas e amigos, que em mim confiaram e aos demais que, direta ou indiretamente, contribuem para este estudo ou em minha jornada acadêmica.

Conhece-te a ti mesmo e conhecerás o universo e os deuses.
Sócrates.

RESUMO

CORREA, Yuri Reis. IDENTIFICAÇÃO DE SISTEMAS DINÂMICOS NÃO LINEARES PARA O CONTROLE PREDITIVO DE CLIMATIZAÇÃO, USANDO MODELOS *HAMMERSTEIN*.. 67 f. Trabalho de Conclusão de Curso – Coordenadoria do Curso de Engenharia de Software, Universidade Tecnológica Federal do Paraná. Dois Vizinhos, 2019.

Este trabalho visa o desenvolvimento de uma metodologia para obter, de forma automática, a parte não-linear estática do modelo em *Hammerstein*, para o controle preditivo (*MPC*) de climatização em edificações não-residenciais, baseando-se em aprendizado de máquina. O objetivo da realização deste trabalho é auxiliar na possível obtenção de uma alternativa ao modelo já existente, desenvolvido através de Programação Genética. O trabalho é motivado pela necessidade de redução no consumo de energia e emissão de gás dióxido de carbono (CO_2) na atmosfera pelos edifícios, tendo em vista o aumento considerável na quantidade de edificações construídas e seu atual impacto no meio ambiente.

Palavras-chave: Inteligência Artificial, Aprendizado de máquina, Mat. Computacional

ABSTRACT

CORREA, Yuri Reis. IDENTIFICATION OF DYNAMIC NON-LINEAR SYSTEMS FOR CLIMATIZATION PREDICTIVE CONTROL, USING HAMMERSTEIN MODELS.. 67 f. Trabalho de Conclusão de Curso – Coordenadoria do Curso de Engenharia de Software, Universidade Tecnológica Federal do Paraná. Dois Vizinhos, 2019.

This work aims at the development of a methodology to obtain, automatically, the non-linear static part of a model in Hammerstein for the predictive control of climatization in non-residential buildings, based on machine learning. The objective of this work is to help the achievement of an eventual alternative to the existing model, developed through Genetic Programming. The work is motivated by the need to reduce the energy consumption and emissions of carbon dioxide (CO_2) into the atmosphere by buildings, given the considerable increase in the number of buildings built and their current impact on the environment.

Keywords: *Artificial Inteligence*, *Machine Learning*, *Computing Mathematics*

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

APRBS	*Amplitude Modulated Pseudo-random Binary Sequence* (Sequência Binária Pseudo-randômica de Amplitude Modulada)
E+	Simulador de Edificações *Energy Plus*
GP	*Genetic Programming* (Programação Genética)
MPC	*Model Predictive Control* (Controle por Modelo Preditivo)
OE	*Output Errors* (Erros de Saída)
PRBS	*Pseudo-random Binary Sequence* (Sequência Binária Pseudo-randômica)
TCC	Trabalho de Conclusão de Curso
UTFPR	Universidade Tecnológica Federal do Paraná

SUMÁRIO

1 INTRODUÇÃO

Um dos maiores problemas na climatização de edifícios é o alto consumo de energia e a emissão de dióxido de carbono (CO_2) na atmosfera. Esse problema é evidente pelo fato de muitas dessas construções não possuírem uma estratégia de controle voltada para a climatização ou, ainda que possuam tal estratégia, esta é baseada em controle reativo, não sendo otimizada para obter maior eficiência aplicando menor quantidade de esforço. Esse fator faz com que estes sistemas de climatização não otimizados, na maioria dos casos, empreguem quantidade desnecessária de energia para a adequação do ambiente à temperatura desejada.

Objetivando a redução do consumo de energia e da emissão de CO_2 na atmosfera terrestre, muitos dos métodos de controle reativo empregados por edificações, controlam o esforço aplicado pelo sistema de forma a reagir a estímulos recebidos do ambiente, tais como aumento de temperatura externa, maior irradiação solar, entre outros. De acordo com Rockett e Hathway (2017, p.2, tradução nossa), "Atualmente, quase todas as edificações não-domésticas empregam sistemas baseados em regras, também conhecidos como sistemas de gerenciamento de edifícios (*BMSs*) ou sistemas de gerenciamento energético de edifícios (*BEMs*).", para realizar o controle, afim de atingir os *setpoints* previamente definidos. [1]

Nesse contexto, se enquadram os sistemas de controle reativo, fortemente criticados por autores, como Privara et al. (2013), por possuírem complexidade exponencialmente crescente para o conjunto de regras utilizados. A dificuldade se torna cada vez maior quando se adicionam fatores aquém da temperatura para o controle da climatização, sendo a estratégia de reação, caracterizada pelo conjunto de instruções lógicas "se-senão"(*if-else*), ineficiente na maioria dos casos.

Rockett e Hathway (2017) propõem sistemas de controle preditivo da climatização, tais como o *Model Predictive Control* (*MPC*), que realizam uma previsão da temperatura ambiente a ser atingida, no caso da aplicação de determinada quantidade de calor, também

[1] "*Currently, almost all non-domestic buildings employ rule-based controllers, also known as building management systems (BMSs) or building energy management systems (BEMs).*"

definida pelo sistema, em função de um conjunto de outras variáveis, para uma quantidade 'n' de espaços de tempo futuros (o que se denomina tempo discreto). Estes espaços de tempo podem ser medidos em segundos, minutos, horas ou qualquer outra unidade temporal (T+1, T+2, ... T+n).

O sistema preditivo em questão, que atua no controle do aquecimento de edificações não residenciais (em países de clima frio). utiliza um modelo matemático capaz de realizar previsão das variáveis de saída do sistema com base nas variáveis passadas e nas entradas de controle a serem aplicadas. Especificamente neste projeto, a saída representa a temperatura do ambiente e a variável definida como entrada de controle é o fluxo de massa de água quente que passa por um radiador. É permitido ainda ao modelo tomar outras variáveis de entrada, como por exemplo a temperatura externa, velocidade do vento e radiação solar. O *MPC*, então, usa estas variáveis para fazer uma otimização da variável controlada a fim de que a saída (temperatura) seja a mais próxima possível do valor de referência (*setpoint*).

No *MPC* além do fluxo de massa de água a ser passada pelo radiador, nos tempos (T, T+1, ..., T+n), são dadas como entrada informações advindas da saída do mesmo procedimento para tempos passados (..., T-2, T-1, T), sendo o modelo, desta maneira, caracterizado como recursivo.

A existência de um bom modelo para a predição é central ao funcionamento do MPC. Este modelo se caracteriza pelo conjunto de regras matemáticas que levam à obtenção do resultado de qual será o estado ou a atuação do sistema nos tempos futuros, partindo-se de variáveis extraídas do ambiente ou de execuções anteriores. Há vários tipos de modelos preditivos que podem ser aplicados ao MPC e cada um utiliza diferentes modelos matemáticos. Atualmente, estão sendo desenvolvidos trabalhos que aplicam diferentes técnicas, tais como, Programação Genética (ainda em desenvolvimento) (ROCKETT et al.,) ou Redes Neurais (ROCKETT; HATHWAY; SYKES, 2018). Estes modelos se caracterizam por serem uma alternativa automatizada para um controle que já é atualmente aplicado, ainda que de forma manual e ineficiente.

Rockett, Lopes, Dou e Hathway relatam a projeção de continuação de seu estudo com *GP*: "Uma área majoritária do nosso trabalho futuro será explorar árvores derivadas de Programação Genética no Modelo de Controle Preditivo (*MPC*), como relatado na seção 1. Resultados serão publicados em outros artigos."(ROCKETT et al., , p.11, tradução nossa) [2]

[2] *"One major area of our future work will be to exploit GP tree derivatives in model predictive control*

Neste trabalho, é abordada a obtenção de um modelo, que será, posteriormente, essencial para a geração de uma solução completa, que poderá ser aplicada ao MPC. Este último poderá utilizar o modelo completo para a avaliação em uma edificação simulada. O estudo presente propõe a obtenção da parte não linear estática do modelo *Hammerstein*, como forma de contribuir para o surgimento de alternativas à Programação Genética e/ou às Redes Neurais para a obtenção do modelo que fará a predição das próximas temperaturas. Este estudo possui seu diferencial inovador pautado na geração e aplicação de uma nova abordagem para o Modelo de Controle Preditivo, o que possibilitará estudos futuros em melhoria do processo de controle preditivo.

A simulação da edificação será realizada com o *software Energy Plus* (*E+*) (CRAWLEY et al., 2017), um sistema de uso industrial, que realiza a simulação de uma edificação, levando em consideração, desde fatores estruturais, até indicadores de habitação e emissão de calor pelos componentes eletrônicos presentes no prédio.

Este trabalho é dividido em dois momentos. No primeiro deles, para a aquisição de dados úteis ao aprendizado de máquina, a edificação foi estimulada com uma sequência binária pseudo-randômica de amplitude modulada (*APRBS*), que permite que os dados coletados sejam relevantes e estejam dentro de condições de operação esperadas. Além disso, foram realizadas as implementações que resultaram no modelo preditivo.

Num segundo momento, posterior à realização deste estudo, já de posse de tal modelo, obtido por aprendizagem de máquina, através de um controlador, correspondente a uma implementação do *MPC*, serão inseridas as variáveis controláveis (i.e. fluxos de massa), outras variáveis de entrada tais como as temperaturas anteriormente registradas, radiação solar, velocidade do vento, etc. O *Energy Plus* utilizará o *MPC* (*Model Predictive Control*), para a obtenção dos fluxos de massa a serem aplicados nos radiadores da edificação para o aquecimento.

1.1 MOTIVAÇÃO

Este trabalho está pautado em um dos problemas ambientais mais discutidos na atualidade: a redução da emissão de gases poluentes e economia de energia. O objetivo é causar impacto nesta redução, no que se refere a edificações de grande porte, principalmente não residenciais. De acordo com Rockett e Hathway (2017), no Reino Unido a emissão de gás carbônico por edifícios não residenciais responde por 18% do total para o país.

(MPC) [3], as set out in Section 1. Results will be published elsewhere."

Além disso, estes autores afirmam que o problema se tornará ainda maior, tendo em vista que a área coberta por essas edificações aumentará em um terço até 2050. Desta maneira, torna-se necessário o empenho em soluções que visem reduzir as emissões de gás carbônico e o consumo de energia em edificações. Rockett e Hathway (2017) propõem, que o controle do consumo de energia em climatização de edifícios deve ser realizado levando-se em consideração o conforto térmico, sendo esse conforto atingido através da especificação de '*setpoints*' de temperatura que devem ser respeitados.

Tendo em vista esse fato, Rockett e Hathway (2017) apontam o Modelo de Controle Preditivo, ou *MPC*, como forma de alcance deste objetivo. Segundo estes autores, foi identificada uma redução de até 50% no consumo de energia com o uso desta técnica. "Assim como estudos baseados em simulação, foi descoberto que o *MPC* realmente reduz o consumo de energia e aumenta o conforto térmico. Comparado a estratégias de controle clássicas, o *MPC* economizou até 30% de energia, ainda considerando a variação desta quantidade, dependente do tipo de edificação e das condições climáticas" (HAZYUK; GHIAUS; PENHOUET, 2012, p.2, tradução nossa) [3].

Contudo, as edificações possuem um longo tempo de resposta (*lag* ou *delay*) à aplicação do calor. Ou seja, é demandado um determinado período até que a temperatura seja efetivamente modificada. Sendo assim, o uso de regras de controle reativo não é suficiente para permitir o gerenciamento otimizado destes sistemas. Este tempo de resposta se deve à natureza dinâmica de primeira ordem dos sistemas térmicos.

Tradicionalmente, em plantas químicas (área de maior aplicação do *MPC*), são investidas várias horas de um time de engenheiros de controle, para a obtenção e calibragem do modelo preditivo. Para a área de climatização, este fato gera um custo financeiro, cujo valor pode ultrapassar o valor economizado. Portanto, o incentivo financeiro ao uso do *MPC* acaba sendo inviabilizado. Para tornar o uso do modelo economicamente viável, se faz necessário investir em estratégias de obtenção automática destes modelos preditivos. Este processo é particularmente importante quando considerado que qualquer alteração física na edificação, ou nas imediações, pode invalidar tal modelo.

O uso de aprendizado de máquina para a obtenção destes modelos surge, portanto, como uma alternativa viável. Outros trabalhos apresentam métodos como programação genética e redes neurais (ROCKETT; HATHWAY; SYKES, 2018). Estes trabalhos tratam do uso de aprendizado de máquina que, baseado em um conjunto de dados

[3] "*Similar to simulation-based studies, it has been found that MPC really reduces energy consumption and improves thermal comfort. Compared to classical control strategies, it saved up to 30% of energy, but this quantity varied depending on building type and weather conditions.*"

coletados, produz um modelo de caixa preta (uma aproximação empírica das dinâmicas do sistema, sem uma relação real com as características físicas). Estes métodos utilizados (Redes Neurais e Programação Genética) pretendem identificar um modelo não linear e dinâmico, que é demasiado complexo. Atendendo à necessidade de separação em blocos não lineares estáticos e dinâmicos, propõe-se o modelo *Hammerstein*. A ideia central é implementar parte da automatização desta aplicação , possibilitando que trabalhos futuros a comparem com modelos automatizados existentes, afim de medir sua viabilidade econômica e tecnológica.

O framework *MPC*, bem como o sistema de coleta de dados, já existem e operam sobre a ferramenta *E+* (CRAWLEY et al., 2017). O diferencial deste trabalho será a proposição e implementação de uma metodologia que auxilie na viabilidade econômica do *MPC*. Este desenvolvimento será realizado e testado em parceria com pesquisadores da *The Univesity of Sheffield*, no Reino Unido, nos departamentos de Engenharia Eletroeletrônica (*EEE*) e Engenharia Civil (*CIV*).

1.2 OBJETIVOS GERAIS

O trabalho tem por objetivo desenvolver uma metodologia de obtenção automática do modelo *Hammerstein* (esclarecido na seção 2.6) para o *MPC* de edificações, baseado em aprendizado de máquina. O modelo *Hammerstein* aplicado abrange um bloco não linear estático, formado por *splines* (curvas definidas por dois ou mais pontos) suaves e monotônicas, seguido de um bloco linear dinâmico, que utiliza o modelo de erros de saída (*OE*, do Inglês, *OutputError*). Os modelos devem ser acoplados, conforme definido no modelo *Hammerstein*, para gerar um modelo final não-determinístico e dinâmico.

Devido ao fato de o modelo ser utilizado pelo *MPC*, dentro de um processo de otimização que utiliza técnicas dependentes de derivadas, a suavidade da curva obtida no bloco não-linear estático é essencial. Uma curva é suave em um intervalo se, para todo e qualquer ponto do intervalo, existem sua primeira e segunda derivadas. O uso de *splines* para a interpolação de conjunto de dados (*dataset*) é proeminente na literatura, contudo, *splines*, em si, não são necessariamente monotônicas, diferentemente da relação entre o aumento de fluxo de massa e a quantidade de calor inserida no ambiente, que é monotônica por definição. Sendo assim, se faz necessária a garantia desta propriedade também nas *splines*. Para isso, será utilizada uma otimização das funções cúbicas que descrevem a *spline*, conforme sugerido por Wolberg e Alfy (2002).

1.3 OBJETIVOS ESPECÍFICOS

Os seguintes pontos foram visados para o sucesso do estudo a ser realizado:

- Identificar e analisar do processo de obtenção de dados de entrada pelo sistema *E+*;
- Analisar e compreender da obtenção da temperatura aplicada nos tempos futuros, utilizando se dos modelos já existentes em estudos relacionados;
- Desenvolver do bloco estático não linear, utilizando *Splines*
- Otimizar para atingir-se *splines* suaves e monotônicas
- Compreender e interpretar os resultados obtidos por este modelo.

2 REVISÃO BIBLIOGRÁFICA

Esta seção visa esclarecer conceitos utilizados para a implementação e aplicação deste trabalho, objetivando aproximar o leitor do contexto nele abordado.

2.1 CLIMATIZAÇÃO DE EDIFÍCIOS

Com o advento de novas tecnologias e surgimento de novas ideias de negócios no mercado, bem como a expansão constante de espaços corporativos, educacionais, entre outros não destinados à habitação residencial (priorizadas para este estudo), nota-se, proporcionalmente, aumento na necessidade de adequar estes ambientes ao conforto inerente às atividades neles desempenhadas. Neste contexto está a climatização de edificações, uma necessidade cada vez mais evidente, tendo em vista as mudanças climáticas ocorridas nas últimas décadas no planeta e a necessidade de adequação em construções que estejam situadas em locais de temperaturas extremas.

No contexto deste trabalho, são estudadas edificações carentes de aquecimento, que se situam, em geral, países de clima predominantemente frio, tais como o Reino Unido (tomado como referência para a aplicação futura deste estudo).

Existe literatura referente ao tema de climatização e seus impactos no meio ambiente, através do consumo de energia e emissão de gás dióxido de carbono (CO_2) na atmosfera terrestre, como é o exemplo de Rockett e Hathway (2017). Essa preocupação se torna ainda maior no tangente à climatização de edificações, por se tratarem de ambientes amplos, que demandam maior quantidade de tempo, esforço e, consequentemente, de energia para alcançarem a condição de conforto térmico.

2.2 ENGENHARIA DE *SOFTWARE*

Segundo Pressman (2011), a Engenharia de *Software* "é o estabelecimento e o emprego de sólidos princípios de engenharia de modo a obter *software* de maneira

econômica, que seja confiável e funcione de forma eficiente em máquinas reais".

Neste contexto, diversas diretrizes e métodos são aplicados ao processo de produção de um software para que se adquira o nível de qualidade e eficiência esperado.

As principais áreas relativas à Engenharia de Software abordadas neste trabalho são a Codificação (programação) e a Modelagem de Sistemas, através de diagramas da *Unified Modeling Language* (*UML*), que é empregada na representação da solução desenvolvida.

2.3 MODELOS DE SISTEMAS

O conceito de modelo matemático é fundamental em muitos ramos da ciência e da engenharia. (BILLINGS, 2013, p.1, tradução nossa). [4]

Praticamente todo sistema existente pode ser modelado matematicamente, sendo descrito como um conjunto de equações matemáticas, que pode ser extraído do problema abstraído, havendo, para tal, uma série de estruturas e métodos que podem ser seguidos e utilizados. Um sistema é descrito por Oliveira (2002) como um conjunto de partes interagentes e interdependentes que, conjuntamente, formam um todo unitário com determinado objetivo e efetuam determinada função . De forma geral, o modelo matemático de um sistema qualquer descreve seu comportamento, baseando-se em suas entradas.

De acordo com Oliveira (2002), refere-se à entrada de um sistema como o conjunto de estímulos externos aos quais este sistema é exposto. Uma entrada pode ser exemplificada por uma variável cujo valor é dado por um usuário do sistema ou, ainda, dados que são recolhidos de análises ao ambiente ou contexto no qual o sistema está inserido. Como saída de um sistema pode se definir a resposta que este entrega, que leva em consideração as entradas dadas e o processamento que o sistema aplica nestas últimas.

Ao se possuir o modelo de um sistema definido, torna-se possível a obtenção da lei de controle do mesmo, através da inversão do modelo. No entanto, a esta inversão, algumas vezes pode não ser factível. Por este motivo, pode ser realizada a otimização do modelo.

De acordo com Billings (2013, p.3), a identificação de um sistema consiste em um método de medição da descrição deste sistema, processando-se suas entradas e saídas observadas, sendo o complemento do problema da simulação.

[4] "*The concept of a mathematical model is fundamental in many branches of science and engineering.*"

2.3.1 SISTEMAS LINEARES E NÃO-LINEARES

De acordo com Casillo (2009, p.11) um sistema não-linear pode ser descrito como um sistema no qual o princípio da superposição não se aplica. Em outras palavras, se as variáveis que o definem aparecem na equação sob a forma de funções não lineares.

2.3.2 SISTEMAS ESTÁTICOS E DINÂMICOS

De acordo com Cassandras e Lafortune (2007, p.45), em sistemas estáticos, os elementos de saída sempre serão independentes dos valores passados de entrada.

Esta definição permite a visualização da não aplicabilidade de sistemas estáticos individualmente neste estudo, por se tratar de um caso onde é necessária a relação entre os dados passados que foram obtidos do sistema em tempos anteriores e a saída do sistema no tempo atual, pois deste princípio dependerá a otimização do modelo, que buscará a evolução contínua.

Por outro lado, ainda segundo Cassandras e Lafortune (2007, p.45), em sistemas dinâmicos, esta relação de dependência entre a saída e os dados obtidos de tempos passados existe e mostra-se essencial, o que ilustra sua aplicabilidade no modelo Hammerstein completo.

O *Hammerstein* utiliza uma aplicação em blocos de um sistema estático (não linear), seguido de um sistema dinâmico (linear).

2.4 MODELO DE CONTROLE PREDITIVO *MPC*

O *Model Predictive Control* (*MPC*) é uma metodologia de controle utilizada em sistemas que apresentam certas restrições em sua aplicação.

O *MPC*, de acordo com Camacho e Bordons (2004), tem origem na década de 1970 e não se refere a apenas uma estratégia específica de controle e sim a um amplo e abrangente leque de métodos utilizados para obtenção de sinais de controle através de otimização e simplificação de funções.

Ainda segundo Camacho e Bordons (2004), o *MPC* abrange uso explícito de um modelo para predição da saída do processo em instantes futuros, além do cálculo de uma sequência de controle, minimizando a função objetivo.

No caso da climatização de edifícios não residenciais, em geral, o sistema de

controle tradicional possui um atraso considerável entre um estímulo de entrada (neste caso, a passagem da água pelo radiador, que ocasiona o esforço no sistema) e a resposta deste sistema (representado pela real percepção de aquecimento no edifício). Este atraso, aliado a capacidade do *MPC* de predição dos dados para tempos futuros, justifica modelos como o proposto neste estudo.

2.5 ALGORITMOS GENÉTICOS E PROGRAMAÇÃO GENÉTICA

Algoritmos genéticos se baseiam nos processos evolucionários da natureza, onde os indivíduos de cada espécie se multiplicam e fazem com que seu grupo se perpetue, através de métodos como a seleção natural, onde sobrevivem aqueles que melhor se adaptam ao meio em que vivem, e mutação, onde se alteram alguns genes de alguns indivíduos, o que faz com que a espécie se reproduza de forma diferenciada e novas características desejadas sejam adquiridas. Essas características desejadas são codificadas em termos de uma função de *fitness*. Esta função retorna a qualidade ou o erro do modelo. Por exemplo: o erro entre o valor obtido do modelo e o esperado contido na base de dados. Esses algoritmos são também chamados de Algoritmos Evolucionários.

A programação genética realiza o mesmo processo para buscar uma solução em uma estrutura de árvore que representa uma equação matemática. A Figura 1 demonstra a equação

$$(3 + a) \,/\, (1 + b) \tag{1}$$

em forma de árvore, utilizada na Programação Genética.

O processo atual para a obtenção do modelo utilizado é realizado utilizando-se Programação Genética, onde o modelo é aprimorado a cada nova execução minimizando a predição do modelo com os dados da base de treinamento.

No contexto deste trabalho, será utilizada, para o desenvolvimento da otimização do modelo não-linear estático, a *GAlib*, implementada na linguagem *C++*, que, de acordo com seu manual de operação (Wall (1996)), consiste em uma biblioteca para objetos de algoritmos genéticos e possui ferramentas que possibilitam otimizar qualquer programa escrito em *C++*, utilizando qualquer representação e qualquer operador genético.

A figura 2 mostra um exemplo de programa derivado de uma implementação da GAlib, que realiza a maximização de uma função contínua em duas variáveis.

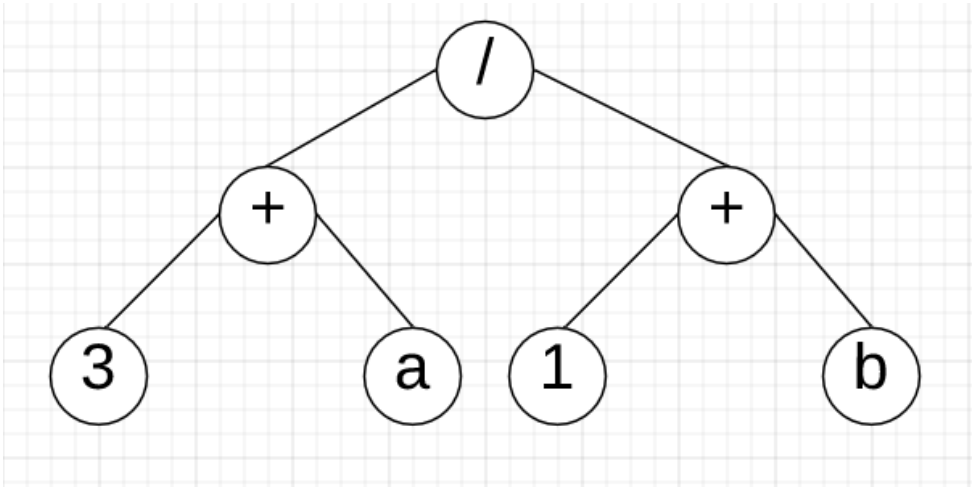


Figura 1: Equação $(3 + a)/(1 + b)$ em forma de árvore, utilizada pela *GP*
Fonte: Autoria Própria

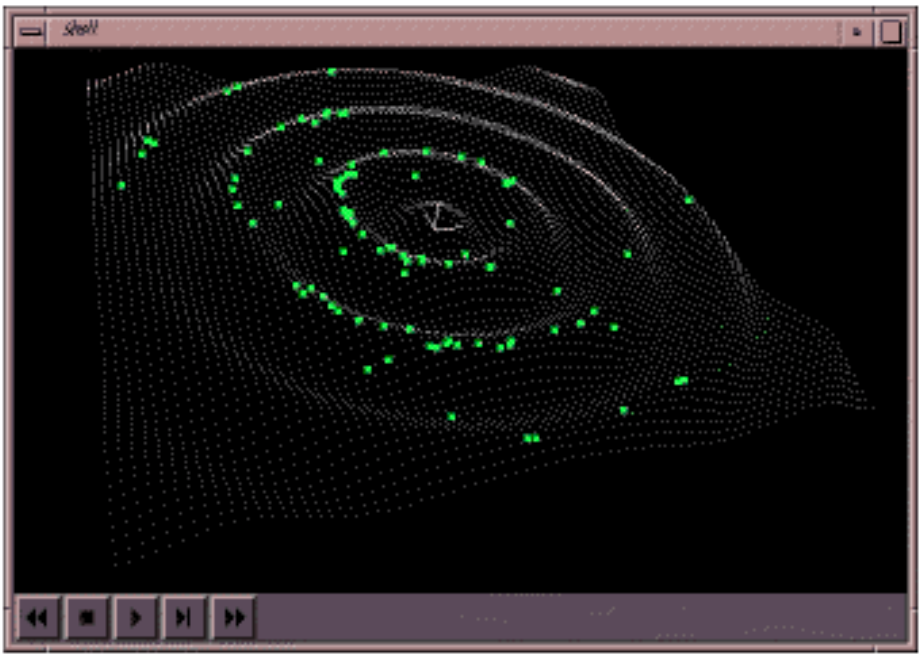


Figura 2: Exemplo de programa derivado de implementação da *GAlib*
Fonte: (GALIB. . . ,)

2.6 MODELO *HAMMERSTEIN*

O *Hammerstein* faz parte dos modelos de blocos interconectados (CASILLO, 2009) (ou Modelo Estruturado por Blocos (BILLINGS, 2013, p.31)). De acordo com Casillo (2009), nesse tipo de sistema um bloco não-linear estático precede um bloco linear

dinâmico, como ilustrado na Figura 3.

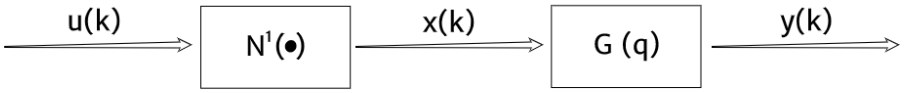


Figura 3: Modelo *Hammerstein*
Fonte: Adaptado de Casillo (2009)

Neste trabalho é utilizado uma implementação de múltiplas entradas e uma saída (MISO) do Hammerstein. Porém, após as sucessivas utilizações do modelo MISO se obterá um modelo de múltiplas entradas e múltiplas saídas (MIMO). São consideradas entradas as variáveis não controláveis, tais como temperaturas e incidências da radiação solar obtidas em 'n' estados passados e a variável controlável de fluxo de massa. Dessa forma, é inserido no sistema o conjunto desses dados obtidos no tempo imediatamente anterior (t-1), o que tornará o sistema recursivo.

O bloco não linear estático é responsável por gerar *splines* e realizar a otimização nas mesmas para que sejam sempre suaves e monotônicas. O bloco dinâmico linear é responsável pela validação dos valores obtidos por meio de testes de Erro de Saída (OE).

De acordo com Billings (2013, p.32), a ideia básica por trás dos modelos como o *Hammerstein* é a identificação de blocos dentro do sistema, com base apenas nas medições externas de entrada e saída, sem acesso a nenhum sinal interno de modo a manter o relacionamento de volta para o sistema subjacente.

2.7 *SPLINES* CÚBICAS

As *Splines* cúbicas são amplamente utilizadas para adequar uma função contínua suave em dados discretos. Elas possuem um papel importante em campos como a computação gráfica e o processamento de imagens, onde a interpolação suave é essencial na modelagem, animação e escalonamento de imagens. (WOLBERG; ALFY, 2002, p.145, tradução nossa).[6]

Ainda de acordo com Wolberg e Alfy (2002, p.145), as *splines* cúbicas são atrativas

[6]"*Cubic splines are widely used to fit a smooth continuous function through discrete data.They play an important role in such fields as computer graphics and image processing, where smooth interpolation is essential in modeling, animation, and image scaling*"

para a adequação de uma curva, pela razão de causarem considerável redução no uso de recursos computacionais. Para estes autores (WOLBERG; ALFY, 2002, p. 150), uma curva é monotônica em um intervalo se, e somente se, não existir uma mudança de sinal no valor derivado em nenhuma parte da curva no intervalo

Um conjunto de $k-1$ curvas *splines* são utilizadas para interpolar dados na forma de uma função definida em trechos (*piecewise function*), conforme ilustrado na Figura 4. Cada um destes trechos é uma curva *spline* cúbica que é definida no intervalo $[x_k, x_{k+1}]$ e possui a seguinte forma cúbica (WOLBERG; ALFY, 2002, p. 146-148):

$$f_k(x) = a_k(x - x_k)^3 + b_k(x - x_k)^2 + c_k(x - x_k)^2 + d_k \tag{2}$$

onde:

$a_k = \frac{1}{\Delta x_k}(-2\frac{\Delta y_k}{\Delta x_k} + y'_k + y'_{k+1})$;

$b_k = \frac{1}{\Delta x_k^2}(3\frac{\Delta y_k}{\Delta x_k} - 2y'_k - y'_{k+1})$;

$c_k = y'_k$;

$d_k = y_k$.

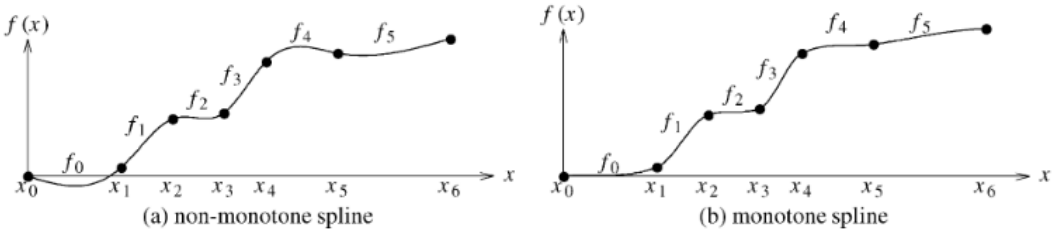


Figura 4: Conjuntos de *splines* monotônicas (a) e não monotônicas (b)
Fonte: Wolberg e Alfy (2002, p.150)

2.8 APRENDIZAGEM DE MÁQUINA

A aprendizagem de máquina pode ser definida pela sua capacidade de assimilar e tornar padronizados - o que se pode entender por 'aprender' - procedimentos, rotinas e métodos, sem a necessidade de que estes sejam programados previamente.

Este trabalho estuda a otimização de um modelo matemático por Aprendizagem de Máquina. Para este fim, existem diversas opções plausíveis de método que podem ser utilizados, tais como Programação Genética, Regressão Linear, Regressão Não-linear,

Redes Neurais, entre outras.

A otimização deste modelo pode ser realizada através da quantificação de perdas obtidas, ou seja, das situações onde a saída prevista se difere em grande escala da saída obtida. O sistema deve ser 'treinado' para corrigir a predição nas próximas iterações, considerando estas perdas.

Para valores reais, que não sejam binários, este aprendizado pode ser realizado utilizando-se Erro Quadrático Médio, que consiste na média entre todos os quantitativos de erro computados. Já para valores classificadores binários é utilizado o quantitativo por Perda de Entropia Cruzada Binária.

Para o fim da aprendizagem de máquina, no caso aqui estudado, são utilizados algoritmos que busquem por bons parâmetros no espaço de busca para o modelo matemático, a cada iteração do sistema. Para tal, pode-se combinar algoritmos de otimização global, nos quais é procurado um resultado ótimo global em um espaço de busca, com algoritmos de otimização local, os quais procuram por pontos ótimos locais, que não necessariamente representem o ponto ótimo global, como ilustrado na Figura 5.

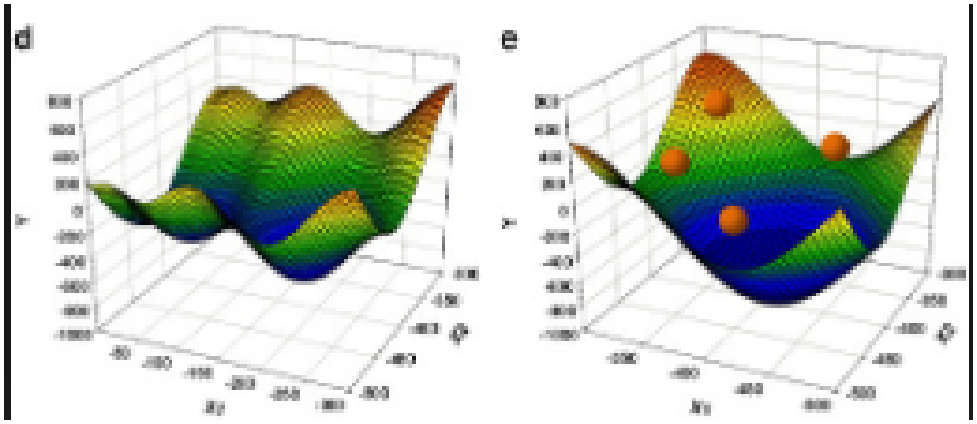


Figura 5: Exemplo de Espaço de Busca
Fonte: Adaptado de Li et al. (2013)

2.8.1 BIBLIOTECA *NLOPT*

O *NLopt* se trata de uma biblioteca, disponível para diversas linguagens de programação, tais como C++ (utilizada neste trabalho), *Matlab*, *Fortran*, *Python*, entre outras, que permite a otimização de funções matemáticas. Esta biblioteca possui uma interface comum para diversos algoritmos, o que permite realizar a mudança de um para outro de maneira fácil e rápida. O *NLopt* também suporta tanto a otimização global

quanto a otimização local de funções.

O *NLopt* é uma ferramenta de *Software* Livre (*Open Source*), que encontra-se disponível sob a licença *GNU LGPL*.

Os algoritmos do *NLopt* são identificados por uma constante única, que é utilizada pelas linguagens de programação para realizar a chamada de determinado algoritmo.

2.8.1.1 ALGORITMOS DE OTIMIZAÇÃO GLOBAL

- *Multi-Level Single-Linkage* (*MLSL*) (KAN; TIMMER, 1987)

  Este algoritmo realiza uma sequência de otimizações locais, iniciando-se de locais aleatórios ou com baixa discrepância. É caracterizado por ser uma heurística de "*clustering*", que evita a obtenção repetida dos mesmos ótimos locais.

- *Improved Stochastic Ranking Evolution Strategy* (*ISRES*) (RUNARSSON; YAO, 2018)

  O *ISRES* é implementado sob a premissa de combinações de regras de mutação, com suavização exponencial, e variações no diferencial. A função de "*fitness*"ocorre pela função objetiva para problemas sem regras não lineares.

- *Evolutionary Algorithm* (*ESCH*) (SANTOS; GONCALVES; FIGUEROA, 2010)

  Este algoritmo é utilizado somente para suporte de restrições vinculadas e não suporta restrições não lineares.

2.8.1.2 ALGORITMOS DE OTIMIZAÇÃO LOCAL

- *Constrained Optimization by Linear Aproximations* (*COBYLA*) (POWELL, 1994)

  O *COBYLA* constrói sucessivas aproximações lineares da função objetivo e restrições e otimiza essas aproximações a cada passo.

- *Principal AXIS* (*PRAXIS*) (BRENT, 1973)

  Projetado para otimização irrestrita, o *PRAXIS* implementa um expediente de retorno fácil ao infinito quando suas restrições são violadas.

2.9 TRABALHOS RELACIONADOS

Este trabalho é norteado e embasado em outros estudos, os quais permitem a compreensão do universo de aplicabilidade da solução.

Os trabalhos de Rockett e Hathway (2017) e Rockett, Hathway e Sykes (2018) concedem uma visão geral acerca de implementações do modelo *MPC* para melhoria do controle de edifícios não-residenciais. Rockett e Hathway (2017) realizam uma revisão crítica do uso deste modelo, atendo-se à ênfase do mesmo em implementações práticas, ao invés de aspectos teóricos. Rockett, Hathway e Sykes (2018) realizam um estudo relacionado à técnica de *Big Data*, aliada ao modelo preditivo, afim de otimizar o uso de energia nas edificações, bem como as condições internas do mesmo.

Vand et al. (2019) oferece uma visão mais clara do impacto do uso do *MPC* na redução do gasto energético e da emissão de gás CO_2 na atmosfera, relacionando o uso deste modelo com uma redução considerável do uso de energia e, consequentemente, da liberação de dióxido de carbono.

Outros estudos, tais como Rockett e Dou (2005), Rockett (2018), Rockett, Lopes e Dou (2018) e Dou e Rockett (2017) ilustram o uso de Programação Genética para a obtenção do modelo preditivo do sistema.

Em Rockett, Lopes e Dou (2018) é apresentado um padrão desenvolvido pelos autores, baseado na linguagem de marcação *XML*, que é aplicada para árvores genéticas e oferece uma padronização de sua implementação.

Dou e Rockett (2017) realizam uma série de experimentos, utilizando-se de um *framework* desenvolvido em Programação Genética, para obtenção de árvores genéticas, o que virá a ser a base de seu trabalho com *MPC* e *GP* combinados. Rockett e Dou (2005), por sua vez, comparam diferentes formas de seleção de subárvores genéticas para busca local, assim como diferentes formas de realização destas buscas.

Rockett (2018) realiza testes de permutação para otimização por 'poda' de subárvores genéticas, observando redução de, aproximadamente 20% no tamanho das mesmas.

3 METODOLOGIA

A Figura 6 contextualiza todo o processo no qual este estudo se encaixa para o *MPC* de edificações. Os dados coletados, por meio de uma simulação, com uso do software *Energy Plus* são utilizados para a aprendizagem de um modelo *Hammerstein*. Consequentemente, o modelo é utilizado pelo sistema *MPC* para realizar o controle do sistema de climatização da edificação. O MPC opera através de otimização preditiva sobre tal modelo, apenas aplicando uma configuração na edificação caso esta satisfaça a temperatura desejada em um horizonte futuro para o modelo.

É importante ressaltar que o MPC encontra um planejamento de configuração em um horizonte futuro, o que significa que este realiza uma sequência de acionamentos em uma janela de tempo futura. Contudo, após a aplicação do primeiro valor deste planejamento, um novo é realizado, contendo os dados atualizados pela execução anterior. Esta condição se deve ao fato de o sistema nunca atingir a perfeição e acumular erros, sendo essencial a geração de um novo planejamento a cada passo de controle

Este estudo foca na obtenção da parte não linear estática do modelo Hammerstein (destacada em verde na Figura 6). O modelo especificado na imagem é composto por esta implementação não linear estática, assim como por uma implementação linear dinâmica. Outros trabalhos relacionados a este, como Rockett, Lopes e Dou (2018) mantém seu foco na obtenção de um único modelo não linear dinâmico, utilizando para tal objetivo a programação genética.

3.1 ATIVIDADES

Para o desenvolvimento do trabalho, foi, inicialmente, realizado um estudo da literatura existente relacionada ao tema, afim de identificar conceitos e compreender as tecnologias utilizadas. Após, foi estudada a estrutura dos dados obtidos através a ferramenta *E+*, um simulador de edificações em tempo real.

Foram identificadas as variáveis a serem utilizadas, além dos testes e experimentos

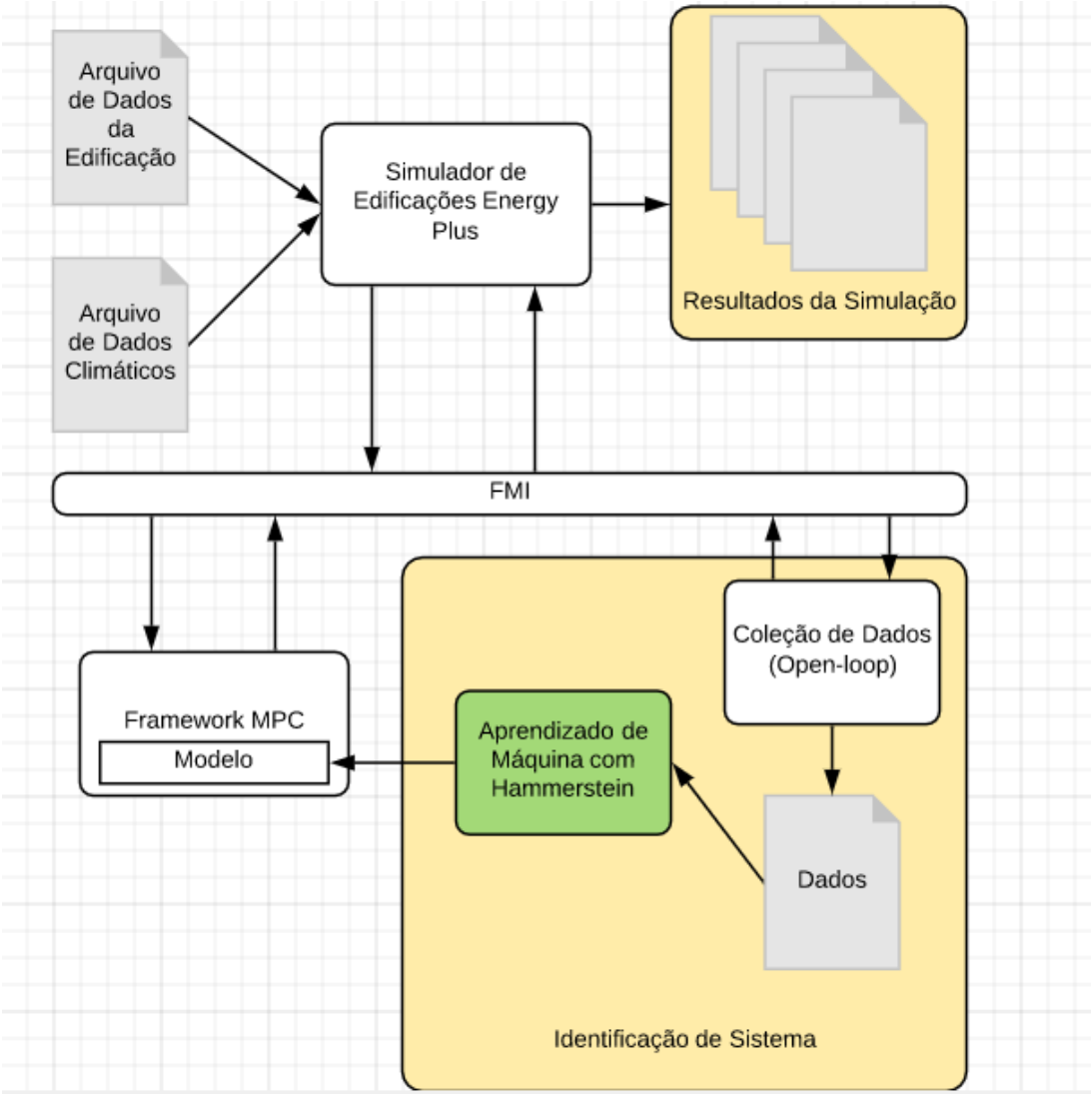


Figura 6: Fluxograma do Processo do *MPC*, utilizando *Hammerstein*
Fonte: Adaptado de Rockett, Lopes e Dou (2018)

com o aprendizado de máquina.

Na ferramenta, foi usado um modelo das configurações do edifício, bem como da conjuntura climática, no ambiente externo. Estes valores tem como referência as condições apresentadas durante o período de um ano, em uma edificação simulada pelo *E+*, situada na cidade de *Manchester*, Reino Unido.

A partir desses dados, um modelo de aprendizagem de máquina realiza um processo sobre os mesmos, através de seleção e *fitting*, dando origem ao conjunto estático não linear, que fará parte do modelo *Hammerstein*, sendo implantado no controlador do sistema de climatização.

3.2 UTILIZAÇÃO DO SIMULADOR *ENERGY PLUS*

Para a obtenção dos dados para o aprendizado de máquinas deste trabalho deve ser utilizado o software industrial *Energy Plus*, um simulador de edificações utilizado para a modelagem do consumo de energia nesse tipo de construção.

O *E+* recebe um arquivo em texto plano, conhecido como ***Input Data File***, Arquivo de Entrada de Dados, em tradução livre, que deve conter todas as informações e o estado preliminar do edifício. Estas informações abrangem pontos determinantes acerca da estrutura física da edificação, tais como geometria, materiais utilizados em sua construção, dentre outros. Também são introduzidas informações operacionais do edifício, como, por exemplo, as atividades nele realizadas. Virtualmente, qualquer informação relevante acerca do edifício pode ser introduzida neste arquivo.

Outro arquivo importante que deve ser adicionado é o arquivo de dados de climatização, chamado de ***Weather File*** (Arquivo de Clima), que contém informações relativas ao levantamento das condições climáticas externas a edifício ao longo de um determinado período de tempo. Muitas entradas podem ser consideradas neste arquivo, tais como temperatura externa, nível de radiação solar, velocidade do vento, etc.

3.3 DESCRIÇÃO DA EDIFICAÇÃO DE TESTES

A coleta de dados se dá em um experimento simulado no E+ em um edifício de sala única (representado na Figura 7), um radiador pelo qual será conduzido determinado fluxo de massa de água quente e um valor variável de ar fresco que será introduzido ao sistema. A edificação considerada possui dimensões de dez metros de comprimento, por dez metros de largura por três metros de altura. Essa edificação possui também duas janelas de tamanhos iguais, posicionadas ao centro das paredes. Foi considerada como localização da edificação a cidade de *Manchester*, no Reino Unido.

3.4 AQUISIÇÃO DE DADOS

Inicialmente, o sistema deve passar por uma fase de excitação, onde sinais são introduzidos, seguindo uma ideia prévia do esperado para conforto térmico. É importante que estes sinais de excitação sejam apropriados para o contexto, pois a informação obtida deve ser relevante para qualquer condição de operação. A amplitude do sinal de excitação deve ser suficiente para abranger todas as frequências interessantes para o estudo.

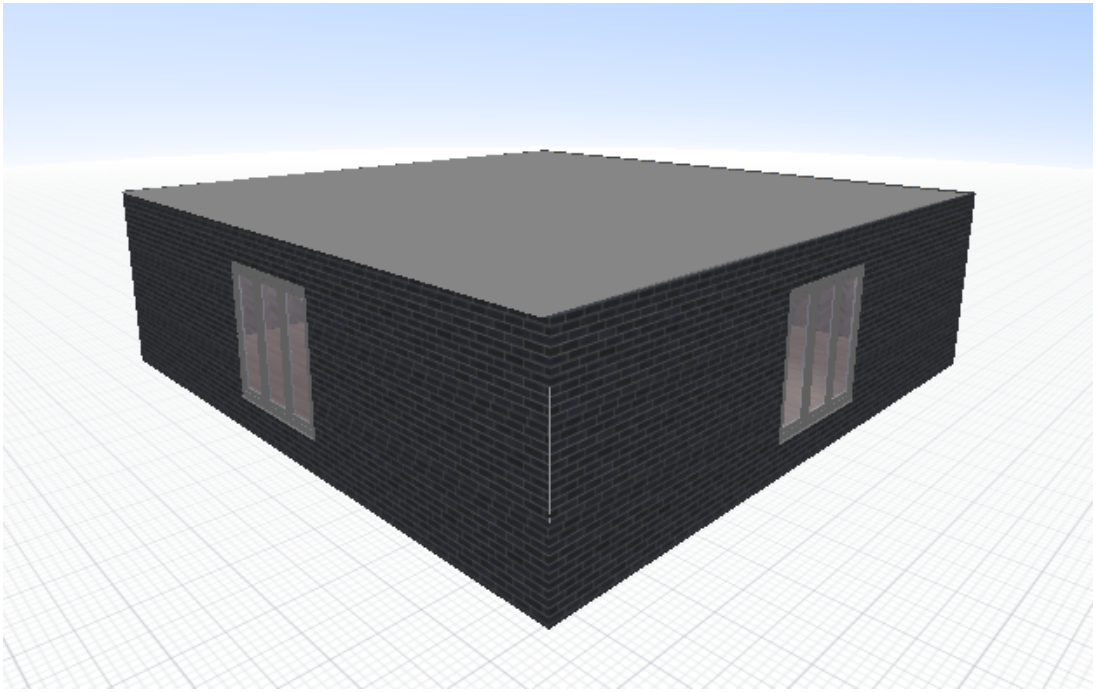


Figura 7: Representação da edificação estudada
Fonte: Autoria Própria

Em geral, são utilizadas, para tal fim, sequências binárias pseudo-randômicas, também conhecidas por "*PRBS*", nas quais um sinal oscila entre duas amplitudes fixas e sua função de autocorrelação se aproxima de ruído branco.

No entanto, não é possível capturar um comportamento não-linear utilizando-se desta abordagem. Por esta razão, em sistemas não-lineares são utilizadas sequências binárias pseudo-randômicas de amplitude modulada, também conhecidas por "*APRBS*". Sua principal particularidade é a randomização da amplitude de uma *PRBS* tradicional.

3.5 SELEÇÃO DE ENTRADAS

Para que a identificação de sistemas seja eficiente, variáveis relevantes devem ser levadas em consideração. Basear-se em variáveis que não apresentam relevância ou omitir variáveis úteis pode comprometer fortemente a precisão de predição do modelo, além de ocasionar erros sistemáticos no mesmo. A Tabela 1 apresenta as variáveis consideradas para o desenvolvimento deste trabalho, acompanhadas do fator de escalonamento utilizado para cada uma delas, afim de manter seu valor entre 0(zero) e 1(um).

Tabela 1: Variáveis utilizadas na identificação do sistema

VARIÁVEL	NOME DA VARIÁVEL	TIPO	FATOR ESC.
T_{ext}	Temperatura Externa à Zona	Entrada	21.0
Q_{solar}	Radiação Solar Direta e Difusa (W/m^2)	Entrada	839.8
T.F.M (MFR)	Taxa de Fluxo de Massa (kg/s)	Entrada	0.11
y	Temperatura do Ar na Zona (C)	Saída	21.0
Q	Fluxo de Calor do Radiador para a Zona (W)	Entrada /Saída	3385.4

Fonte: Autoria própria.

3.6 MODELO NÃO LINEAR ESTÁTICO DO *HAMMERSTEIN*

Nesta etapa, foi automaticamente gerado, através de aprendizado de máquina, um modelo não linear estático, representado por uma função definida em trechos, a partir de um conjunto de *splines* cúbicas (*piecewise cubic spline functions*). Para tal modelo, a entrada é representada pelo fluxo de massa de água quente que é passado pelo radiador (*MFR - Mass Flow Rate*). Já a saída se dá pelo fluxo de calor gerado (Q), demonstrado em *Watts* (W).

Para tal obtenção, o processo se vale de duas otimizações aninhadas, uma externa e outra interna. A otimização externa é realizada através de um algoritmo genético (AG), sobre uma população de funções definidas em trechos (*piecewise*). O AG define a quantidade e a posição dos nós de cada indivíduo da população. Cada um destes indivíduos, que é representado por uma função em trechos de *splines* cúbicas, é submetido, posteriormente, à otimização interna, realizada visando obter-se um conjunto de splines que atendam às restrições de suavidade e monotonicidade.

3.6.1 CODIFICAÇÃO DO MODELO

A Figura 8 representa o Diagrama *UML* de Classes que abstrai os códigos gerados para obtenção do modelo não-linear estático. As classes *CPWSPLine* e *CPWSPLineGA* implementam os métodos necessários tanto para a geração quando para a otimização das *splines*. A Figura 9, por sua vez, demonstra, através do Diagrama de Sequência da *UML*, o fluxo de chamadas de funções pelo qual passa a execução do modelo.

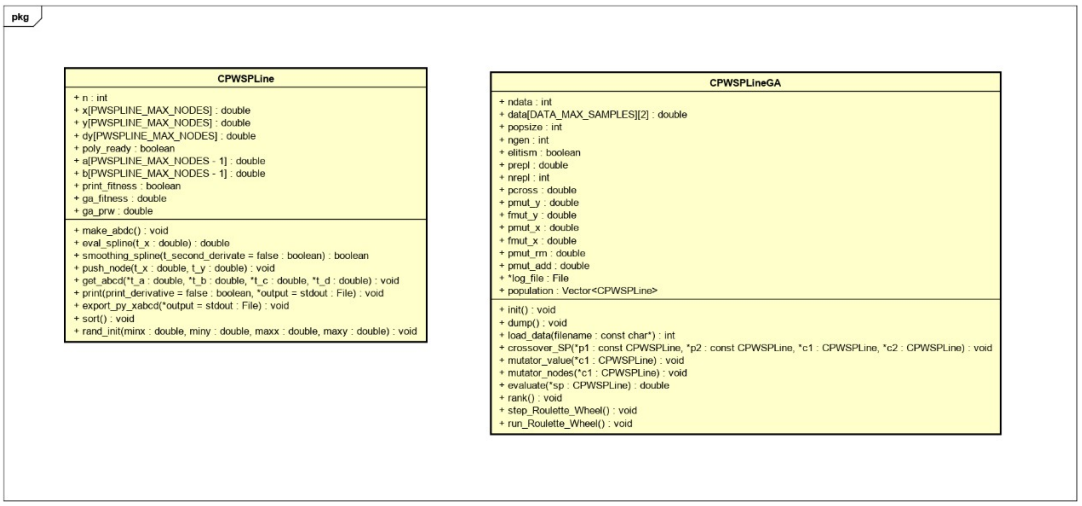


Figura 8: Representação do modelo em Diagrama de Classes
Fonte: Autoria Própria

3.6.2 OTIMIZAÇÃO EXTERNA COM ALGORITMOS GENÉTICOS

Inicialmente uma população aleatória é gerada e um número de nós iniciais é uniformemente sorteado, entre três e oito. Para cada novo nó, um novo ponto de duas dimensões é sorteado dentro do espaço (0,0) a (1,1). O algoritmo prossegue com um número determinado de gerações.

Em cada geração da otimização externa, o algoritmo genético, primeiramente, seleciona, através do método de seleção por roleta do *GAlib* (*GARouletteWheelSelector*), dois indivíduos da população, que serão os geradores de dois novos indivíduos (através da operação de *crossover*).

De acordo com Wall (1996, p.77), o *GARouletteWheelSelector* seleciona o indivíduo através da magnitude do valor de fitness do mesmo, em relação ao restante da população, sendo o indivíduo mais propenso a ser escolhido quando este valor é mais alto.

Após esta seleção, o algoritmo realiza o *crossover* a partir de um um corte simples, em determinado ponto de cada um dos indivíduos geradores, afim de determinar quais partes de cada um deles serão passadas a cada filho.

Cada um dos filhos gerados, receberá a parte anterior ao corte de um dos indivíduos pais e a posterior do outro, formando-se, desta maneira, dois indivíduos para cada par de pais.

O algoritmo genético também realiza durante o *crossover* o controle de tamanho

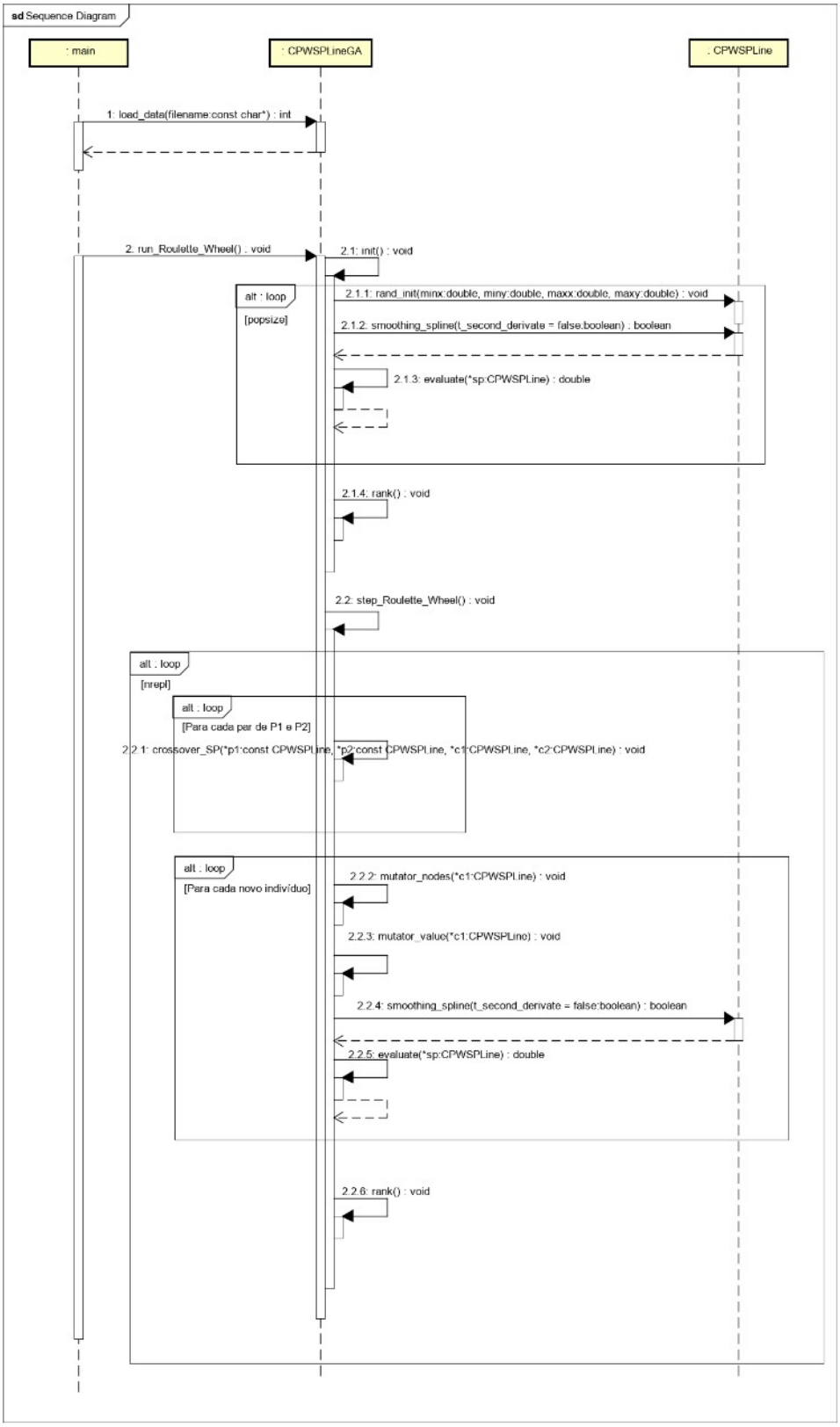


Figura 9: Representação do modelo em Diagrama de Sequência

Fonte: Autoria Própria

de cada filho gerado, impedindo que o corte seja realizado nos pais, de forma a gerar filhos que possuam tamanho maior que um limite pré-estabelecido de 8 nós e nem inferior a 2 nós.

Após o *crossover*, duas mutações: a primeira pode inserir novos nós em um intervalo aleatório ou remover nós aleatórios. A outra mutação pode deslocar os pontos no espaço.

Todos os operadores de *crossover* e mutação são projetados para garantir a ordem da dimensão x por construção. Já a monotonicidade da dimensão y não é garantida, uma vez que a otimização interna o fará.

Os operadores de *crossover* e mutação são então utilizados para gerar novos indivíduos da população. Em cada um destes novos indivíduos é aplicada a otimização interna para garantir a monotonicidade e suavidade da função, conforme descrito na Seção 3.6.3. Os novos indivíduos são gerados em número suficiente para substituir 25% da população. Os indivíduos removidos são aqueles com o menor valor de fitness.

3.6.2.1 REPRESENTAÇÃO (GENÓTIPO) DOS INDIVÍDUOS

Cada indivíduo é representado por um valor n, o número de nós, e n pares de nós, nas posições (x, y). Por exemplo, a representação $n = 4, \{(0.0, 0.0), (0.5, 0.5), (0.6, 0.8), (1.0, 1.0)\}$ define os nós da Figura 10

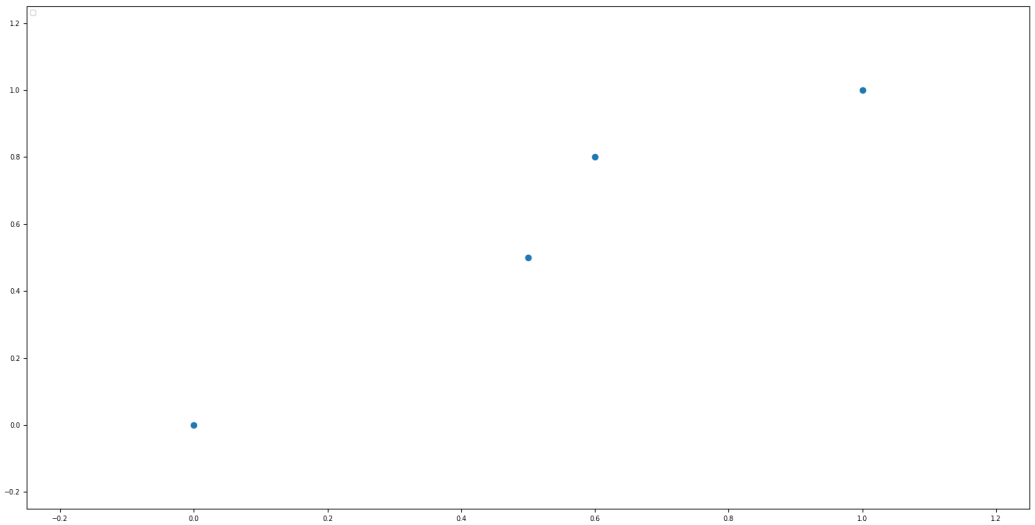


Figura 10: Representação dos nós da sequência

Fonte: Autoria Própria

3.6.2.2 OPERADOR DE *CROSSOVER*

O algoritmo que realiza os cortes nos indivíduos pais e a geração dos filhos é denominado Operador de *Crossover*.

A Figura 11 representa um exemplo de configuração de dois indivíduos pais (A e B), que serão submetidos ao Crossover, através do operador especificado. Cada indivíduo da população possui um conjunto de pontos, contendo dois valores reais em cada um, que representam, respectivamente, a posição x e y de cada ponto, em um plano cartesiano.

Figura 11: Indivíduos pais A e B selecionados pelo algoritmo
Fonte: Autoria Própria

A partir destes indivíduos, pais, o operador realizará, nos mesmos, um corte em posição aleatória, desde que respeitados o seguinte critério: o corte deverá ser realizado em posição posterior à primeira e anterior à ultima do conjunto de pontos do indivíduo. Além dessa restrição, deve-se observar, também, que os cortes deverão ser realizados de modo que os filhos a serem gerados não ultrapassem um tamanho máximo (MAX) definido. Para tal restrição ser atendida, o algoritmo realiza o corte do primeiro pai (A), de forma a atender apenas à primeira restrição e, após, realiza verificações antes do corte do segundo indivíduo pai (B),

A Figura 12 demonstra o corte realizado nos mesmos indivíduos apresentados anteriormente, garantindo que a restrição para o tamanho dos filhos seja cumprida.

É possível observar que o algoritmo garante a ordenação dos pontos em cada indivíduo, em referência ao valor de x. Por sua vez, a ordenação do valor y, que corresponde ao fator que garante monotonicidade ao conjunto de curvas, será ajustada pela otimização interna de cada indivíduo (ver seção 3.6.3).

Esta ordenação se dá através do uso do algoritmo de intercalação que é parte do

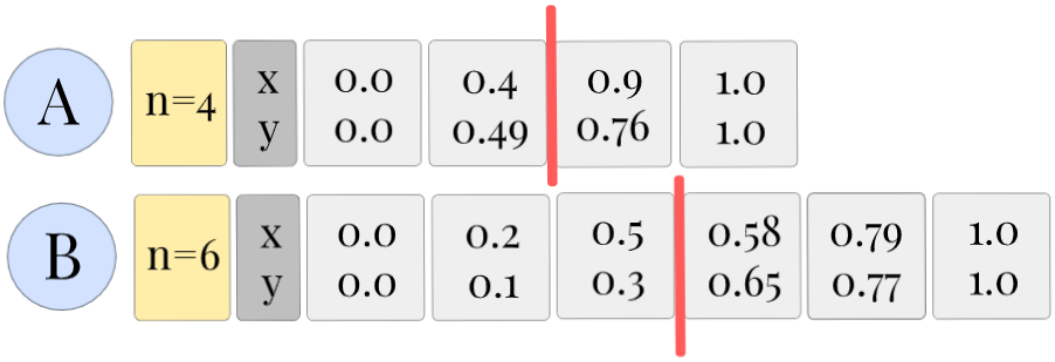


Figura 12: Corte realizado nos indivíduos A e B

Fonte: Autoria Própria

algoritmo de ordenação *Merge Sort*. O *Merge Sort* consiste, de acordo com Cormen et al. (2009, p.30-32), em um algoritmo que realiza a divisão de uma sequência de n elementos a ser ordenada em duas subsequências de tamanho $n/2$. Após isto, o algoritmo é executado, recursivamente, sobre estas sequencias menores, realizando o mesmo procedimento até que se obtenham diversas sequências de apenas um elemento e, posteriormente, insere cada um dos elementos individuais obtidos em uma única estrutura de sequência, que possuirá como tamanho a mesma quantidade de elementos da sequência inicial.

A Figura 13 demonstra o funcionamento básico do algoritmo *Merge Sort*. O reagrupamento (ou intercalação) ordenado se dá através da comparação dos menores valores de cada um dos dois vetores que será agrupado. Como a ordem dos vetores é garantida por execuções anteriores deste mesmo processo, o primeiro valor ainda não utilizado de cada vetor é escolhido. O menor valor entre os comparados é inserido na próxima posição livre do vetor resultante e o próximo valor do vetor em que estava o número passará a ser comparado. Esta implementação é demonstrada na Figura 14.

A partir dos indivíduos pais, A e B, são gerados dois filhos (A' e B'), ilustrados na Figura 15 e que, ao fim da execução do operador, devem haver atendido todas as especificidades supracitadas.

Por meio do processo de intercalação do algoritmo *Merge Sort*, o operador de *Crossover* realiza a construção ordenada dos filhos resultantes dos cortes em P1 e P2 em termos do valor de x. Para tal operação, são realizadas comparações entre o valor dado ao eixo x em cada um dos elementos de um dos trechos advindos dos indivíduos pais que compõe o filho com os elementos do trecho advindo do outro pai, um a um. No caso de um dos elementos comparados ser menor que o outro, este é reposicionado a frente do último na composição do filho. No caso de a diferença entre os dois elementos estar compreendida

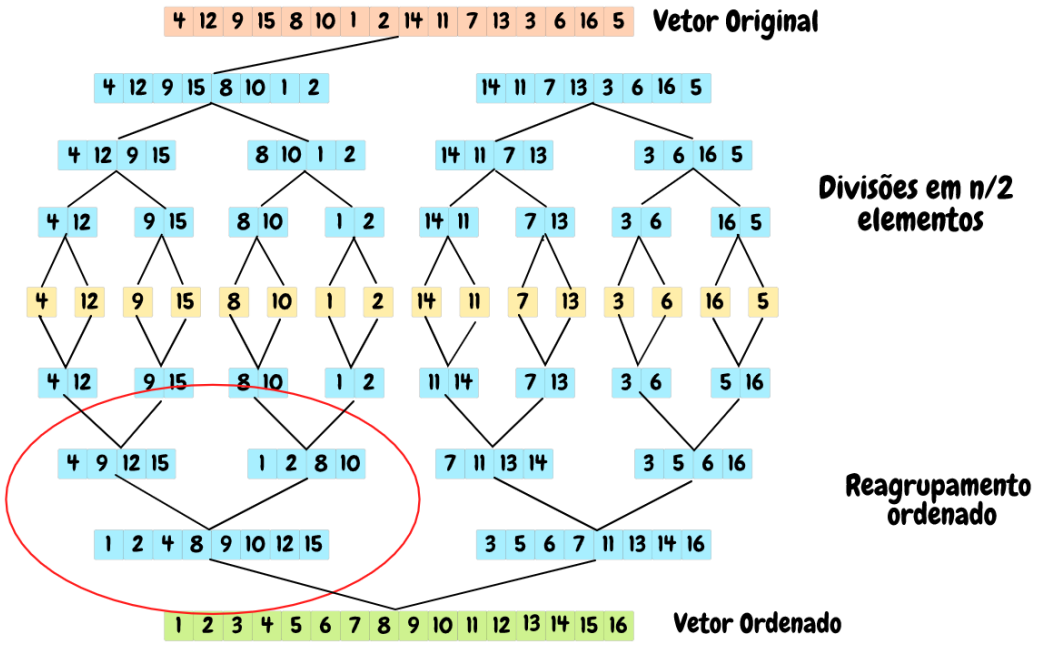


Figura 13: Exemplo de execução do *Merge Sort*

Fonte: Autoria Própria

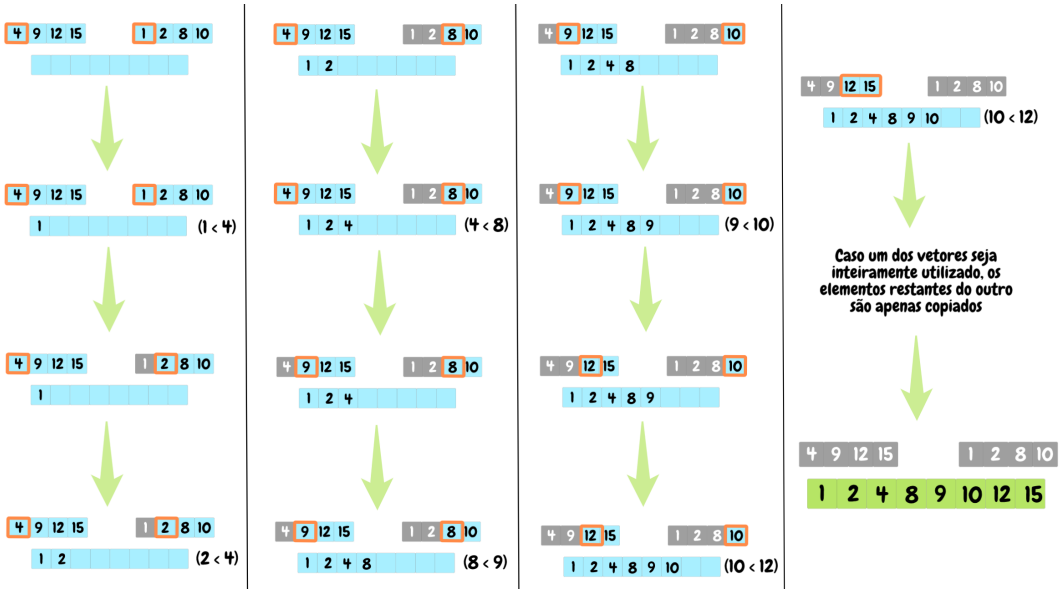


Figura 14: Execução do agrupamento ordenado de um dos trechos do vetor utilizado na Figura 13

Fonte: Autoria Própria

dentro de um pequeno intervalo, previamente definido e denominado *threshold*, o elemento que deveria ser inserido posteriormente no elemento filho é descartado. Este processo evita que variações com diferença insignificante sejam considerados desnecessariamente.

Figura 15: Conjunto de indivíduos filhos A' e B' gerados a partir do corte nos indivíduos A e B

Fonte: Autoria Própria

As Figuras 16, 17 e 18 representam a operação do algoritmo de *crossover* de forma abstrata. A partir dos elementos pais P1 e P2, na Figura 16, os filhos C1 e C2 são gerados. Nota-se, no entanto, que C2 seria um filho não ideal, por possuir valores que estão desordenados, além de possuir mais de um elemento que apresentem o valor para o eixo x com uma diferença insignificante. O algoritmo de *crossover* garante que este filho tenha sua ordem reestabelecida, como na Figura 17 e que o último valor semelhante inserido seja removido, como demonstra a Figura 18. Vale ressaltar que os passos descritos nas figuras são realizado durante a formação dos filhos no algoritmo. As figuras ilustram as alterações de forma diferenciada para melhor compreensão.

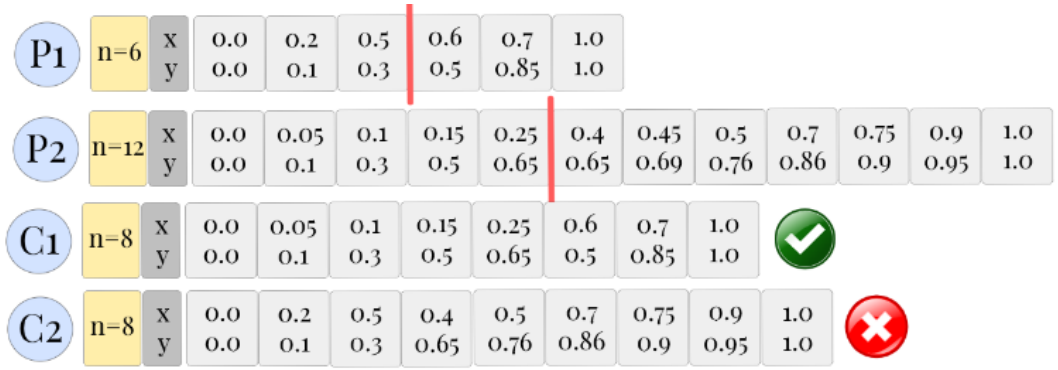


Figura 16: A geração de filhos seria errônea, caso as restrições de ordem e repetição de valores não fossem atendidas pelo algoritmo

Fonte: Autoria Própria

Figura 17: Uma das correções do algoritmo para a geração de filhos errôneos é a ordenação

Fonte: Autoria Própria

Figura 18: Outra correção do algoritmo para a geração de filhos errôneos é a remoção do ultimo valor inserido que esteja distante de seu antecessor em valor menor ou igual ao *threshold*

Fonte: Autoria Própria

3.6.2.3 OPERADOR DE MUTAÇÃO

O operador de mutação é o algoritmo responsável por adicionar alterações nos conjuntos de *splines* gerados a partir da execução do Operador de *Crossover* (indivíduos filhos). O algoritmo deve sortear probabilidades aleatórias para baseado, nas mesmas, realizar operações como adição, remoção e alteração de nós. Este procedimento permitirá que sejam testados diferentes configurações e valores, afim de se obter os melhores resultados de predição possíveis. Ele adiciona uma variação genética na população, o que auxilia na exploração global do espaço de busca.

O algoritmo percorrerá os pontos de cada conjunto gerado pelo operador de *Crossover*, sorteando a probabilidade de realização ou não de cada operação.

A operação de alteração modifica o valor das coordenadas x e y do ponto em questão para valores que estejam compreendidos dentro do intervalo do valor de cada variável no ponto anterior e no ponto posterior ao atual.

Por sua vez, a operação de adição, caso esta seja realizada para o ponto atual,

insere na sequência um novo ponto, com coordenadas de x e y, logo após o ponto atual. O valor de x deve ser adicionado levando em consideração a ordenação exigida para a otimização externa. O ordenação de y não é necessária neste momento. No caso da operação de remoção, o ponto atual é removido do sistema. O algoritmo deliberadamente não remove os pontos extremos.

As Figuras 19, 20 e 21 demonstram as mutações possíveis sendo realizadas para o filho B', definido anteriormente.

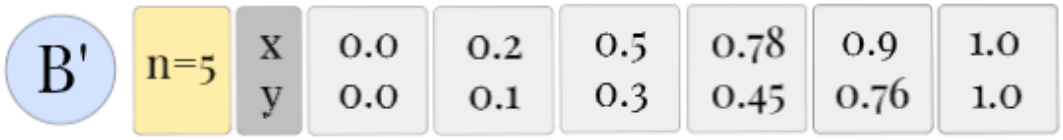


Figura 19: Demonstração da adição de um novo ponto no conjunto B'
Fonte: Autoria Própria

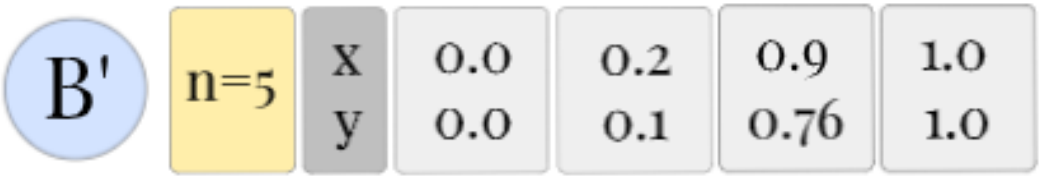


Figura 20: Demonstração da adição de um novo ponto no conjunto B'
Fonte: Autoria Própria

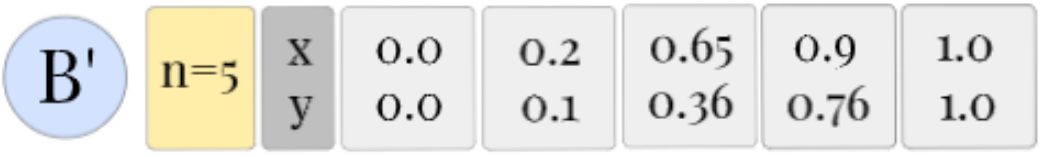


Figura 21: Demonstração da adição de um novo ponto no conjunto B'
Fonte: Autoria Própria

3.6.3 OTIMIZAÇÃO INTERNA: CURVAS SUAVES E MONOTÔNICAS

A otimização interna consiste no segundo passo da obtenção do modelo. Conforme descrito em Wolberg e Alfy (2002, p.175), a otimização deve ser realizada através das

etapas de minimização e de garantias da continuidade da segunda derivada para que se atinga a monotonicidade.

A minimização se dá através da seguinte fórmula:

$$E_s = \sum_{k=0}^{n-1} |(y_k - f(k))| \tag{3}$$

A continuidade da segunda derivada é calculada, caso possível, conforme:

$$\begin{aligned} -y'_{k-1}\frac{1}{\Delta x_{k-1}} - y'_k\left[\tfrac{2}{\Delta x_k} + \tfrac{2}{\Delta x_{k-1}}\right] - y'_{k+1}\frac{1}{\Delta x_k} - y_{k-1}\frac{3}{\Delta x_{k-1}^2} + \\ y_k\left[\tfrac{3}{\Delta x_{k-1}^2} - \tfrac{3}{\Delta x_k^2}\right] + y_{k+1}\frac{3}{\Delta x_k^2} = 0 \end{aligned} \tag{4}$$

Da mesma forma, é considerada a garantia da monotonicidade:

$$sign(m_k)\begin{bmatrix} -1 & 0 \\ 0 & -1 \\ 1 & -1 \\ -1 & 1 \\ 2 & 1 \\ 1 & 2 \end{bmatrix}\begin{bmatrix} y'_k \\ y'_{k+1} \end{bmatrix} \leq |m_k|\begin{bmatrix} 0 \\ 0 \\ 3 \\ 3 \\ 9 \\ 9 \end{bmatrix} \tag{5}$$

Em relação à Equação (4), é possível que não exista solução possível. Neste caso, a execução é realizada novamente, desta vez, sem possuir a restrição de continuidade da segunda derivada.

A restrição de monotonicidade limita a escolha a curvas monotônicas dentro da família de todas as possíveis curvas entre dois pontos. a Figura 22 ilustra algumas das possibilidades passíveis de curvas definidas entre os pontos (x_k, y_k) e (x_{k+1}, y_{k+1}).

(WOLBERG; ALFY, 2002) provam que uma *spline*, definida no intervalo (x_k, y_k) e (x_{k+1}, y_{k+1}), contando com primeiras derivadas em cada nó y'_k e y'_{k+1}, respectivamente, é monotônica se os valores de α e β estiverem nos intervalos definidos na Figura 23. É possível destacar, ainda, na Figura 22, que para um determinado valor de $\alpha_k = 0.0$ e $\beta_k = 5.0$ a curva não é monotônica. Contudo, para os valores $\alpha_k = 0.0$ e $\beta_k = 1.0$ obtém-se uma curva com esta característica.

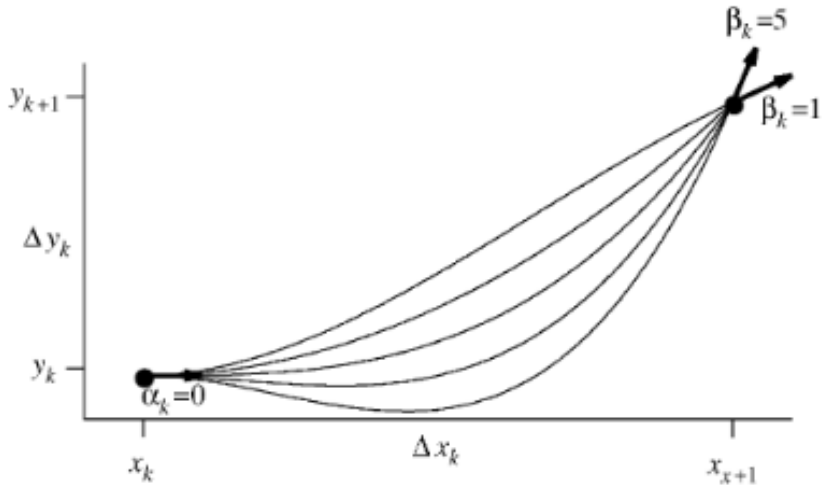


Figura 22: Família de polinômios cúbicos interpolados
Fonte: Wolberg e Alfy (2002, p.150)

Os valores de α e β são definidos como:

$$\alpha_k = y'_k/m_k \tag{6}$$

$$\beta_k = y'_{k+1}/m_k \tag{7}$$

Onde m_k representa o declive (ou gradiente) entre os nós, sendo:

$$m_k = \frac{\Delta_y}{\Delta_x} = \frac{y_{k+1} - y_k}{x_{k+1} - x_k} \tag{8}$$

Deve-se observar que, por tratar-se de uma função definida em trechos de *splines* cúbicas, o último ponto, assim como as derivadas de sua k-ésima *spline*, corresponde ao primeiro ponto da k-ésima *spline*. É possível notar, portanto, que *splines* consecutivas compartilham pontos e derivadas.

3.7 APLICAÇÃO

Os resultados desta etapa poderão ser utilizados como parâmetros de entrada para a implementação e aplicação do segundo bloco do modelo *Hammerstein*, correspondente a um modelo dinâmico, que poderá ser modelado através do uso da estrutura de erros de

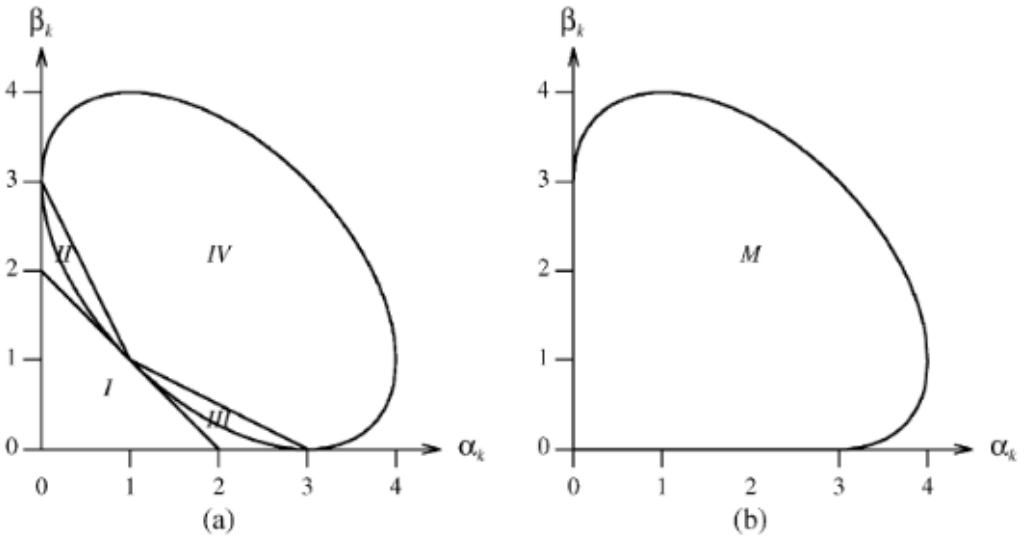


Figura 23: Pares (α, β) para uma curva monotônica
Fonte: Wolberg e Alfy (2002, p.153)

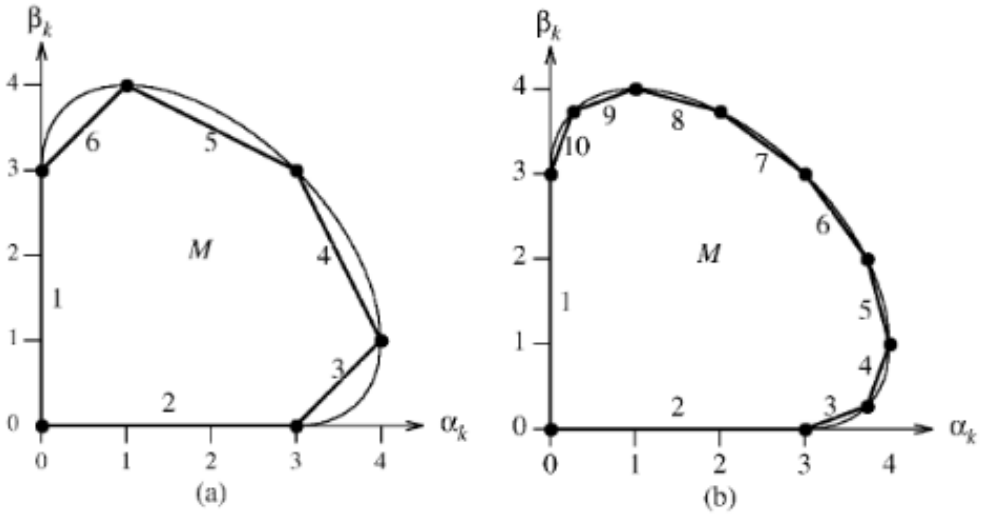


Figura 24: Aproximação linear para uma região M: $(a) n = 6; (b) n = 10$.
Fonte: Wolberg e Alfy (2002, p.157)

saída (através do modelo *OE*).

Ao fim de uma execução dos dois blocos combinados, obter-se-á o resultado da temperatura obtida para o tempo imediatamente futuro ao da execução do processo. Este conjunto resultante deverá ser comparado com valores de referência, obtidos por sistemas preditivos que possuem base em conforto térmico. A função de otimização resultante desta

comparação será dada como descrito a seguinte fórmula:

$$\left(\sum_i (Y_i - R_i)^2\right) + \left(\sum_i |X_i - X_{i-1}|\right) + \left(\sum_i X_i\right) \tag{9}$$

Neste cálculo, primeiro bloco representa o somatório das diferenças de todos os valores de temperaturas obtidas pelo modelo (representados por 'Y') e os valores de referencia encontrados baseados em conforto térmico (dadas por 'R'). O bloco seguinte consiste no somatório de controle forte para cada valor encontrado, dado pelo módulo da diferença deste, representado por 'X_i' para seu antecessor direto, dado por 'X_{i-1}'. O último bloco indica o somatório das quantidades de energias gastas em cada descoberta, determinado, como forma simplificada, por 'X_i'.

4 RESULTADOS

Nesta seção, estão descritos os dados obtidos na experimentação exposta de acordo com o capítulo 3.

4.1 AQUISIÇÃO DE DADOS

Na Figura 25 é apresentada a sequência de excitação aplicada ao sistema do *MPC*. A sequência em azul representa o fluxo de massa de água (*MFR*) aplicado, que interfere nas sequências em vermelho, indicativas da quantidade de calor (Q) resultante. O trecho em preto simboliza a temperatura atingida na sala. Nesta sequência, 54% dos valores registrados durante o período das 17:00h às 6:00h do dia posterior, correspondem a zeros. Os demais valores utilizam a sequência APRBS apresentada na Figura 25. Isto se deve ao fato de, neste estudo, estarem sendo abordados sistemas de aquecimento de edificações não residenciais, conforme disposto na seção 2.1. Neste tipo de construção, devido a jornada de trabalho, a ocupação se dá, majoritariamente, na faixa de horário das 9:00h até às 17:00h. O início antecipado em 3 horas se deve à estimativa do *delay* do sistema, que é estimado em 3 horas.

A Figura 26 tem por objetivo representar o fato de a mudança na quantidade de calor gerada no ambiente tender a apresentar leve diminuição, quase imperceptível, quando mantido um fluxo constante de massa de água passando pelo radiador. Isso gera um aumento também sutil na temperatura atingida na sala.

4.2 MODELO NÃO LINEAR ESTÁTICO

Uma vez que o objetivo do modelo é obter *splines* suaves e monotônicas, por questões de parametrização, a Figura 27 demonstra um exemplo que, por não ser monotônico, não atende a esta definição.

A Figura 28 demonstra um exemplo de spline que atende o estado de suave e

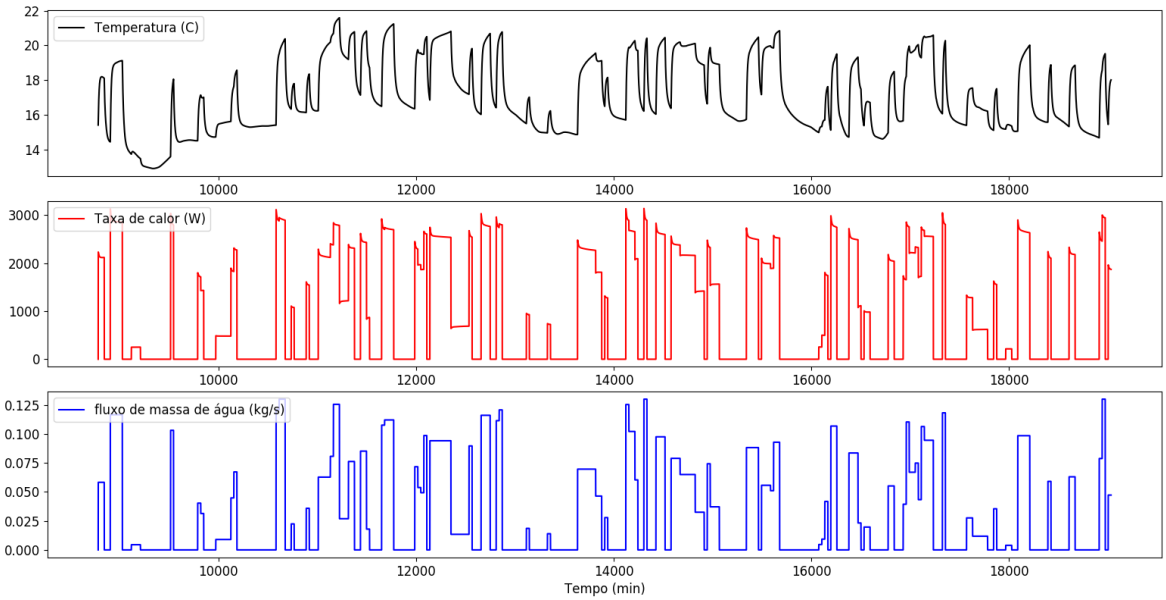


Figura 25: Representação dos dados aplicados ao sistema

Fonte: Autoria Própria

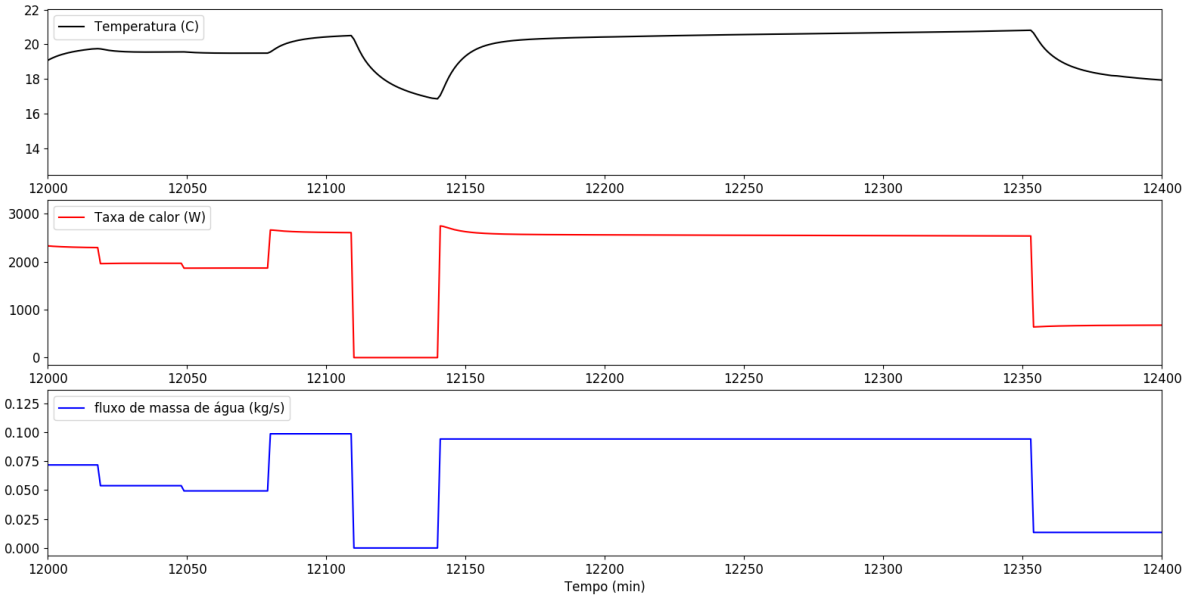


Figura 26: Demostração do comportamento da quantidade de calor e da temperatura com a manutenção constante do fluxo de massa de água

Fonte: Autoria Própria

monotônica, após haver sido submetia às otimizações pela primeira e segunda derivativa.

Para atestar a assertividade do modelo não-linear estático, foram realizados 3 (três) experimentos distintos, realizados, no entanto, utilizando-se do mesmo conjunto de dados - a sequência de excitação em fluxo de massa de água ao longo do tempo (ilustrado

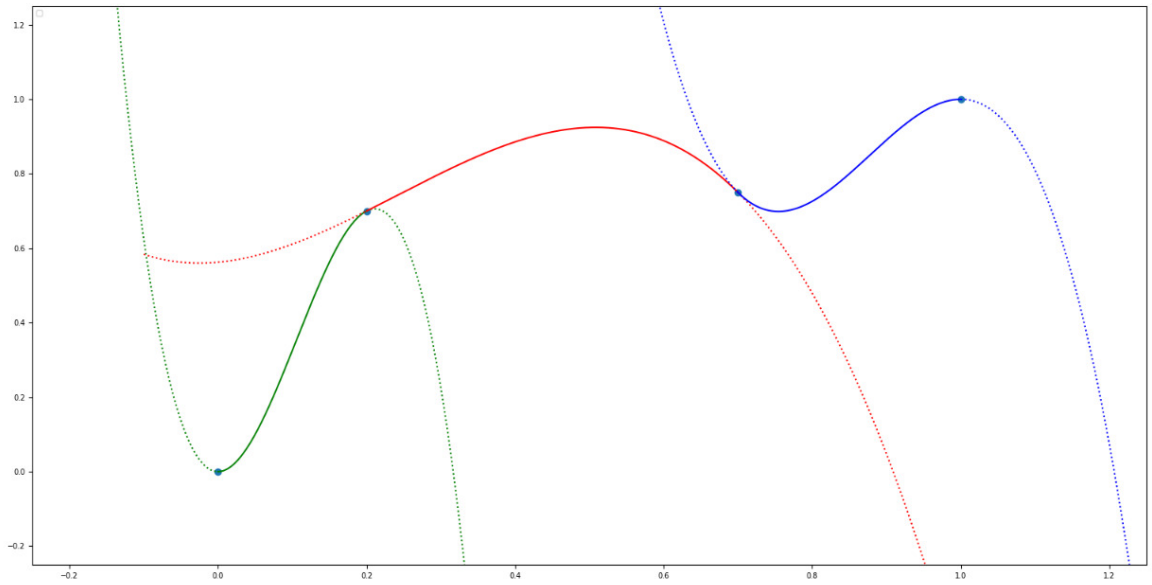


Figura 27: Exemplo de *spline* não monotônica
Fonte: Autoria Própria

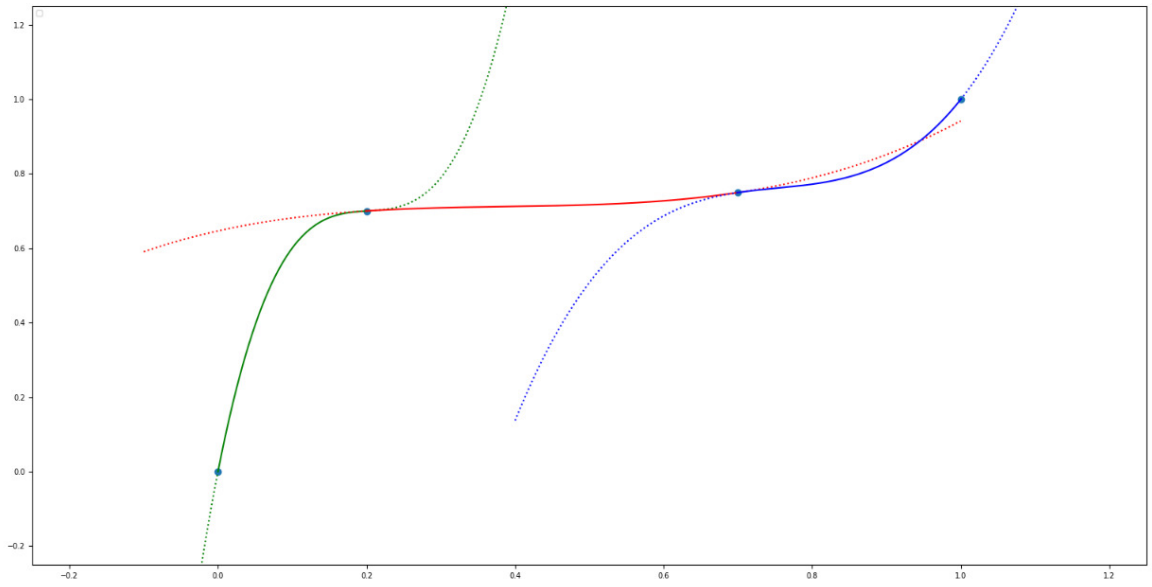


Figura 28: Exemplo de *spline* suave e monotônica
Fonte: Autoria Própria

na Figura 25).

Notou-se que o modelo foi capaz de obter uma otimização eficiente dos conjuntos de *splines* obtidos na fase anterior, obtendo valores de *fitness* que convergiram a quantidades cada vez mais próximas de zero.

O Experimento 1 inicializou vinte indivíduos, nos quais os valores de *fitness* variaram entre 0,00225408 e 0,13974185. Estes indivíduos deram origem a 507 (quinhentas e sete) gerações de indivíduos filhos. O menor valor de *fitness* encontrado foi de 0,00014053, observado desde a 424ª (quadringentésima vigésima quarta) geração.

A Figura 29 ilustra a convergência das *splines* obtidas, para as gerações 3, 15, 30 e 500, respectivamente, de acordo com o esperado. Os pontos em azul demonstram por quais valores é esperado que o conjunto de *splines* passe. Cada curva de cor distinta representa uma das *splines* do conjunto. Vale ressaltar que apenas um trecho, suave e monotônico de cada *spline* é aproveitado no conjunto final.

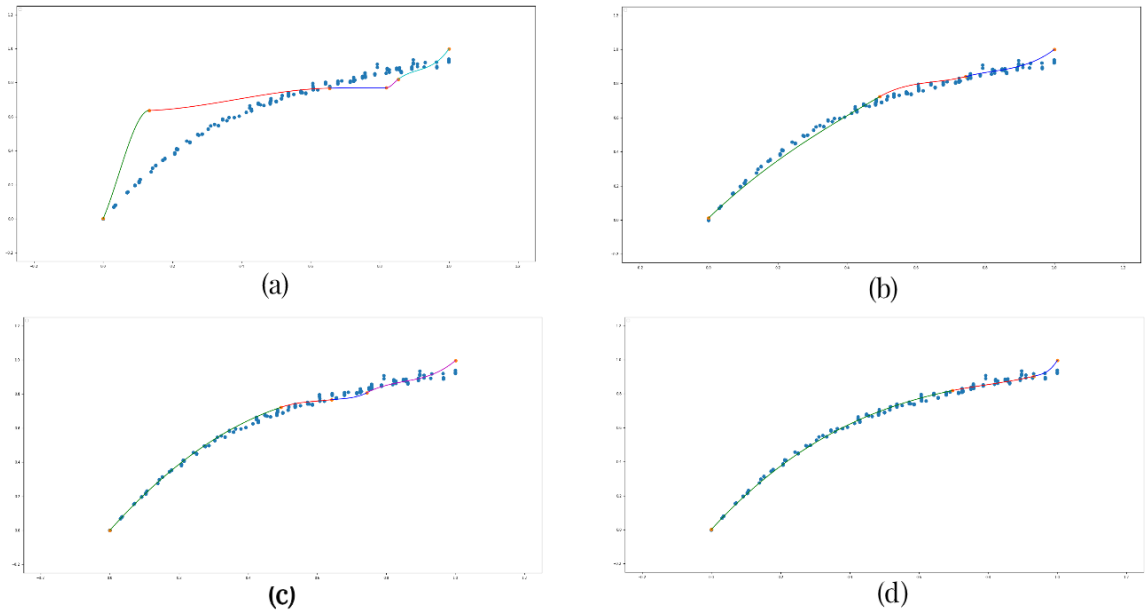


Figura 29: Conjuntos de *splines* obtidos para as gerações 3 (a), 15 (b), 30 (c) e 500 (d), no Experimento 1

Fonte: Autoria Própria

A Figura 30 apresenta os mesmos conjuntos das gerações 15 e 500, da Figura 29. Neste caso, no entanto, os trecho pontilhados de cada linha de cor distinta representam as curvas originais de cada *spline*, das quais apenas um trecho foi aproveitado.

O Experimento 2, também implementou vinte indivíduos, com *fitness* entre 0,00122997 e 0,17648850. Foram obtidos 1225 (mil duzentos e vinte e cinco) gerações de conjuntos de *splines*, com menor *fitness* observado de 0,00011517, desde a 731ª (septingentésima trigésima primeira) geração. As *splines* obtidas nas gerações 1, 10, 156 e 471 do Experimento 2 são representadas na Figura 31 e suas curvas originais são ilustradas na Figura 32.

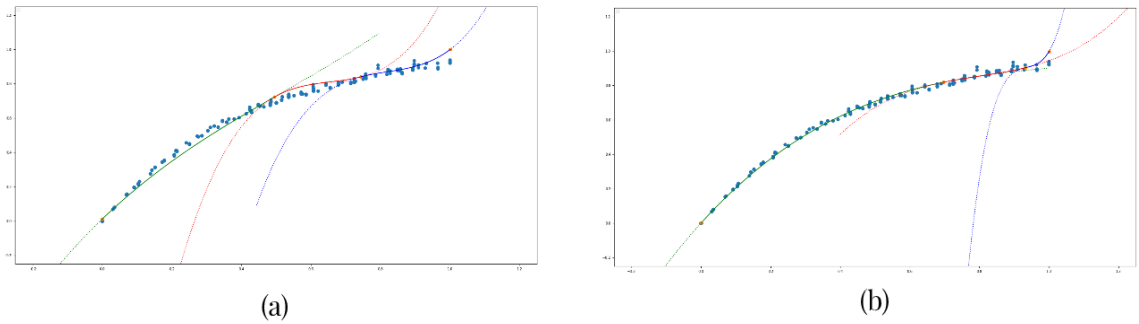


Figura 30: Conjuntos de *splines* obtidos para as gerações 15 (a) e 500 (b), no Experimento 1, com suas continuações

Fonte: Autoria Própria

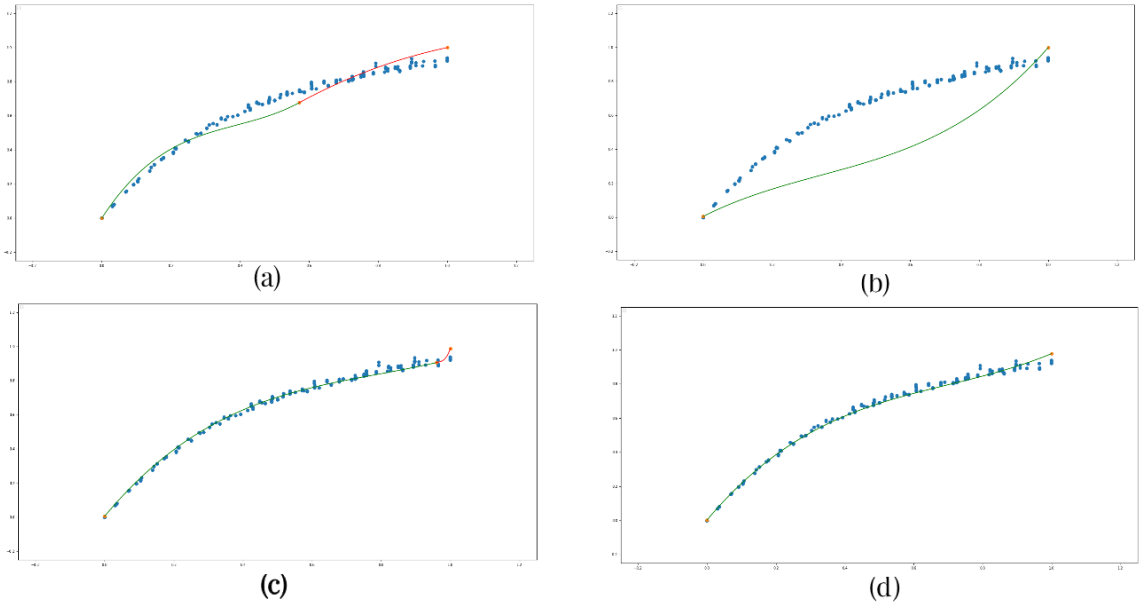


Figura 31: Conjuntos de *splines* obtidos para as gerações 1 (a), 10 (b), 156 (c) e 500 (d), no Experimento 2

Fonte: Autoria Própria

O Experimento 3, por sua vez apresentou *fitness* entre 0,00062183 e 0,14569326 nos vinte indivíduos inicializados. Após a geração de 571 (quinhentos e setenta e uma) gerações de indivíduos filhos, o menor *fitness* foi registrado na 471ª (quadringentésima septuagésima primeira) geração, e teve o valor de 0,00009572. o conjunto de *splines* obtidas para as gerações 1, 10, 355 e 471 é mostrada na Figura 33. Suas curvas originais são ilustradas na figura 34.

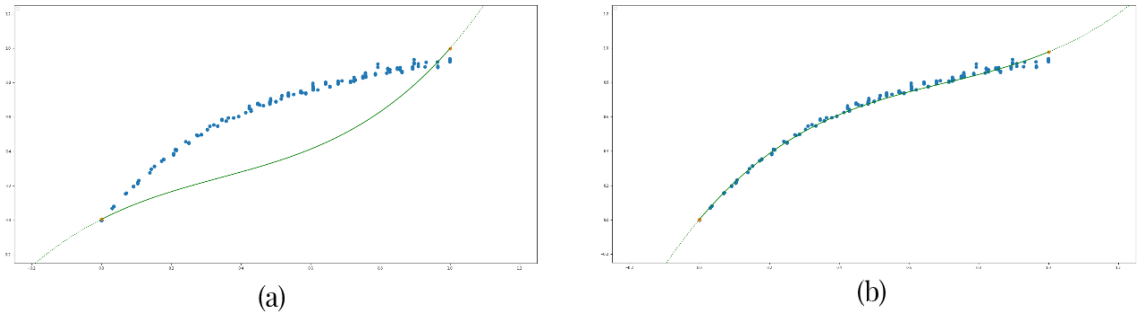


Figura 32: Conjuntos de *splines* obtidos para as gerações 10 (a) e 500 (b), no Experimento 2, com suas continuações

Fonte: Autoria Própria

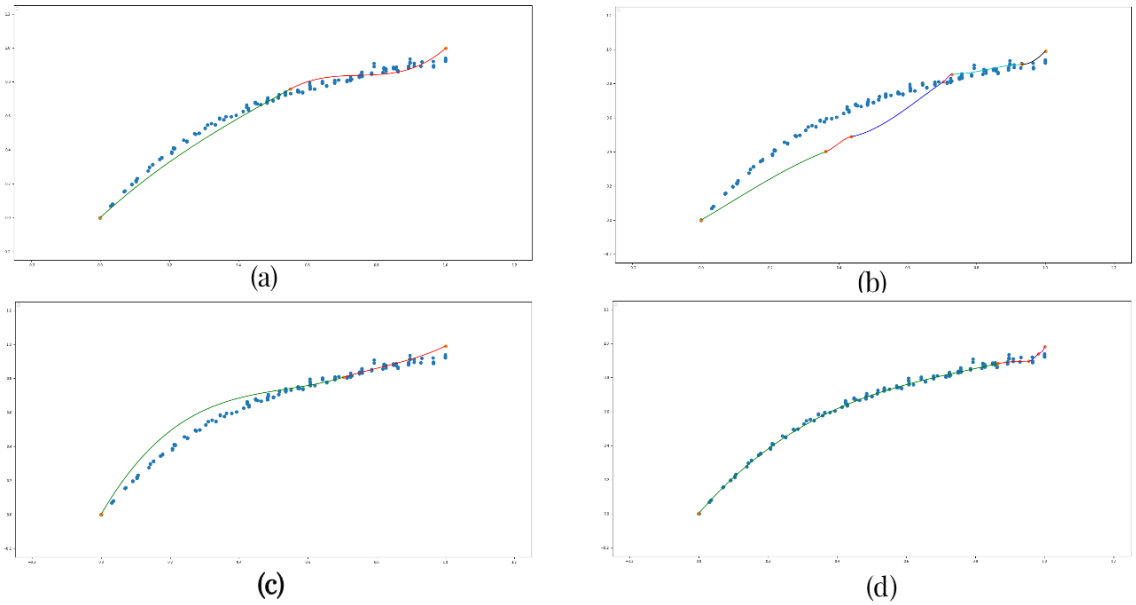
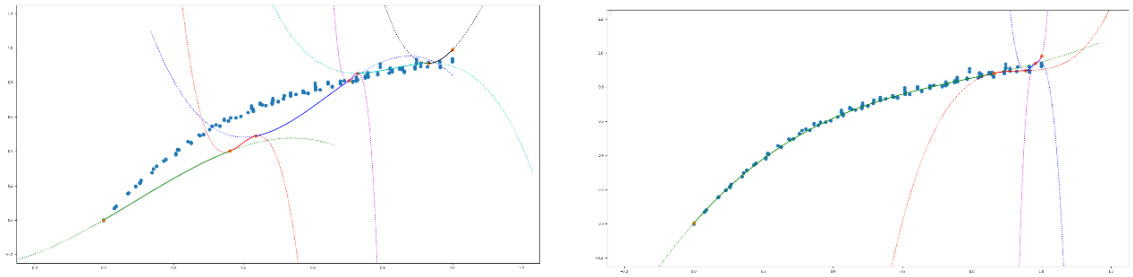


Figura 33: Conjuntos de *splines* obtidos para as gerações 1 (a), 10 (b), 355 (c) e 471 (d), no Experimento 3

Fonte: Autoria Própria

A Figura 35 apresenta o gráfico indicador da convergência gradual do valor de *fitness*, em função das gerações de conjuntos de *splines* geradas pelo modelo para o Experimento 1. É possível notar que o valor apresentou quedas cada vez menores ao decorrer das gerações. Isto se deve ao fato de os conjuntos evoluírem em termos de conformidade com a expectativa, ao decorrer do tempo.

Os Apêndices A, B e C demonstram as saídas da execução do modelo para cada



(a) (b)

Figura 34: Conjuntos de *splines* obtidos para as gerações 10 (a) e 471 (b), no Experimento 3, com suas continuações

Fonte: Autoria Própria

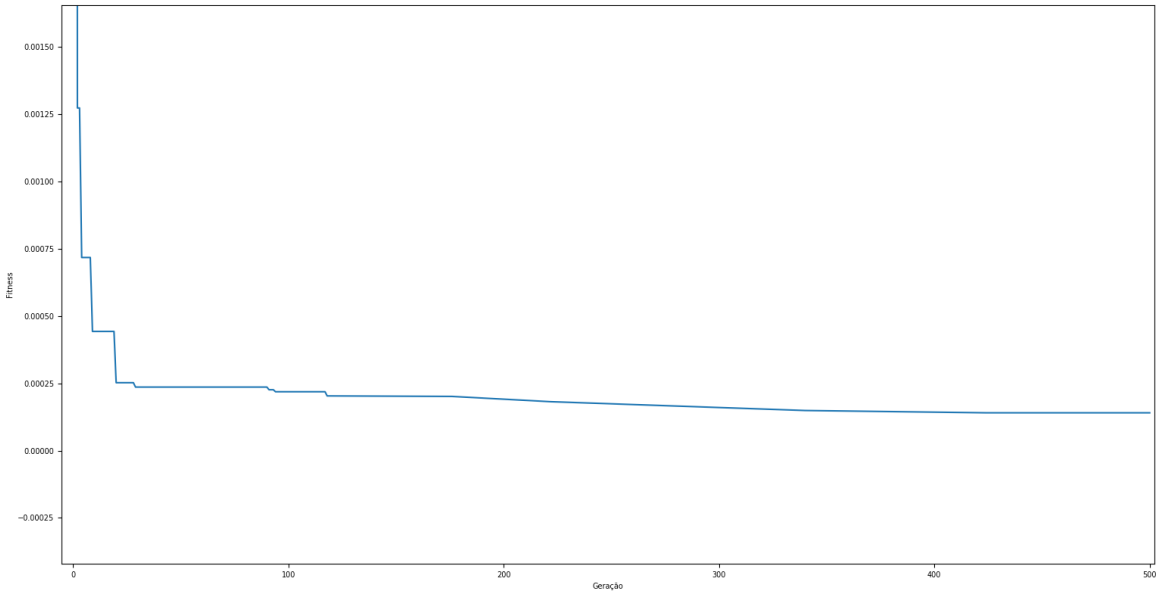


Figura 35: Gráfico de convergência dos valores de *fitness* para o Experimento 1

Fonte: Autoria Própria

um dos Experimentos realizados.

Percebe-se que os conjuntos de *splines* obtidos ao fim de cada Experimento, tidos como ótimos pelo modelo, consistem em soluções bastante próximas, ainda que ligeiramente diferentes entre si. Isto permite aferir que o modelo atingiu a assertividade esperada e poderá atuar como predecessor do modelo linear dinâmico.

Notou-se uma dificuldade do algoritmo em otimizar o último ponto de cada conjunto de *splines*, que tende a ficar muito próximo do valor $x = 1.0$, $y = 1.0$, não sendo

otimizado e convergindo para dentro do intervalo esperado. Não foi possível aferir, com clareza, causas para este impasse, que pode haver sido gerado por erro de implementação.

5 CONCLUSÕES

Este estudo obteve resultados satisfatórios, na obtenção de um modelo não-linear estático, conciso e assertivo, que foi capaz de executar, com considerável exatidão, a convergência das curvas definidas entre os pontos, realizando uma otimização, tanto interna quanto externa, destas curvas, afim de garantir que atendam as especificidades previstas no escopo. Este modelo relaciona a variável de entrada, fluxo de massa de água (MFR), com a variável de saída, quantidade de calor (Q).

Conforme observado nos dados adquiridos (ver Figuras 25 e 26), a relação *MFR* x Q é claramente não-linear mas não estritamente estática. Porém, é verificado que um modelo estático oferece uma boa aproximação, já que após alguns poucos minutos, (uma fração da constante de tempo do sistema) a variação de Q se estabiliza, conforme ilustrado na Figura 26. Nota-se que a constante de tempo do sistema em estudo é de aproximadamente 30 minutos.

A otimização externa trata-se de um algoritmo genético onde a função de *fitness* é a média do erro quadrático entre o valor previsto pelo modelo e o valor real. A otimização interna encontra valores para cada nó, de forma a garantir a suavidade e monotonicidade da função conforme descrito por (WOLBERG; ALFY, 2002).

Este estudo pode, ainda, contribuir para uma sequência na obtenção do modelo *Hammerstein* completo e sua aplicação no sistema do *MPC*, possibilitando e viabilizando estudos futuros para atender esta necessidade. Assim, o resultado deste trabalho poderá ser complementado com um modelo linear dinâmico que relacionará a variável controlável de entrada: quantidade de calor (Q), a variável auto regressiva: temperatura da sala e outras variáveis não controláveis, tais como temperatura externa e radiação solar. Um possível candidato para este modelo linear dinâmico é o modelo de Erro de Saída ou *Output Error* (OE). Em trabalhos futuros, sugere-se, portanto, a finalização do modelo *Hammerstein* com OE e seu comparativo com resultados preliminares de um modelo desenvolvido em *GP*.

Indiretamente, o sucesso deste trabalho impacta positivamente o esforço para redução de custos operacionais e financeiros da aplicação do *MPC* em edificações não-residenciais, contribuindo para que a aplicação de sistemas robustos de controle seja economicamente viável e amplamente aplicada, tornando a redução de emissão de CO_2 possível, a partir do menor consumo de energia.

REFERÊNCIAS

BILLINGS, S. A. **Nonlinear System Identification NARMAX Methods in the Time, Frequency, and Spatio-Temporal Domains**. 1. ed. [S.l.]: John Wiley & Sons, Ltd., 2013.

BRENT, R. **Algorithms for minimization without derivatives**. [S.l.]: Prentice-Hall, 1973.

CAMACHO, E. F.; BORDONS, C. **Model Predictive Control**. 2. ed. London: Springer-Verlag, 2004.

CASILLO, D. S. da S. **Controle preditivo não linear baseado no modelo de Hammerstein com prova de estabilidade**. Tese (Doutorado) — Doutorado em Automação e Sistemas; Engenharia de Computação; Telecomunicações - Universidade Federal do Rio Grande do Norte, 2009.

CASSANDRAS, C. G.; LAFORTUNE, S. **Introduction to Discrete Event Systems**. 2. ed. [S.l.]: Springer, 2007.

CORMEN, T. et al. **Introduction to Algorithms**. 3. ed. Cambridge, Estados Unidos: Elsevier, 2009.

CRAWLEY, D. B. et al. Energy plus: Energy simulation program. **ASHRAE Journal**, ASHRAE, v. 42, n. 4, p. 49–56, 2017.

DOU, T.; ROCKETT, P. Semantic-based local search in multiobjective genetic programming. In: **Genetic and Evolutionary Computation Conference Companion (GECCO '17)**. Berlin, Germany: [s.n.], 2017. p. 225–226. Disponível em: <https://doi.org/10.1145/3067695.3076015>.

GALIB: Screenshots. Disponível em: <http://lancet.mit.edu/ga/ScreenShots.html>. Acesso em: 15 de novembro de 2019.

HAZYUK, I.; GHIAUS, C.; PENHOUET, D. Optimal temperature control of intermittently heated buildings using model predictive control: Part ii e control algorithm. **Building and Environment**, Lausanne, Elsevier Sequoia, v. 51, n. 0, p. 388–394, 2012.

KAN, A. H. G. R.; TIMMER, G. T. Stochastic global optimization methods part i: Clustering methods. **Genetic Programming and Evolvable Machines**, Springer, v. 39, n. 1, p. 27–56, 1987.

LI, H. et al. Genetic algorithm search space splicing particle swarm optimization as general-purpose optimizer. **Chemometrics and Intelligent Laboratory Systems**, Elsevier, v. 128, n. 15, p. 153–159, 2013.

OLIVEIRA, D. de Pinho Rebouças de. **Sistemas de informação gerenciais: estratégias, táticas, operacionais**. 16. ed. São Paulo: Atlas, 2002.

POWELL, M. J. D. A direct search optimization method that models the objective and constraint functions by linear interpolation. In: SPRINGER, DORDRECHT. **Gomez S., Hennart JP. (eds) Advances in Optimization and Numerical Analysis. Mathematics and Its Applications**. Dordrecht, 1994. v. 275, p. 51–67. Disponível em: <https://doi.org/10.1007/978-94-015-8330-5_4>. Acesso em: 8 jun. 2019.

PRESSMAN, R. S. **Engenharia de software: uma abordagem profissional**. 7. ed. São Paulo: AMGH, 2011.

PRIVARA, S. et al. Building modeling as a crucial part of building predictive control. **Energy and Buildings**, Lausanne, Elsevier Sequoia, v. 56, n. 0, p. 8–22, 2013.

ROCKETT, P. **Pruning of Genetic Programming Trees Using Permutation Tests**. [S.l.], 09 2018.

ROCKETT, P.; DOU, T. Comparison of semantic-based local search methods for multiobjective genetic programming. **IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)**, Springer, v. 35, n. 2, p. 233–243, 2005.

ROCKETT, P.; HATHWAY, E. A. Model-predictive control for non-domestic buildings: a critical review and prospects. **Building Research & Information**, Routledge, v. 45, n. 5, p. 556–571, 2017.

ROCKETT, P.; HATHWAY, E. A.; SYKES, J. Big data and model predictive control for improved building performance. In: **2018 Stretching the Envelope**. Londres: CIBSE Technical Symposium 2018. The Chartered Institution of Building Services Engineers (CIBSE), 2018.

ROCKETT, P.; LOPES, Y. K.; DOU, T. **GPML: An XML-based Standard for the Interchange of Genetic Programming Trees**. [S.l.], 08 2018.

ROCKETT, P. et al. d(tree)-by-dx: Automatic and exact differentiation of genetic programming trees.

RUNARSSON, T.; YAO, X. Search biases in constrained evolutionary optimization. **Genetic Programming and Evolvable Machines**, Springer, v. 19, n. 4, p. 535–563, 2018.

SANTOS, C. H. da S.; GONCALVES, M. S.; FIGUEROA, H. E. H. Designing novel photonic devices by bio-inspired computing. **IEEE Photonics Technology Letters**, Springer, v. 22, n. 15, p. 1177– 1179, 2010.

VAND, B. et al. Sensitivity analysis of building physical parameters to maximize heating energy saving using mpc. In: **3rd International Conference of Energy Harvesting, Storage, and Transfer (EHST'19)**. [s.n.], 2019. Disponível em: <http://dx.doi.org/10.11159/ehst19.136>.

WALL, M. **GAlib: A C++ Library of Genetic Algorithm Components**. Estados Unidos: Massachusetts Institute of Technology, 1996.

WOLBERG, G.; ALFY, I. An energy-minimization framework for monotonic cubic spline interpolation. **Journal of Computational and Applied Mathematics**, Elsevier, v. 143, p. 145–188, 2002.

6 APÊNDICE A - SAÍDA DE EXECUÇÃO DO EXPERIMENTO 1

```
Loading data from sqlite (v. 3.30.1)...Loaded 672 records[done]
initing individual 1/20 ... Fitness = 0.13974185
...
initing individual 8/20 ... Fitness = 0.00225408
initing individual 9/20 ... Fitness = 0.02031246
...
initing individual 14/20 ... Fitness = 0.13505901
initing individual 15/20 ... Fitness = 0.00130623
initing individual 16/20 ... Fitness = 0.01668795
...
initing individual 20/20 ... Fitness = 0.02084923

Gen 1:
BEST: 19 0.02031246
x = [0.0000000000, 0.0392218835, 0.7223970215, 1.0000000000]
a = [-900.9520337271, 1.9432669581, -11.9753771126]
b = [-146.2613386384, -1.5597558450, 5.2250116669]
c = [16.0941035368, 0.4628578676, 1.0526179100, 1.1849851921]
d = [0.0001018350, 0.3519799138, 0.5598353189, 0.9985127816]

...

Gen 423:
BEST: 16 0.00020504
x = [0.0000000000, 0.5585057543, 0.9319044181, 1.0000000000]
a = [1.9668983560, 2.6688564865, -509.3513588669]
b = [-3.0632334684, -2.1945067236, 50.1014561669]
```

c = [2.4367927522, 0.8557217239, 0.3331984214, 0.0709746464]
d = [0.0024899204, 0.7506034838, 0.9031014884, 0.9972790242]

Gen 424:
BEST: 1 0.00014053
x = [0.0000000000, 0.6965681417, 0.9319044181, 1.0000000000]
a = [0.6120192095, 3.3797814015, 230.8581452959]
b = [−1.9408614839, −1.0345928522, −2.6208197902]
c = [2.2279484860, 0.4149322687, 0.4895267626, 3.3440676564]
d = [0.0014138496, 0.8184620339, 0.9028625709, 0.9969401341]

Gen 425:
BEST: 1 0.00014053
x = [0.0000000000, 0.6965681417, 0.9319044181, 1.0000000000]
a = [0.6120192095, 3.3797814015, 230.8581452959]
b = [−1.9408614839, −1.0345928522, −2.6208197902]
c = [2.2279484860, 0.4149322687, 0.4895267626, 3.3440676564]
d = [0.0014138496, 0.8184620339, 0.9028625709, 0.9969401341]

...

Gen 507:
BEST: 1 0.00014053
x = [0.0000000000, 0.6965681417, 0.9319044181, 1.0000000000]
a = [0.6120192095, 3.3797814015, 230.8581452959]
b = [−1.9408614839, −1.0345928522, −2.6208197902]
c = [2.2279484860, 0.4149322687, 0.4895267626, 3.3440676564]
d = [0.0014138496, 0.8184620339, 0.9028625709, 0.9969401341]

7 APÊNDICE B - SAÍDA DE EXECUÇÃO DO EXPERIMENTO 2

```
Loading data from sqlite (v. 3.30.1)...Loaded 672 records[done]
initing individual 1/20 ... Fitness = 0.01425027
...
initing individual 5/20 ... Fitness = 0.00122997
initing individual 6/20 ... Fitness = 0.07752595
...
initing individual 16/20 ... Fitness = 0.09696315
initing individual 17/20 ... Fitness = 0.00695745
initing individual 18/20 ... Fitness = 0.17648850
initing individual 19/20 ... Fitness = 0.01216109
initing individual 20/20 ... Fitness = 0.05784300

Gen 1:
BEST: 1 0.00122997
x = [0.0000000000, 0.5715200335, 1.0000000000]
a = [5.6794808070, 0.1255657969]
b = [-6.6380071601, -0.8629513209]
c = [3.1222135272, 1.1000595520, 0.4297046342]
d = [0.0000700715, 0.6765083301, 0.9993060531]

Gen 2:
BEST: 1 0.00122997
x = [0.0000000000, 0.5715200335, 1.0000000000]
a = [5.6794808070, 0.1255657969]
b = [-6.6380071601, -0.8629513209]
c = [3.1222135272, 1.1000595520, 0.4297046342]
d = [0.0000700715, 0.6765083301, 0.9993060531]
```

```
Gen 3:
BEST: 1 0.00122997
x = [0.0000000000, 0.5715200335, 1.0000000000]
a = [5.6794808070, 0.1255657969]
b = [-6.6380071601, -0.8629513209]
c = [3.1222135272, 1.1000595520, 0.4297046342]
d = [0.0000700715, 0.6765083301, 0.9993060531]

...

Gen 730:
BEST: 18 0.02246656
x = [0.0000000000, 0.9591526159, 1.0000000000]
a = [0.4616903147, 2624.3668837037]
b = [-0.3445546844, -93.4402955755]
c = [0.8489494356, 1.4622172793, 6.9649716097]
d = [0.0018535460, 0.9065376828, 0.9892211554]

Gen 731:
BEST: 1 0.00011517
x = [0.0000000000, 1.0000000000]
a = [1.2904481194]
b = [-2.7553857926]
c = [2.4396241479, 0.8001969209]
d = [0.0018340776, 0.9765205523]

Gen 732:
BEST: 1 0.00011517
x = [0.0000000000, 1.0000000000]
a = [1.2904481194]
b = [-2.7553857926]
c = [2.4396241479, 0.8001969209]
d = [0.0018340776, 0.9765205523]
```

...

```
Gen 1225:
BEST: 1 0.00011517
x = [0.0000000000, 1.0000000000]
a = [1.2904481194]
b = [-2.7553857926]
c = [2.4396241479, 0.8001969209]
d = [0.0018340776, 0.9765205523]
```

8 APÊNDICE C - SAÍDA DE EXECUÇÃO DO EXPERIMENTO 3

```
Loading data from sqlite (v. 3.30.1)...Loaded 672 records[done]
initing individual 1/20 ... Fitness = 0.14569326
initing individual 2/20 ... Fitness = 0.11467105
...
initing individual 7/20 ... Fitness = 0.00062183
initing individual 8/20 ... Fitness = 0.07716246
initing individual 9/20 ... Fitness = 0.02615525
...
initing individual 18/20 ... Fitness = 0.01314021
initing individual 19/20 ... Fitness = 0.04463564
initing individual 20/20 ... Fitness = 0.01447240

Gen 1:
BEST: 1 0.00062183
x = [0.0000000000, 0.5503562989, 1.0000000000]
a = [0.8066750405, 9.1692568801]
b = [-1.3473012906, -5.4244952774]
c = [1.8725050672, 1.1225199476, 1.8058458383]
d = [0.0002986076, 0.7572282687, 0.9988061817]

Gen 2:
BEST: 1 0.00062183
x = [0.0000000000, 0.5503562989, 1.0000000000]
a = [0.8066750405, 9.1692568801]
b = [-1.3473012906, -5.4244952774]
c = [1.8725050672, 1.1225199476, 1.8058458383]
d = [0.0002986076, 0.7572282687, 0.9988061817]
```

```
Gen 3:
BEST: 1 0.00062183
x = [0.0000000000, 0.5503562989, 1.0000000000]
a = [0.8066750405, 9.1692568801]
b = [-1.3473012906, -5.4244952774]
c = [1.8725050672, 1.1225199476, 1.8058458383]
d = [0.0002986076, 0.7572282687, 0.9988061817]

...

Gen 470:
BEST: 18 0.00024096
x = [0.0000000000, 0.7038069668, 0.7299224776]
a = [1.9996278654, 328.5599030212]
b = [-3.2405940824, -20.1487994607]
c = [2.4524731096, 0.8624802009, 0.4823410027]
d = [0.0030482434, 0.8210299970, 0.8356643041]

Gen 471:
BEST: 1 0.00009572
x = [0.0000000000, 0.8639462024, 0.9536965963, 0.9816707732,
1.0000000000]
a = [0.9801914399, 21.5560457644, -1809.6265252524, 8952.6189514305]
b = [-2.3531813192, -5.0806461586, 103.9208707269, -124.3270149564]
c = [2.3173062865, 0.4461157714, 0.0550462433, 1.6208533416,
6.0863977197]
d = [0.0034700834, 0.8811548390, 0.8958525966, 0.9391011092,
0.9821704320]

Gen 472:
BEST: 1 0.00009572
x = [0.0000000000, 0.8639462024, 0.9536965963, 0.9816707732,
1.0000000000]
a = [0.9801914399, 21.5560457644, -1809.6265252524, 8952.6189514305]
```

b = [−2.3531813192, −5.0806461586, 103.9208707269, −124.3270149564]
c = [2.3173062865, 0.4461157714, 0.0550462433, 1.6208533416, 6.0863977197]
d = [0.0034700834, 0.8811548390, 0.8958525966, 0.9391011092, 0.9821704320]

...

Gen 571:
BEST: 1 0.00009572
x = [0.0000000000, 0.8639462024, 0.9536965963, 0.9816707732, 1.0000000000]
a = [0.9801914399, 21.5560457644, −1809.6265252524, 8952.6189514305]
b = [−2.3531813192, −5.0806461586, 103.9208707269, −124.3270149564]
c = [2.3173062865, 0.4461157714, 0.0550462433, 1.6208533416, 6.0863977197]
d = [0.0034700834, 0.8811548390, 0.8958525966, 0.9391011092, 0.9821704320]