

UNIVERSIDADE ESTADUAL DE MARINGÁ
CENTRO DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA QUÍMICA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA QUÍMICA

Hanniel Ferreira Sarmiento de Freitas

**Desenvolvimento de uma ferramenta em Python
orientada a equações aplicada à simulação de processos
biotecnológicos**

Maringá - PR - Brasil

Junho de 2019

UNIVERSIDADE ESTADUAL DE MARINGÁ
CENTRO DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA QUÍMICA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA QUÍMICA

**Desenvolvimento de uma ferramenta em Python
orientada a equações aplicada à simulação de processos
biotecnológicos**

Hanniel Ferreira Sarmiento de Freitas
Eng^o Químico, UFRN, 2011
Me. Eng^a Química, UFRN, 2013
Orientador: Prof. Cid M. G. Andrade
Coorientador: Prof. José Eduardo Olivo

Tese de Doutorado submetida à Universidade Estadual de Maringá, como parte dos requisitos necessários à obtenção do Grau de Doutor em Engenharia Química, área de Desenvolvimento de Processos.

Maringá - PR - Brasil

Junho de 2019

UNIVERSIDADE ESTADUAL DE MARINGÁ
CENTRO DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA QUÍMICA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA QUÍMICA

Esta é a versão final da Tese de Doutorado apresentada por Hanniel Ferreira Sarmiento de Freitas perante a Comissão Julgadora do Curso de Doutorado em Engenharia Química em 28 de junho de 2019.

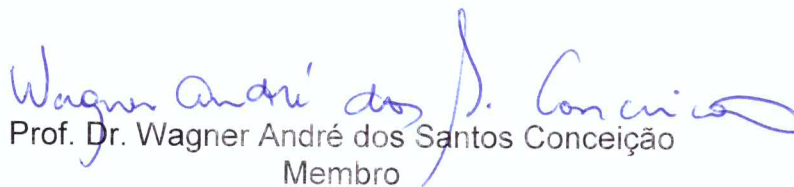
COMISSÃO JULGADORA



Prof. Dr. Cid Marcos Gonçalves Andrade
Presidente/Orientador



Prof. Dr. José Eduardo Olivo
Coorientador



Prof. Dr. Wagner André dos Santos Conceição
Membro



Prof. Dr. Paulo Eduardo Polon
Membro



Prof. Dr. Rafael Krummenauer
Membro



Prof.^a Dr.^a Mônica Ronobo Coutinho
Membro

Agradecimentos

Agradeço à minha família, a qual sempre me brindou com a compreensão e o apoio que eu por tantas vezes precisei, ao longo da construção desse trabalho. Em especial, à minha companheira Isis, cujo amor, carinho e compreensão foram alicerce para esse e tantos outros projetos que houveram, e que ainda haverão.

Agradeço ao meu orientador, Professor Cid Marcos Gonçalves Andrade, pelas orientações e em especial, pela confiança em nosso trabalho, pela compreensão das minhas limitações e pelas valiosas lições ao longo dessa caminhada. Agradeço também ao co-orientador, o Professor José Eduardo Olivo, o qual sempre se mostrou acessível, humano, e fez considerações valiosas para o crescimento desse trabalho.

Agradeço a todos que de contribuíram direta ou indiretamente para o desenvolvimento deste trabalho. Foi uma longa jornada, árdua e desafiadora em muitos aspectos, que contribuiu para o meu crescimento intelectual, profissional e, em última análise, enquanto pessoa.

“Um livro é a prova de que os homens são capazes de fazer magia.”
(CARL SAGAN)

DESENVOLVIMENTO DE UMA FERRAMENTA EM PYTHON ORIENTADA A EQUAÇÕES APLICADA À SIMULAÇÃO DE PROCESSOS BIOTECNOLÓGICOS

AUTOR: HANNIEL FERREIRA SARMENTO DE FREITAS

ORIENTADOR: PROF. CID M. G. ANDRADE

COORIENTADOR: PROF. JOSÉ EDUARDO OLIVO

Tese de Doutorado; Programa de Pós-Graduação em Engenharia Química; Universidade Estadual de Maringá; Av. Colombo, 5790, BL E46 – 09; CEP: 87020-900 – Maringá – PR, Brasil, defendida em 28 de junho de 2019. 148 p.

RESUMO

Os processos biotecnológicos (ou bioprocessos) tem se tornado cada vez mais importantes sob o ponto de vista industrial nas últimas décadas, em razão não só da necessidade de diversificação das matrizes energéticas rumo a adoção de fontes renováveis, cujo representante absoluto é o bioetanol, mas também motivado por um esforço em substituir toda a economia atual que é seriamente dependente de combustíveis fósseis (não só no aspecto energético, mas também como matéria-prima para a produção de plásticos, solventes, embalagens, etc) em uma economia baseada na biotecnologia, em que rotas produtivas industriais são continuamente substituídas pelas suas contrapartes baseadas em bioprocessos. Nesse sentido, o estudo da modelagem, simulação, controle e otimização dessa classe de processos industriais mostra-se como um importante tema de pesquisa, haja vista que os processos biotecnológicos apresentam diversas complexidades inerentes ao fato de que são baseados, em última análise, no metabolismo de seus micro-organismos de trabalho, o que traduz-se em uma resposta do processo por vezes altamente não-linear e transiente, em que o próprio produto desejado no bioprocessos apresenta uma ação inibitória ao metabolismo microbiano, a exemplo do que ocorre no processo produtivo do bioetanol por meio de fermentação.

Assim, o presente trabalho objetiva documentar o desenvolvimento de uma ferramenta orientada a equações para a modelagem e simulação de processos, capaz de executar estudos de otimização e controle, e utilizá-la em casos de estudo de otimização e controle de processos biotecnológicos, a partir de dois modelos muito populares para a descrição do processo de produção de bioetanol *in-silico* (portanto, de cunho computacional) nos regime de batelada alimentada e de fermentação contínua, envolvendo dois dos mais populares micro-organismos produtores de bioetanol em âmbito industrial, respectivamente: a levedura *Saccharomyces cerevisiae*, e a bactéria *Zymomonas mobilis*.

Convém mencionar que a ferramenta não foi desenvolvida com o intuito de rivalizar com outras ferramentas gratuitas de modelagem e simulação de processos, tais como ASCEND, EMSO, GAMS, DAETOOLS, mas para servir de base para a compreensão das complexidades do desenvolvimento de uma ferramenta de modelagem e simulação de processos orientada a equações, ou mesmo como a base para outra ferramenta, uma vez que o concebeu-se o projeto como um código a ser contribuído colaborativamente pelos interessados no futuro. Nesse sentido, a ferramenta foi desenvolvida utilizando a linguagem computacional *Python*, de altíssimo

nível, que conta com um grande ecossistema de bibliotecas que permitem que suas capacidades sejam expandidas para os mais diversos propósitos.

Foram realizados estudos de otimização em malha aberta, utilizando rotinas de otimização meta-heurística, e estudos de controle adotando a abordagem DRTO, utilizando uma rotina meta-heurística como algoritmo de otimização. A escolha pelos métodos meta-heurísticos deu-se em função das propriedades dos mesmos de robustez quando aplicados a problemas altamente não-lineares, a exemplo daqueles oriundos de bioprocessos. Os resultados mostram que a ferramenta foi capaz de desenvolver os casos de estudo desejados, entretanto fazem-se necessários alguns aperfeiçoamentos em sua estrutura computacional, especificamente no que diz respeito às rotinas empregadas para a integração e resolução de problemas diferenciais, que não mostraram-se robustas o suficiente para lidar com problemas para os quais são esperadas oscilações e bifurcações nas soluções. Ainda assim, pode-se afirmar que por ter o código aberto e uma premissa de projeto colaborativo, a ferramenta desenvolvida mostrou-se promissora, e pode servir como base para projetos futuros na área da modelagem e simulação de processos sob uma abordagem de orientação a equações.

Palavras-chaves: Modelagem e simulação, Controle de processos, Biorreatores, Otimização, Simuladores orientados a equações, Python

DESENVOLVIMENTO DE UMA FERRAMENTA EM PYTHON ORIENTADA
A EQUAÇÕES APLICADA À SIMULAÇÃO DE PROCESSOS BIOTECNOLÓGICOS

DEVELOPMENT OF AN EQUATION-ORIENTED TOOL IN PYTHON
APPLIED TO BIOTECHNOLOGICAL PROCESS SIMULATION

AUTHOR: HANNIEL FERREIRA SARMENTO DE FREITAS
SUPERVISOR: PROF. CID M. G. ANDRADE
COSUPERVISOR: PROF. JOSÉ EDUARDO OLIVO

Doctoral Thesis; Chemical Engineering Graduate Program; State University of Maringá;
Av. Colombo, 5790, BL E46 – 09; CEP: 87020-900 – Maringá – PR, Brasil, presented on
28th June 2019. 148 p.

ABSTRACT

Biotechnological processes (or bioprocesses) have become increasingly important from the industrial point of view in the last decades, due not only to the need to diversify the energy matrices towards the adoption of renewable sources, whose absolute representative is bioethanol, but also motivated by an effort to replace the entire current economy that is seriously dependent on fossil fuels (not only in the energy aspect but also as raw material for the production of plastics, solvents, packaging, etc.) in a biotechnology-based economy, in which industrial production routes are continually replaced by their bioprocess-based counterparts. In this sense, the study of modeling, simulation, control and optimization of this class of industrial processes is an important research topic, since the biotechnological processes present several complexities inherent to the fact that they are based, ultimately, on the metabolism of their working microorganisms, which translates itself into a response of the process that is sometimes highly nonlinear and transient, in which the desired product in the bioprocess has an inhibitory action on microbial metabolism, as observed in the production process of bioethanol through fermentation.

Thus, the present work aims to document the development of an equation-oriented tool for modeling and simulation of processes, capable of performing optimization and control studies, and to use it in case studies of optimization and control of biotechnological processes, from two very popular models for the description of the *in-silico* (hence computational-based) bioethanol production process in fed batch and continuous fermentation regimes, involving two of the most popular bioethanol producing micro-organisms in respectively, *Saccharomyces cerevisiae* yeast, and the *Zymomonas mobilis* bacterium.

It should be mentioned that the tool was not developed in order to rival other free process modeling and simulation tools, such as ASCEND, EMSO, GAMS, DAE-TOOLS, but to serve as a basis for understanding the complexities of developing a modeling and simulation of processes oriented to equations, or even as the basis for another tool, since the project was conceived as a code to be contributed collaboratively by stakeholders in the future. In this sense, the tool was developed using the high-level computational language *Python*, which has a large ecosystem of libraries that allow its capabilities to be expanded for a variety of purposes.

Were performed open-loop optimization using meta-heuristic optimization routines, and control studies adopting the DRTO approach, using a meta-heuristic routine as an optimization algorithm. The choice of meta-heuristic methods was based on their properties of robustness when applied to highly non-linear problems, such as those derived from bioprocesses. The results show that the tool was able to develop the desired case studies, however, some improvements are necessary in its computational structure, specifically with respect to the routines used for the integration and resolution of differential problems, which were not shown robust enough to handle problems for which oscillations and bifurcations in the solutions are expected. Nevertheless, it can be said that, because it has open source and a premise of collaborative design, the developed tool has shown to be promising, and can serve as a basis for future projects in the area of modeling and simulation of processes under a equation-oriented approach.

Key-words: Modeling and simulation, Process control, Biorreactors, Optimization, Equation-oriented simulators, Python

Lista de ilustrações

Figura 1 – Representação esquemática da interseção entre as diversas áreas que compõem a biotecnologia.	8
Figura 2 – Representação esquemática dos tipos de modelos mecanísticos utilizados na descrição dos processos biotecnológicos, destacando a diferença entre cada um deles	9
Figura 3 – Representação esquemática do processo cíclico e contínuo de implementação do PAT.	19
Figura 4 – Representação esquemática de algumas das diferentes configurações possíveis para a aplicação de sensores para o monitoramento de bioprocessos.	19
Figura 5 – Representação esquemática de uma estrutura de monitoramento e controle de um processo biotecnológico, a partir de uma multitude de sensores diferentes.	20
Figura 6 – Representação gráfica da forma da função <i>sphere</i> (esfera), ilustrando a aparência gráfica de uma função unimodal. A referida função matemática possui apenas um mínimo global.	28
Figura 7 – Representação gráfica da forma da função <i>holder table</i> (mesa de suporte), ilustrando a complexidade inerente à otimização de funções multimodais. A referida função matemática possui diversos pontos de mínimo local e quatro pontos de mínimos globais.	28
Figura 8 – Representação gráfica da forma da função <i>egg holder</i> (caixa de ovos), ilustrando a complexidade inerente à otimização de funções multimodais. A referida função matemática possui diversos pontos de mínimo local e um mínimo global.	29
Figura 9 – Representação gráfica da <i>maldição da dimensionalidade</i>	30
Figura 10 – Imagem da plataforma MIDAS, desenvolvida a partir do trabalho de LaTorre et al. (2015), que permite a comparação de resultados obtidos com algoritmos de otimização a partir de diversos problemas de referência. Disponível em: < http://vps128.cesvima.upm.es/lab/ >.	35
Figura 11 – Representação esquemática do procedimento da modelagem de um processo.	39
Figura 12 – Representação esquemática da relação entre os diferentes componentes da ferramenta que são voltados a interação direta com o usuário.	43
Figura 13 – Representação esquemática da relação entre os diferentes componentes da ferramenta que compõem o núcleo interno da mesma.	44
Figura 14 – Representação esquemática do processo de realização de operações aritméticas na ferramenta (no exemplo, a adição) a partir de duas variáveis diferentes (a), duas variáveis equivalentes (b), uma variável e um parâmetro especificado ou constante (c) e entre uma variável e um parâmetro não especificado (d), com o resultado simbólico em destaque.	45

Figura 15 – Representação esquemática do processo de conversão entre uma quantidade (objeto <i>Quantity</i>) e um termo simbólico equacional (objeto <i>EquationNode</i>), por meio da rotina interna específica a esse propósito.	47
Figura 16 – Representação esquemática do processo de conexão entre dois modelos (objetos <i>Model</i>), por meio de um objeto <i>Connection</i> , destacando o conjunto final de equações obtido, em um objeto do tipo <i>EquationBlock</i>	48
Figura 17 – Representação esquemática do processo de otimização na ferramenta.	50
Figura 18 – Representação da relação de interdependência dos módulos que compõem o pacote <i>core</i>	51
Figura 19 – Diagrama de classes do subpacote <i>unit</i>	52
Figura 20 – Diagrama de classes do subpacote <i>quantity</i>	52
Figura 21 – Diagrama de classes do subpacotes <i>variable</i> , <i>parameter</i> e <i>constant</i> , derivados da classe <i>Quantity</i>	53
Figura 22 – Diagrama da classe <i>Domain</i> , definida no subpacote <i>core.domain</i>	54
Figura 23 – Diagrama das classes definidas no subpacote <i>core.equation</i> , correspondendo as classes <i>Equation</i> e <i>Connection</i> (a qual deriva da primeira).	55
Figura 24 – Diagrama das classes definidas no subpacote <i>core.equation_block</i>	55
Figura 25 – Diagrama das classes definidas no subpacote <i>core.expression_evaluation</i>	56
Figura 26 – Diagrama das classes definidas no subpacote <i>core.error_definitions</i>	57
Figura 27 – Diagrama das classe <i>Model</i> , definida no pacote <i>model</i>	58
Figura 28 – Diagrama das classe <i>Problem</i> , definida no pacote <i>problem</i>	59
Figura 29 – Diagrama da classe <i>Simulation</i>	59
Figura 30 – Diagramas de classe daquelas definidas no pacote <i>optimization</i>	60
Figura 31 – Diagramas de classe daquelas definidas no pacote <i>solver</i> . As classes <i>LASolver</i> , <i>NLASolver</i> , <i>DSolver</i> e <i>DaeSolver</i> são utilizadas respectivamente para a resolução de sistemas de equações lineares, não-lineares, puramente diferenciais e algébrico-diferenciais.	61
Figura 32 – Diagramas de atividade representando o fluxo informativo e procedimento realizado na definição de um modelo.	62
Figura 33 – Diagramas de atividade representando o fluxo informativo e o procedimento realizado na conexão de dois modelos	63
Figura 34 – Diagramas de atividade representando o fluxo informativo e o procedimento realizado na obtenção das informações de um modelo.	63
Figura 35 – Diagramas de atividade representando o fluxo informativo e o procedimento realizado na definição de um caso de estudo.	64
Figura 36 – Diagramas de atividade representando o fluxo informativo e o procedimento realizado na obtenção das informações de um caso de estudo.	64
Figura 37 – Diagramas de atividade representando o fluxo informativo e o procedimento realizado na definição de uma simulação.	65
Figura 38 – Diagramas de atividade representando o fluxo informativo e o procedimento realizado na definição de um caso de estudo de otimização.	66

Figura 39 – Diagramas de atividade representando o fluxo informativo e o procedimento realizado na produção de gráficos a partir dos resultados de uma simulação.	66
Figura 40 – Representação esquemática dos tipos de modelos mecanísticos utilizados na descrição dos processos biotecnológicos, destacando a diferença entre cada um deles, apresentado novamente para a conveniência do leitor.	68
Figura 41 – Representação esquemática do procedimento empregado na otimização em malha aberta do processo de produção do bioetanol.	73
Figura 42 – Perfil temporal da vazão de enchimento do fermentador, obtido a partir da rotina GA, para a expressão paramétrica polinomial (a) e cossenoidal (b).	78
Figura 43 – Perfil temporal da vazão de enchimento do fermentador, obtido a partir da rotina DE, para a expressão paramétrica polinomial (a) e cossenoidal (b).	78
Figura 44 – Perfil temporal das concentrações de biomassa, substrato e bioetanol, obtido a partir da rotina GA, para a expressão paramétrica polinomial.	79
Figura 45 – Perfil temporal das concentrações de biomassa, substrato e bioetanol, obtido a partir da rotina GA, para a expressão paramétrica cossenoidal.	79
Figura 46 – Perfil temporal das concentrações de biomassa, substrato e bioetanol, obtido a partir da rotina DE, para a expressão paramétrica polinomial.	80
Figura 47 – Perfil temporal das concentrações de biomassa, substrato e bioetanol, obtido a partir da rotina DE, para a expressão paramétrica cossenoidal.	80
Figura 48 – Comparação de rotas metabólicas (simplificadas) envolvidas no metabolismo da produção do bioetanol, para os micro-organismos <i>Saccharomyces cerevisiae</i> (A) e <i>Zymomonas mobilis</i> (B).	84
Figura 49 – Fluxograma de operação para o problema de controle de um bioprocesso por meio de DRTO, para o problema de busca por <i>set-point</i>	88
Figura 50 – Resultado obtidos para a simulação dinâmica do modelo de produção de bioetanol pela em fermentação contínua, em termos do perfil temporal da concentração do produto (50a), concentração de biomassa (50b), concentração de substrato (50c) e concentração de componentes-chave (50d). Todos os resultados foram obtidos para um valor de concentração de substrato na alimentação do biorreator (S_0) equivalente a $150,3 \text{ kg m}^{-3}$	91
Figura 51 – Resultado obtidos para a simulação dinâmica do modelo de produção de bioetanol pela em fermentação contínua, em termos do perfil temporal da concentração do produto (51a), concentração de biomassa (51b), concentração de substrato (51c) e concentração de componentes-chave (51d). Todos os resultados foram obtidos para um valor de taxa de diluição na alimentação do biorreator (D_{in}) equivalente a $0,5 \text{ h}^{-1}$	92

- Figura 52 – Resultados obtidos para o estudo de controle do processo de produção de bioetanol *in-silico* por meio de fermentação contínua, utilizando a abordagem DRTO. Foram empregados 20 intervalos para a otimização da variável manipulada ao longo dos horizontes de predição, correspondendo a uma duração para o horizonte de predição de 1 h. Nas Figuras 52a, 52b e 52c são retratados respectivamente os perfis temporais de concentração de biomassa celular (X), substrato (S) e componentes-chave para o metabolismo celular (E). 94
- Figura 53 – Resultados obtidos para o estudo de controle do processo de produção de bioetanol *in-silico* por meio de fermentação contínua, utilizando a abordagem DRTO. Foram empregados 40 intervalos para a otimização da variável manipulada ao longo dos horizontes de predição, correspondendo a uma duração para o horizonte de predição de 0,5 h (trinta minutos). Nas Figuras 53a, 53b e 53c são retratados respectivamente os perfis temporais de concentração de biomassa celular (X), substrato (S) e componentes-chave para o metabolismo celular (E). 95
- Figura 54 – Resultados obtidos para o estudo de controle do processo de produção de bioetanol *in-silico* por meio de fermentação contínua, utilizando a abordagem DRTO. Foram empregados 80 intervalos para a otimização da variável manipulada ao longo dos horizontes de predição, correspondendo a uma duração para o horizonte de predição de 0,25 h (15 minutos). Nas Figuras 54a, 54b e 54c são retratados respectivamente os perfis temporais de concentração de biomassa celular (X), substrato (S) e componentes-chave para o metabolismo celular (E). 96
- Figura 55 – Resultados obtidos para os perfis temporais da variável controlada, a concentração de bioetanol (P), utilizando a abordagem DRTO, com destaque para o valor do *set-point* estabelecido para a mesma ($P_{sp} = 65 \text{ kg m}^{-3}$). Foram empregados 20, 40 e 80 intervalos para a otimização da variável manipulada ao longo dos horizontes de predição, correspondendo a uma duração para o horizonte de predição de 1 h (60 minutos), 0,5 h (30 minutos) e 0,25 h (15 minutos), retratados respectivamente nas Figuras 55a, 55b e 55c. 97

Lista de tabelas

Tabela 1 – Representação matemática dos termos utilizados na definição de um problema de otimização, e seu significado.	22
Tabela 2 – Diferentes tipos de equações que podem figurar nos modelos utilizados na ferramenta <i>sloth</i> , em termos do tipo de equação, a forma por meio da qual são referidas no <i>software</i> e a sua expressão matemática canônica	42
Tabela 3 – Representação matemática dos termos utilizados no modelo de produção de bioetanol, e seu significado.	69
Tabela 4 – Parâmetros do modelo para a produção de bioetanol em batelada-alimentada (HONG, 1986).	70
Tabela 5 – Valores iniciais das variáveis empregadas no PVI resultante do processo de produção <i>in-silico</i> do bioetanol	72
Tabela 6 – Restrições utilizadas na resolução dos problemas de otimização no presente caso de estudo.	74
Tabela 7 – Configuração empregada no algoritmo GA.	76
Tabela 8 – Configuração empregada no algoritmo DE.	76
Tabela 9 – Resultados obtidos para os parâmetros das funções de enchimento da dorna e o subsequente valor obtido para a função objetivo, por meio da rotina GA.	77
Tabela 10 – Resultados obtidos para os parâmetros das funções de enchimento da dorna e o subsequente valor obtido para a função objetivo, por meio da rotina DE.	77
Tabela 11 – Parâmetros e condições iniciais do modelo para a produção de bioetanol, por meio de fermentação contínua, utilizando a <i>Zymomonas mobilis</i> como micro-organismo de trabalho (MIRLEKAR et al., 2017).	86
Tabela 12 – Concentração inicial para as variáveis do modelo de produção de bioetanol por meio de fermentação contínua, utilizando a cultura de <i>Zymomonas mobilis</i> (MIRLEKAR et al., 2017).	87
Tabela 13 – Configuração empregada no algoritmo DE, para a resolução dos problemas de otimização oriundos do DRTO.	89

Lista de abreviaturas e siglas

1G	Bioetanol de primeira geração, obtido diretamente a partir de fontes primárias de açúcares fermentescíveis
2G	Bioetanol de segunda geração, obtido a partir de fontes lignocelulósicas
ABC	<i>Artificial Bee Colony optimization</i> , ou otimização por colônia de abelhas
AAA	<i>Artificial Algae Algorithm</i> , ou algoritmo de algas artificiais
BSA	<i>Backtracking Search Algorithm</i> , ou algoritmo de busca retrocessiva
CMAES	<i>Covariance Matrix Adaptation Evolution Strategy</i> , ou estratégia de evolução por adaptação da matriz de covariância
CPP	<i>Critical Process Parameters</i> , ou parâmetros críticos do processo
CQA	<i>Critical Quality Attributes</i> , ou atributos de qualidade crítica
CSTR	<i>Continuous Stirred Tank Reactor</i> , ou tanque reator agitado contínuo
DE	<i>Differential Evolution</i> , ou evolução diferencial
DECC	<i>Differential Evolution with Cooperative Coevolution</i> , ou evolução diferencial com coevolução cooperativa
DFBA	<i>Dynamic Flux Balance Analysis</i> , ou análise de balanço de fluxo dinâmico
DRTO	<i>Dynamic Real-Time Optimization</i> , ou otimização dinâmica em tempo real
EMEA	Agência governamental análoga ao FDA, para a europa
ELISA	<i>Enzyme-Linked ImmunoSorbent Assay</i> , ou ensaio de imunoabsorção enzimática
FDA	Agência americana de administração de alimentos e remédios (<i>Food and Drug Administration</i>)
FTIR	<i>Fourier-Transform Infrared Spectroscopy</i> , ou espectroscopia no infravermelho com transformada de Fourier
GA	<i>Genetic Algorithm</i> , ou algoritmo genético
GC	<i>Gas Chromatography</i> , ou cromatografia gasosa
GPU	<i>Graph Processing Unit</i> , ou unidade de processamento gráfico
HPLC	<i>High Performance Liquid Chromatography</i> , ou cromatografia líquida de alta eficiência
KDPG	2-ceto-3-desoxi-6-fosfogluconato

MHLW	Agência governamental análoga ao FDA, para o Japão
MIR	<i>Mid-Infrared Spectroscopy</i> , ou espectroscopia no infravermelho intermediário
M-MOPSO	<i>Modified Multi-Objective Particle Swarm Optimization</i> , ou otimização multi-objetivo por enxame de partículas
MS	<i>Mass Spectrometry</i> , ou espectrometria de massa
NADH	<i>Nicotinamide Adenine Dinucleotide</i> , ou dinucleótido de nicotinamida e adedina, em sua forma reduzida
NIR	<i>Near-Infrared Spectroscopy</i> , ou espectroscopia no infravermelho próximo
NMPC	<i>Non-linear Model Predictive Control</i> , ou modelo preditivo baseado em modelo não-linear
OBL	<i>Opposition-Based Learning</i> , ou aprendizado baseado em oposição
PAT	<i>Process Analytical Technology</i> , ou tecnologia analítica de processo, uma iniciativa de monitoramento de processos biotecnológicos
pH	Potencial hidrogeniônico
PSO	<i>Particle Swarm Optimization</i> , ou otimização por enxame de partículas
UML	<i>Unified Modeling Language</i> , ou linguagem unificada de modelagem
UPLC	<i>Ultra Performance Liquid Chromatography</i> , ou cromatografia líquida de ultra performance
RAM	<i>Random Access Memory</i> , ou memória de acesso aleatório
SA	<i>Simulated Annealing</i> , ou recozimento simulado

Lista de símbolos

J	Função objetivo referente a um problema de otimização genérico
u	Parâmetros a serem otimizados, referente a um problema de otimização genérico
x	Variáveis de estado do modelo, referente a um problema de otimização genérico
\tilde{p}	Função (ou o conjunto delas) que define os parâmetros por meio de relações constitutivas, referente a um problema de otimização genérico
f	Modelo matemático referente a um problema de otimização genérico
\dot{f}	Equações diferenciais que definem o modelo, referente a um problema de otimização genérico
c	Função que define as restrições dos parâmetros, referente a um problema de otimização genérico
t	Tempo, para modelos dinâmicos, referente a um problema de otimização genérico
$f(x_1, x_2)$	Função genérica de dois parâmetros
r	Razão entre o volume de uma hiperesfera, representando o espaço de optimalidade em um espaço de busca <i>n-dimensional</i> , e o volume de um hipercubo, representando esse espaço de busca
$g(x_i)$	Função separável de um problema de otimização, definida a partir da <i>i-ésima</i> variável conjunto para o qual uma função multidimensional genérica é definida
$h(\tilde{x})$	Função separável de um problema de otimização, definida a partir do conjunto de variáveis para o qual uma função multidimensional genérica é definida
\tilde{x}	Conjunto de variáveis para o qual uma função multidimensional genérica é definida
x_i	<i>i-ésima</i> variável do conjunto para o qual uma função multidimensional genérica é definida
f_i	<i>i-ésima</i> função que compõe a função global f , definida em termos de x_i
$f(\tilde{x})$	Função genérica a ser otimizada, a partir de um conjunto de variáveis a partir do qual a mesma está definida
n	Dimensionalidade de uma função genérica a ser otimizada
x	Variável genérica de uma função linear, não-linear, diferencial ou algébrico-diferencial

y	Variável genérica de uma função linear, não-linear, diferencial ou algébrico-diferencial
z	Variável genérica de uma função linear, não-linear, diferencial ou algébrico-diferencial
V	Volume contido no biorreator (L) para o modelo empregado no caso de estudo I
X	Concentração de biomassa celular contida no biorreator ($g L^{-1}$) para o modelo empregado no caso de estudo I
S	Concentração de substrato contida no biorreator ($g L^{-1}$) para o modelo empregado no caso de estudo I
P	Concentração de bioetanol contida no biorreator ($g L^{-1}$) para o modelo empregado no caso de estudo I
$u(t)$	Taxa de alimentação do biorreator ($L h^{-1}$) para o modelo empregado no caso de estudo I
μ	Taxa de crescimento da biomassa (h^{-1}) para o modelo empregado no caso de estudo I
μ_0	Taxa máxima de crescimento da biomassa (h^{-1}) para o modelo empregado no caso de estudo I
q	Taxa de produção de bioetanol (h^{-1}) para o modelo empregado no caso de estudo I
q_0	Taxa máxima de produção de bioetanol (h^{-1}) para o modelo empregado no caso de estudo I
K_P	Constante de Monod para o modelo empregado no caso de estudo I, com unidades de $g L^{-1}$
K_{PI}	Constante de inibição do produto para o modelo empregado no caso de estudo I, com unidades de $g L^{-1}$
K_S	Constante de Monod para o modelo empregado no caso de estudo I, com unidades de $g L^{-1}$
K_{SI}	Constante de inibição do produto para o modelo empregado no caso de estudo I, com unidades de $g L^{-1}$
x_{30}	Concentração de substrato na corrente de alimentação do biorreator para o modelo empregado no caso de estudo I, com unidades de $g L^{-1}$
y_r	Fator de rendimento biomassa/substrato para o modelo empregado no caso de estudo I, com unidades de $g g^{-1}$
a	Parâmetro a ser determinado para as expressões paramétricas para o cálculo da vazão de alimentação do biorreator, para o modelo empregado no caso de estudo I (adimensional)

b	Parâmetro da ser determinado para as expressões paramétricas para o cálculo da vazão de alimentação do biorreator, para o modelo empregado no caso de estudo I (adimensional)
c	Parâmetro da ser determinado para as expressões paramétricas para o cálculo da vazão de alimentação do biorreator, para o modelo empregado no caso de estudo I (adimensional)
d	Parâmetro da ser determinado para as expressões paramétricas para o cálculo da vazão de alimentação do biorreator, para o modelo empregado no caso de estudo I (adimensional)
T_f	Tempo final do processo fermentativo para o modelo empregado no caso de estudo I, com unidade de h
S	Concentração de substrato contida no biorreator ($kg\ m^{-3}$) para o modelo empregado no caso de estudo II
X	Concentração de biomassa celular contida no biorreator ($kg\ m^{-3}$) para o modelo empregado no caso de estudo II
E	Concentração de componentes-chave para o metabolismo celular contida no biorreator ($kg\ m^{-3}$) para o modelo empregado no caso de estudo II
P	Concentração de bioetanol contida no biorreator ($kg\ m^{-3}$) para o modelo empregado no caso de estudo II
Y_x	Taxa específica de crescimento máxima (h^{-1}) para o modelo empregado no caso de estudo II
Y_{sx}	Fator de rendimento baseado em substrato ($kg\ kg^{-1}$) para o modelo empregado no caso de estudo II
Y_{px}	Fator de rendimento baseado em produto ($kg\ kg^{-1}$) para o modelo empregado no caso de estudo II
D_{in}	Taxa de diluição na corrente de alimentação do biorreator (h^{-1}) para o modelo empregado no caso de estudo II
D_{out}	Taxa de diluição na corrente de saída do biorreator (h^{-1}) para o modelo empregado no caso de estudo II
K_s	Constante de Monod ($kg\ m^{-3}$) para o modelo empregado no caso de estudo II
S_0	Concentração de substrato na corrente de alimentação do biorreator ($kg\ m^{-3}$) para o modelo empregado no caso de estudo II
X_0	Concentração de biomassa celular na corrente de alimentação do biorreator ($kg\ m^{-3}$) para o modelo empregado no caso de estudo II

E_0	Concentração de componentes-chave para o metabolismo celular na corrente de alimentação do biorreator ($kg\ m^{-3}$) para o modelo empregado no caso de estudo II
P_0	Concentração de bioetanol na corrente de alimentação do biorreator ($kg\ m^{-3}$) para o modelo empregado no caso de estudo II
m_p	Fator de manutenção baseado em produto (h^{-1}) para o modelo empregado no caso de estudo II
m_s	Fator de manutenção baseado em substrato (h^{-1}) para o modelo empregado no caso de estudo II
k_1	Constante empírica (h^{-1}) para o modelo empregado no caso de estudo II
k_2	Constante empírica ($m^3\ kg^{-1}\ h^{-1}$) para o modelo empregado no caso de estudo II
k_3	Constante empírica ($m^6\ kg^{-2}\ h^{-1}$) para o modelo empregado no caso de estudo II
P_{sp}	Set-point para a concentração de bioetanol contida no biorreator ($kg\ m^{-3}$) para o estudo de controle <i>in-silico</i> do modelo empregado no caso de estudo II
n	Número de intervalos em que o tempo de fermentação está dividido, para o estudo de controle <i>in-silico</i> do modelo empregado no caso de estudo II
t_k	Intervalo de tempo para o <i>k-ésimo</i> intervalo no qual o tempo total de fermentação está dividido, para o estudo de controle <i>in-silico</i> do modelo empregado no caso de estudo II

Sumário

	1 INTRODUÇÃO	1
1.1	Introdução geral	1
1.2	Objetivos	4
1.2.1	Objetivo geral	4
1.2.2	Objetivos específicos	4
	2 PROCESSOS BIOTECNOLÓGICOS	7
2.1	Acerca dos processos biotecnológicos de interesse industrial	7
2.1.1	O conceito de processo biotecnológico	7
2.1.2	Modelagem de processos biotecnológicos	8
2.2	Produção de bioetanol por via fermentativa	11
2.2.1	Introdução	11
2.2.2	Aspectos operacionais importantes no processo de produção de etanol	14
2.2.2.1	Escolha da matéria-prima	14
2.2.2.2	Escolha dos micro-organismos de trabalho	15
2.2.2.3	Parâmetros físico-químicos	15
	3 OTIMIZAÇÃO E CONTROLE APLICADOS A PROCES-	
	SOS BIOTECNOLÓGICOS UTILIZANDO TÉCNICAS META-	
	HEURÍSTICAS	17
3.1	Monitoramento e controle dos bioprocessos	17
3.2	Otimização de bioprocessos utilizando técnicas meta-heurísticas	21
3.3	Problemas de otimização de alta dimensionalidade	27
3.3.1	Características matemáticas dos problemas de otimização	27
3.3.2	Breve revisão da literatura acerca do tema	31
	4 DESENVOLVIMENTO DA FERRAMENTA DE SIMULA-	
	ÇÃO ORIENTADA À EQUAÇÕES	37
4.1	Importância da modelagem e simulação de processos	37
4.2	Acerca da UML	38
4.3	Metodologia utilizada no desenvolvimento da ferramenta	40
4.3.1	Estrutura conceitual da ferramenta	40
4.3.1.1	Perspectiva geral	41
4.3.1.2	Definição de variáveis, parâmetros e constantes	45
4.3.1.3	Inserção e resolução de equações	46
4.3.1.4	Definição de modelos, e composição de um problema de estudo	47
4.3.1.5	Simulação	48
4.3.1.6	Otimização	49
4.3.2	Diagramas de classe	49
4.3.2.1	Pacote <i>core</i>	50
4.3.2.2	Pacote <i>model</i>	56
4.3.2.3	Pacote <i>problem</i>	57
4.3.2.4	Pacote <i>simulation</i>	58
4.3.2.5	Pacote <i>optimization</i>	60

4.3.2.6	Pacote <i>solver</i>	61
4.3.3	Diagramas de atividades	61
4.3.3.1	Definição de um modelo	62
4.3.3.2	Conexão entre dois modelos	62
4.3.3.3	Exibição das informações pertinentes do modelo	63
4.3.3.4	Definição de um caso de estudo (problema)	63
4.3.3.5	Exibição das informações pertinentes do problema (caso de estudo)	64
4.3.3.6	Definição de uma simulação	65
4.3.3.7	Definição de um caso de estudo de otimização e sua execução	65
4.3.3.8	Produção de gráficos a partir dos resultados de uma simulação	66
5	CASO DE ESTUDO I: OTIMIZAÇÃO EM MALHA ABERTA DE UM PROCESSO DE PRODUÇÃO DE BIOETANOL EM BATELADA ALIMENTADA	67
5.1	Apresentação da problemática	67
5.1.1	Modelagem do processo fermentativo	68
5.1.2	Otimização de processos biotecnológicos	70
5.2	Métodos utilizados	72
5.2.1	Simulação da produção de etanol	72
5.2.2	Otimização do processo fermentativo <i>in-silico</i>	72
5.3	Resultados e discussões	76
6	CASO DE ESTUDO II: SIMULAÇÃO E CONTROLE DE PROCESSO DE PRODUÇÃO DE BIOETANOL POR FERMENTAÇÃO CONTÍNUA	83
6.1	Apresentação da problemática	83
6.1.1	Modelagem do problema	85
6.2	Métodos utilizados	86
6.2.1	Simulação dinâmica do problema	86
6.2.2	Problema de controle	87
6.3	Resultados e discussões	90
6.3.1	Simulação dinâmica do problema	90
6.3.2	Controle da produção de bioetanol	93
7	CONCLUSÃO	99
7.1	Conclusão geral	99
7.2	Perspectivas para trabalhos futuros	100
	REFERÊNCIAS	101
	Apêndice A CÓDIGO FONTE UTILIZADO NO CASO DE ESTUDO I	115
A.1	Otimização em malha aberta	115
	Apêndice B CÓDIGO FONTE UTILIZADO NO CASO DE ESTUDO II	129
B.1	Estudo de simulação dinâmica da produção de bioetanol	129
B.2	Estudo de controle do processo de produção de bioetanol	138

CAPÍTULO 1

Introdução

1.1 Introdução geral

A cada dia, os chamados processos biotecnológicos tornam-se mais importantes para a sociedade moderna, seja diretamente, por meio da fabricação de produtos utilizados no dia-a-dia - a exemplo de alimentos, fármacos, combustíveis - ou de maneira indireta, na qual produtos fabricados a partir de rotas tradicionais - as quais muitas vezes não são eficientes ou ambientalmente sustentáveis - são paulatinamente substituídos pelas suas contrapartes advindas de bioprocessos. Em termos financeiros, estima-se que o volume dos produtos oriundos de bioprocessos atinja o patamar de 515 bilhões de euros em 2020 (FESTEL et al., 2012).

Em razão das já mencionadas motivações econômicas acerca dos processos biotecnológicos, bem como da complexidade inerente ao processo de desenvolvimento microbiano, especialmente em cultivos descontínuos ou semidescontínuos, o monitoramento e controle dos bioprocessos representa um tema cada vez mais desafiador no rol dos problemas da engenharia moderna. Com o intuito de obter-se exploração ótima de um determinado organismo produtivo em paralelo com a redução de custos operacionais e o aumento no rendimento do processo, mas ao mesmo tempo prezando pela manutenção da qualidade do produto metabólico de interesse e a sua consistência, é importante que exista um contínuo aprimoramento das capacidades de controle e monitoramento destes. No entanto, um grande número dos bioprocessos ainda são operados de maneira distante do seu ótimo, principalmente em virtude das limitações de seu monitoramento (CLEMENTSCHITSCH; BAYER, 2006; SIMUTIS; LÜBBERT, 2015).

Conforme mencionado anteriormente, em virtude das inúmeras complexidades intrínsecas aos bioprocessos, especialmente quando escalados à dimensões industriais, o controle destes representa um grande desafio, configurando um importante nicho de pesquisa. Uma vez que as técnicas de controle de processos tradicionais mostram-se por vezes incapazes de resolver satisfatoriamente os problemas advindos dos bioprocessos, o controle inteligente e a computação evolutiva, também chamada de meta-heurística (VITALIY, 2006), figuram como alternativas para este intento. Ambas as técnicas fazem parte de um grupo de métodos que se utiliza de paradigmas análogos àqueles utilizados pelos processos cognitivos humanos, chamado de inteligência artificial (IA). Esse conjunto de métodos, utilizado desde tarefas de otimização a identificação de processos, mostra-se robusta e bem adequada para a modelagem, otimização e controle de bioprocessos (BAUGHMAN; LIU, 1995; DOCHAIN, 2008; TIAN et al., 2013).

Como fruto da paulatina substituição das rotas usuais de produção de *commodities* e químicos refinados baseados por processos baseados em biotecnolo-

gia, tem ocorrido uma mudança de paradigma em termos das margens econômicas destes procedimentos tradicionais. Por exemplo, quando comparados a processos para a produção de fármacos, a competitividade destes processos depende mais do preço final dos produtos do que da proteção de patentes, o que implica que qualquer aumento na eficiência ou produtividade dos mesmos pode impactar fortemente a sua rentabilidade (ROCHA et al., 2014). Neste sentido, fica clara a emergência na busca pela busca do melhor perfil para os parâmetros operacionais de modo a maximizar a rentabilidade de um processo biotecnológico.

Uma vez que os bioprocessos empregam entidades biológicas (sejam eles micro-organismos como um todo ou apenas estruturas especializadas, como antígenos ou ácidos nucleicos) como catalisadores dos processos de produção e os produtos biotecnológicos são, em última análise, resultados de sua atividade, o controle dessa classe de processos apresentam características específicas. O frequente comportamento não-linear do metabolismo microbiano e a forte relação entre este processo e as características ambientais e operacionais do biorreator tornam o desenvolvimento de modelos matemáticos descritivos para bioprocessos e o próprio controle do processo uma tarefa laboriosa (WANG et al., 2009; GADKAR et al., 2005). Apesar das dificuldades inerentes a esse propósito, a otimização baseada em modelo do processo biotecnológico tem sido objeto de várias pesquisas (BANGA et al., 2005; OCHOA, 2016). Nesse sentido, os processos fermentativos conduzidos em biorreatores do tipo batelada alimentada representam um tópico importante, dada a utilização generalizada desta classe de reatores no campo industrial bioquímico, o que pode ser explicado pela menor probabilidade de inibição por substrato devido à sobrealimentação (quando comparado ao modo de operação em batelada comum) e a preservação de um ambiente estéril dentro do fermentador (quando comparado a equipamentos contínuos) (RANI; RAO, 1999).

Em um aspecto geral, o objetivo final de um bioprocessos é maximizar a produtividade deste por meio da manipulação do perfil da taxa de alimentação, apesar da possível presença de produtos inibitórios, especialmente em se tratando daqueles processos conduzidos em batelada alimentada (PIMENTEL et al., 2015). O comportamento complexo já descrito dos bioprocessos no escopo de seu controle constitui um problema de otimização dinâmica não trivial, que exige o uso de técnicas robustas capazes de encontrar uma solução viável que leve a uma produtividade ótima em paralelo com um não aprisionamento em um ponto ótimo local, aliado a uma demanda de esforço computacional razoável. Nesse sentido, as técnicas de computação evolutiva e meta-heurísticas representam importantes alternativas, pois esses métodos geralmente obtêm boas soluções com tempos de computação modestos, muito embora o caráter global da solução não possa ser garantido (ROCHA et al., 2014).

O monitoramento e controle dos bioprocessos apresenta desafios intrínsecos não só no tocante à complexidade do crescimento microbiano em si, mas a própria tarefa de mensura requer não só a implementação de uma série de sistemas sensores, os quais podem estar fisicamente instalados em diferentes configurações (seja no interior do reator ou instalado externamente ao mesmo) (NAJAFPOUR, 2007), bem como sofrerem influência de efeitos deletérios, a exemplo da dependência das propriedades do instrumento com a temperatura, formação de co-produtos no meio reacional, ocorrência bolhas de gás, ou outro comportamento complexo, resultando em informação de mensura corrompida (KRAUSE et al., 2015). Nesse sentido, a utilização de problemas de referência (ou *benchmark*, em inglês) constitui uma importante ferramenta, haja vista que por meio das condições bem estabelecidas nestes, torna-se possível empregar técnicas de simulação para o estudo de cenários alternativos, modelagem dinâmica de sistemas e controle e otimização de processos, atividades estas que muitas vezes mostram-se proibitivas em escala de produção. No tocante aos bioprocessos, estes problemas de referência buscam representar as já mencionadas complexidades inerentes a esta classe de processos industriais, nas suas diversas aplicações, a exemplo dos trabalhos de Arnell et al. (2017), Ochoa et al. (2010), Rocha et al. (2007), Jayaraman et al. (2001), Maurer et al. (2006), entre outros. Muito embora notáveis avanços tenham ocorrido no âmbito da modelagem e simulação de processos biotecnológicos, incluindo-se ferramentas e metodologias para este intento, a carência de problemas de referência reproduzíveis ainda é que convém ser mencionado (VILLAVERDE et al., 2015; MEARS et al., 2017).

Nesse sentido, os simuladores de processo mostram-se como importantes alternativas, tendo em vista que muitas vezes eles permitem que sejam estudados cenários produtivos que seriam proibitivos sob o ponto de vista da segurança do processo, ou mesmo do custo em se destinar a instalação industrial para a realização de testes, especialmente no âmbito dos bioprocessos, nos quais frequentemente os produtos possuem um alto valor agregado. No âmbito dos simuladores de processo, existem duas classes de duas abordagens diferenciadas, a saber: a abordagem sequencial-modular, em que cada uma das operações unitárias do processo é resolvida sequencialmente; abordagem simultânea, ou orientada a equações, em que os modelos que descrevem as operações são unidos, formando um grande bloco de equações, o qual é solucionado empregando técnicas computacionais adequadas. Embora implique em um maior esforço computacional, o uso da abordagem simultânea possui diversas vantagens, incluindo a robustez numérica, a facilidade de resolução de problemas não-lineares e altamente acoplados (SHACHAM et al., 1982; PATTISON; BALDEA, 2014; KAMATH et al., 2010), fatores especialmente relevantes para estudos de otimização e controle de tais processos biotecnológicos, conforme já foi mencionado.

Diante do exposto, o presente trabalho objetiva desenvolver uma ferramenta para modelagem e simulação de processos, a partir da abordagem de orientada a equações. A ferramenta deve ser capaz de fornecer a estrutura básica para que o usuário realize a declaração de seus modelos, agregue aqueles que fazem parte do problema em estudo e proceda a solução do mesmo. Convém mencionar também a importância de que seja possível que o usuário reutilize e estenda os modelos definidos quando necessário. Diante dessas capacidades, essa ferramenta pode ser utilizada em estudos de controle e otimização de processos biotecnológicos, em que os entes microbiológicos são vistos como uma unidade ou um subprocesso que pode ser controlado e otimizado, um tema que ainda suscita relevantes pesquisas atualmente.

1.2 Objetivos

Na presente seção serão explanados o objetivo geral do presente estudo, bem como os objetivos específicos nos quais o desenvolvimento do presente trabalho foi norteado.

1.2.1 Objetivo geral

O objetivo geral do presente trabalho reside no desenvolvimento de uma ferramenta de código aberto utilizando a linguagem computacional Python, por meio da qual seja possível declarar modelos e simular processos empregando a abordagem orientada a equações. A ferramenta será empregada em estudos de modelagem, controle e otimização de processos biotecnológicos, a partir do exemplo dos processos produtivos de produção de bioetanol por via fermentativa, os quais mostram-se especialmente desafiadores para essas tarefas, em razão de suas características inerentes.

1.2.2 Objetivos específicos

A partir do objetivo geral do trabalho, definido anteriormente, podem ser estabelecidos os seguintes objetivos específicos:

- Desenvolvimento de uma ferramenta para modelagem e simulação de processos orientada a equações
- Adotar uma estrutura modular para o desenvolvimento da ferramenta, de modo que suas funções sejam compartimentalizadas

- Realização de estudos de controle e otimização aplicados a bioprocessos, utilizando a ferramenta desenvolvida
- Análise dos resultados obtidos, sob a perspectiva da funcionalidade da ferramenta

CAPÍTULO 2

Processos biotecnológicos

No presente capítulo uma breve revisão acerca dos processos biotecnológicos será realizada, compreendendo desde o conceito definidor dessa importante classe de processos industriais, bem como as estratégias constantes na literatura para modelagem desse tipo de sistema. Em seguida, serão discutidos aspectos de interesse que tangenciam o tema da produção de bioetanol por via fermentativa, processo esse que consta no cerne do presente trabalho, compreendendo desde uma breve revisão acerca da importância do mesmo, bem como estudos de sua modelagem.

2.1 Acerca dos processos biotecnológicos de interesse industrial

2.1.1 O conceito de processo biotecnológico

Em função de sua abrangência, o conceito de biotecnologia não é apresentado de maneira uniforme na literatura, contudo um ponto comum entre autores remete às ciências que interseccionam-se para dar origem à referida disciplina: Química, biologia, engenharia, botânica, entre outras. Conforme descreve Borzani et al. (2001, p. VI), trata-se de um campo de trabalho em essência multidisciplinar, que demanda a atuação sinérgica e integrada de diversos profissionais. Ainda de acordo com os autores, embora a biotecnologia tenha assumido um papel prioritário nos esforços de pesquisa no meio científico, os processos biotecnológicos ou bioprocessos tem sido utilizados desde a antiguidade na produção de diversos produtos, tais como o vinho, queijo, pão, entre outros. A Figura 1 mostra a interseção entre as diferentes áreas do conhecimento para compor o domínio de estudo da biotecnologia.

Em termos históricos, atribui-se a Karl Ereky, um engenheiro agrícola húngaro, a primeira definição de biotecnologia, como a “ciência e os métodos que permitem a obtenção de produtos a partir de matéria-prima mediante a intervenção de organismos vivos”. (MALAJOVICH, 2012, p. 1). Essa atividade, a qual assumia um aspecto essencialmente artesanal de outrora, vai dar lugar a uma aplicação laboratorial e industrial, implicando em diversos processos que compreendem alimentos, combustíveis, medicamentos, dentre uma miríade de produtos obtidos a partir da atividade de micro-organismos ou demais entes biológicos. As vendas relacionadas ao setor biotecnológico, excetuando-se o setor farmacêutico, atingiram a quantia de 48 bilhões de euros em 2007, com um rendimento esperado de 340 bilhões de euros em 2017, valor esse que corresponderia a cerca de 15% da venda de químicos em escala mundial (FESTEL, 2010 apud TAKORS, 2012).

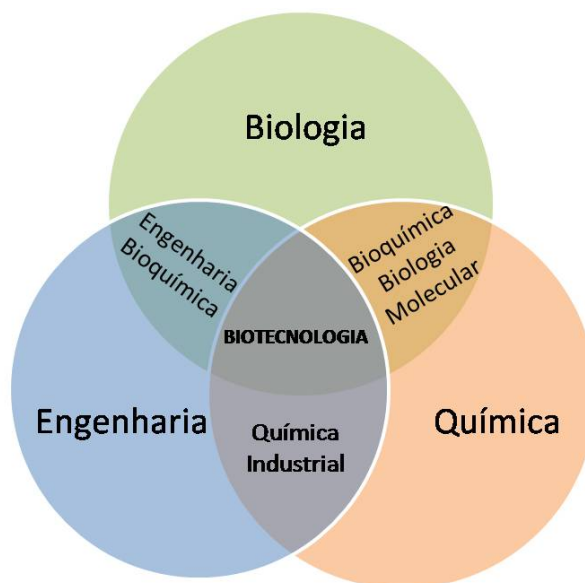


Figura 1 – Representação esquemática da interseção entre as diversas áreas que compõem a biotecnologia.

Fonte – Adaptado de Borzani et al. (2001)

2.1.2 Modelagem de processos biotecnológicos

Dentre os diversos tipos de modelos que utilizam relações matemáticas para a descrição de bioprocessos, os chamados modelos mecanísticos (ou fenomenológicos) representam um ponto de bastante interesse, considerando que estes são construídos com base nos processos incidentais do sistema. Estes constituem uma das principais ferramentas para a descrição de sistemas de interesse, oferecendo diversas vantagens como capacidade de extrapolação e controle de processo, fomentando assim atividades de planejamento de experimentos, bem como a determinação de quais variáveis críticas do processo devem ser monitoradas e controladas com minúcia (FERNANDES et al., 2013). Baseados em princípios determinísticos, tais modelos permitem que, a partir de condições iniciais estabelecidas em torno dos ditos processos incidentais (as variáveis do modelo), estados futuro do presente sistema sejam preditos, incorporando relações matemáticas entre entradas do processo (variáveis críticas) e saídas (concentração de produtos e atributos de qualidade dos mesmos) (GERNAEY et al., 2010; FERNANDES et al., 2013). Tais modelos, juntamente com os empíricos (caixa-preta), compõe a chamada classe de modelo baseados em relações matemáticas. Somam-se a essa classe de modelos os estatísticos, e os qualitativos (CRAVEN et al., 2013). Contudo, conforme já foi mencionado, os modelos matemáticos – e em especial os mecanísticos – oferecem diversas vantagens como capacidade de extrapolação e controle de processo, fomentando assim atividades de planejamento de experimentos, bem como a determi-

nação de quais variáveis críticas do processo devem ser monitoradas e controladas com minúcia (FERNANDES et al., 2013).

De acordo com (GERNAEY et al., 2010), os modelos mecanísticos para processos fermentativos e biocatalíticos são desenvolvidos através de balanços de massa, quantidade de movimento e energia, acrescidos de relações constitutivas apropriadas para a representação dos mecanismos intrínsecos, a exemplo das expressões cinéticas para a representação do comportamento dinâmico do processo. Por sua vez, os modelos mecanísticos são classificados à depender da abordagem utilizada na formulação do modelo, ou mais especificamente na maneira com que o comportamento celular é descrito. Várias abordagens estão disponíveis para a descrição matemática dos processos biológicos, que podem ser classificados em termos da descrição das propriedades individuais de células ou subpopulações microbianas - segregadas, para as quais as propriedades individuais são contabilizadas e não segregadas, para as quais um comportamento médio é considerado para toda a população - e no que se refere ao nível de detalhe para a composição da biomassa - estruturado, para o qual o material biológico é examinado entre vários componentes e não-estruturado, para o qual a biomassa é combinada em um único termo macroscópico (GERNAEY et al., 2010; FERNANDES et al., 2013). Essa classificação é representada esquematicamente na Figura 2.



Figura 2 – Representação esquemática dos tipos de modelos mecanísticos utilizados na descrição dos processos biotecnológicos, destacando a diferença entre cada um deles

Fonte – Adaptado de Fernandes et al. (2011), Gernaey et al. (2010).

Quanto à abordagem utilizada na descrição da natureza da biomassa celular, os modelos podem ser classificados como *estruturados*, para os quais esta é composta por múltiplos componentes (por exemplo, proteínas, membrana celular, etc) e *não-estruturados*, para os quais a biomassa celular é aglutinada em um termo único. Os modelos não-estruturados não consideram os processos intracelulares e relacionam implicitamente às mudanças na fisiologia celular com o ambiente (FREDRICKSON, 1976 apud CRAVEN et al., 2013), e baseiam-se na hipótese de crescimento balanceado, a qual postula que durante o crescimento celular, a concentração de metabólitos internos estão em estado quasi-estacionário, exibindo uma taxa líquida de conversão nula (PROVOST; BASTIN, 2004 apud CRAVEN et al., 2013). Por outro lado, os modelos estruturados são mais complexos, uma vez que incorporam um conhecimento bioquímico incidental ao descrever a biomassa celular em termos de compartimentos que se diferenciam físico e quimicamente, cujas interações são expressas por equações estequiométricas que consideram diversas rotas metabólicas ou expressões de taxas cinéticas (CRAVEN et al., 2013).

Quanto à consideração da heterogeneidade celular, os modelos podem ser classificados como *não-segregados*, para os quais o estabelecimento de subpopulações de micro-organismos não é considerada, e apenas um comportamento “médio” é descrito, ou *segregados*, para os quais considera-se que a população microbiana exibe uma compartimentalização em termos de subpopulações com comportamentos heterogêneos entre si. Os modelos não-segregados baseiam-se em uma representação média do comportamento celular, utilizando uma variável concentrada, como a quantidade de biomassa por unidade de volume, para a descrição de toda a população de micro-organismos, conforme apresentado nos trabalhos de Craven et al. (2013) e Gernaey et al. (2010). Em contrapartida, os modelos segregados consideram os efeitos de variação entre as células no que tange às suas propriedades, em contraste com a consideração de um comportamento médio adotado para os modelos não-segregados.

Entre as classes de modelos mencionadas anteriormente, existem os grupos intermediários. Os modelos não-estruturados e não-segregados representam a abordagem mais simples dentre os modelos mecanísticos para a descrição de bioprocessos, de modo que nenhum mecanismo cinético intracelular é adotado, e apenas as entradas e saídas são consideradas, bem como assume-se que a biomassa está uniformemente distribuída ao longo do sistema. (MENDESA et al., 2011a; GERNAEY et al., 2010; MENDESA et al., 2011b).

Modelos estruturados e não-segregados representam uma importante classe de modelos mecanísticos, descrevendo a biomassa celular em termos de diversas variáveis (metabólitos, membrana celular, organelas, etc), valendo-se no entanto da premissa de que essa distribuição da biomassa é a mesma para toda a população

celular. Apesar das limitações dessa classe de modelos, no trabalho de Royle et al. (2013) o autor menciona importantes trabalhos desenvolvidos a partir desta ferramenta, destacando a sua utilidade na análise de dados e otimização de processos biotecnológicos.

Os modelos chamados de segregados e não-estruturados descrevem as subpopulações de micro-organismos existentes em uma cultura, sem contudo analisar os detalhes do seu metabolismo celular. Contudo, em se tratando do cultivo de um único organismo, estes subgrupos expressam as diferentes fases do desenvolvimento celular, sendo necessária a existência de uma variável descritiva no modelo afim de quantificar a transição entre as subpopulações. Assim, torna-se virtualmente impossível a predição da dinâmica de um micro-organismo isolado utilizando um modelo segregado e não-estruturado (GERNAEY et al., 2010).

Por fim, os modelos chamados de segregados e estruturados apresentam um elevado grau de complexidade, uma vez que a distribuição de uma ou mais variáveis intracelulares é considerada, bem como ocorre a divisão dos micro-organismos em subpopulações (MENDESA et al., 2011b) A utilização desta classe de modelos permite que seja obtido um nível apreciável de realismo na descrição dos bioprocessos, especialmente se o histórico progresso individual das células torna-se foco de análise, a exemplo da consideração os efeitos cumulativos de desnutrição celular durante processos do tipo batelada alimentada, ou a estabilidade de micro-organismos contendo plasmídeos para a produção de proteínas recombinantes, conforme descrito no trabalho de Lapin et al. (2006).

2.2 Produção de bioetanol por via fermentativa

2.2.1 Introdução

A crescente demanda por fontes de energia tem liderado a sociedade moderna em busca contínua de processos mais eficientes, bem como uma pesquisa apreciável sobre alternativas de produção. A matriz energética atual, baseada principalmente nos combustíveis fósseis, foi gradualmente substituída por um novo paradigma, que depende de muitas fontes sustentáveis, como o uso de energia solar, eólica, hidrotermal e biomassa, entre muitas alternativas. Essas fontes emergentes têm uma característica ecológica e podem ajudar a mitigar os problemas ambientais decorrentes da utilização da matriz energética atual, como a liberação de grandes quantidades de dióxido de carbono e outros poluentes para a atmosfera, potencializando o problema do efeito-estufa, bem como a poluição do ar e a chuva ácida (FREITAS et al., 2017).

Entre muitos biocombustíveis, o bioetanol representa uma alternativa importante para a substituição do combustível fóssil, considerando-se o biocombustível mais utilizado para o transporte mundial (BALAT, 2011). Os Estados Unidos e o Brasil representam os maiores produtores de bioetanol do mundo, representando aproximadamente 85% da produção global do produto (POPP et al., 2014). No entanto, é importante destacar a implementação de um programa governamental de incentivo de fontes de energia alternativas através da utilização do bioetanol, chamado Programa Nacional de Álcool (PROALCOOL), em 1975 (MAYER et al., 2015), levando o país a um cenário de vanguarda na tecnologia atual de produção de bioetanol. Esta fonte de combustível, obtida a partir de um processo de fermentação realizado por vários micro-organismos, dentre os quais sem dúvida a *Saccharomyces cerevisiae* aparece como a plataforma de produção biotecnológica mais comum, tem também na multiplicidade de matérias-primas para sua obtenção uma vantagem notável. O bioetanol pode ser obtido a partir de matérias-primas contendo sacarose, amido, resíduos agroindustriais em geral ou mesmo fontes alternativas, tais como biomassa oriunda de algas (BALAT, 2011; VOHRA et al., 2014; BAEYENS et al., 2015; CHEN et al., 2015a). De acordo com Demirbas (2008), combustíveis baseados em biomassa ou *biocombustíveis*, tais como o bioetanol, oferecem diversas vantagens sobre combustíveis baseados no petróleo, a exemplo da grande disponibilidade de matérias primas, caráter amigável ao meio ambiente, são biodegradáveis e contribuem para o desenvolvimento econômico em termos da substituição de rotas produtivas tradicionais por aquelas baseadas em biotecnologia, dentre outros pontos de mérito.

Desde a crise do petróleo em 1970, a pesquisa e desenvolvimento de rotas de produção de bioetanol mais eficientes tem sido constante, tal como a utilização de biomassa lignocelulósica, à despeito de um curto período de redução do óleo cru durante as décadas de 80 e 90 (BAI et al., 2008). O Brasil e os Estados Unidos figuram como os dois maiores produtores mundiais de etanol, sendo responsáveis por cerca de 85% da produção mundial (RFA, 2017). Como fruto de um novo ciclo de aumento no preço das fontes energéticas baseadas no petróleo nos últimos anos, há um estímulo na adoção de fontes energéticas alternativas, dentre as quais convém destacar a bioenergia, a qual mostra-se cada vez mais como uma alternativa viável em países emergentes, devido a condições climáticas favoráveis, uma relativa grande quantidade de áreas disponíveis para o cultivo de matérias-primas, implicando em um conseqüente custo reduzido - em termos relativos - na produção de biomassa (WICKE et al.; SMEETS et al., 2011, 2007 apud EIJK et al., 2014).

Grande parte do bioetanol produzido mundialmente pode ser caracterizado como um biocombustível de primeira geração, obtido diretamente a partir de matérias-primas vegetais, as quais apresentam a imediata desvantagem de demandarem

áreas agricultáveis as quais poderiam ser utilizadas no cultivo de alimentos para a provisão da população mundial. A produção mundial em grande escala do bioetanol seja dependente de insumos obtidos diretamente por meio da agricultura, a exemplo da sacarose presente na cana-de-açúcar no Brasil, ou do amido presente no milho, nos Estados Unidos, os já referidos dois maiores produtores mundiais deste combustível (GUPTA; VERMA, 2015). Uma alternativa às matérias-primas que demandam a utilização de áreas agricultáveis é a substituição pelo chamado bioetanol de segunda geração, o qual baseia-se na utilização de materiais residuais lignocelulósicos, especialmente resíduos da agroindústria, para produção do biocombustível por meio da ação de enzimas especializadas.

No trabalho de Khatiwada et al. (2016), os autores apresentam uma análise técnico-econômica acerca da utilização de plantas de produção de bioetanol modernas que contam com a capacidade de alternar entre a utilização do bagaço e resíduos agro-industriais da cana-de-açúcar na obtenção de bioeletricidade ou na utilização deste material na obtenção de bioetanol por meio do processo de segunda geração (2G). Convém mencionar, a análise realizada por Khatiwada et al. tomou os dados de regiões produtoras de cana-de-açúcar e indústrias localizada em São Paulo (SP), estudando cenários que consideram a exportação do bioetanol 2G para a União Européia, bem como o consumo no mercado interno deste. De acordo com os autores, a implementação total da rota e segunda geração nas instalações industriais só é justificada para cenários específicos de preços de exportação desse combustível, uma vez que para valores inferiores, esse processo mostra-se contraproducente, e para preços de eletricidade que excedam certo patamar, torna-se vantajosa a conversão integral da utilização do material lignocelulósico a produção de bioeletricidade.

Nos trabalhos de Koppram et al. (2014), Eijck et al. (2014), Gupta & Verma (2015), os autores discutem diversas dificuldades inerentes ao processo de produção de bioetanol de segunda geração (os quais estão relacionados com a dispendiosa etapa de pré-tratamento da biomassa), aquelas relativas à transferência de massa para altas cargas de sólidos, bem como o fato de que esta consiste em uma tecnologia ainda em desenvolvimento. Especialistas indiquem que no futuro os recursos atualmente alocados na pesquisa e desenvolvimento ligados à produção de bioetanol de primeira geração serão alocados paulatinamente na produção deste baseada em tecnologias de segunda geração (ADITIYA et al., 2016). Contudo, atualmente a matriz baseada na tecnologia predecessora ainda é muito utilizada, especialmente no Brasil, uma vez que a produção local de bioetanol é baseada na cana-de-açúcar e não implica em restrições severas na cadeia produtiva das matérias-primas alimentícias, bem como a existência de um mercado interno bem estabelecido e o incipiente incentivo no investimento nas tecnologias de produção

baseadas em matérias-primas lignocelulósicas (SALLES-FILHO et al., 2017).

2.2.2 Aspectos operacionais importantes no processo de produção de etanol

Na presente seção, serão discutidos brevemente alguns dos aspectos operacionais importantes no processo de produção do bioetanol por meio da fermentação de matérias-primas pelos micro-organismos de trabalho. Essa análise compreende desde parâmetros de *upstream* tais como a natureza da matéria-prima, a escolha do micro-organismo fermentador, até *downstream*, nas etapas de processamento do biocombustível produzido por meio de destilação.

2.2.2.1 Escolha da matéria-prima

Em linhas gerais, a biomassa que serve como matéria prima para a produção de bioetanol, fonte de carboidratos fermentescíveis, pode ser classificada em três categorias abrangentes: fontes ricas em açúcares simples, a exemplo da cana-de-açúcar, sorgo sacarino, frutas, entre outros; materiais amiláceos, a exemplo do milho, trigo, arroz, batata e o malte; materiais lignocelulósicos, categoria na qual figuram os exemplos dos resíduos vegetais tais como aparas de madeira, folhas, caule, bagaço oriundo do processamento de alguns cultivares, entre outros (BALAT; BALAT, 2009; ZABED et al., 2017).

A escolha da matéria-prima traz grandes implicações acerca das etapas de processamento necessárias para o processo fermentativo em si, uma vez que a utilização de fontes ricas em amido implica na necessidade da disponibilização dos açúcares fermentescíveis a partir do amido, ocorrendo por meio da sacarificação deste último (AWG-ADENI et al., 2013). De maneira semelhante, o emprego de materiais lignocelulósicos como matéria-prima demanda etapas preliminares de natureza química, física, biológica ou a associação destas com vistas a desestruturar a complexa estrutura da celulose em açúcares fermentescíveis (CHEN; FU, 2016; SINDHU et al., 2016). Incluem-se nessa categoria as pesquisas que versam sobre a utilização de resíduos alimentares para a produção de bioetanol, conforme apresentado nos trabalhos de Kiran & Liu (2015), Han et al. (2019), Loizidou et al. (2017). No trabalho de (ZABED et al., 2017), os autores apresentam extensivamente uma relação de diversas matérias-primas para a produção deste biocombustível, relacionando-as com sua produtividade e aspectos relevantes que outros trabalhos apontam acerca das mesmas.

Convém mencionar que embora a utilização de cultivares com alto teor de açúcares fermentescíveis, a exemplo da cana-de-açúcar, torna-se paulatinamente

vista como contraproducente, uma vez que os problemas oriundos da necessidade de implementar o cultivo dessas plantas em vastas extensões de terra, especialmente na esfera ambiental (GOLDEMBERG et al., 2008). Isso contrapõe-se ao fato de essa matéria-prima não demandar as etapas de processamento prévio observadas nas fontes amiláceas e lignocelulósicas, implicando em menores custos de operação e implementação da estrutura industrial.

2.2.2.2 Escolha dos micro-organismos de trabalho

Conforme já foi mencionado, considerando que a produção do bioetanol é fruto do metabolismo microbiano (a exemplo de outros produtos de processos biotecnológicos), o emprego de certos agentes microbiológicos nesse intento implicará diretamente na resposta obtida para o bioprocessamento. Diversos micro-organismos podem ser utilizados com esse propósito, tais como a levedura *Saccharomyces cerevisiae* (a qual desponta como a plataforma biológica mais tradicional para a produção de bioetanol por via fermentativa), as bactérias *Zymomonas mobilis*, *Lactobacillus fermentum*, entre outras (ELSHAGHABEE et al., 2016; AZHAR et al., 2017; LIAO et al., 2016). De acordo com o trabalho de Liao et al. (2016), a despeito do micro-organismo escolhido, há o desafio em comum a todos que consiste no aumento das rotas metabólicas de assimilação de carbono por parte destes, concentrando o fluxo metabólico destas cadeias de reações químicas intra-celulares na obtenção do produto de interesse, sejam estas rotas disponíveis naturalmente ou concebidas artificialmente.

Conforme apresentado no trabalho de (MIELENZ, 2001) e Liao et al. (2016), a aplicação de técnicas genômicas para o aprimoramento contínuo do metabolismo microbiano com o fim último de potencializar a produção de bioetanol representa um importante tópico de estudo, haja vista que a grande quantidade de entes biológicos viáveis para essa tarefa permite que não só açúcares simples sejam processados, como outros constituintes da biomassa, a exemplo da celulose e hemicelulose. Assim, o organismo produtor de bioetanol ideal seria capaz de fermentar todos os tipos de açúcares advindos da biomassa, exibindo grande resistência aos monômeros de lignina, acetatos e outros produtos inibitórios, produzindo uma combinação sinérgica de enzimas celulolíticas, necessárias para a completa hidrólise da celulose a partir da matéria-prima (MIELENZ, 2001).

2.2.2.3 Parâmetros físico-químicos

Diversos parâmetros físico-químicos exibem grande importância na performance do processo produtivo do bioetanol. Uma vez que este produto é obtido a

partir do metabolismo dos micro-organismos de trabalho, é natural que as características ambientais que afetam este último impliquem diretamente na produtividade do processo fermentativo.

O aumento na concentração de substrato (ou, em termos simples, açúcares fermentescíveis) implica diretamente no aumento da taxa de fermentação. Contudo, uma concentração excessiva deste pode implicar em uma estabilização da taxa de fermentação, uma vez que nesse cenário, a concentração de substrato ultrapassa a taxa de absorção das células microbianas. Geralmente, a concentração típica de açúcares utilizada é de cerca de 150 g L^{-1} (AZHAR et al., 2017; ZHANG et al., 2015).

O valor do potencial hidrogeniônico (pH) do meio fermentativo também exibe influência na produção do etanol, considerando que esta característica afeta a taxa de contaminação bacteriana, o crescimento celular, a taxa de fermentação e a de formação de produtos. A permeabilidade de certos nutrientes essenciais no meio intra-celular é influenciado pela concentração do íon hidrônio (H_3O^+) no meio fermentativo (ou em outras palavras, pelo pH). A faixa de pH ótimo para a produção de bioetanol por meio da *Saccharomyces cerevisiae* é de 4 a 5 (ZABED et al., 2014; ZABED et al., 2017). De maneira semelhante, a taxa de agitação exibe a mesma relação com o controle da permeabilidade dos nutrientes através da membrana celular, bem como a excreção do bioetanol do ambiente intra-celular para o meio fermentativo; no entanto, o excesso de agitação pode comprometer o metabolismo celular, ou mesmo danificar fisicamente a célula. A taxa de agitação tipicamente empregada para a fermentação por meio de leveduras é de 150 a 200 *rpm* (ZABED et al., 2014).

Outro importante aspecto para a produção do bioetanol por meio da ação dos micro-organismos é a temperatura, uma vez que esse parâmetro exibe uma grande influência nas atividades metabólicas destes organismos. Contudo, há uma temperatura máxima, a partir da qual o aumento neste parâmetro implica no comprometimento dos processos biológicos intracelulares, culminando na morte celular. Assim, a temperatura ideal para a produção de bioetanol em um processo fermentativo está relacionada diretamente a natureza do micro-organismo de trabalho. Tipicamente, para os processos que se valem da utilização do *Saccharomyces cerevisiae*, a temperatura é mantida entre 28 e 30 °C, e para a fermentação utilizando *K. marxianus*, em 42 °C (AZHAR et al., 2017; ZABED et al., 2017). Convém mencionar que temperaturas ligeiramente superiores são reportadas para os processos produtivos baseados em células de micro-organismos imobilizadas, o que acredita-se estar relacionado com a transferência de calor entre a superfície particular e o meio intra-celular (LIU; SHEN, 2008 apud ZABED et al., 2017).

CAPÍTULO 3

Otimização e controle aplicados a processos
biotecnológicos utilizando técnicas
meta-heurísticas

No presente capítulo, serão discutidos os princípios acerca da otimização de processos biotecnológicos a partir da utilização de algoritmos estocásticos, ou meta-heurísticos. Conforme será discutido ao longo deste capítulo, esta classe de técnicas utilizadas para a otimização de problemas apresenta diversas características que vão ao encontro das propriedades intrínsecas aos bioprocessos. Inicialmente, será apresentada uma breve discussão acerca do monitoramento e controle dos bioprocessos, aspectos cruciais para exequibilidade destes no âmbito industrial, os quais apresentam complexidades em virtude das características inerentes aos bioprocessos. Em seguida, será apresentada uma revisão do referencial teórico acerca da otimização de bioprocessos, com ênfase na utilização de técnicas meta-heurísticas com este propósito. Por fim, serão discutidos os problemas de otimização de alta dimensionalidade, os quais representam um importante aspecto das práticas atuais de controle e otimização de processos industriais.

3.1 Monitoramento e controle dos bioprocessos

Com o passar dos anos, os processos biotecnológicos tem sido empregados cada vez mais amplamente na indústria, em razão e diversas razões, dentre as quais podem ser citadas o aumento da qualidade e rentabilidade dos produtos, mudanças na estrutura legislativa em torno do sistema produtivo, entre outros (DOCHAIN, 2008; FESTEL et al., 2012). No entanto, em razão das características intrínsecas a esta classe de processos, surgem diversos desafios para o controle e otimização destes. Em última análise, isso deve-se ao fato de que os micro-organismos (cerne dos processos biotecnológicos) apresentam um comportamento complexo e, portanto, sua modelagem acurada torna-se uma tarefa não trivial, implicando também na dificuldade da reprodutibilidade de experimentos. Como implicação dessa complexidade, pode-se mencionar o fato de que os conjuntos de parâmetros e configurações do modelos de bioprocessos podem sofrer grandes modificações durante o tempo, as quais podem ser consequência de mudanças metabólicas na biomassa (por exemplo, modificações na morfologia celular, implicando em uma variação reológica do sistema) ou mesmo a nível genético. Outra dificuldade reside na ausência de sensores adequados ao monitoramento de processos biotecnológicos, especialmente no que tange a propriedades cruciais para o entendimento do funcionamento interno das entidades biológicas, as quais frequentemente são analisadas em laboratório, possuindo um elevado custo de aquisição e manutenção (DOCHAIN, 2008; NAJAFPOUR, 2007).

Face às dificuldades descritas anteriormente, percebe-se que o desenvolvimento de sistemas de controle e otimização aplicados a sistemas biotecnológicos

depende fortemente de estratégias para o monitoramento destes. Diversos trabalhos enfatizam as vantagens de um monitoramento em tempo real dos bioprocessos – a exemplo de Havlik et al. (2013), Dietzsch et al. (2013), Freitas & Andrade (2015), Sommeregger et al. (2017), entre outros – e ressaltam a sua implicação direta em seu rendimento, produtividade e confiabilidade por meio de um arranjo de controle adequado, valendo-se desta estrutura de monitoramento. Nesse sentido, a análise do posicionamento dos sensores de monitoramento, a estrutura física destes e seu método de medição são aspectos determinantes para o acurado controle dos resultados obtidos com os bioprocessos. Essa busca pela reprodutibilidade e um controle adequado pelo processo têm como grande expoente a subárea da biotecnologia ligada a produção de fármacos, uma vez que a indústria farmacêutica intrinsecamente volta-se de maneira constante a busca por procedimentos inovadores com vistas a maximização do rendimento de produto e sua qualidade.

Com esse intento, diversos procedimentos oriundos de agências reguladoras foram desenvolvidas com vistas a garantir esses resultados desejáveis, dentre as quais podemos mencionar como exemplo relevante a iniciativa PAT (*Process Analytical Technology*, ou tecnologia analítica de processo) do FDA (*United States Food and Drug Administration*), iniciativa esta que foi seguida imediatamente pelas agências análogas européia e japonesa, respectivamente, EMEA e MHLW (GNOTH et al., 2007; SIMON et al., 2015). Uma vez que os já mencionados princípios biotecnológicos são comuns a outros tipos de processos, tais como o tratamento de resíduos, indústria alimentícia, bem como a produção de biocombustíveis (tendo na produção de etanol por via fermentativa o exemplo mais icônico desta), os procedimentos necessários para a obtenção dos desejáveis resultados para a indústria de fármacos é transferida naturalmente para estas outras subáreas da biotecnologia. Em linhas gerais, a iniciativa PAT reside na análise da implicação dos parâmetros de processo críticos (CPP, do inglês *critical process parameters*) nos atributos de qualidade críticos (CQA, do inglês *critical quality attributes*), monitorando os CPP ao longo de toda a estrutura produtiva (FREITAS; ANDRADE, 2015). Na Figura 3, a implementação do PAT é representada esquematicamente por meio de um processo cíclico.

Em termos da configuração física dos sensores aplicados ao monitoramento de bioprocessos, estes podem ser classificados como *ex-situ* ou de mensura indireta, no qual o material é retirado diretamente do meio reacional e transportado para uma estrutura externa para a medição das propriedades de interesse; *in-situ* ou de medida direta, no qual a medida é realizada a partir da amostragem no próprio meio reacional. Os sensores também podem ser dispostos a partir de uma configuração híbrida, em combinação dessas duas classes, por exemplo em que o elemento sensor está localizado externamente ao meio reacional entretanto a amostra é retirada diretamente do meio reacional (SIMON et al., 2015; FREITAS; ANDRADE, 2015;

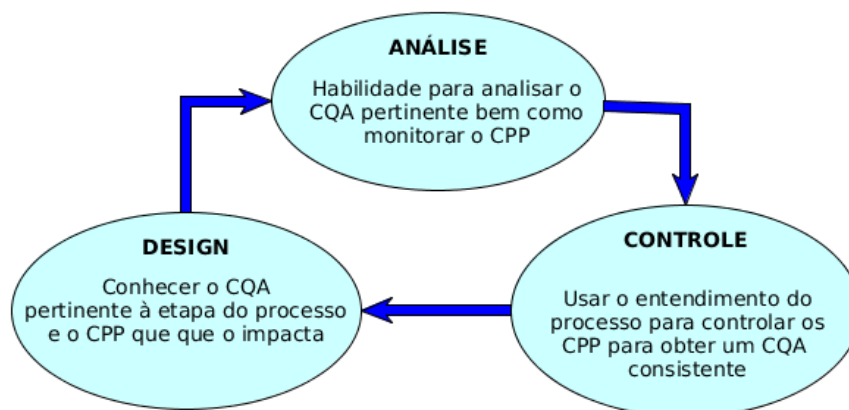


Figura 3 – Representação esquemática do processo cíclico e contínuo de implementação do PAT.

Fonte – Adaptado de Gnoth et al. (2007).

CHRISTIAN et al., 2018). Na Figura 4, exemplos de diferentes configurações são representadas esquematicamente.

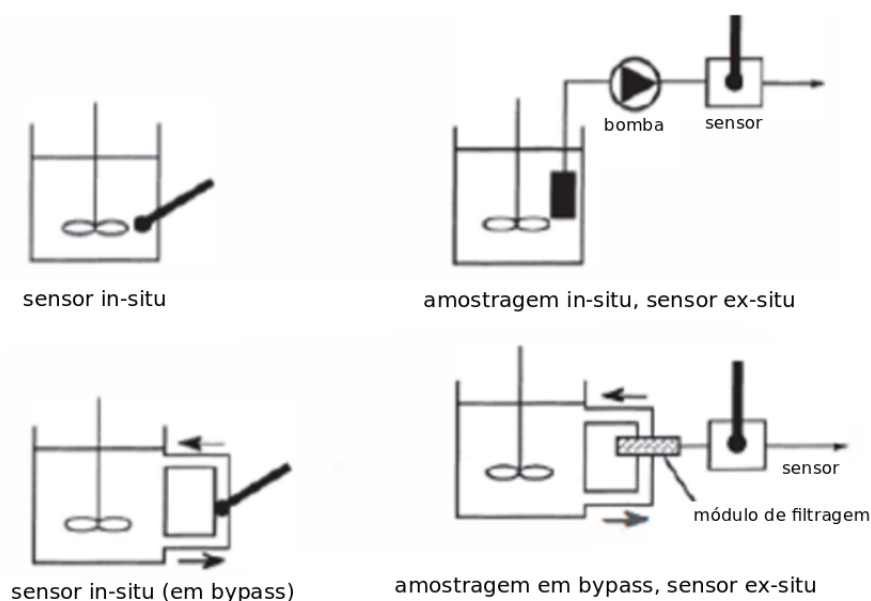


Figura 4 – Representação esquemática de algumas das diferentes configurações possíveis para a aplicação de sensores para o monitoramento de bioprocessos.

Fonte – Adaptado de Najafpour (2007).

Por outro lado, os sensores podem também serem classificados em termos do mecanismo de mensura. Nesse sentido, os referidos elementos de mensura podem ser classificados em: métodos ópticos, dentre cujas aplicações podemos destacar as técnicas espectroscópicas (FTIR, MIR, NIR, Raman, etc); métodos eletroquí-

micos, referidos como uma das técnicas comuns para o sensoriamento dos processos biotecnológicos, baseados na transferência de carga de um eletrodo para o meio de amostragem; métodos baseados em análise separativa, cujas aplicações de destaque são a análise da fase gasosa do biorreator e/ou a fase líquida do meio reacional, por meio de técnicas como a cromatografia gasosa (GC), espectroscopia de massa (MS) ou cromatografia líquida de alta performance (HPLC) Freitas & Andrade (2015), Zhao et al. (2015), Harms et al. (2002). Convém mencionar que frequentemente múltiplos sensores são empregados, inclusive como entrada para modelos que relacionam diversas medidas em prol de um resultado de uma outra propriedade de interesse. Por exemplo, a partir dos dados de pH e teor de um metabólito de interesse, podem ser inferidos valores de NADH, uma grandeza intracelular que de outra forma teria a sua mensura direta dificultada. Esses “sensores virtuais” são chamados de *soft-sensors* (ou *software-sensors*) (LUTTMANN et al., 2012). Na Figura 5, uma representação de uma estrutura de monitoramento e controle de um bioprocessamento estabelecida a partir da utilização simultânea de diversos tipos de sensores pode ser visualizada, a qual representa esquematicamente uma malha de controle fechada.

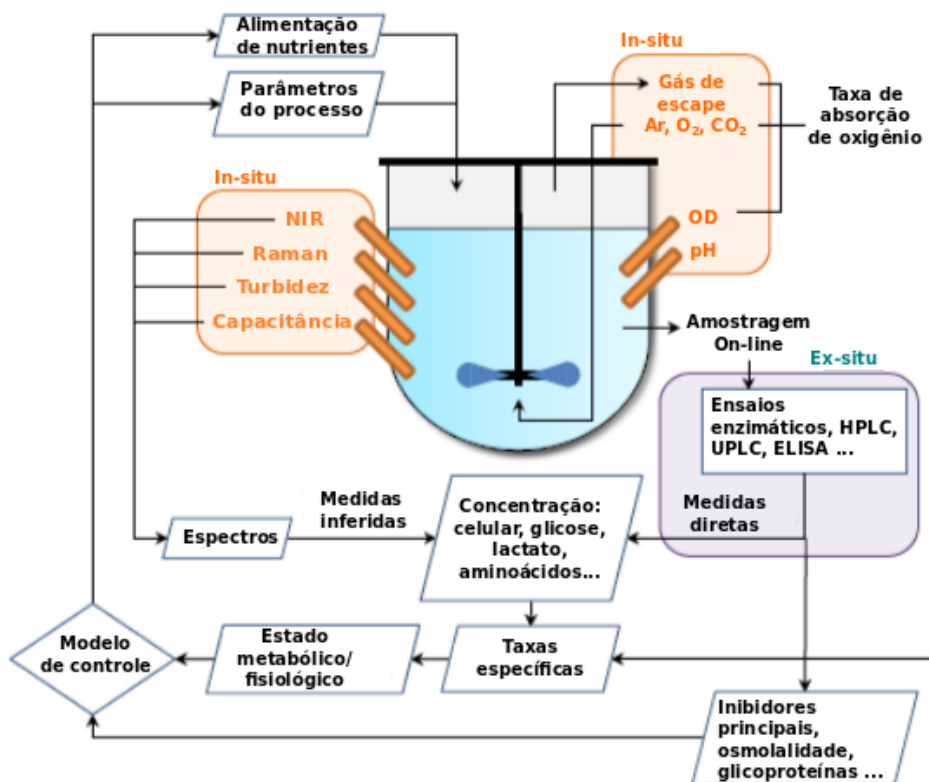


Figura 5 – Representação esquemática de uma estrutura de monitoramento e controle de um processo biotecnológico, a partir de uma multitude de sensores diferentes.

Fonte – Adaptado de Zhao et al. (2015).

3.2 Otimização de bioprocessos utilizando técnicas meta-heurísticas

Conforme já foi descrito anteriormente, os bioprocessos apresentam uma série de características intrínsecas a sua classe. Nesse sentido, estes processos podem ser bastante beneficiados em termos da sua viabilidade econômica, controlabilidade e segurança por meio de sua otimização. Esta tem sido empregada em aplicações relacionadas a processos biotecnológicos no âmbito da identificação de parâmetros cinéticos reacionais, resolução de problemas de otimização dinâmica em processos conduzidos no regime de batelada (ou batelada alimentada), no planejamento de plantas integradas, de sistemas de controle baseados em otimização, em estudos de condições operacionais ótimas (OCHOA et al., 2009; BANGA et al., 2005; JABARIVELISDEH; WALDHERR, 2016).

Em síntese, a otimização pode ser definida pela busca de uma solução que minimize ou maximize uma determinada função de custo (relacionada direta ou indiretamente ao processo em tela), também chamada de *função objetivo*, atendendo a um conjunto de restrições quando este é determinado. Matematicamente, este pode ser definido conforme mostram as Equações 1-5, a seguir. O significado dos termos apresentados nas referidas equações é apresentado na Tabela 1.

$$\min J(u, x, \tilde{p}, f, c, t) \quad (1)$$

seja:

$$f(\tilde{p}, u, x, t) = 0 \quad (2)$$

$$\dot{f}(\tilde{p}, u, t) = 0 \quad (3)$$

$$\tilde{p}(u, x, t) = 0 \quad (4)$$

sujeito a:

$$c(u, t) = 0 \quad (5)$$

Em termos matemáticos, as abordagens para o tratamento do problema de otimização podem ser divididas entre determinísticas, em que se utilizam operações específicas em termos da escolha conjuntos-solução candidatos e sua implicação no resultado da função objetivo; estocástica, na qual são empregadas operações de natureza randômica, em paralelo às implicações da escolha dos conjuntos-solução

Tabela 1 – Representação matemática dos termos utilizados na definição de um problema de otimização, e seu significado.

Termo	Significado
J	Função objetivo a ser minimizada
f	Modelo matemático do processo
\dot{f}	Equações diferenciais que definem o modelo
c	Restrições para a otimização dos parâmetros
\tilde{p}	Relações constitutivas que definem os parâmetros
u	Variáveis a serem otimizadas
x	Variáveis de estado do modelo
t	Tempo (para modelos dinâmicos)

como empregadas nas abordagens determinísticas. No que tange aos bioprocessos, suas características de não linearidade, grande número de variáveis nos conjuntos-solução e frequente ocorrência de funções-objetivo multimodais acarretam em uma performance superior para as técnicas estocásticas em termos da sua taxa de convergência e menor complexidade em sua implementação, quando comparadas com as suas variantes determinísticas (BANGA et al., 2005; OCHOA et al., 2009; ROCHA et al., 2014; JABARIVELISDEH; WALDHERR, 2016; OCHOA, 2016).

Sob a luz dos processos biotecnológicos, o desafio consiste no controle de um processo produtivo para seu estado ótimo a fim de que seja atingida sua máxima produtividade com o menor custo possível. Recentemente o foco dos estudos acerca da aplicação de abordagens de engenharia de processos na otimização de bioprocessos ainda tem consistido na otimização dinâmica (ou controle ótimo em malha aberta), especialmente em processos conduzidos em batelada alimentada, nos quais a taxa de alimentação de substrato consiste na principal variável manipulada (ROCHA et al., 2014; BANGA et al., 2005); não obstante, diversos trabalhos tem apontado a importância de empregar técnicas de otimização e controle mais robustas, a exemplo do controle por modelo não linear, ou NMPC, do inglês *Non-linear Model Predictive Control* (Ochoa et al. (2010), Pantano et al. (2017)); análise de fluxo dinâmica, ou DFBA, do inglês *Dynamic Flux Balance Analysis* (Nikdel et al. (2018), Chowdhury et al. (2014)); otimização e baseada em sistemas *fuzzy* (Wang et al. (2014), Márquez-Vera et al. (2018), Davidson (2018)), entre outras.

Os bioprocessos apresentam características intrínsecas que os diferenciam de outros processos químicos, especialmente no que tange ao grau de complexidade em termos dos modelos utilizados em sua descrição. Conforme já foi discutido nas seções anteriores, as características inerentes aos processos biotecnológicos

(comportamento dinâmico e frequentemente não linear, inibição por parte de produtos e subprodutos, entre outros) implicam em dificuldades para as técnicas clássicas para a otimização global destes, de modo que as meta-heurísticas figuram como alternativas notáveis para tal objetivo. Apesar de técnicas analíticas para otimização de bioprocessos serem utilizadas em casos mais simples, estas tornam-se excessivamente complexa quando o número de variáveis de estado e controle aumenta, de modo que as abordagens numéricas apresentam-se como candidatas fiáveis para a tarefa. Dentre essas, as abordagens estocásticas – a exemplo das técnicas evolutivas e meta-heurísticas – destacam-se em razão das já mencionadas características de facilidade de implementação, taxa de convergência e relativa boa performance em problemas envolvendo funções multimodais (ROCHA et al., 2014). Nesse sentido, serão relatados a seguir alguns exemplos de utilização destas referidas técnicas na otimização de processos biotecnológicos, contudo sem a pretensão de esgotar a multitude de referências disponíveis na literatura.

No trabalho de Egea et al. (2007), os autores apresentam um estudo acerca da utilização da meta-heurística de busca por dispersão aplicada a problemas dentre os quais destacam-se a estimação de parâmetros de um problema de rotas metabólicas e a síntese de uma planta de tratamento de esgotos. Os referidos tópicos estudados correspondem a problemas de referência, apresentados nos trabalhos de (MOLES et al., 2003b) e (MOLES et al., 2003a), respectivamente. Convém mencionar que o estudo da estrutura genética e da expressão de proteínas nos micro-organismos de trabalho (respectivamente, a genômica e proteômica), a exemplo do que pode ser visto no trabalho de Egea et al. (2007) e outros que serão citados doravante, representam importantes ferramentas para a otimização de bioprocessos em termos de sua produtividade. Somam-se a essas técnicas a utilização de fluidodinâmica computacional, que permite a análise da influência de determinadas configurações operacionais na reologia dos biorreatores, bem como uma maior compreensão da transferência de massa e energia durante o processo (WANG et al., 2009).

No trabalho de Yüzgeç et al. (2009), os autores apresentam a aplicação de uma estratégia baseada em algoritmos genéticos para a otimização de um processo em batelada-alimentada para produção de biomassa e outros produtos celulares a partir do micro-organismo *Saccharomyces cerevisiae*, contudo buscando minimizar a formação de etanol como subproduto indesejável, uma vez que esse comprometeria a qualidade e quantidade final dos produtos desejados ao fim do cultivo. Para esse intento, os autores ressaltam a importância de manter a taxa de crescimento próxima ao seu valor crítico, avaliando os perfis determinados para as variáveis manipuladas por meio da otimização em biorreatores em escala industrial, incluindo fontes de perturbação características deste bioprocessos tipicamente encontradas no âmbito industrial. Os resultados obtidos endossam a robustez da meta-heurística

aplicada para o controle do processo em tela.

O trabalho de Ochoa et al. (2009) apresenta a implementação de um algoritmo estocástico para otimização de bioprocessos, baseados no comportamento observado em moléculas em solução, chamado de têmpera paralela inspirada em moléculas. O desempenho da meta-heurística implementada pelos autores foi avaliada por meio de sua aplicação em dois casos de estudo, a otimização dinâmica de um processo em batelada alimentada utilizando perfis de alimentação não lineares baseados em funções cossenoidais, e a identificação dos parâmetros cinéticos referentes a um modelo do processo de produção de bioetanol. A rotina desenvolvida teve seus resultados comparados a meta-heurísticas clássicas como o SA (KIRKPATRICK et al., 1983) (do inglês *simulated annealing*, ou recozimento simulado) e GA (do inglês *genetic algorithm*, ou algoritmo genético), tendo sido obtidos resultados superiores para a implementação apresentada neste trabalho. Convém mencionar que analisando o algoritmo desenvolvido quanto às bases de seu funcionamento, este se assemelha a uma versão modificada da rotina de otimização por PSO (KENNEDY; EBERHART, 1995) (do inglês *particle swarm optimization*, ou enxame de partículas) acrescida do critério de aceitação de Metropolis (METROPOLIS et al., 1953) para as novas soluções no processo iterativo.

No trabalho de Egea et al. (2010), os autores apresentam um estudo acerca da utilização de uma meta-heurística baseada na busca por dispersão, chamada pelos autores de *evolutionary algorithm for complex-process optimization*, ou *algoritmo evolutivo para otimização de processos complexos*. A técnica baseia-se principalmente na implementação de operadores para combinação de soluções candidatas baseadas em hiper-retângulos, e na atualização da população de soluções candidatas por meio de uma estratégia desenvolvida no referido trabalho. Os autores avaliaram o desempenho da meta-heurística por eles desenvolvida por meio de dois problemas de referência, a síntese de uma planta de tratamento de esgotos real (MOLES et al., 2003a) e a secagem de placas de celulose embebidas com ácido ascórbico (BANGA; SINGH, 1994). Quando comparados a outros algoritmos de meta-heurística, a rotina desenvolvida pelos autores apresentou resultados competitivos para os problemas em tela.

O trabalho de Rocha et al. (2014) apresenta um estudo acerca da otimização de quatro diferentes processos biotecnológicos em batelada-alimentada na forma de problemas de referência: obtenção de uma proteína recombinante por meio do cultivo de *E. Coli* (ROCHA; FERREIRA, 2002; ROCHA et al., 2004); produção de etanol por meio do cultivo de *Saccharomyces Cerevisiae* (CHEN; HWANG, 1990); produção de anticorpos monoclonais por meio de hibridoma (ROUBOS et al., 1999); desenvolvimento de esquemas otimizados de controle para a produção induzida de proteínas heterólogas em bactérias (LEE; RAMIREZ, 1994). Tal estudo dá-se

por meio da aplicação de três meta-heurísticas bastante referenciadas na literatura, algoritmos genéticos (HOLLAND et al., 1992) (GA, do inglês *Genetic Algorithm*), evolução diferencial (STORN; PRICE, 1997) (DE, do inglês *Differential Evolution*) e otimização por enxame de partículas (KENNEDY; EBERHART, 1995) (PSO, do inglês *Particle Swarm Optimization*). Os autores relatam que a evolução diferencial utilizando a variante aleatória na mutação das soluções candidatas exibiu os melhores resultados para todos os casos de estudo, com resultados pouco superiores à técnica GA. Por fim, os autores discutem brevemente acerca da natureza *gulosa* (em inglês, *greedy*) das meta-heurísticas representa um obstáculo para seu desempenho, enfatizando a necessidade de que se estabeleça um balanceamento entre este comportamento e a preservação da diversidade nas soluções candidatas (ou, em outras palavras, sua aleatoriedade), o que é referenciado por outros autores como a ponderação entre exploração e exploração do espaço de busca no decorrer da otimização, ou ainda entre a busca local e global neste.

No trabalho de Zain et al. (2018b), os autores apresentam uma aplicação do algoritmo de busca por retrocesso (BSA, do inglês *backtracking search algorithm*), o qual é referido como uma evolução da meta-heurística de evolução diferencial, que por sua vez como uma técnica robusta e eficiente na otimização de problemas no âmbito dos processos biotecnológicos. De acordo com os autores, a meta-heurística de BSA realiza um balanceamento entre busca local e global, aliado a um menor número de parâmetros de controle (apenas um), ainda que apresente desvantagens quanto a taxa de convergência e precisão. No trabalho, estes comparam o desempenho do BSA com quatro meta-heurísticas, escolhidas em razão de suas características de interesse: a estratégia de adaptação da evolução de matriz de covariância (HANSEN; OSTERMEIER, 2001) (CMAES, do inglês *covariance matrix adaptation evolution strategy*), que é apresentada como uma técnica recente de inteligência de enxame, com boa taxa de convergência global; otimização por colônias de abelhas artificiais (KARABOGA; BASTURK, 2007) (ABC, do inglês *artificial bee colony*), referida como uma técnica de inteligência de enxame amplamente utilizada, apresentando resultados promissores em problemas variados; algoritmo de algas artificiais (UYMAZ et al., 2015) (AAA, do inglês *artificial algae algorithm*), apresentada como a evolução das meta-heurísticas modernas baseadas em inteligência de enxame; evolução diferencial, referida como uma técnica bem estabelecida no campo da otimização de problemas de fermentação do tipo batelada-alimentada. Os estudos de caso utilizados pelos autores na comparação das meta-heurísticas correspondem a diversos problemas de referência de fermentação em batelada alimentada, compreendendo diversos aspectos da aplicação em âmbito industrial dos mesmos: produção de etanol por meio de *Saccharomyces cerevisiae* (CHEN; HWANG, 1990); produção induzida de proteínas heterólogas em bactérias recombinantes (LEE; RA-

MIREZ, 1994); produção de penicilina (BANGA et al., 2005); estágios do tratamento de efluentes da indústria vinícola por meio de lagoas aeradas em escala piloto (MONTALVO et al., 2010); fermentação metanogênica a partir do lodo encontrado na fração sólida de esgotos, um problema estendido pelos autores a partir do trabalho original de (SOSNOWSKI et al., 2008). A partir dos resultados, os autores relatam que para o problema cuja solução é do tipo não limitada, referente à produção de proteínas heterólogas, o desempenho das meta-heurísticas analisadas foi semelhante. De modo geral, para os outros problemas de referência, nos quais constam limitações para os parâmetros a serem otimizados, o BSA apresentou os melhores resultados.

A otimização multiobjetivo também figura como um tópico de interesse no âmbito dos bioprocessos. No trabalho de Zain et al. (2018a), os autores apresentam uma meta-heurística implementada a partir do algoritmo de otimização por enxame de partículas especialmente desenvolvida para problemas multiobjetivo (chamado de M-MOPSO, do inglês *Modified Multi-Objective Particle Swarm Optimization*), considerando também as deficiências do referido algoritmo quando estendido a problemas de alta dimensionalidade e na atenção às restrições nas variáveis de decisão do problema. O algoritmo desenvolvido teve seu desempenho analisado por meio de problemas de referência no âmbito de bioprocessos e tratamento tumoral, a saber: produção de lisina por meio de fermentação (OHNO et al., 1976), objetivando maximizar a produtividade específica ($g L^{-1} h^{-1}$) em termos do produto, bem como o rendimento percentual do processo em termos do reagente; produção induzida de proteínas heterólogas em bactérias recombinantes (LEE; RAMIREZ, 1994), buscando maximizar a produção de proteínas em paralelo à minimização da quantidade de indutor alimentada no biorreator; modelo para tratamento quimioterápico a tumores (PILLIS; RADUNSKAYA, 2003), no qual o intento é de reduzir a concentração de células cancerosas e o volume de medicação fornecida ao paciente. Com o objetivo de fornecer meios para a comparação do M-MOPSO com outros algoritmos, o desempenho deste também foi avaliado a partir dos problemas de teste referidos como de grande dificuldade (CF1-CF10) referentes ao *Congress of Evolutionary Computation 2009* (CEC2009) (ZHANG et al., 2008). Os resultados mostram que para os problemas de referência do CEC2009, o M-MOPSO apresentou no geral resultados melhores em comparação as outras meta-heurísticas avaliadas, exceto para duas das dez funções avaliadas (CF1 e CF10), para as quais figurou como o segundo melhor algoritmo. Quanto aos problemas de referência correspondentes aos bioprocessos, o M-MOPSO obteve uma frente de Pareto rivalizável com o algoritmo mais bem sucedido para o primeiro problema, e um desempenho indistinto dos demais para o segundo, entretanto os autores ressaltam que a rotina por eles desenvolvida obteve uma maior abrangência em sua frente de Pareto. Em termos do problema

de tratamento tumoral, um comportamento semelhante ao observado no segundo problema de fermentação (produção induzida de proteínas heterólogas) foi obtido, para três diferentes cenários de tratamento quimioterápicos avaliados.

3.3 Problemas de otimização de alta dimensionalidade

Os problemas de otimização de alta dimensionalidade (POAD) e suas características intrínsecas constituem um dos ramos de maior interesse no âmbito do estudo de meta-heurísticas, especialmente no que concerne problemas em domínio contínuo, conforme descreve LaTorre et al. (2015). Face à crescente integração material e energética observada em geral nos processos industriais, aliada a um significativo incremento no grau de automação destes, o estudo sistemático de sua otimização invariavelmente conta com um grau de complexidade significativo, o qual pode ser traduzido na dimensão do problema matemático intrínseco, ou seja, em sua dimensionalidade. A seguir, serão brevemente discutidas três características bastante importantes para o estudo da otimização de funções: a modalidade, separabilidade e dimensionalidade. Convém mencionar que as referidas propriedades matemáticas mostram-se bastante determinantes no desempenho da resolução de problemas de otimização, independente de sua escala, entretanto quando estes são escalados à alta dimensionalidade, essas características podem afetar de maneira significativa o desempenho de algumas meta-heurísticas.

3.3.1 Características matemáticas dos problemas de otimização

A modalidade refere-se a existência ou não de múltiplos picos ou vales no gráfico funcional de uma função em termos de suas variáveis, característica essa que representa um desafio ao desempenho das técnicas meta-heurísticas, haja vista que estas podem reter o desenvolvimento das soluções ao longo das iterações do algoritmo, levando a solução a ficar retida em pontos de ótimo locais, ao invés atingir o ótimo global (MUKHOPADHYAY; DAS, 2016; JAMIL; YANG, 2013; CHANG, 2015). Quando uma função apresenta apenas um ponto ótimo, diz-se que esta é *unimodal*. Quando estão presentes mais de um destes, a função é referida como *multimodal*.

Na Figura 6, são representados graficamente os valores obtidos para uma função unimodal, cujo ponto de mínimo global correspondem a $f(x_1, x_2) = 0$ para $(0,0)$. É possível perceber que o comportamento da função é bastante diverso daquele representado nas Figuras 7 e 8, nas quais são representados graficamente os valores obtidos para funções multimodais, de modo a ilustrar a complexidade envolvida na otimização de modelos de processos biotecnológicos.

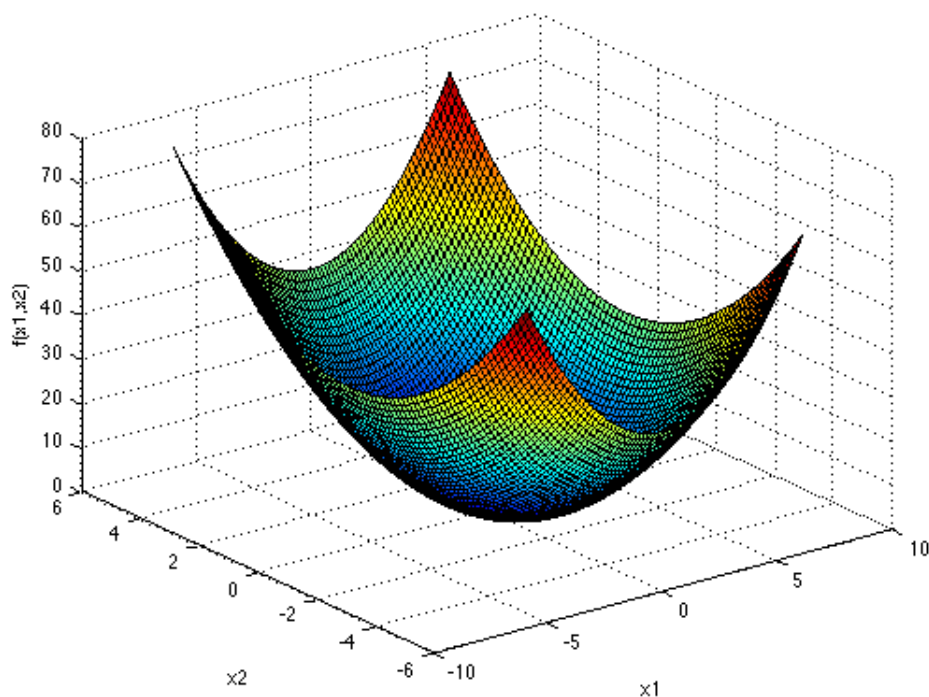


Figura 6 – Representação gráfica da forma da função *sphere* (esfera), ilustrando a aparência gráfica de uma função unimodal. A referida função matemática possui apenas um mínimo global.

Fonte – Adaptado de Surjhanovic & Bingham (2013).

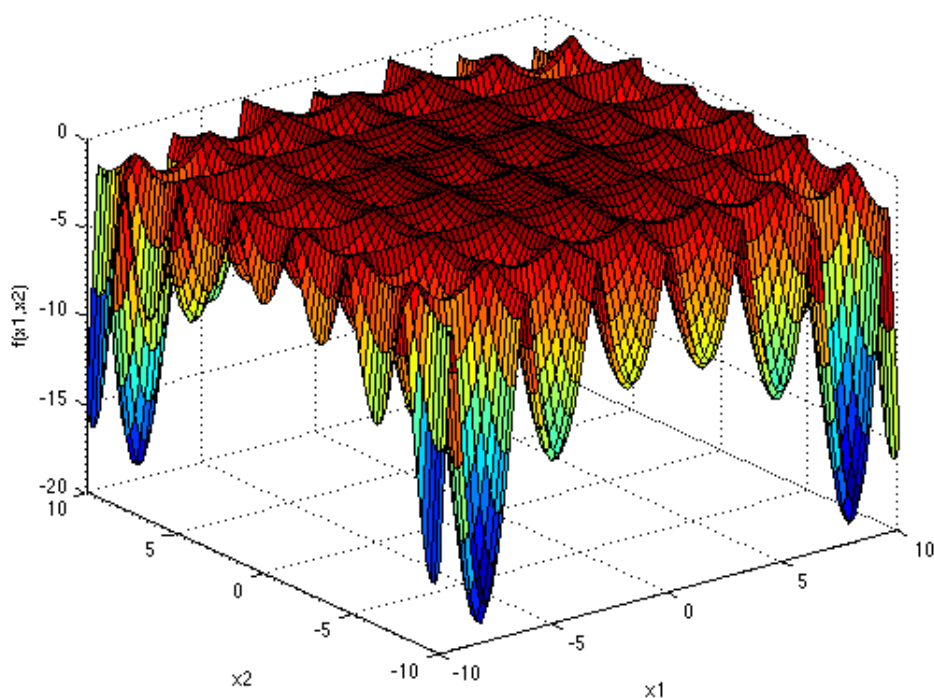


Figura 7 – Representação gráfica da forma da função *holder table* (mesa de suporte), ilustrando a complexidade inerente à otimização de funções multimodais. A referida função matemática possui diversos pontos de mínimo local e quatro pontos de mínimos globais.

Fonte – Adaptado de Surjhanovic & Bingham (2013).

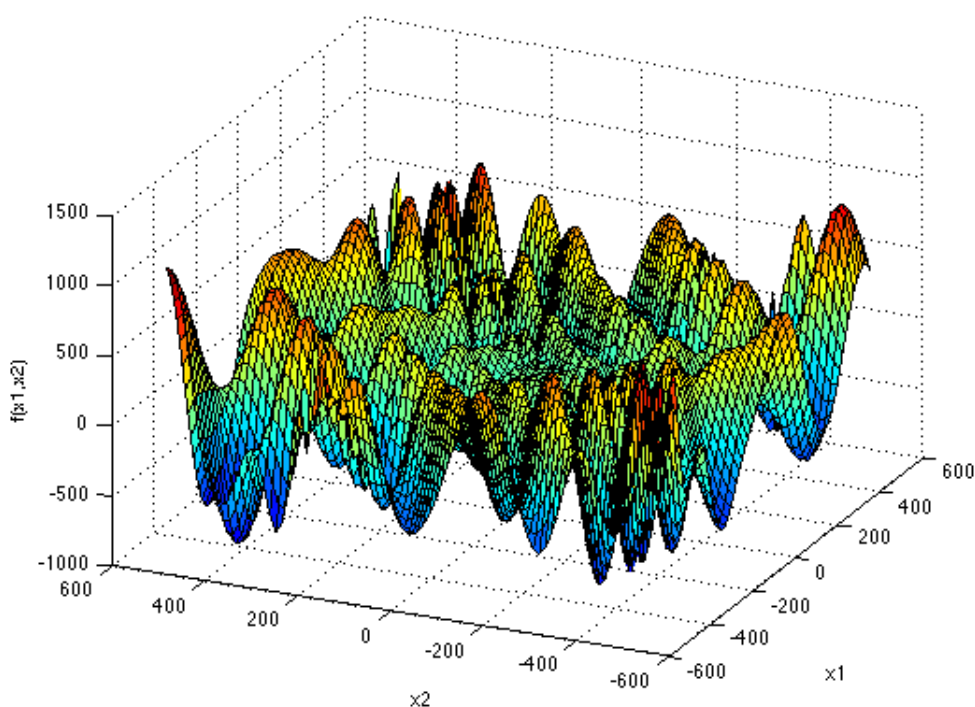


Figura 8 – Representação gráfica da forma da função *egg holder* (caixa de ovos), ilustrando a complexidade inerente à otimização de funções multimodais. A referida função matemática possui diversos pontos de mínimo local e um mínimo global.

Fonte – Adaptado de Surjhanovic & Bingham (2013).

Acerca das Figuras 6 e 7, convém mencionar que a existência de diversos “picos” (ou “vales”) na topografia funcional – característica essa que corresponde à *modalidade* – representa um obstáculo ao desempenho do algoritmo de busca, uma vez que existe uma tendência que estes fiquem “presos” nestes locais, e não consigam encontrar os pontos de máximo ou mínimos globais. Posteriormente, na Figura 8, uma função ainda mais complexa sob o ponto de vista da otimização é mostrada, a função *egg holder*. Ambas estas funções tem o seu comportamento conhecido e são utilizadas como problemas de referência (ou *benchmark*) de algoritmos de otimização.

À medida que aumentam-se o número de variáveis de decisão do problema de otimização, além do incremento exponencial no esforço computacional, observa-se uma deterioração no desempenho dos mesmos, um conceito introduzido no trabalho de Bellman (1957), chamado de *maldição da dimensionalidade* (BERGH; ENGELBRECHT, 2004; WANG et al., 2011; HUANG; MOHAN, 2006; WANG et al., 2013). Este conceito compreende dois aspectos: um referente à deterioração da capacidade das soluções iniciais geradas pelo algoritmo se aproximarem da região de ótimo, uma região definida como uma hipersfera que circunda o ponto de ótimo

global no espaço-solução do problema; e a natureza mutável de algumas funções, que podem apresentar características diferentes quando são expandidas para um número superior de dimensões, a exemplo da função de *Rosenbrock*, que é unimodal para duas dimensões, tornando-se multimodal a medida que esta é ampliada para um número superior de dimensões (BERGH; ENGELBRECHT, 2004; LI et al., 2013; SHANG; QIU, 2006). Diversos autores classificam os problemas de otimização como de baixa-dimensionalidade quando o número de variáveis de decisão nestes é igual ou inferior a uma centena (WANG et al., 2013; CHEN et al., 2015b; MAHDAVI et al., 2015), e de alta dimensionalidade quando este número é superior ao referido patamar.

Considerando o conceito de *maldição da dimensionalidade*, uma representação esquemática do princípio do referido conceito pode ser visualizada na Figura 9. Em termos de um espaço euclidiano $\{x \forall x \subseteq \mathcal{R}^n\}$, imagina-se uma hiperesfera que circunda a região do ótimo global, chamada de região de otimalidade, e um hiper-cubo representando o espaço de busca, e a razão de seus volumes definida como r . Para um espaço de baixa dimensionalidade, o valor de r é próximo à unidade, valor este que vai se distanciando, culminando em valores diminutos para números elevados no número de dimensões (ou variáveis de decisão no problema em tela). Conforme descrito em Bergh & Engelbrecht (2004), a medida que aumenta a dimensionalidade do problema, a probabilidade de que as soluções candidatas geradas por um algoritmo estocástico por meio de uma distribuição uniformemente aleatorizada, incorra na referida hiperesfera que circunda o ponto de ótimo global são determinadas por r , e vão decrescendo, explicando a deterioração no desempenho do algoritmo.

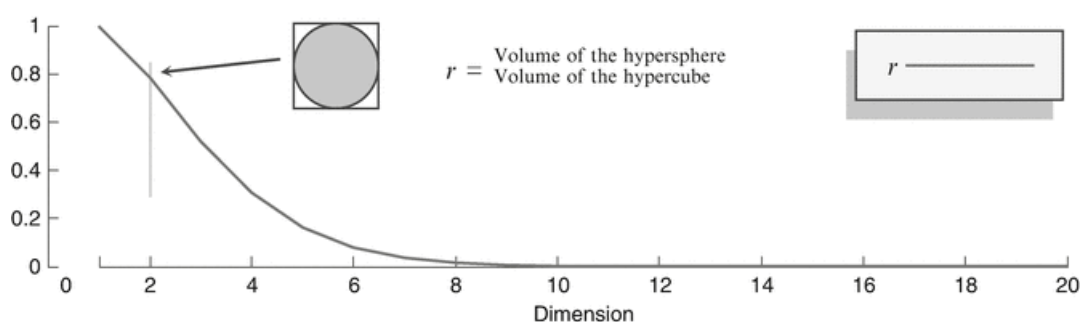


Figura 9 – Representação gráfica da *maldição da dimensionalidade*.

Fonte – Adaptado de Keogh & Mueen (2017).

Outro importante conceito no estudo das características das funções as quais deseja-se analisar sob o aspecto da otimização consiste na separabilidade. Esse conceito representa a dificuldade inerente à otimização da função em tela, uma vez que este remete a interdependência entre as variáveis funcionais (GODA, 2013; MUKHOPADHYAY; DAS, 2016; JAMIL; YANG, 2013). Conforme descrevem Mukho-

padhyay & Das (2016) e Jamil & Yang (2013), as funções ditas completamente separáveis podem ser otimizadas a partir de n processos de otimização independentes, correspondendo ao somatório de contribuições individuais para cada função univariável, sendo n a dimensionalidade da referida função. De acordo com os autores, essas afirmações podem ser representadas matematicamente por meio das Equações 6 e 7.

$$\frac{\partial f(\bar{x})}{\partial x_i} = g(x_i) h(\bar{x}) \quad \forall i \in \{1, 2, \dots, n\} \quad (6)$$

$$f(\bar{x}) = \sum_{i=1}^n f_i(x_i) \quad (7)$$

Nas Equações 6 e 7, os termos \bar{x} , x_i , f , f_i e n representam respectivamente o conjunto de variáveis para o qual a função é definida, a i -ésima variável deste conjunto, a função a ser otimizada (definida em termos de \bar{x}), a i -ésima função que compõe a função global (definida em termos de x_i), e a dimensionalidade da função $f(\bar{x})$.

3.3.2 Breve revisão da literatura acerca do tema

Abaixo, serão discutidos alguns trabalhos que se debruçam sobre o tema da utilização de meta-heurísticas em problemas de otimização de alta dimensionalidade, em termos das principais conclusões estabelecidas pelos autores, e aspectos importantes suscitados por estes para a compreensão do tema.

No trabalho de Mahdavi et al. (2015), os autores relatam que as meta-heurísticas desenvolvidas para problemas de otimização global podem ser divididas em duas grandes categorias: algoritmos de co-evolução cooperativa, baseados na estratégia de decomposição dos problemas em subcomponentes a partir da redução dimensional, utilizando-se do paradigma de dividir para conquistar; algoritmos baseados em métodos de não-decomposição, os quais resolvem os problemas de otimização de grande dimensionalidade como um todo, empregando operadores específicos ou combinações com métodos alternativos para a exploração de espaços de busca complexos. Os autores ressaltam a importância da utilização de técnicas eficientes para a otimização de funções não-separáveis ou parcialmente não-separáveis, nas quais ocorre uma forte interação entre algumas (ou todas) das variáveis de decisão do problema. Também salienta-se a necessidade de avaliar o desempenho de meta-heurísticas dedicadas a problemas de alta dimensionalidade em problemas oriundos de aplicações reais, ao invés de limitar a sua análise a problemas de referência.

Em Chen et al. (2015b), os autores apresentam análises acerca da influência da dimensionalidade dos problemas de otimização e o desempenho dos algoritmos em tela, para duas meta-heurísticas bastante populares na literatura especializada: a evolução diferencial (DE) e a otimização por enxame de partículas (PSO). Como conclusão de seu estudo, os autores apontam que tempo de convergência parece crescer linearmente com o aumento da população. Segundo os autores, o simples aumento populacional não torna os mecanismos de busca capazes de um desempenho satisfatório em problemas de alta dimensionalidade, uma vez que utilizando os mecanismos básicos do PSO e DE, estes convergirão prematuramente em um hiperplano de dimensionalidade inferior ao espaço de busca global, afetando a convergência dos algoritmos.

No trabalho de Yang et al. (2007), duas implementações modificada da meta-heurística clássica de evolução diferencial são apresentadas, chamadas de evolução diferencial com coevolução cooperativa (DECC, do inglês *Differential Evolution with Cooperative Coevolution*). De acordo com os autores, esta implementação visa tratar da escalabilidade dos algoritmos derivados de evolução diferencial aplicados a problemas de alta dimensionalidade, utilizando princípios de coevolução cooperativa baseados na estratégia de decomposição do problema em subcomponentes, os quais são otimizados separadamente e então combinados. O primeiro algoritmo implementado pelos autores, chamado de DEEC-I, baseia-se na otimização dos subcomponentes obtidos a partir da subdivisão do problema principal, a atribuição de um peso para cada um destes subcomponentes na reconstrução da solução para o problema principal, e a posterior otimização destes pesos; o DEEC-II baseia-se em um princípio mais simples, em que um subcomponente é escolhido aleatoriamente, e otimizado, e por fim os subcomponentes são combinados no conjunto solução para o problema principal. Os algoritmos foram avaliados a partir de problemas de referência, Suganthan et al. (2005), e resultados obtidos mostram que o desempenho dos dois algoritmos foi bastante próximo, com alguma vantagem para o DEEC-I.

De maneira semelhante, no trabalho de Omidvar et al. (2014) os autores apresentam uma estratégia para a decomposição automática do conjunto de variáveis de decisão do problema aliada à coevolução cooperativa, chamada de agrupamento diferencial, obtido a partir da definição matemática de funções aditivamente separáveis. No referido trabalho, consta também uma análise das diferentes categorias de estratégias para a decomposição do problema de otimização. De acordo com o trabalho de Omidvar et al. (2014), os resultados obtidos indicam que o método de agrupamento diferencial permite uma melhor escalabilidade de meta-heurísticas aplicadas a problemas de alta dimensionalidade, e que esta estratégia de decomposição possibilita que as contribuições individuais de cada subcomponente sejam quantificadas em termos do valor global da função objetivo.

Alguns autores adotam a estratégia de aprendizado baseado em oposição (OBL, do inglês *Opposition-Based Learning*), – classificada como técnica de não-decomposição, de acordo com a classificação apresentada em Mahdavi et al. (2015) – para o aumento da taxa de convergência. No trabalho de Wang et al. (2011), os autores empregam a referida estratégia para expandir a meta-heurística de evolução diferencial. o desempenho da rotina implementada foi comparado ao de outras meta-heurísticas frente a uma série de problemas de referência, apresentadas em (TANG et al., 2007; HERRERA; LOZANO, 2009; HERRERA et al., 2010), tendo sido obtidos valores que demonstram que esta é no geral superior ou equivalente às demais meta-heurísticas avaliadas. Entretanto, para alguns problemas de teste específicos, o algoritmo apresentou dificuldades na busca pelo ótimo global.

No trabalho de Wang et al. (2013), os autores tratam do problema da alta demanda computacional e dificuldades inerentes aos problemas de otimização de alta-dimensionalidade por meio da paralelização dos cálculos na meta-heurística de evolução diferencial, acrescida de aprendizado baseado em oposição (OBL) e uma estratégia de ajuste adaptativo dos parâmetros do algoritmo. Em seu trabalho, Wang et al. direcionaram o processamento numérico para a unidade de processamento gráfico (GPU, do inglês *Graph Processing Unit*) do computador, explorando a sua capacidade de execução de cálculos em paralelo. Os resultados obtidos pelos autores, a partir das funções de referência apresentadas em Herrera et al. (2010), indicam que o algoritmo apresentou performance superior à versão canônica da meta-heurística, e que a estratégia de OBL mostrou-se significativa para o resultado. Os autores ainda ressaltam que seus experimentos demonstraram que o aumento populacional implicou em uma melhora significativa na taxa de convergência, fator importante para problemas de alta-dimensionalidade.

Outros autores tratam do problema inerente aos problemas de otimização de alta dimensionalidade no tocante à sua demanda computacional, e as estratégias para a minimização desta. No trabalho de Olguin-Carbajal et al. (2013), os autores empregam uma técnica de não-decomposição baseada em busca local, combinada com uma estratégia para minimizar a população utilizada pela meta-heurística de evolução diferencial, chamada de micro-população (ou μ -população, como por vezes a abordagem é referida na literatura especializada), a qual se baseia na manutenção de um número reduzido de soluções candidatas que são constantemente substituídas por novas soluções aleatoriamente obtidas a partir das restrições para as variáveis de decisão do problema, com o intuito de evitar a convergência prematura da rotina em mínimos locais. Os resultados obtidos por Olguin-Carbajal et al., obtidos a partir de problemas de referência de alta dimensionalidade apresentados em Herrera et al. (2010), indicam que o algoritmo por eles desenvolvido mostrou-se superior ao algoritmo canônico de evolução diferencial. Uma estratégia seme-

lhante àquela adotada por Olguin-Carbajal et al. (2013) é apresentada no trabalho de Huang & Mohan (2006), nos quais os autores apresentam uma meta-heurística baseada em otimização por enxame de partículas chamada de μ PSO, nos quais é mantida uma diminuta população de 5 soluções candidatas, e quando ocorre estagnação destas ou a convergência a um conjunto solução para o qual o valor da função objetivo é pior, emprega-se uma estratégia de repulsão destas utilizando princípios da Lei de Coulomb. Os resultados obtidos apontam para uma performance superior ao algoritmo canônico do PSO. Contudo, convém mencionar que os resultados foram obtidos a partir do estudo de apenas quatro problemas de referência escalados para 1000 dimensões, que não representam em si toda a complexidade inerente aos problemas de otimização de alta dimensionalidade.

No trabalho de LaTorre et al. (2015), os autores apresentam uma extensiva análise de diversos algoritmos (meta-heurísticas e também rotinas determinísticas) utilizados para a resolução de problemas de otimização de alta dimensionalidade, avaliando sua performance frente a problemas de referência coletados em Li et al. (2013), Tang et al. (2010), Herrera et al. (2010). Em sua obra, LaTorre et al. concluem que o algoritmo MOS-CEC2013 apresentou melhor performance em geral para os problemas analisados, entretanto este se mostrou incapaz de determinar o ótimo global para problemas específicos. Os autores ainda enfatizam que a partir dos resultados de sua análise, é possível perceber que os algoritmos formam grupos, nos quais diversos deles apresentam performances semelhantes para problemas em específico. Uma consideração semelhante quanto às meta-heurísticas já é descrita no trabalho de Gandomi et al. (2013), no qual os autores mencionam que em razão de sua concepção residir na heurística, não existe uma técnica que seja universalmente superior, de modo que sua performance está condicionada ao problema a ser avaliado. Uma vez que o algoritmo MOS-CEC2013 consiste em uma composição dinâmica adaptativa de diversas meta-heurísticas que são aplicadas, faz sentido referido como superior em termos de performance no levantamento realizado por LaTorre et al., endossando a ressalva mencionada por Gandomi et al. em seu trabalho.

Convém mencionar que a partir do trabalho de LaTorre et al. (2015), os autores desenvolveram uma importante plataforma on-line para a comparação de resultados obtidos a partir de algoritmos de otimização a partir de diversos problemas de referência, incluindo aqueles que tratam do tema de otimização de alta dimensionalidade, apresentados em Li et al. (2013), Tang et al. (2010), Herrera et al. (2010), Tang et al. (2007), bastando que o usuário formate seus resultados em um arquivo de texto a partir do modelo fornecido na plataforma. O usuário também pode comparar entre si os resultados dos algoritmos apresentados nestes referidos trabalhos. Na Figura 10 é exibida uma imagem da plataforma, referente ao campo para a dis-

ponibilização de resultados obtidos pelo usuário frente a um dos conjuntos de problemas de referência, permitindo que estes sejam comparados facilmente com os resultados apresentados na literatura.

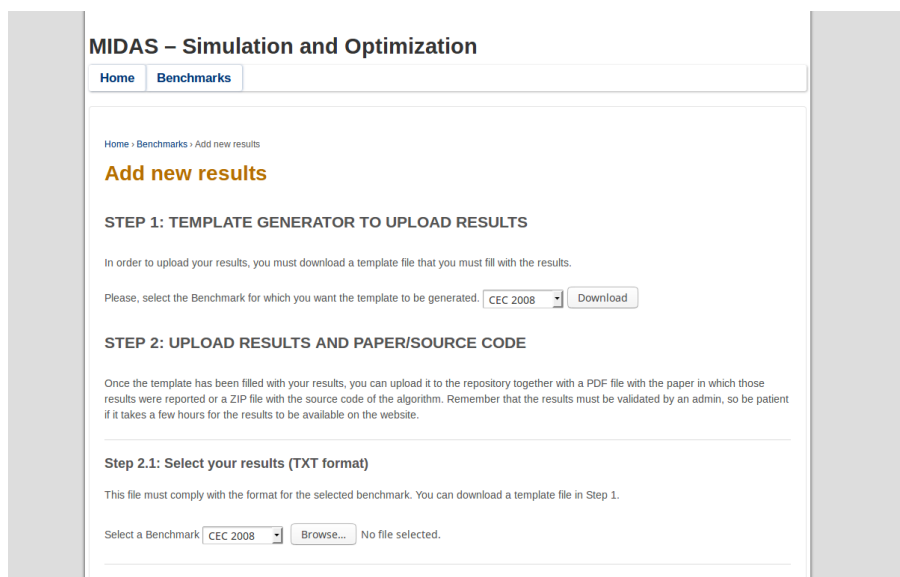


Figura 10 – Imagem da plataforma MIDAS, desenvolvida a partir do trabalho de La-Torre et al. (2015), que permite a comparação de resultados obtidos com algoritmos de otimização a partir de diversos problemas de referência. Disponível em: <<http://vps128.cesvima.upm.es/lab/>>.

Fonte – Próprio autor.

CAPÍTULO 4

Desenvolvimento da ferramenta de simulação
orientada à equações

No presente capítulo será apresentada a metodologia utilizada no desenvolvimento da ferramenta de simulação orientada à equações que compõe o *corpus* do presente trabalho, nomeada de *sloth* (FREITAS, 2019). Inicialmente, um breve referencial teórico acerca da importância da modelagem e simulação no âmbito dos processos. A seguir, será brevemente discutida a metodologia utilizada na concepção da estrutura da ferramenta, bem como os diagramas UML (do inglês *Unified Modeling Language*, ou linguagem de modelagem unificada) referentes a esta.

Convém mencionar que existem diversas ferramentas gratuitas desenvolvidas com o mesmo propósito daquela apresentada neste trabalho, entre as quais podemos destacar os software EMSO (SOARES; SECCHI, 2003), DAETOOLS (NIKOLÍĆ, 2016), ASCEND IV (WESTERBERG et al., 1994), OPEN MODELLICA (FRITZSON et al., 2002), entre outros. Contudo, conforme foi discutido na seção referente aos objetivos pertinentes a este trabalho na seção, a ferramenta aqui apresentada propõe-se a representar uma implementação realizada em linguagem computacional de alto nível (Python), utilizando as bibliotecas de referência para cada um de seus subcomponentes (conforme será discutido na subseção 4.3) que interfaceiam-se com linguagens de mais baixo-nível e que enfatizam a performance, tais como C, C++ e FORTRAN. Assim, o propósito da ferramenta aqui é apresentada reside na concepção de um software gratuito, de código aberto e que possa ser utilizado para a compreensão do conceito de simulação de processos orientada a equações e os procedimentos compreendidos neste intento, especialmente no que tange aos processos de cunho biotecnológico.

4.1 Importância da modelagem e simulação de processos

A modelagem de um processo reside na concepção de um ente – frequentemente de natureza quantitativa – que descreva o comportamento deste, em termos dos fenômenos intrínsecos à sua operação. De acordo com Ingham et al. (2008), os tipos de modelos mais úteis sob o aspecto técnico e científico são aqueles expressos em termos matemáticos. Dentre estes, os chamados modelos dinâmicos, que compreendem os aspectos transientes do processo, são especialmente valiosos na tarefa de prover informações sobre o funcionamento do sistema em estudo. As aplicações dos modelos matemáticos e simulação destes são inúmeras, a exemplo de: pesquisa e desenvolvimento, compreendendo a determinação de parâmetros cinéticos reacionais a partir de dados de laboratório ou de escala piloto, bem como estudos de otimização, controle e aumento de escala; *design* industrial, nos quais podem ser incluídos os estudos acerca dos efeitos do dimensionamento e arranjo dos equipamentos e a performance dinâmica do processo, avaliação da integração

material e energética das várias partes deste, bem como a simulação de cenários de parada, início de operação (*start-up*) e situações emergenciais; operação de plantas industriais, compreendendo o treinamento de operadores e na identificação sistemática de problemas de controle, otimização da operação do processo industrial, bem como a possibilidade de realizar análises exploratórias com segurança por meio de um modelo matemático ao invés de se utilizar a própria instalação industrial para este propósito, dentre outras (LUYBEN, 1989; OGUNNAIKE; RAY, 1994).

De acordo com Ingham et al. (2008), as etapas referentes ao procedimento de modelagem de um processo podem ser identificados, em linhas gerais, conforme a seguir: definição do problema, em termos dos objetivos do estudo; levantamento acerca da compreensão acerca dos fenômenos intrínsecos ao sistema em estudo; formulação do problema em termos matemáticos, e posterior resolução do modelo por simulação; validação das previsões realizadas a partir do modelo frente a resultados conhecidos, e reavaliação das considerações feitas na concepção do mesmo, caso esta mostre-se necessária; por fim, o modelo está pronto para ser utilizado para os propósitos de *design* de processos, controle ou outros propósitos. Na Figura 11, o procedimento pode ser visualizado de maneira esquemática. Esse fluxograma, como todos os demais que figuram no presente capítulo, foram produzidos por meio do *software* livre yED (YWORKS, 2018).

4.2 Acerca da UML

A UML (do inglês *Unified Modeling Language*, ou Linguagem Unificada de Modelagem) é uma família de notações gráficas, apoiada por um meta-modelo único que ajuda na descrição e no projeto de sistemas de *software*, particularmente daqueles construídos utilizando o paradigma de orientação a objetos ou OO (do inglês *Object-Oriented*). Embora tenha sido concebida para a tarefa de desenvolver, compreender, manter e testar um sistema complexo de *software*, a UML tem sido utilizada na modelagem de sistemas de um modo mais amplo e geral (FOWLER; SCOTT, 2007; BOOCH et al., 2006). De acordo com o trabalho de Fowler & Scott (2007), a UML originou-se da unificação de diferentes linguagens gráficas de modelagem de objetos, em meados de 1997. Conforme descrito nos trabalhos de Booch et al. (2006) e Miles & Hamilton (2006), o advento dessa metodologia na representação dos sistemas foi reflexo da efervescência do surgimento de linguagens computacionais adotando o paradigma OO, ocorrido à época.

Convém mencionar que um modelo é, essencialmente, uma abstração da entidade real a ser modelada. Assim, abre-se mão dos detalhes que possam mostrar-se irrelevantes ou potencialmente confusos acerca do sistema em estudo, em favor

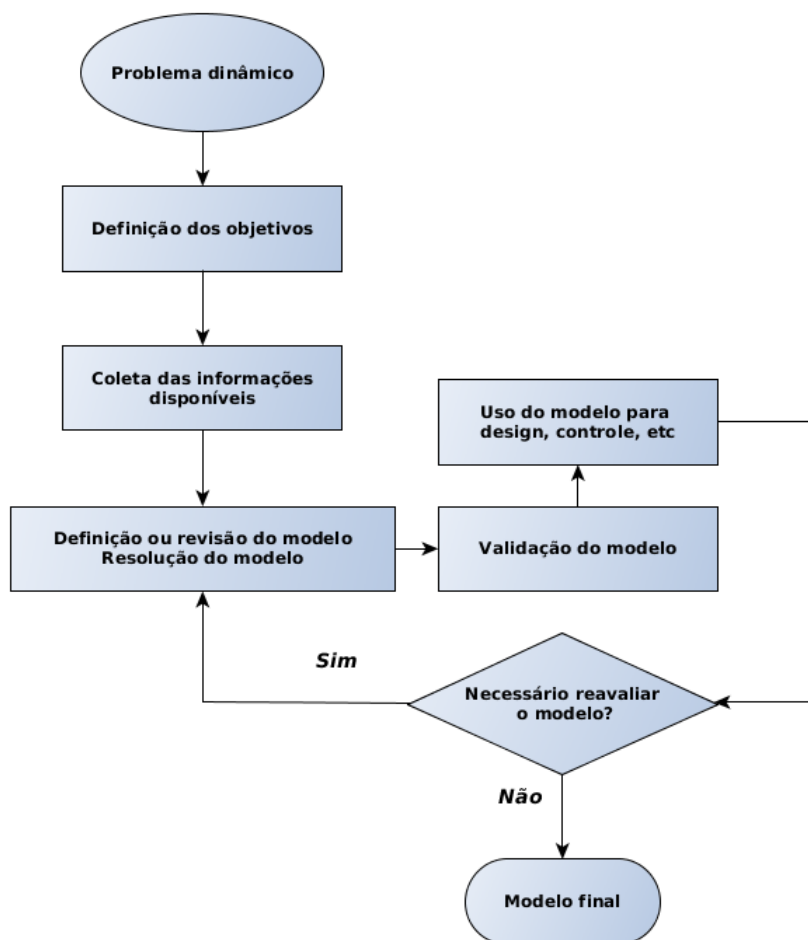


Figura 11 – Representação esquemática do procedimento da modelagem de um processo.

Fonte – Adaptado de Ingham et al. (2008).

da possibilidade de compreender a multitude de processos interentes ao mesmo, bem como a capacidade de interpretação, execução e avaliação deste. Nas versões mais modernas da UML, a atenção é voltada para a definição de aspectos de notação e formalização da definição dos meta-modelos, de modo a permitir que haja o compartilhamento máquina-a-máquina dos modelos desenvolvidos (MILES; HAMILTON, 2006).

O conceito central da UML reside justamente na definição de um modelo abstrato para a caracterização do sistema em estudo, o que no caso apresentado no presente trabalho corresponde à ferramenta *sloth*. Esse modelo pode ser utilizado no processo de levantamento de requisitos do *software* e na construção da documentação do mesmo, processos que são intrinsecamente contínuos (ou dinâmicos, na terminologia proposta pelos autores) em razão da habitual tendência de que os sistemas tornem-se mais complexos a medida que novas funções mostram-se necessárias ao mesmo.

A associação dos elementos preconizados pela UML permite que sejam representados diferentes tipos de diagramas, os quais podem ser classificados em duas amplas categorias: diagramas estruturais, os quais mostram a estrutura estática do sistema e suas partes em diferentes níveis de abstração, bem como a inter-relação entre as mesmas; diagramas comportamentais, os quais voltam-se à natureza dinâmica dos objetos do sistema, ou em outras palavras, para a representação das mudanças ocorridas no sistema ao longo do tempo (MILES; HAMILTON, 2006; FOWLER; SCOTT, 2007). A definição de todos os tipos de diagramas contidos nessas duas grandes categorias está além do escopo deste texto. Dentre os diversos tipos de diagramas preconizados pela UML, dois tipos de diagramas mostram-se mais úteis ao propósito deste trabalho dentre os conjuntos dos diagramas estruturais e comportamentais, respectivamente: o diagrama de classes, e o diagrama de atividade. Essas ferramentas serão utilizadas na representação das estruturas que compõem o *software* apresentado neste trabalho, constantes nas subseções 4.3.2 e 4.3.2. A descrição minuciosa das regras de nomenclatura para as mesmas dentro dos padrões UML pode ser vista no trabalho de Guedes (2011).

4.3 Metodologia utilizada no desenvolvimento da ferramenta

Esta seção intenta apresentar a metodologia utilizada no desenvolvimento da ferramenta de simulação orientada a equações. Para tanto, serão discutidos brevemente os aspectos teóricos referentes ao paradigma de descrição de processos UML, o qual será utilizado na representação diagramática dos mecanismos computacionais utilizados no desenvolvimento e operação da ferramenta. A estrutura conceitual da ferramenta será apresentada, e a inter-operação entre os diferentes módulos que a compõem será discutida. A seguir, serão apresentados os diagramas de classe e de atividades referentes a ferramenta, de modo a apresentar respectivamente a estrutura computacional utilizada na formulação desta, bem como o fluxo de informações no *software* durante os processos típicos realizados pelo usuário.

4.3.1 Estrutura conceitual da ferramenta

Na presente subseção, a estrutura da ferramenta desenvolvida será apresentada, dando ênfase aos princípios de inter-conectividade entre os diferentes módulos que compõem a mesma. Convém mencionar que a minuciosa descrição do código-fonte da ferramenta está além do escopo desta seção, e também não será realizada no presente trabalho, considerando que o *software* é disponibilizado gratuitamente no repositório da ferramenta na plataforma *Github*. A seguir, serão discutidos os componentes da ferramenta sob o ponto de vista de *software*, os quais

serão referidos como *módulos* a fim de proporcionar maior legibilidade do texto, sem contudo ater-se ao significado deste sob o aspecto computacional.

Além da estrutura computacional que será mostrada no presente capítulo, referente a ferramenta em si, convém reiterar que a ferramenta `sloth` utiliza-se de diversas bibliotecas computacionais integrantes do ecossistema da linguagem *Python* que tratam dos requisitos específicos para o *software*, os quais são enumerados a seguir, em conjunto com as devidas referências:

- Computação simbólica: biblioteca `sympy` (MEURER et al., 2017)
- Resolução de sistemas lineares e não-lineares: bibliotecas `sympy` e `scipy` (MEURER et al., 2017; JONES et al., 2001)
- Cálculo numérico em geral, operação em baixo-nível das operações matemáticas: biblioteca `numpy` (WALT et al., 2011)
- Resolução de sistemas diferenciais: bibliotecas `assimulo` e `scipy` (ANDERSON et al., 2015; JONES et al., 2001)
- Resolução de sistemas algébrico-diferenciais: biblioteca `assimulo` (ANDERSON et al., 2015)
- Otimização: biblioteca `pagmo` (BISCANI et al., 2019)
- Produção de gráficos: biblioteca `matplotlib` (HUNTER, 2007)

4.3.1.1 Perspectiva geral

Sob a perspectiva do usuário, os principais módulos da ferramenta correspondem àqueles voltados às seguintes tarefas: criação dos modelos (módulo *model*); criação dos casos de estudo (ou problemas) que contém esses modelos (módulo *problem*); criação das simulações que são realizadas a partir de problemas (*simulation*); as rotinas utilizadas para a resolução dos sistemas de equação oriundas das simulações (módulo *solvers*); o módulo *optimization*, utilizado no estudos de otimização. Os problemas são formados a partir de um modelo ou da conjunção de vários, dando origem a um sistema de equações a ser resolvido. Este pode ser de natureza linear, não-linear, diferencial ou algébrico diferencial. A diferença entre as classes de equações a serem inseridas nos modelos utilizados na ferramenta `sloth` pode ser visualizada na Tabela 2. Por fim, o módulo de operações unitárias (*unit_op*) também integra o rol de componentes do *software* imediatamente acessível ao usuário, no qual estão definidos diferentes tipos de operações unitárias de uso comum em processos industriais tais como tanques, bombas, reatores, entre outros.

Tipo de equação	Referência no <i>software</i>	Forma canônica
Linear	<i>linear</i>	$x + y + z - 5 = 0$ $2x - 0,7y + 2z = 0$ $y + z - 2 = 0$
Não-linear	<i>nonlinear</i>	$x^2 + y^2 + z - 5 = 0$ $2x - 0,7y + 2z = 0$ $y^2 + z - 2 = 0$
Diferencial	<i>differential</i>	$\frac{dx}{dt} = 3,5x - y$ $\frac{dy}{dt} = \text{sen}(y) + x$
Algébrico-diferencial	<i>algebraic-differential</i>	$\frac{dx}{dt} = x + y - \log(2.3)$ $x + 0,7y = 0$ $\frac{dy}{dt} = \text{sen}(y) + x$

Tabela 2 – Diferentes tipos de equações que podem figurar nos modelos utilizados na ferramenta *sloth*, em termos do tipo de equação, a forma por meio da qual são referidas no *software* e a sua expressão matemática canônica

Fonte – Próprio autor.

Na Figura 12, a relação entre os módulos que compõem a parte do *software* voltada a interação direta com o usuário pode ser visualizada esquematicamente. O módulo *model* permite que o usuário defina os modelos a serem resolvidos, e o módulo *unit_op* conta com uma biblioteca de modelos pré-determinados correspondentes a diversas operações unitárias; o módulo *problem* define um caso de estudo a partir de um ou mais modelos interconectados por meio de suas equações; em seguida, o caso de estudo definido irá compor uma simulação por meio do módulo *simulation*, o qual se utilizará do módulo *solver* para resolver o sistema de equações correspondente; o módulo *optimization* pode ser utilizado para conduzir estudos de otimização, a partir de um caso de estudo com um número de graus de liberdade adequado. Convém mencionar que a minuciosa descrição do fluxo lógico de operação da ferramenta está além do escopo da presente subseção do texto, e conforme já foi mencionado, será descrito apropriadamente na subseção 4.3.3 a seguir. Convém mencionar que na referida imagem, os módulos destacados com uma linha tracejada – respectivamente *simulation* e *solvers* – são utilizados de maneira conjunta, requerendo do usuário apenas a configuração dos parâmetros da simulação.

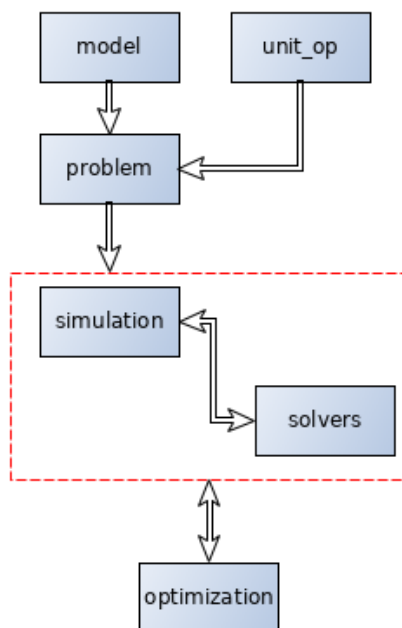


Figura 12 – Representação esquemática da relação entre os diferentes componentes da ferramenta que são voltados a interação direta com o usuário.

Fonte – Próprio autor.

A definição e operação entre as equações definidas para os modelos utiliza-se de um arcabouço conceitual definido na ferramenta por meio dos seguintes aspectos: definição dos modelos em termos de variáveis (para as quais um valor é determinado), parâmetros (valor previamente especificado ou a determinar, no caso de otimizações) e constantes (valor previamente especificado), respectivamente provida pelos módulos *variable*, *parameter* e *constant* (respectivamente, objetos do tipo *Variable*, *Parameter* e *Constant*). Todos essas grandezas são derivados dos objetos do tipo quantidade (ou *Quantity*), a cargo do módulo *quantity*, por meio do qual podem ser atribuídas as dimensões das quantidades em termos das unidades do SI (sistema internacional de unidades); processamento de grandezas em termos integrantes das equações por meio do módulo de avaliação de expressões (*evaluation_expression*), obtendo objetos do tipo integrantes de equações (ou *Equation-Node*); processamento das equações que definem um modelo a partir de computação simbólica, por meio do módulo *equation*, correspondendo a um grupo de objetos do tipo *Equation*; formação de um bloco de equações, correspondendo a um objeto do tipo *EquationBlock*; coerência dimensional nas operações, garantida pelo fato de que os dois lados das equações devem ser dimensionalmente equivalentes, a cargo do módulo de unidades (*unit*); disponibilização das unidades mais usuais na forma de objetos (tipo *Unit*); tratamento de erros por meio do módulo (*error_definitions*). Na Figura 13, a relação entre os referidos módulos pode ser visualizada esquematicamente. Na referida figura, os módulos destacados com uma linha contínua

indicam uma relação de derivação entre os módulos *quantity* e os derivados *variable*, *parameter* e *constant*. Aqueles destacados por meio de uma linha tracejada – respectivamente *equation* e *equation_block* – são utilizados de maneira conjunta, requerendo do usuário apenas a entrada das equações no modelo, para que seja gerado o objeto *EquationBlock*.

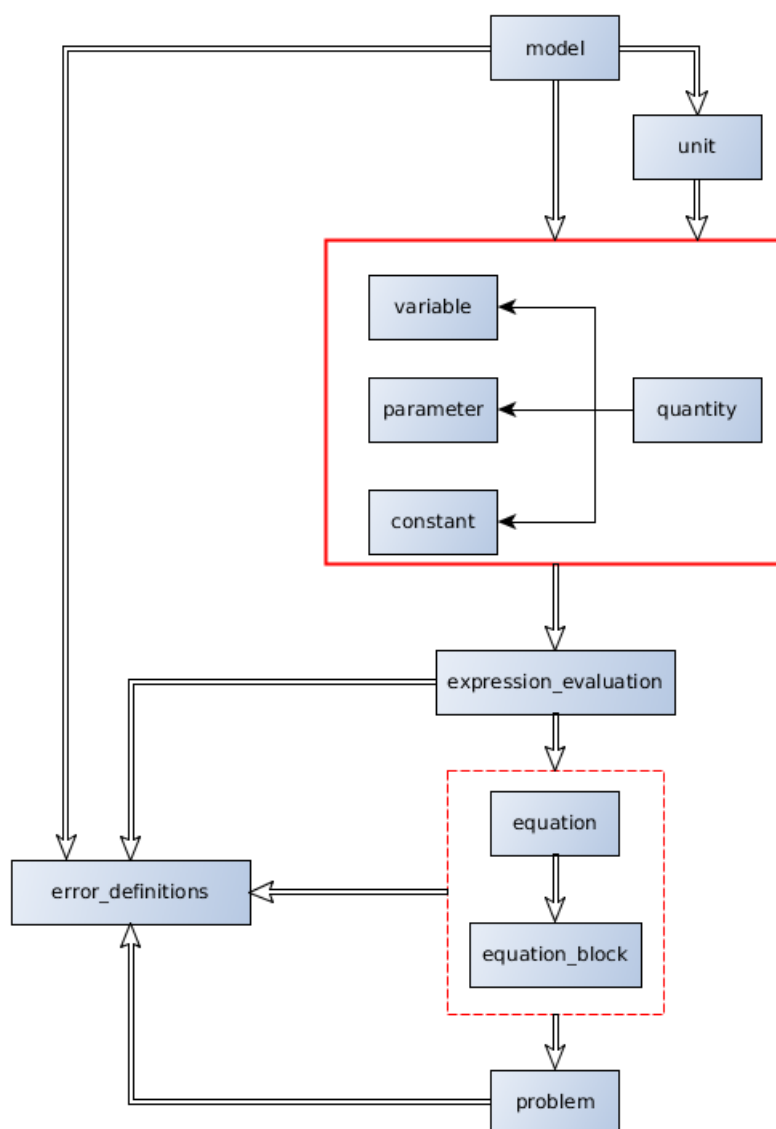


Figura 13 – Representação esquemática da relação entre os diferentes componentes da ferramenta que compõem o núcleo interno da mesma.

Fonte – Próprio autor.

As operações entre os grupos dimensionais utilizados na declaração das equações sob forma de quantidades (variáveis, parâmetros ou constantes, correspondendo respectivamente a objetos do tipo *Variable*, *Parameter* e *Constant*), que representam rotinas basilares ao funcionamento da ferramenta, podem ser descritos conforme apresentado em Soares (2003): caso uma operação de adição ou subtração de duas quantidades ocorra, avalia-se se ambas apresentam coerência

dimensional, isto é, se suas dimensões são equivalentes; em caso positivo, o valor numérico das quantidades é calculado a partir da operação, dando origem a uma nova quantidade; caso negativo, um erro é retornado ao usuário. Nas operações de multiplicação, divisão, potenciação e radiciação essa checagem não é necessária, bastando que as dimensões da quantidade resultante sejam calculadas adequadamente.

Convém mencionar que para processar as operações entre quantidades de natureza distintas (variáveis e parâmetros ou constantes, por exemplo), emprega-se a computação simbólica para representar a soma destas quantidades, de modo que a operação entre duas variáveis (ou entre uma variável e um parâmetro não especificado) produzirá uma expressão simbólica consistindo na soma destas, ao passo que quando somadas a um parâmetro especificado (ou constante), simbolicamente o segundo termo é considerado como um número. Na Figura 14, esse procedimento é representado esquematicamente.



Figura 14 – Representação esquemática do processo de realização de operações aritméticas na ferramenta (no exemplo, a adição) a partir de duas variáveis diferentes (a), duas variáveis equivalentes (b), uma variável e um parâmetro especificado ou constante (c) e entre uma variável e um parâmetro não especificado (d), com o resultado simbólico em destaque.

Fonte – Próprio autor.

4.3.1.2 Definição de variáveis, parâmetros e constantes

As variáveis, parâmetros e constantes representam os diferentes tipos de entidades matemáticas que podem ser utilizadas na declaração as equações (na

ferramenta, um objeto do tipo *Equation*) que compõem um modelo (na ferramenta, um objeto do tipo *Model*). A diferenciação entre essas classes de entidades é muito importante para a adequada resolução do sistema de equações que constituirão o problema de estudo, conforme será visto em seções posteriores. As principais características de cada uma destas categorias pode ser vista a seguir:

- **Variáveis:** Devem ter seu valor determinador mediante a resolução das equações que compõem o sistema, contribuindo portanto para o número de graus de liberdade do modelo. São declaradas no módulo *variable* da ferramenta, dando origem a objetos do tipo *Variable*.
- **Parâmetros:** Podem ter o seu valor determinado (parâmetro especificado) ou não (parâmetro não-especificado), deste modo respectivamente acrescentando ou não o número de graus de liberdade do modelo. Estudos de otimização requerem pelo menos um parâmetro não especificado a ser tratado como variável manipulada. São declarados no módulo *parameter* da ferramenta, dando origem a objetos do tipo *Parameter*.
- **Constantes:** Possuem um valor determinado no momento de sua definição. São declarados no módulo *constant*, dando origem a objetos do tipo *Constant*.

Essas três classes são chamadas (no âmbito da descrição conceitual da ferramenta) de quantidades, derivando dos objetos *Quantity*, definidos no módulo *quantity*. Todos os objetos do tipo *Quantity* (e seus derivados) apresentam valor (definido ou não) e dimensões. Este último atributo representa um importante aspecto para a formulação das equações, uma vez que a coerência dimensional é um dos pilares da ferramenta na composição destas. Outro aspecto importante destes objetos é a possibilidade de conversão para termos simbólicos na composição de equações, gerando os objetos do tipo *EquationNode* que irão formar as equações, conforme será discutido a seguir.

4.3.1.3 Inserção e resolução de equações

A inserção de equações que compõem os modelos representa um dos principais aspectos da ferramenta. Essas constituem um tipo de objeto específico (tipo *Equation*), que é definido por meio da conversão das quantidades constituintes das equações em objetos do tipo *EquationNode*, por meio de rotina interna dos objetos *Quantity*, conforme pode ser visto esquematicamente na Figura 15. Essa rotina é definida a partir da sobrecarga do operador `__call__()`, um recurso da linguagem computacional (*operator overloading*) que propicia uma sintaxe clara e simplificada para declaração das equações constituintes dos modelos em questão.

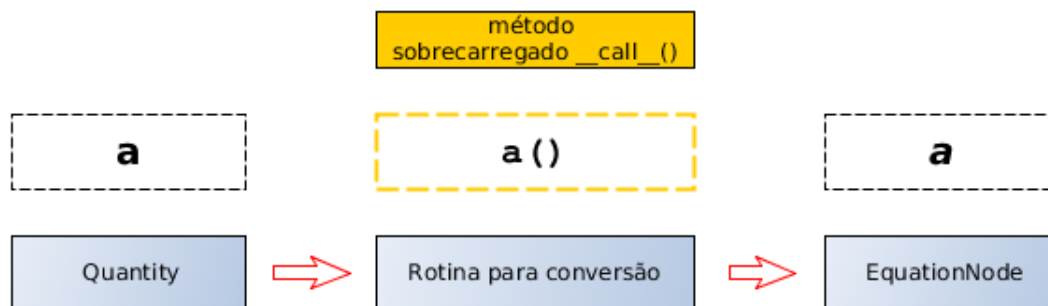


Figura 15 – Representação esquemática do processo de conversão entre uma quantidade (objeto *Quantity*) e um termo simbólico equacional (objeto *EquationNode*), por meio da rotina interna específica a esse propósito.

Fonte – Próprio autor.

A coerência dimensional é um aspecto muito importante na ferramenta, haja vista que esta propriedade é fundamental para garantir a coesão matemática dos termos que compõem as equações. Conforme já foi descrito, para que ocorra uma operação de adição ou subtração de duas quantidades, avalia-se se suas dimensões são equivalentes; em caso positivo, o valor numérico das quantidades é calculado a partir da operação, dando origem a uma nova quantidade; caso negativo, um erro é retornado ao usuário. Nas operações de multiplicação, divisão, potenciação e radiciação essa checagem não é necessária, bastando que as dimensões da quantidade resultante sejam calculadas adequadamente. Convém mencionar ainda que as operações transcendentais (exponenciação, logaritmos, etc) partem do princípio que o operando é adimensional, retornando um erro ao usuário caso essa premissa não seja observada.

4.3.1.4 Definição de modelos, e composição de um problema de estudo

Os modelos podem ser definidos por meio de objetos do tipo *Model*, mediante a declaração das equações que os compõem. No contexto da ferramenta *sloth*, um modelo pode ser definido como um conjunto de equações que formam um sistema bem-posto. Quanto ao número de graus de liberdade, para estudos de otimização, é necessário que existam graus de liberdade positivos, os quais são utilizados como variáveis manipuladas pela rotina de otimização, conforme será descrito a seguir. Do contrário, em se tratando de casos de estudo de simulação dos modelos desenvolvidos, o número de graus de liberdade deve ser nulo.

A conexão entre diferentes modelos – por exemplo, a corrente de saída de uma operação unitária é a entrada de outro processo – pode ser realizada por meio dos objetos do tipo *Connection*, derivados de *Equation*. Assim, cria-se uma equação

de conexão no modelo cuja informação é recebida (ou no exemplo anteriormente mencionado, aquele que recebe a corrente oriunda da operação unitária) de modo a acoplar as variáveis entre os dois modelos. Na Figura 16, a conexão entre dois modelos é representada de maneira esquemática, obtendo-se um conjunto final de equações a ser resolvido, que na ferramenta é tratado como um objeto do tipo *EquationBlock*. Esse último aglutina todas as equações a serem resolvidas pela ferramenta posteriormente.

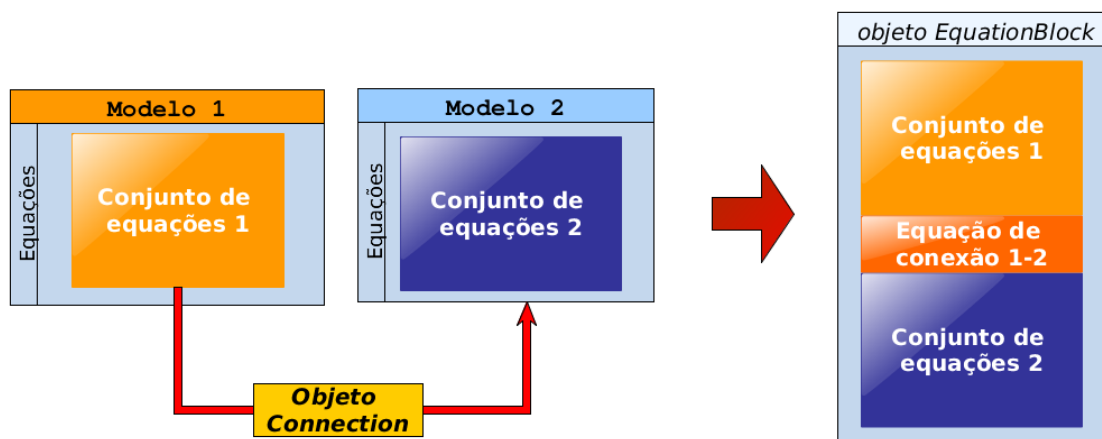


Figura 16 – Representação esquemática do processo de conexão entre dois modelos (objetos *Model*), por meio de um objeto *Connection*, destacando o conjunto final de equações obtido, em um objeto do tipo *EquationBlock*.

Fonte – Próprio autor.

Uma vez que os modelos definidos pelo usuário já estejam estabelecidos, um caso de estudo (ou problema) deve ser criado. Para tanto, o usuário cria um objeto do tipo *Problem*, e declara os modelos integrantes do mesmo. A conexão entre os modelos ocorre por meio de uma rotina específica dessa referida classe de objetos, e implicitamente uma equação de acoplamento é criada entre os dois modelos por meio das variáveis envolvidas nesta conexão (por exemplo, uma vazão mássica de saída de uma primeira operação unitária, e uma de saída na segunda).

4.3.1.5 Simulação

As simulações ocorrem a por meio de objetos do tipo *Simulation*, definidos a partir de objetos do tipo *Problem*. No contexto da ferramenta, a definição da simulação é a etapa na qual serão definidas as condições iniciais necessárias à resolução do sistema de equações a ser resolvido, bem como configurações adicionais. Conforme já foi mencionado, uma vez definidas as equações constituintes dos modelos em tela, e sendo definido um caso de estudo, obtém-se um conjunto de equações (objeto *EquationBlock*). A partir desse conjunto de equações, ocorre uma detecção

automática do tipo de rotina de resolução necessária do mesmo, em termos das classes já definidas na Tabela 2.

Após a simulação, a depender da natureza do sistema de equações, serão obtidos os valores estáticos da resposta para as variáveis (problemas estacionários) ou um perfil temporal de resposta para as variáveis (problemas dinâmicos). É possível representar essa resposta graficamente por meio do módulo *plotter*, que permite que sejam obtidos gráficos a partir dos resultados obtidos pela simulação.

4.3.1.6 Otimização

Com vistas a fornecer o subsídio para o outro aspecto que integra o cerne do presente trabalho, o de otimização de problemas biotecnológicos, foi concebido um componente dedicado à realização de estudos de otimização na ferramenta *sloth*, o módulo *optimization*. Por meio desse, é gerado um objeto do tipo *Optimization* ligado a uma simulação de um caso de estudo com um número de graus de liberdade positivo, os quais são utilizados como variáveis manipuladas para a otimização.

O processo de otimização na ferramenta consiste na avaliação de uma função objetivo – definida na etapa de criação do objeto *Optimization* – em termos da especificação das variáveis manipuladas no valor definido pelo algoritmo, realizando a simulação do problema (então, bem-posto) e obtendo um resultado. Esse ciclo é repetido a depender do algoritmo de otimização empregado, sendo determinados os resultados pertinentes ao problema. Esse procedimento é representado esquematicamente na Figura 17. Após a definição do conjunto dos modelos que irão compor o caso de estudo, são avaliados os valores das variáveis manipuladas e o resultado na função objetivo, até que seja atingido o critério de parada do algoritmo de otimização.

4.3.2 Diagramas de classe

Na presente seção, serão apresentados os diagramas de classe referentes a ferramenta *sloth*. Conforme descrito em Soares (2003), uma documentação completa acerca do tema, bem como demais aspectos da UML podem ser encontrados em OMG (2000). Os diagramas aqui apresentados foram obtidos diretamente a partir do código-fonte da ferramenta, a qual foi desenvolvida em linguagem computacional *Python*. Essa escolha dá-se em razão da simplicidade na compreensão da sintaxe que é característica da linguagem, o que tornaria os diagramas muito semelhantes àqueles que seriam obtidos caso optássemos pelo português estruturado. A seguir, são discutidos os componentes integrantes da ferramenta, em termos dos pacotes computacionais e os seus respectivos diagramas de classe.

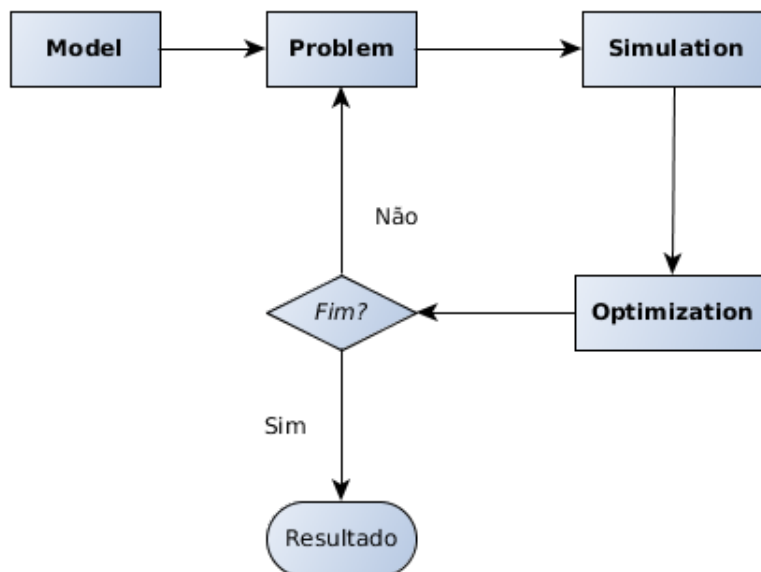


Figura 17 – Representação esquemática do processo de otimização na ferramenta.

Fonte – Próprio autor.

4.3.2.1 Pacote *core*

Nesse pacote estão contidos os principais conceitos envolvendo a modelagem dos sistemas na ferramenta *sloth*, desde a definição de variáveis a serem resolvidas, parâmetros e constantes, até a própria definição dos termos simbólicos que irão compor as equações dos modelos. Na Figura 18, a relação de interdependência entre os módulos que compõem o pacote *core* é representada. Essa figura, assim como as demais mostradas na seção 4.3.2 forma produzidas a partir do próprio código fonte da ferramenta, utilizando a biblioteca *pyreverse*, em conjunto com o software livre *yEd*.

A seguir, serão descritos os subpacotes que integram o pacote principal, em termos de seus respectivos diagramas de classes.

Subpacote *core.unit*

No subpacote *core.unit* está definida a classe *Unit*, que é responsável pela definição da estrutura de coerência dimensional dos objetos de quantidade (*Variable*, *Parameter* e *Constant*). Esse princípio é muito importante para a composição dos modelos a serem estudados na ferramenta, uma vez que a coesão dimensional é premissa na definição de uma equação matemática, conforme descrito na seção 4.3.1.3. O diagrama de classe do subpacote *core.unit* pode ser visto na Figura 19.

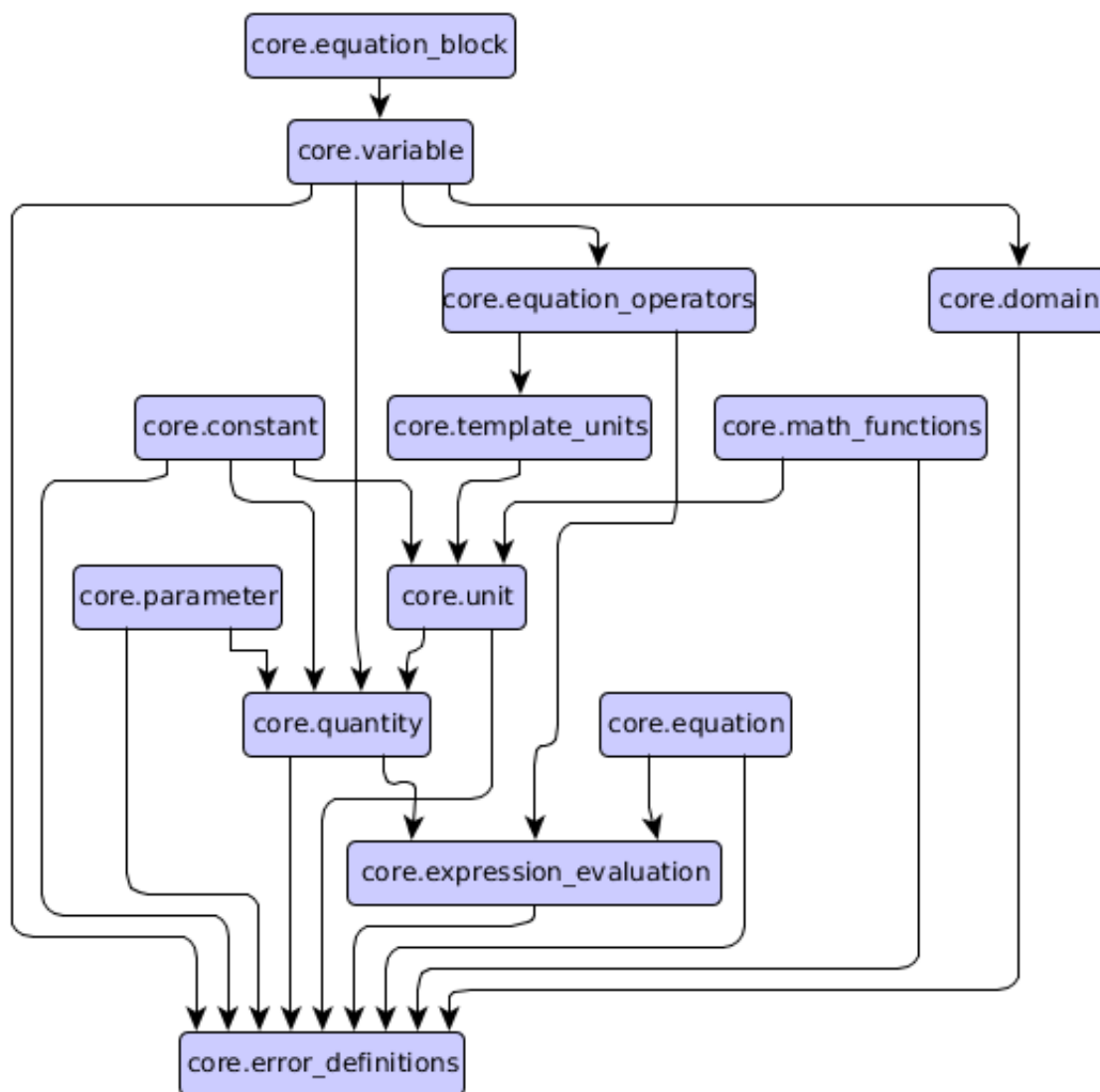


Figura 18 – Representação da relação de interdependência dos módulos que compõem o pacote *core*.

Fonte – Próprio autor.

Subpacote *core.quantity*

No subpacote *core.quantity* está definida a classe *Quantity*, responsável pela definição de objetos portadores de um valor quantitativo. Essa, figura como a classe base para as classes *Variable*, *Parameter* e *Constant*, e seus respectivos objetos. O diagrama de classe do subpacote em tela pode ser visto na Figura 20.

Subpacotes *core.variable*, *core.parameter* e *core.constant*

Nos subpacotes *core.variable*, *core.parameter* e *core.constant*, são definidos respectivamente as classes *Variable*, *Parameter* e *Constant*, dando origem aos ob-

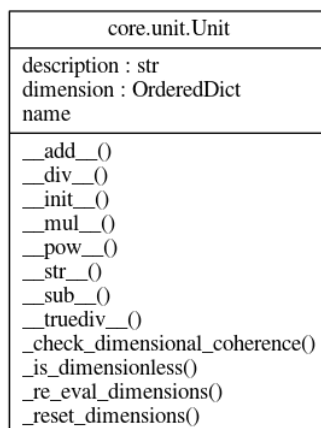


Figura 19 – Diagrama de classes do subpacote *unit*.

Fonte – Próprio autor.

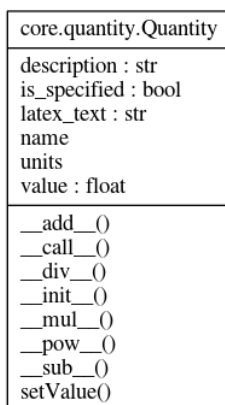


Figura 20 – Diagrama de classes do subpacote *quantity*.

Fonte – Próprio autor.

jetos de mesmo nome, derivados da classe *Quantity*. Estas três classes de objetos representam os componentes das equações, que por sua vez irão compor os modelos definidos na ferramenta. Seus diagramas de classe podem ser visualizados na Figura 21.

Os objetos *Parameter* podem ser especificados em termos de seu valor, o que reduz o número de graus de liberdade do modelo, ou não serem especificados, prática que só é utilizada em estudos de otimização por meio da classe *Optimization*, descrita posteriormente na seção 4.3.2.5; os objetos do tipo *Constant* devem ter seu valor definido. Por fim, objetos do tipo *Variable* não devem ter seu valor definido, uma vez que devem ser determinados pelas rotinas internas da ferramenta de resolução das equações. Convém notar a ausência do método para definição de valores (`setValue`) no diagrama de classes.

As três categorias de objetos referidas nessa seção apresentam uma rotina para a sua conversão em objetos simbólicos, os quais são processados para compor

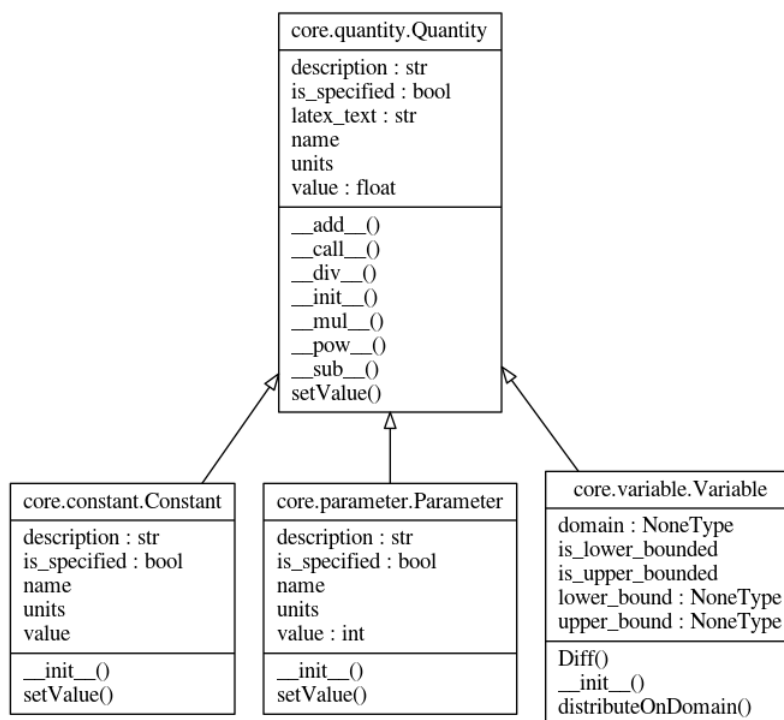


Figura 21 – Diagrama de classes do subpacotes *variable*, *parameter* e *constant*, derivados da classe *Quantity*.

Fonte – Próprio autor.

as equações, conforme descrito na seção 4.3.1.3. Essa rotina corresponde a um método sobrecarregado, que utiliza os recursos da biblioteca de cálculo simbólico *sympy* (MEURER et al., 2017).

Subpacote *core.domain*

O subpacote *core.domain* define a classe *Domain*, por meio da qual são estruturados os domínios matemáticos em que objetos *Variable* podem ser distribuídos. Além de permitirem que sejam calculados os operadores diferenciais para objetos nele distribuídos, um objeto *Domain* armazena as informações de resposta desse, em função de uma determinada variável independente, a exemplo do perfil temporal (o tempo figurando como variável independente) de um objeto *Variable* referente à concentração de um dado reagente. O diagrama da classe *Domain* pode ser visualizado na Figura 22.

Subpacote *core.equation*

O subpacote *core.equation* define a classe *Equation*, por meio da qual estruturam-se as equações que compõem os modelos declarados, formando o sistema a ser

core.domain.Domain
dependent_objs : OrderedDict description : str independent_vars : dict is_set : bool lower_bound : dict name units upper_bound : dict values : NoneType
__call__() __getitem__() __init__() _createDataFramePrototype() _distributeOnDomain() _register() _renameHeaders() _reset() _setDomain()

Figura 22 – Diagrama da classe *Domain*, definida no subpacote *core.domain*.

Fonte – Próprio autor.

resolvido pelas rotinas internas da ferramenta. Os objetos *Equation* são definidos por meio das operações entre os objetos simbólicos, resultando em uma expressão computacional a ser avaliada, cujo tipo será inferido automaticamente dentre aqueles já apresentados na Tabela 2. A classe *Connection* deriva de *Equation*, e encarrega-se de estabelecer a ligação entre as variáveis definidas para dois modelos distintos, por meio de uma equação de conexão, conforme descrito na seção 4.3.1.4, e representado esquematicamente por meio da Figura 16. O diagrama de classes daquelas definidas no subpacote *core.equation* pode ser visualizado na Figura 23.

Subpacote *core.equation_block*

A classe *EquationBlock*, definida neste subpacote, é responsável pelo armazenamento do conjunto de equações (sob a forma de objetos do tipo *Equation*) a serem resolvidas pelas rotinas internas da ferramenta. Na referida classe, também são definidas as rotinas para a detecção automática do tipo de sistema de equações, implicando na escolha da rotina a ser utilizada em sua resolução. O diagrama de classes daquelas definidas no subpacote pode ser visualizado na Figura 24.

Subpacote *core.expression_evaluation*

A classe *EquationNode*, definida neste subpacote, é responsável pela conversão dos objetos quantitativos (*Variable*, *Parameter*, *Constant*) em objetos simbólicos, conforme descrito na seção 4.3.1.3, e representado esquematicamente nas Figuras 14 e 15. O diagrama de classe referente ao subpacote pode ser visualizado na Figura 25.

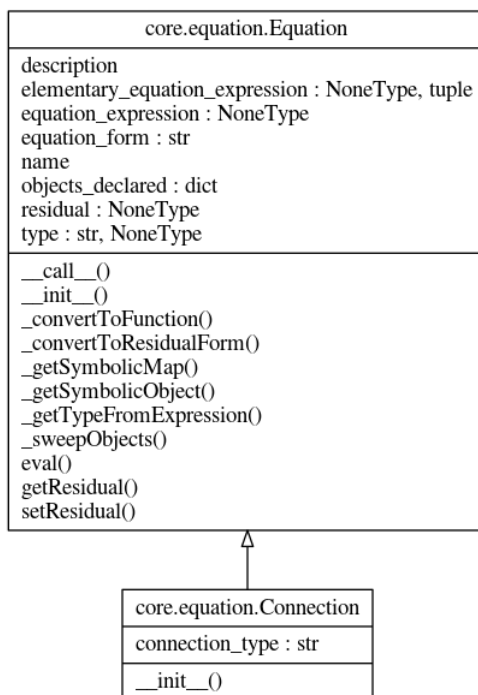


Figura 23 – Diagrama das classes definidas no subpacote *core.equation*, correspondendo as classes *Equation* e *Connection* (a qual deriva da primeira).

Fonte – Próprio autor.

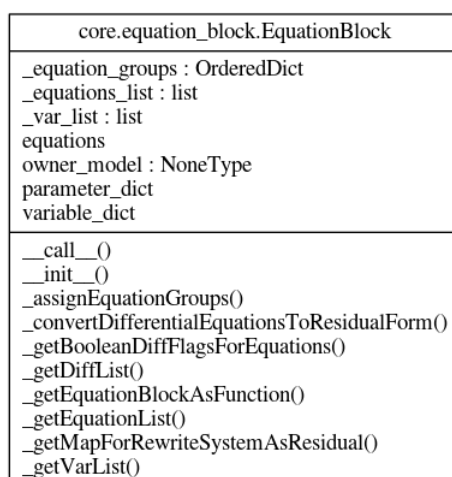


Figura 24 – Diagrama das classes definidas no subpacote *core.equation_block*.

Fonte – Próprio autor.

core.expression_evaluation.EquationNode
args : list equation_type : dict latex_text : str name : str repr_symbolic : NoneType symbolic_map : dict symbolic_object : NoneType unit_object : NoneType variable_map : dict
__add__() __div__() __eq__() __init__() __mul__() __pow__() __radd__() __rdiv__() __repr__() __rmul__() __rsub__() __rtruediv__() __str__() __sub__() __truediv__() _checkEquationTypePrecedence()

Figura 25 – Diagrama das classes definidas no subpacote *core.expression_evaluation*.

Fonte – Próprio autor.

Subpacote *core.error_definitions*

Este subpacote define as classes referentes a captura de exceções (erros) que ocorrem durante as operações com a ferramenta *sloth*. Essas classes retornam mensagens elucidativas para o usuário, bem como fornecem estruturas para que este realize um tratamento adequado durante uma possível extensão dos códigos da ferramenta. Nesse sentido, esse subpacote exibe um papel importante, uma vez que todas as mensagens de erro referem-se as classes nele definidas, conforme pode ser visto por meio do diagrama de inter-dependência do pacote *core*, na Figura 18. O diagrama de classes definidas no subpacote *core.error_definitions* pode ser visualizado na Figura 26.

4.3.2.2 Pacote *model*

Nesse pacote está contida a classe *Model*, responsável pela criação de objetos do tipo de mesmo nome na ferramenta, os quais representam os modelos utilizados na descrição do processo em estudo. Nesses modelos são inseridos os objetos *Variable*, *Parameter* e *Constant*, especificam-se os valores dos argumentos pertinentes, bem como são definidas as equações (objetos *Equation*). Todos esses processos são realizados por meio de rotinas próprias da classe *Model*. Convém

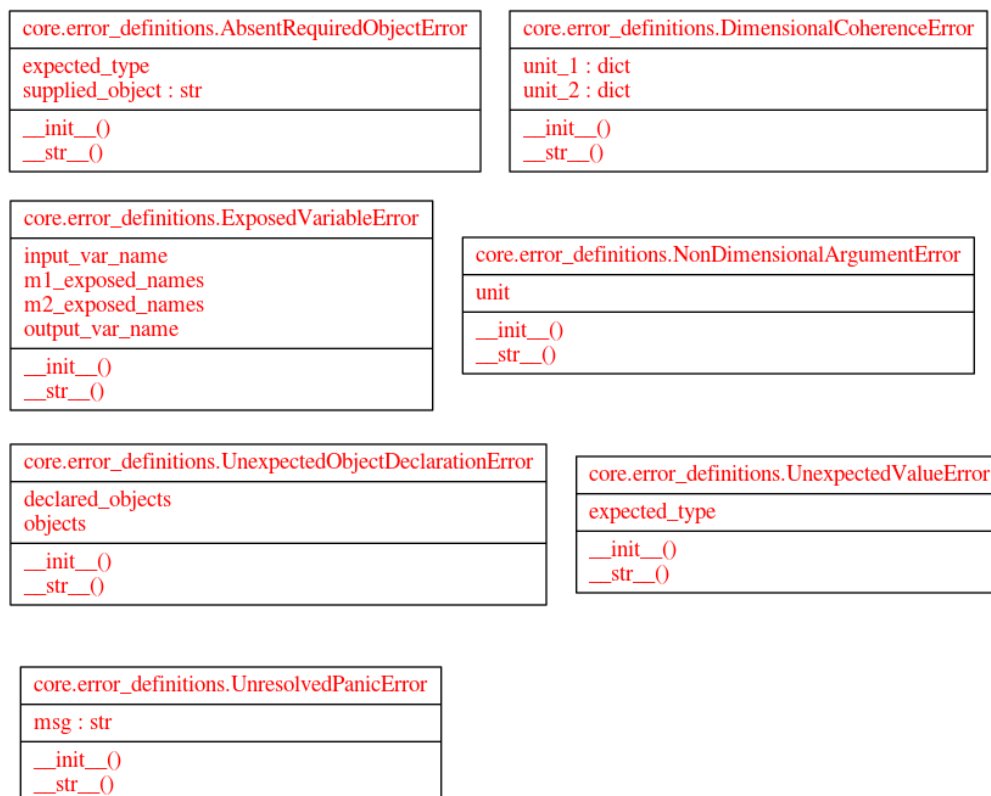


Figura 26 – Diagrama das classes definidas no subpacote `core.error_definitions`.

Fonte – Próprio autor.

mencionar que os objetos do tipo *Variable* podem ser declarados como *exposed*, informando à ferramenta que estes representam a entrada ou saída de outro modelo.

Por meio do método sobrecarregado `__call__`, o usuário pode consolidar apropriadamente o modelo declarado em um objeto do tipo *Model*. Também são definidas rotinas para a exibição de informações acerca das variáveis que são declaradas como expostas (ou *exposed*) – conforme descrito no parágrafo anterior – bem como a quantificação do número de graus de liberdade do modelo. O diagrama de classe em questão pode ser visualizado na Figura 27.

4.3.2.3 Pacote *problem*

Nesse pacote está contida a classe *Problem*, responsável pela criação de objetos do tipo de mesmo nome no contexto da ferramenta `sloth`. Esses representam os casos de estudo (ou problemas) a serem trabalhados pelo usuário, constituídos por um ou mais objetos *Model*. A classe *Problem* também define uma rotina por meio da qual ocorre a conexão entre as variáveis declaradas em objetos *Model*, conforme discutido anteriormente. Por meio do método `resolve`, o usuário pode consolidar apropriadamente o modelo declarado em um objeto do tipo *Problem*, registrando

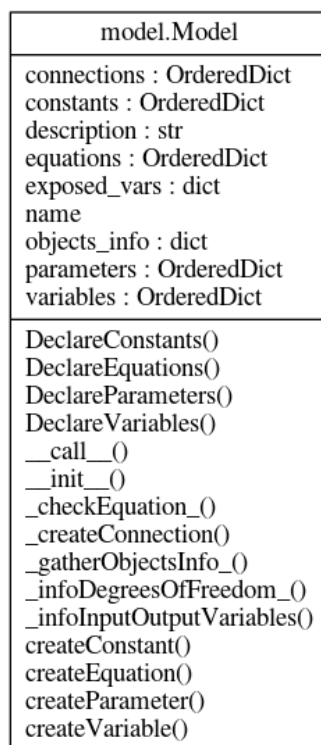


Figura 27 – Diagrama da classe *Model*, definida no pacote *model*.

Fonte – Próprio autor.

também o sistema de equações em um objeto do tipo *EquationBlock*.

O diagrama de classe em tela pode ser visualizado na Figura 27. Convém mencionar que a classe *Problem* define uma estrutura para que o sistema de equações seja redefinido manualmente, como consequência de eventuais modificações nos objetos *Model* atrelados ao objeto *Problem* em questão.

4.3.2.4 Pacote *simulation*

O pacote *simulation* define a classe *Simulation*. Essa classe encarrega-se de criar objetos de tipo do mesmo nome, os quais por sua vez recebem um objeto do tipo *Problem*. Por meio de configurações estabelecidas por meio de rotinas internas (método `setConfigurations`), o usuário pode definir uma série de características para a simulação do sistema de equações formado pelo caso de estudo em questão, bem como definir opções para a saída dos resultados. Nesse sentido, um objeto do tipo *Plotter* pode ser definido, por meio do qual podem ser produzidos diferentes gráficos.

Convém mencionar que para a conveniência do usuário, também foi implementado um método que exporta as configurações de um objeto *Simulation* para um

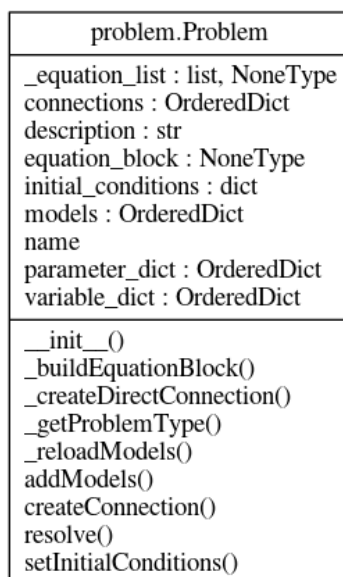


Figura 28 – Diagrama das classe *Problem*, definida no pacote *problem*.

Fonte – Próprio autor.

arquivo JSON (método `dumpConfigurations`), um formato multiplataforma e que permite legibilidade humana, muito utilizado para o armazenamento de configurações em *softwares*. De modo semelhante, o usuário pode definir rapidamente as configurações de uma simulação importando estas a partir de um arquivo. O diagrama da classe *Simulation* pode ser visualizado na Figura 29.

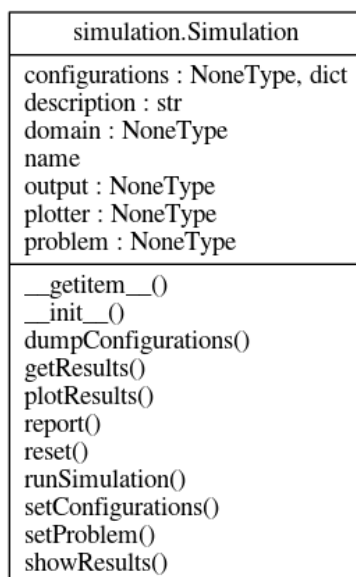


Figura 29 – Diagrama da classe *Simulation*.

Fonte – Próprio autor.

4.3.2.5 Pacote *optimization*

No pacote *optimization* estão definidas as classes *OptimizationProblem* e *Optimization*. A primeira, define os objetos de mesmo nome, os quais são utilizados para caracterização de problemas de otimização sujeita à restrições, a partir de um objeto do tipo *Simulation* definido previamente; a classe *Optimization* define um estudo de otimização, a partir de um objeto *OptimizationProblem*. Conforme descrito na seção 4.3.1.6, a partir de um ou mais graus de liberdade (parâmetros não especificados) de um caso de estudo, as rotinas de otimização realizam um processo iterativo de avaliar novos valores para estes graus de liberdade e o consequente resultado na função objetivo. Na Figura 30, o diagrama de classes daquelas referidas neste parágrafo pode ser visualizado.

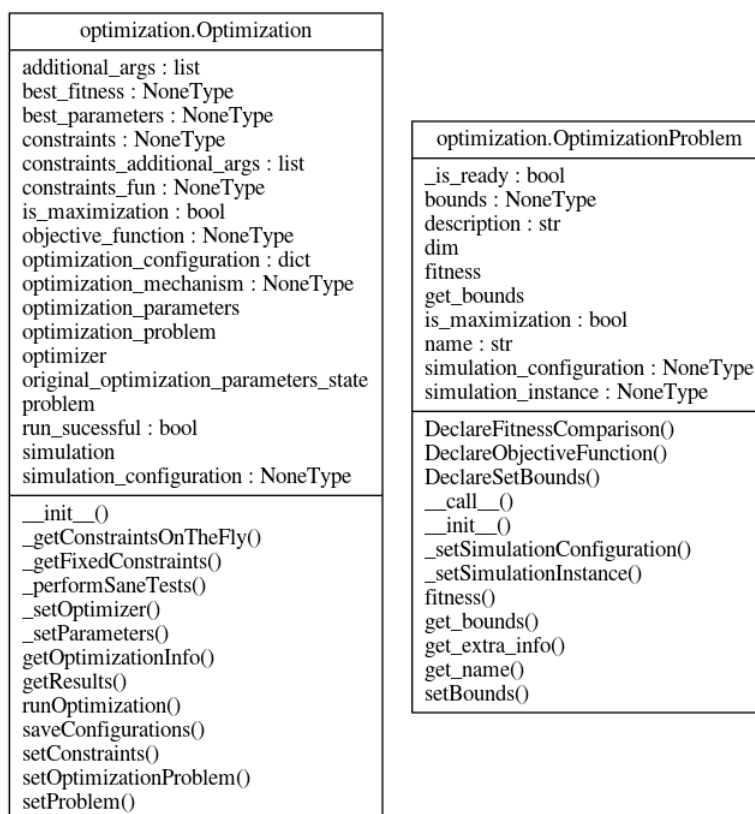


Figura 30 – Diagramas de classe daquelas definidas no pacote *optimization*.

Fonte – Próprio autor.

Em termos das rotinas disponíveis para otimização, a biblioteca *pagmo* (BIS-CANI et al., 2019) foi utilizada por meio da sua interface para a linguagem computacional Python. A referida biblioteca é endossada pela ESA (*European Spacial Agency*, ou agência espacial europeia), consistindo em uma ferramenta estado-da-arte para estudos de otimização.

4.3.2.6 Pacote *solver*

O pacote *solver* reúne as rotinas para a resolução dos sistemas de equações constantes no objeto *EquationBlock* de um dado estudo de caso definido pelo usuário. No pacote, está definida a classe *Solver*, a qual serve de base as classes *LASolver*, *NLASolver*, *DSolver* e *DaeSolver*, as quais são respectivamente encarregadas da resolução de sistemas contendo equações lineares, não-lineares, puramente diferenciais ou algébrico-diferenciais. Convém mencionar que nessas últimas classes, foi implementado um mecanismo para a compilação em tempo de execução das equações definidas na criação dos objetos *Model* em funções matemáticas, o que contribui para a acelerar o processo de integração e, portanto, da obtenção dos resultados. Na Figura 31, o diagrama de classes daquelas referidas neste parágrafo pode ser visualizado.

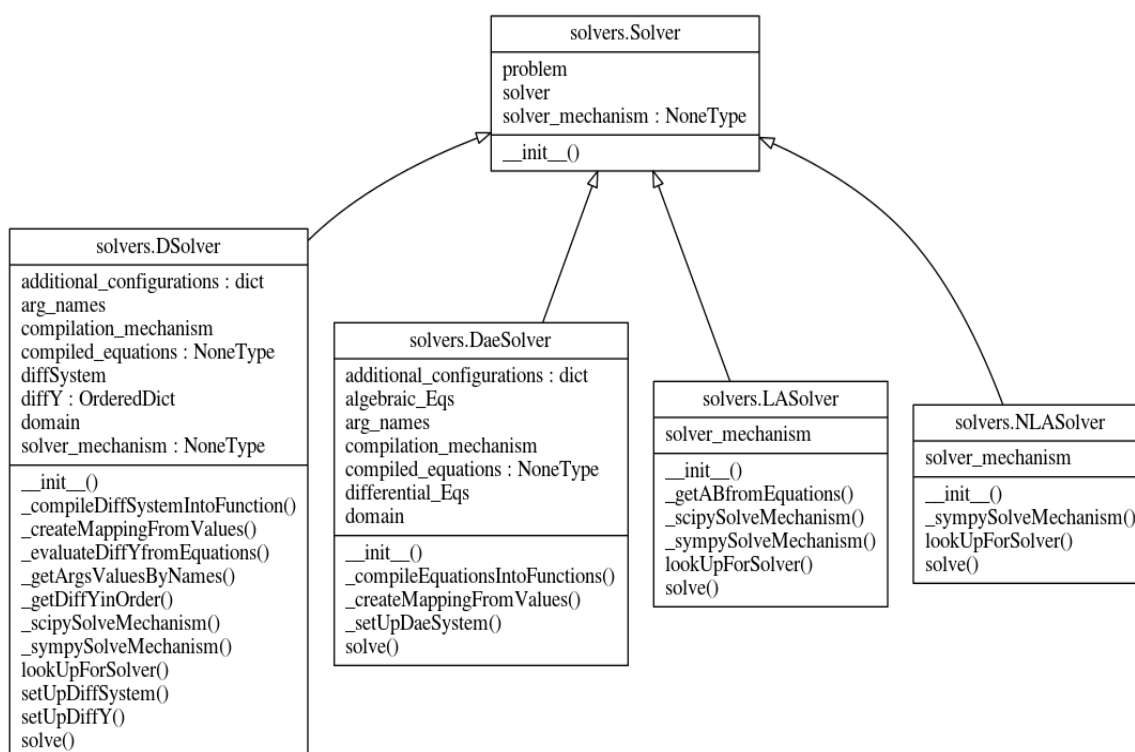


Figura 31 – Diagramas de classe daquelas definidas no pacote *solver*. As classes *LASolver*, *NLASolver*, *DSolver* e *DaeSolver* são utilizadas respectivamente para a resolução de sistemas de equações lineares, não-lineares, puramente diferenciais e algébrico-diferenciais.

Fonte – Próprio autor.

4.3.3 Diagramas de atividades

Na presente subseção, serão representados os processos de fluxo de informações no *software*, por meio de seus respectivos diagramas de atividade. Todos

os diagramas foram produzidos por meio do *software* livre yED (YWORKS, 2018), por meio de sua paleta de componentes de acordo com a padronização para este tipo de diagrama na UML.

4.3.3.1 Definição de um modelo

Na Figura 32, o diagrama de atividade para a definição de um modelo na ferramenta *sloth* é representado, descrevendo as etapas necessárias para a realização dessa tarefa. Na imagem é possível observar a importância da sobrecarga das funções pertinentes a declaração das variáveis, parâmetros e constantes do modelo (respectivamente, *DeclareVariables*, *DeclareParameters* e *DeclareConstant*). Ao fim, para o processamento do modelo, o método `__call__()` deve ser invocado.

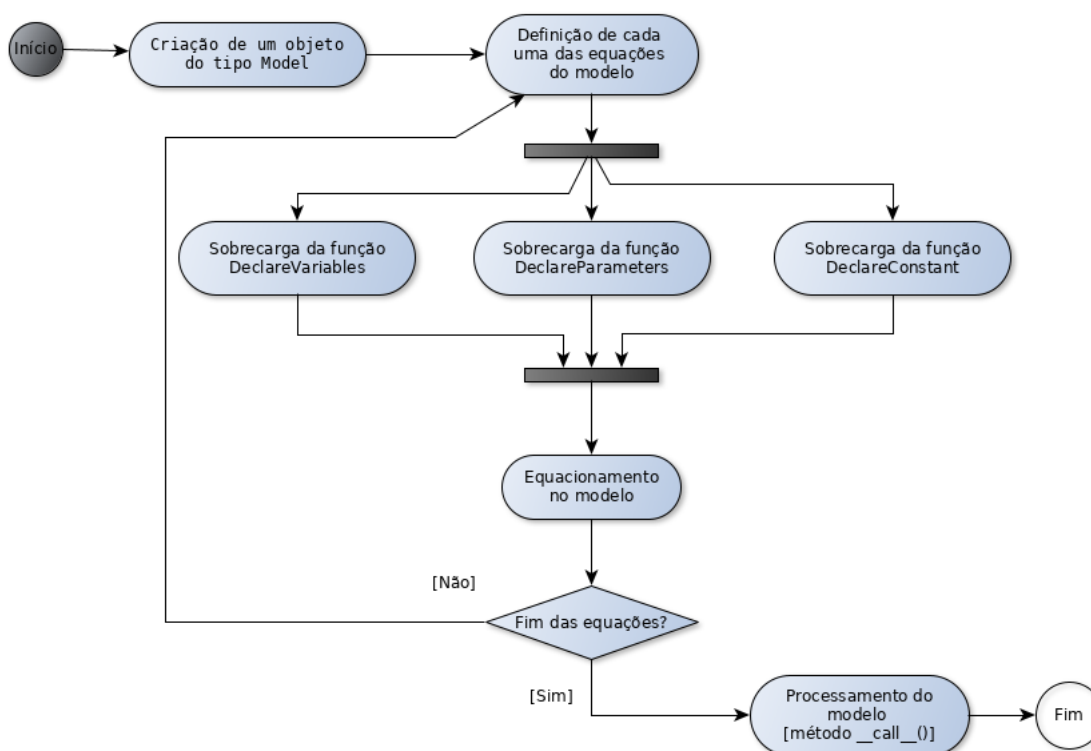


Figura 32 – Diagramas de atividade representando o fluxo informativo e procedimento realizado na definição de um modelo.

Fonte – Próprio autor.

4.3.3.2 Conexão entre dois modelos

Na Figura 33, o diagrama de atividade para a conexão entre dois modelos na ferramenta é representado. Por meio do referido diagrama é possível notar que essa conexão pode ser realizada de duas formas: diretamente a partir de dois mo-

delos (dois objetos *Model*); indiretamente, por meio do método constante no caso de estudo (objeto *Problem*) contendo os dois modelos.

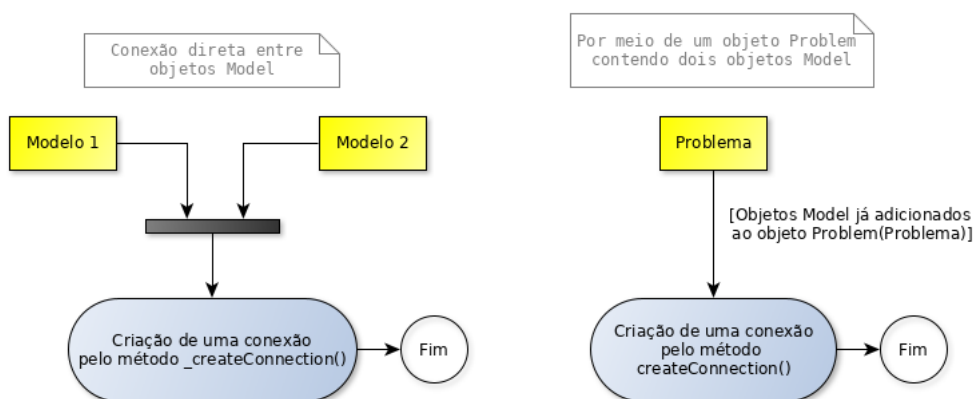


Figura 33 – Diagramas de atividade representando o fluxo informativo e o procedimento realizado na conexão de dois modelos

Fonte – Próprio autor.

4.3.3.3 Exibição das informações pertinentes do modelo

Na Figura 34, o diagrama de atividade para a exibição das informações pertinentes de um dado modelo é representado. Esse procedimento é realizado por meio do método `_infoModelReport_` do objeto *Model* do qual deseja-se obter as informações.

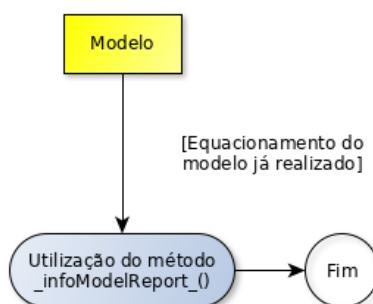


Figura 34 – Diagramas de atividade representando o fluxo informativo e o procedimento realizado na obtenção das informações de um modelo.

Fonte – Próprio autor.

4.3.3.4 Definição de um caso de estudo (problema)

Na Figura 35, o diagrama de atividade para a definição de um caso de estudo é representado. É possível notar que a composição do caso de estudo tem como pré-requisitos a definição dos modelos (objetos *Model*) integrantes do mesmo, e sua

posterior adição ao problema (objeto *Problem*) por meio de seu método `addModels`. Por fim, o caso de estudo deve ser processado por meio de seu método `resolve`.

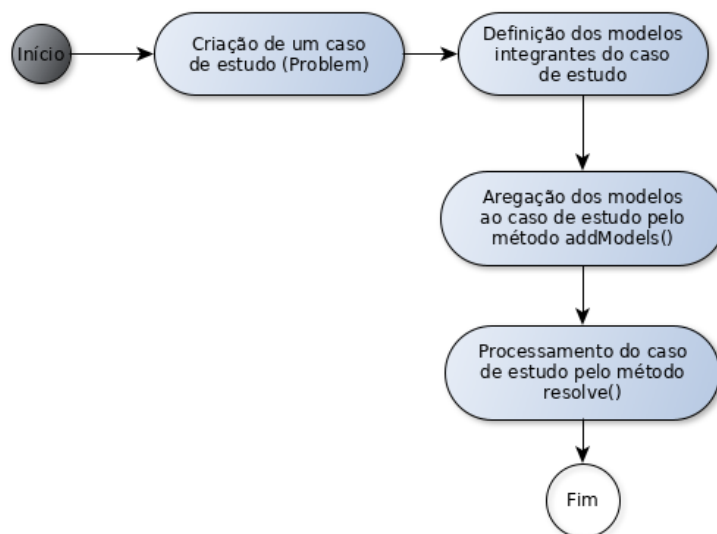


Figura 35 – Diagramas de atividade representando o fluxo informativo e o procedimento realizado na definição de um caso de estudo.

Fonte – Próprio autor.

4.3.3.5 Exibição das informações pertinentes do problema (caso de estudo)

Na Figura 36, o diagrama de atividade para a exibição das informações pertinentes de um caso de estudo é representado. Esse procedimento é realizado por meio do método `_infoProblemReport_` do objeto *Problem* do qual deseja-se obter as informações.

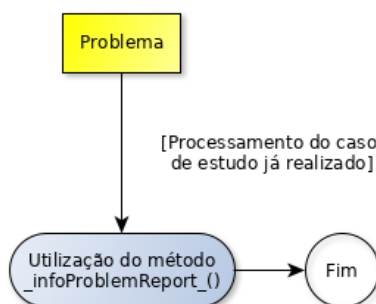


Figura 36 – Diagramas de atividade representando o fluxo informativo e o procedimento realizado na obtenção das informações de um caso de estudo.

Fonte – Próprio autor.

4.3.3.6 Definição de uma simulação

Na Figura 37, o diagrama de atividade para a definição de uma simulação (objeto *Simulation*) é apresentado. É possível notar que a composição do caso de estudo tem como pré-requisitos a definição dos modelos (objetos *Model*) integrantes do mesmo, e sua posterior adição ao problema (objeto *Problem*) por meio de seu método `addModels`. Por fim, o caso de estudo deve ser definido na simulação por meio do método `setProblem`.

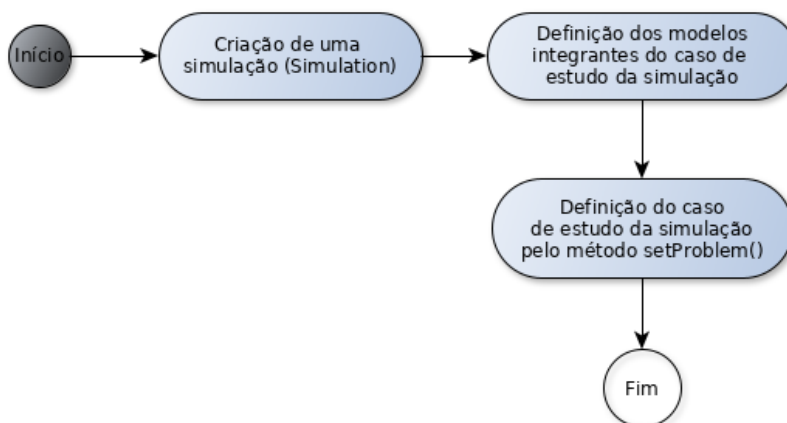


Figura 37 – Diagramas de atividade representando o fluxo informativo e o procedimento realizado na definição de uma simulação.

Fonte – Próprio autor.

4.3.3.7 Definição de um caso de estudo de otimização e sua execução

Na Figura 38, o diagrama de atividade para a definição de um caso de estudo de otimização e sua posterior execução é apresentado. A partir do referido diagrama, é possível notar que a definição de um problema de otimização (objeto *OptimizationProblem*), requisito para a criação de um caso de estudo de otimização (objeto *Optimization*), necessita de um conjunto de informações: a simulação base para o problema de otimização, a qual terá os parâmetros a serem otimizados definidos e, a partir de seus resultados, calcula-se a função objetivo; uma função objetivo, definida a partir do resultado da simulação; as restrições para os parâmetros a serem otimizados.

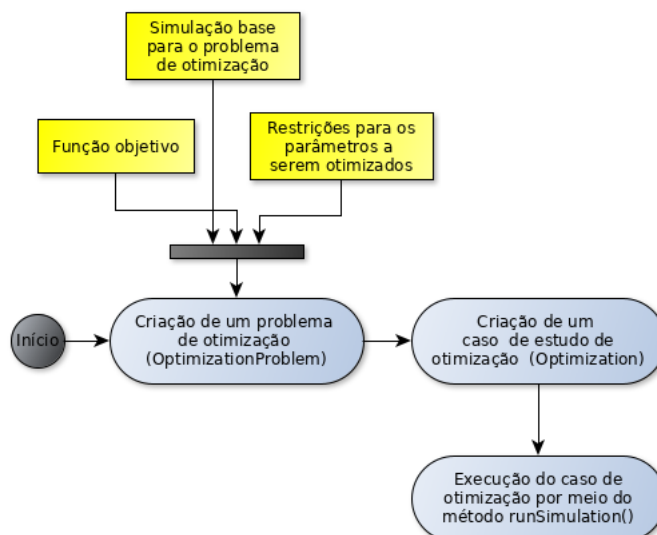


Figura 38 – Diagramas de atividade representando o fluxo informativo e o procedimento realizado na definição de um caso de estudo de otimização.

Fonte – Próprio autor.

4.3.3.8 Produção de gráficos a partir dos resultados de uma simulação

Na Figura 39, o diagrama de atividade para a produção de gráficos a partir dos resultados obtidos em uma simulação é apresentado. A produção dos gráficos a partir dos resultados pode ser obtida a partir do método `plotTimeSeries` (constante nos objetos *Simulation*), utilizando as configurações fornecidas para a obtenção destes.

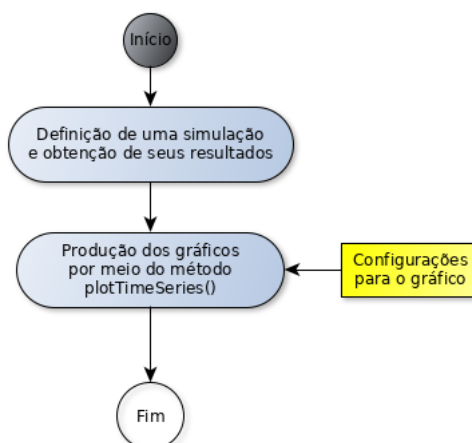


Figura 39 – Diagramas de atividade representando o fluxo informativo e o procedimento realizado na produção de gráficos a partir dos resultados de uma simulação.

Fonte – Próprio autor.

CAPÍTULO 5

Caso de estudo I: otimização em malha aberta de um processo de produção de bioetanol em batelada alimentada

No presente capítulo, será apresentado um caso de estudo para aplicação da ferramenta de simulação orientada à equações descrita neste trabalho, um estudo de otimização de um processo produtivo de bioetanol por meio de fermentação em meio submerso (líquido), em regime de batelada alimentada. Será utilizado um modelo de referência, apresentado no trabalho de HONG (HONG, 1986), o qual figura em diversos trabalhos que tratam do tema de estudos de otimização de processos biotecnológicos, conforme será descrito a seguir. Esse caso de estudo foi publicado (de maneira mais abrangente) no trabalho de Freitas et al. (2017), a partir do qual surgiu a demanda para o desenvolvimento de uma ferramenta de modelagem e simulação orientada a equações. O código-fonte utilizado para a resolução do presente caso de estudo pode ser encontrado no Apêndice A.

5.1 Apresentação da problemática

O presente estudo de caso tem como objetivo analisar o problema de otimização dinâmica, *in-silico* (ou desenvolvido computacionalmente), de um processo de produção de bioetanol conduzido em um biorreator do tipo batelada alimentada, por meio da utilização do conceito de controle preditivo baseado em modelo não-linear. A otimização dinâmica empregará duas técnicas de computação evolutiva, o algoritmo genético (GA) e a evolução diferencial (DE), serão comparados seus desempenhos na manipulação do perfil da taxa de alimentação em termos da produtividade de bioprocessos obtida na utilização da abordagem de otimização em malha aberta, usando um conceito de tempo de terminal livre. À medida que o perfil dinâmico dos parâmetros operacionais apresenta correlação direta com o rendimento do bioprocessos, diferentes perfis de alimentação são avaliados em termos de produtividade de bioetanol (SPADIUT et al., 2013; SPADIUT; HERWIG, 2014).

Devido à relevância já mencionada do tema da geração de bioenergia para a sociedade moderna, a descrição apropriada de um processo de geração de bioenergia em termos de uma modelagem matemática representa um importante assunto de estudo. Neste sentido, várias obras apresentaram modelos para processos de produção de bioenergia através da atividade microbiana, incluindo a geração de biohidrogênio (RIO-CHANONA et al., 2016; WANG et al., 2017; SRIDEVI et al., 2014), a produção de biogás (usado diretamente ou refinado como biometano) através de digestão anaeróbica (KANA et al., 2012; ENITAN et al., 2017; JACOB; BANERJEE, 2016), produção de bioetanol (através da conversão biológica de culturas contendo açúcar, insumos brutos amiláceos ou materiais lignocelulósicos) (OCHOA et al., 2010; BALAT, 2011; ROCHA et al., 2014).

Sob um aspecto de engenharia de processo, a otimização dos processos

biotecnológicos visa o seu aprimoramento da produtividade explorando as capacidades máximas de um microorganismo já selecionado e manipulando variáveis ambientais e operacionais (ROCHA et al., 2014), uma vez que para garantir rendimento da cultura em batelada do produto desejado, a concentração do substrato deve ser controlada para um nível adequado. Esse controle é fundamental para evitar a ocorrência de metabolismo de transbordamento (no inglês, *overflow*) e aumentar a produtividade celular (WECHSELBERGER et al., 2012; YE et al., 2014; HENES; SONNLEITNER, 2007).

5.1.1 Modelagem do processo fermentativo

No caso em estudo, utilizou-se um modelo não segregado e não estruturado para a descrição (*in-silico*) processo de produção do bioetanol, em um biorreactor alimentado pelo microrganismo *Saccharomyces cerevisiae*, conforme apresentado em o trabalho de HONG (HONG, 1986), descrito nas Equações 8-13. Para a conveniência do leitor, a hierarquização entre os diferentes tipos de modelos mecanísticos empregados para a descrição de processos biotecnológicos, já apresentada na Figura 2, será novamente apresentada a seguir.

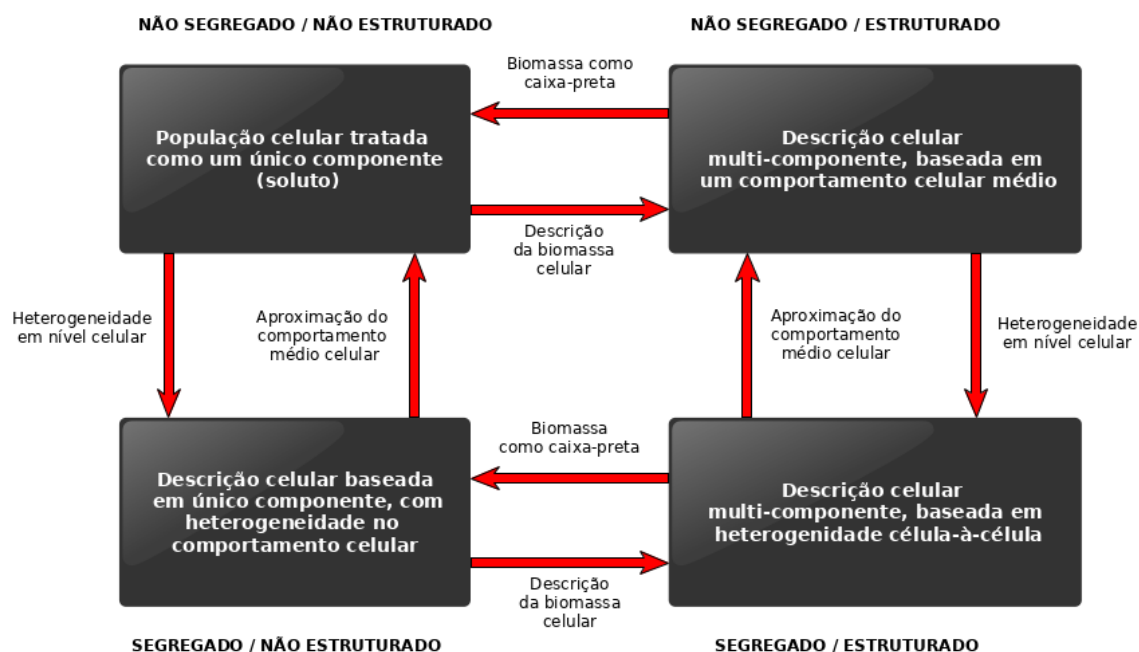


Figura 40 – Representação esquemática dos tipos de modelos mecanísticos utilizados na descrição dos processos biotecnológicos, destacando a diferença entre cada um deles, apresentado novamente para a conveniência do leitor.

Fonte – Adaptado de Fernandes et al. (2011), Gernaey et al. (2010).

O modelo de HONG, empregado no presente caso de estudo, tem sido usado como uma ferramenta de referência para a otimização de produção de bioetanol *in-silico*, conforme apresentado nos trabalhos de vários autores, a exemplo de ROCHA et al.; OCHOA; BANGA et al. (ROCHA et al., 2014; OCHOA, 2016; BANGA et al., 2005), entre outros.

$$\frac{dV}{dt} = u(t) \quad (8)$$

$$\frac{dX}{dt} = \mu X - \frac{uX}{V} \quad (9)$$

$$\frac{dS}{dt} = \frac{-\mu X}{y_r} + \frac{u(S_0 - S)}{V} \quad (10)$$

$$\frac{dP}{dt} = qX - \frac{uP}{V} \quad (11)$$

$$\mu = \left(\frac{\mu_0}{1 + P/K_P} \right) \left(\frac{S}{K_S + S} \right) \quad (12)$$

$$q = \left(\frac{q_0}{1 + P/K_{PI}} \right) \left(\frac{S}{K_{SI} + S} \right) \quad (13)$$

O significado dos termos utilizados nas Equações 8-13 é apresentada na Tabela 3, a partir da representação matemática utilizada nas Equações 8-13. Convém mencionar que foram preservadas as unidades originais do modelo, conforme apresentado no trabalho de de Hong (1986). Por sua vez, os valores dos parâmetros utilizados no modelo são apresentados na Tabela 4.

Tabela 3 – Representação matemática dos termos utilizados no modelo de produção de bioetanol, e seu significado.

Termo	Significado	Unidade
V	Volume	L
X	Concentração de biomassa celular	$g L^{-1}$
S	Concentração de substrato	$g L^{-1}$
P	Concentração de bioetanol	$g L^{-1}$

O modelo utilizado no presente caso de estudo foi desenvolvido com base em considerações biológicas importantes que implicam em sua formulação matemática, como a ocorrência da cinética de Monod para inibição de substrato e produto; Toda a biomassa celular é considerada viável (e, portanto, constituída por microorganismos capazes de converter o substrato em produto de etanol), não é considerada a

Tabela 4 – Parâmetros do modelo para a produção de bioetanol em batelada-alimentada (HONG, 1986).

Parâmetro	Unidade	Definição	Valor
μ_0	h^{-1}	Taxa máxima de crescimento da biomassa	0,408
q_0	h^{-1}	Taxa máxima de produção de bioetanol	1
K_S	$g L^{-1}$	Constante de Monod	0,22
K_{SI}	$g L^{-1}$	Constante de inibição do substrato	0,44
K_P	$g L^{-1}$	Constante de Monod	16
K_{PI}	$g L^{-1}$	Constante de inibição do produto	71,5
y_r	$g g^{-1}$	Fator de rendimento biomassa/substrato	0,1
S_0	$g L^{-1}$	Concentração de substrato na alimentação	150

cinética da morte para a cultura microbiana durante o tempo da batelada e a concentração de biomassa durante a fermentação é diretamente proporcional à taxa de crescimento de biomassa (μ); o meio de fermentação é considerado perfeitamente homogêneo durante o tempo da batelada, portanto, não é observada variação espacial para a concentração de biomassa celular, substrato e bioetanol dentro do fermentador; E, finalmente, o bioetanol representa o produto microbiano mais significativo durante a fermentação, sendo omitidos demais produtos.

5.1.2 Otimização de processos biotecnológicos

Embora os processos biotecnológicos contínuos representem um interesse notório tanto para o meio acadêmico como para a indústria, especialmente para cenários em que a biomassa celular representa o produto de interesse, os biorreatores que operam sob essa configuração por vezes apresentam várias implicações deletérias para o bioprocessamento, tais como a heterogeneidade no interior do equipamento, desafios relacionados à operabilidade a longo prazo e manutenção da esterilidade, entre outros. Outra importante questão consiste no fato de que muitas vezes os custos para atualizar a estrutura de produção para a produção contínua são proibitivos (CROUGHAN et al., 2015; ZYDNEY, 2015). Assim, vários reatores de grande escala ainda operam sob uma configuração alimentada por batelada, de modo que é importante garantir uma metodologia de controle adequada para a taxa de alimentação de substrato (RIO-CHANONA et al., 2016; HENES; SONNLEITNER, 2007), conforme mencionado anteriormente.

Sob um aspecto geral, apesar do rápido desenvolvimento tecnológico observado no campo dos bioprocessos, as otimizações de seus parâmetros operacionais por vezes ainda são baseadas em conhecimento empírico ou por tentativa e erro,

o que mostra-se proibitivo sob o ponto de vista financeiro e operacional (KAWOHL et al., 2007). Os experimentos de otimização *in-silico* representam uma alternativa importante, já que vários cenários operacionais podem ser estudados para buscar a maximização do rendimento do produto desejado e/ou a minimização da obtenção de subprodutos (BANGA, 2008; APEL; WEUSTER-BOTZ, 2015). Vários estudos enfatizam a importância das condições dinâmicas das variáveis operacionais para o bioprocessamento, pois suas características transitórias podem afetar profundamente o rendimento do produto desejado ou a sua pureza (SPADIUT et al., 2013; SPADIUT; HERWIG, 2014; WECHSELBERGER et al., 2012; SANTO et al., 2014). Nesse sentido, a otimização dinâmica do bioprocessamento é fundamental para garantir sua competitividade econômica, incentivando a utilização de técnicas avançadas de controle.

A otimização dinâmica de um determinado processo biotecnológico, em termos de seu modelo descritivo, pode ser vista como um problema de estimação de parâmetros dos perfis dinâmicos das variáveis manipuladas (por exemplo: alimentação de substrato, aeração, agitação e taxas de aquecimento), usando o rendimento específico do(s) produto(s) desejado(s) como função-objetivo. Um problema geral de otimização é descrito matematicamente nas Equações 14-18 (WÜRTH et al., 2009).

$$\min J(x,u,t) \quad (14)$$

sujeito a:

$$\dot{x} = f(x(t),u(t)) \quad (15)$$

$$u_{min} \leq u \leq u_{max} \quad (16)$$

$$t \in [t_{min}, t_{max}] \quad (17)$$

$$x_{min} \leq x \leq x_{max} \quad (18)$$

5.2 Métodos utilizados

5.2.1 Simulação da produção de etanol

Com o intento de simular a produção de etanol, o modelo constituído pelas equações diferenciais apresentadas nas Equações 8-13, acrescidas das condições iniciais que definem o problema de valor inicial (PVI), cujos valores iniciais são descritos na Tabela 5.

Tabela 5 – Valores iniciais das variáveis empregadas no PVI resultante do processo de produção *in-silico* do bioetanol

Variáveis	Significado	Valor
V	Volume (L)	10
X	Concentração de biomassa celular ($g L^{-1}$)	1
S	Concentração de substrato ($g L^{-1}$)	150
P	Concentração de bioetanol ($g L^{-1}$)	0

Com o intuito de realizar a simulação do problema de produção de bioetanol, o modelo foi definido na ferramenta `sloth`. Para tanto, o `solver` do sistema diferencial determinado pelo PVI definido anteriormente, foi empregado a rotina `odeint`, da biblioteca `scipy` (JONES et al., 2001). Esse solver serve como interface para o confiável algoritmo LSODA, da biblioteca `ODEPACK` (HINDMARSH, 1983).

5.2.2 Otimização do processo fermentativo *in-silico*

Durante o procedimento de otimização em malha aberta para a produção de bioetanol *in-silico*, a taxa de alimentação do biorreator figura como a variável manipulada por meio da parametrização da sua vazão volumétrica nominal, e a função objetivo como a produtividade final de bioetanol ($g h^{-1}$), conforme será descrito a seguir. O procedimento para a execução da otimização em malha aberta é esquematicamente representado na Figura 41, no qual u_{opt} representa o valor ótimo para a variável manipulada, \hat{x} a saída do processo (a partir do qual o valor da função objetivo será calculado) e u o valor atual para a variável de manipulada. O processo de otimização foi conduzido por meio da rotina dedicada a esse propósito da ferramenta `sloth`, a qual dispõe de diversas rotinas de otimização, conforme descrito no Capítulo 4.

Na parametrização da vazão da corrente de alimentação do biorreator, foram avaliadas duas diferentes expressões para a determinação do perfil temporal

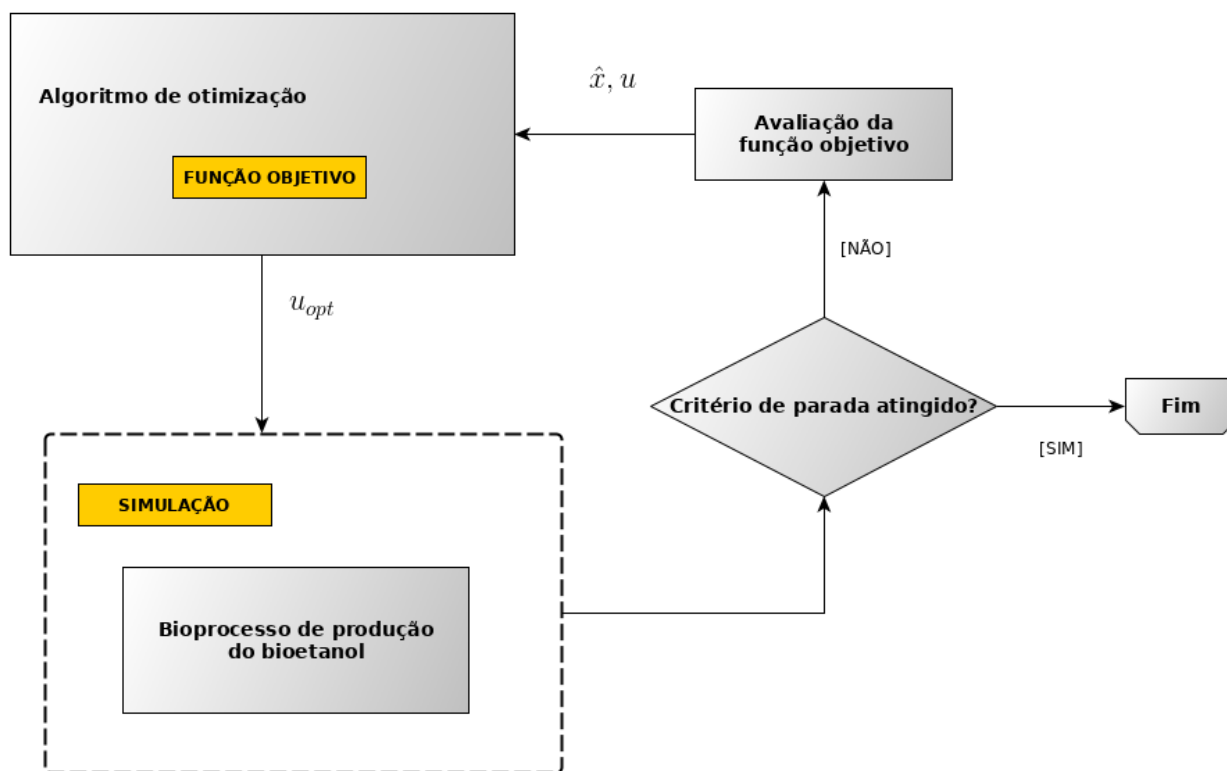


Figura 41 – Representação esquemática do procedimento empregado na otimização em malha aberta do processo de produção do bioetanol.

Fonte – Próprio autor.

da vazão de alimentação do fermentador, a saber: uma expressão polinomial de terceiro grau e uma expressão cossenoidal, esta última sugerida no trabalho de Ochoa (2016). O tempo total de batelada foi tratado como outro parâmetro a ser determinado, em uma abordagem chamada de *open end-time* (ou tempo final em aberto). Assim, as expressões para a parametrização da vazão de alimentação de substrato ao biorreator são apresentadas nas Equações 19-20.

$$u(t) = a \left(\frac{t}{T_f} \right)^3 + b \left(\frac{t}{T_f} \right)^2 + c \left(\frac{t}{T_f} \right) + d \quad (19)$$

$$u(t) = a \cos \left[b \left(\frac{t}{T_f} \right) + c \right] + d \quad (20)$$

Nas Equações 19-20, os termos $u(t)$, a, b, c, d , representam respectivamente o vetor dos valores para a vazão de alimentação parametrizada para um dado instante de tempo t , e os parâmetros a serem determinados para cada uma das expressões de parametrização da vazão de alimentação do biorreator.

Conforme já foi descrito anteriormente, faz-se necessário a determinação dos parâmetros para a expressão de determinação da vazão de alimentação da dorna de fermentação. Este problema reside na maximização de uma função objetivo usando um algoritmo de otimização. Convém mencionar que a escolha da função a ser maximizada representa um aspecto cardinal da resolução do problema, e não há consenso acerca de qual relação matemática é mais apropriada para estudos de otimização envolvendo bioprocessos, muito embora a produtividade específica (unidades de $g h^{-1}$) seja referida como mais adequada para processos de cultura em batelada no qual o biocatalisador não é recuperado após o processo. Não obstante, funções objetivo baseadas em índices volumétricos específicos (unidades de $g L^{-1} h^{-1}$) possuem a vantagem de correlacionar a capacidade da unidade de fermentação e o tempo requerido para a produção de uma dada quantidade de produto (MAURER et al., 2006; WERNER, 2004). Neste sentido, a métrica de produtividade volumétrica específica foi utilizada no presente caso de estudo como função objetivo para a otimização em malha aberta, representada pela Equação 21, em termos da Equação 14 descrita anteriormente.

$$J(x,u,t) = \frac{P(t = T_f) V(t = T_f)}{T_f} \quad (21)$$

Na função objetivo descrita na Equação 21, J representa a produtividade específica de bioetanol ($g h^{-1}$), na qual o termo T_f representa o tempo final do processo de cultivo microbiológico, cujo valor também deve ser determinado pela rotina de otimização (considerando que foi adotado no presente caso de estudo um paradigma de tempo aberto de batelada, ou *open end-time*, conforme já foi mencionado), P a concentração de bioetanol em meio ao ambiente reacional do fermentador, e V o volume do meio reacional contido no equipamento. As restrições para as variáveis são apresentadas na Tabela 6.

Tabela 6 – Restrições utilizadas na resolução dos problemas de otimização no presente caso de estudo.

Parâmetro	Unidades	Valor mínimo	Valor máximo
$u(t)$	$L h^{-1}$	0	12
V	L	0	200
X	$g L^{-1}$	0	-
S	$g L^{-1}$	0	-
P	$g L^{-1}$	0	-

Na determinação do perfil temporal da variável manipulada vazão de alimen-

tação, foram utilizados os algoritmos estocásticos de otimização *algoritmo genético* (GA) e *evolução diferencial* (DE). Conforme descrito no Capítulo 3, ambas as técnicas integram o rol de rotinas meta-heurísticas baseadas em computação evolutiva, embora os princípios norteadores das mesmas difiram entre si. Esses princípios serão sucintamente descritos a seguir.

A técnica GA baseia-se na recombinação das soluções candidatas, codificadas de uma maneira análoga à informação genética. De modo geral, para cada geração (ou iteração do algoritmo), essa recombinação pode ocorrer por meio do cruzamento (*crossover*) entre duas ou mais soluções, produzindo uma nova população. O desempenho dessa nova população é avaliado por meio da função objetivo e, comparando-se com o desempenho da população anterior, são escolhidos as melhores soluções para compor a nova população. Durante o processo de *crossover*, pode ocorrer uma mutação para um ou mais dos parâmetros que compõem a solução candidata (os *gens*, retomando a analogia com a codificação genética), ocorrendo assim a exploração do espaço de busca, o que pode levar a descoberta de soluções candidatas melhores. O procedimento é repetido até que o critério de parada seja atingido (MITCHELL, 1998; HEGERTY et al., 2009).

De maneira semelhante, o algoritmo DE baseia-se na recombinação das soluções candidatas e na possibilidade de ocorrer uma alteração aleatória em uma dos parâmetros das soluções candidatas, também chamada de mutação. Entretanto, convém destacar que a recombinação na evolução diferencial ocorre por meio da projeção dos vetores formados pelas soluções candidatas ao longo do espaço de busca do problema, por meio do parâmetro de fator de escala (*scale factor*) definido para o problema (STORN; PRICE, 1997; HEGERTY et al., 2009; WANG et al., 2013).

Conforme já foi mencionado, a otimização ficou a cargo do módulo de DRTO da ferramenta *sloth*, o qual utiliza incidentalmente nas tarefas de otimização as rotinas disponibilizadas pela biblioteca *pagmo* (BISCANI et al., 2019), e as configurações utilizadas para cada um dos algoritmos são apresentadas nas Tabelas 7 e 8, respectivamente. Convém mencionar que estes valores foram determinados de forma empírica.

Tabela 7 – Configuração empregada no algoritmo GA.

Parâmetro	Valor
Probabilidade de <i>crossover</i>	60%
Probabilidade de mutação	4%
Operador genético para <i>crossover</i>	Aritmético
Operador genético para mutação	Gaussiano
Número de indivíduos	50
Método de seleção	Torneio (4 indivíduos)
Critério de parada	Número de gerações (300)

Tabela 8 – Configuração empregada no algoritmo DE.

Parâmetro	Valor
Probabilidade de <i>crossover</i>	60%
Fator de escala	0.5 (constante)
Variação do algoritmo DE	DE/best/2/bin
Número de soluções candidatas	50
Critério de parada	Número de gerações (300)

5.3 Resultados e discussões

Na presente seção, os resultados obtidos para o estudo de otimização em malha aberta para o processo de produção de bioetanol serão apresentados, em termos dos diferentes algoritmos de otimização utilizados para este intento. Nas Figuras 42 e 43, os perfis temporais para a variável manipulada, a saber a vazão de enchimento do fermentador, são apresentados, sendo determinados respectivamente pelos algoritmos GA e DE, para cada uma das expressões de parametrização adotadas: polinomial e cossenoidal, apresentadas respectivamente nas Equações 19 e 20. Nas Figuras 44- 47, o perfil temporal obtido para as concentrações de biomassa (x_2), substrato (x_3) e bioetanol (x_4) é apresentado, para ambas as expressões de parametrização.

Na Tabela 9, são apresentados os resultados obtidos quanto aos parâmetros da função de parametrização da vazão de alimentação do fermentador, determinados pela rotina GA, para ambas as expressões de parametrização. De maneira similar, na Tabela 10, são apresentados os resultados obtidos a partir da rotina DE.

Tabela 9 – Resultados obtidos para os parâmetros das funções de enchimento da dorna e o subsequente valor obtido para a função objetivo, por meio da rotina GA.

Parâmetro	Polinomial (Equação 19)	Cossenoidal (Equação 20)
a	-100	-46,275964
b	100	-6,057966
c	-88,469655	93,118459
d	52,087491	28,206027
T_f	22,818783	23,453274
J (Equação 21)	568,770075	591,642361

Tabela 10 – Resultados obtidos para os parâmetros das funções de enchimento da dorna e o subsequente valor obtido para a função objetivo, por meio da rotina DE.

Parâmetro	Polinomial (Equação 19)	Cossenoidal (Equação 20)
a	-80,706081	55,896121
b	-25,430220	6,243526
c	-66,621748	35,202551
d	85,137382	27,384719
T_f	23,843537	25,021059
J (Equação 21)	602,759790	582,467526

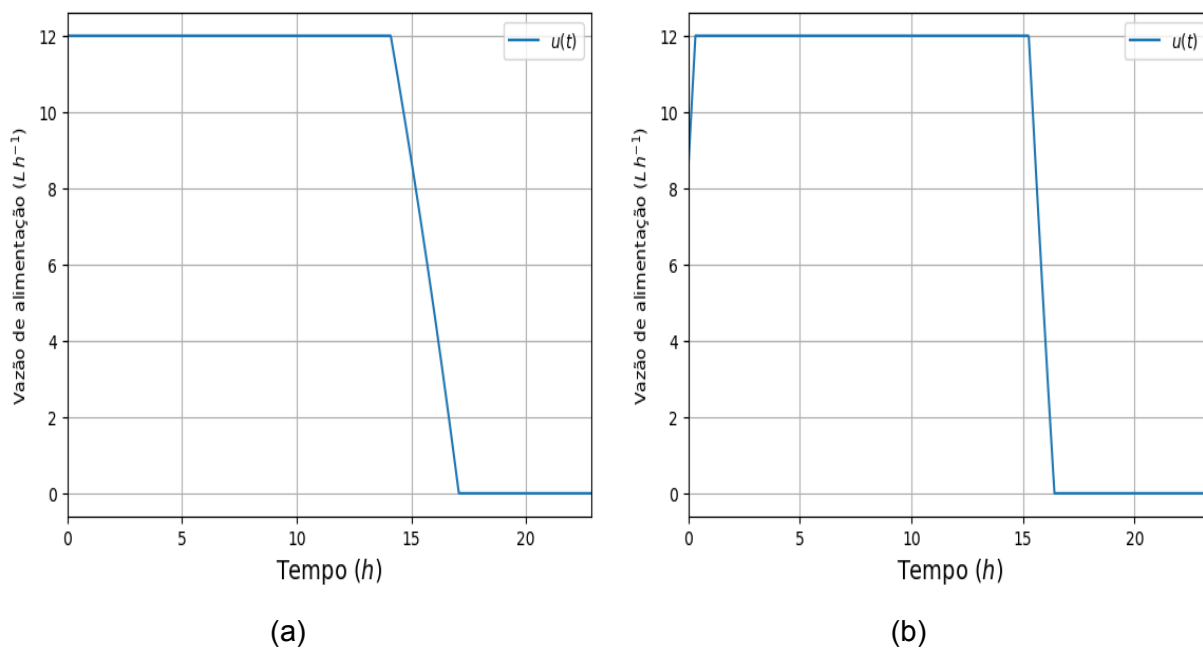


Figura 42 – Perfil temporal da vazão de enchimento do fermentador, obtido a partir da rotina GA, para a expressão paramétrica polinomial (a) e cossenoidal (b).

Fonte – Próprio autor.

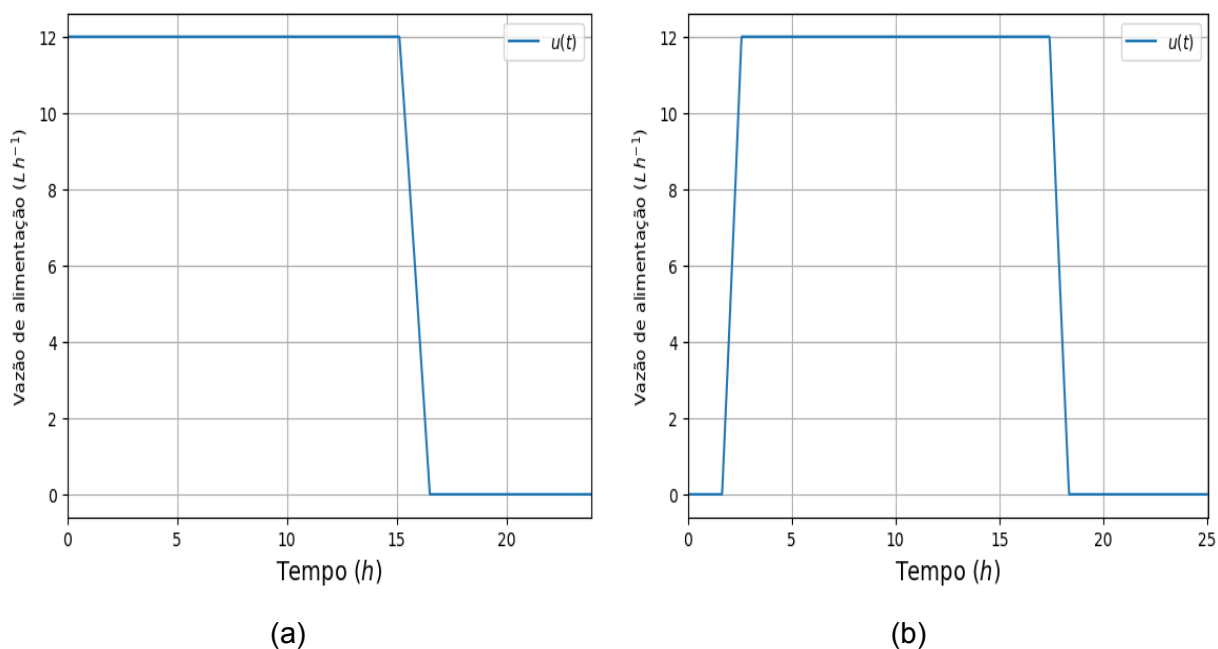


Figura 43 – Perfil temporal da vazão de enchimento do fermentador, obtido a partir da rotina DE, para a expressão paramétrica polinomial (a) e cossenoidal (b).

Fonte – Próprio autor.

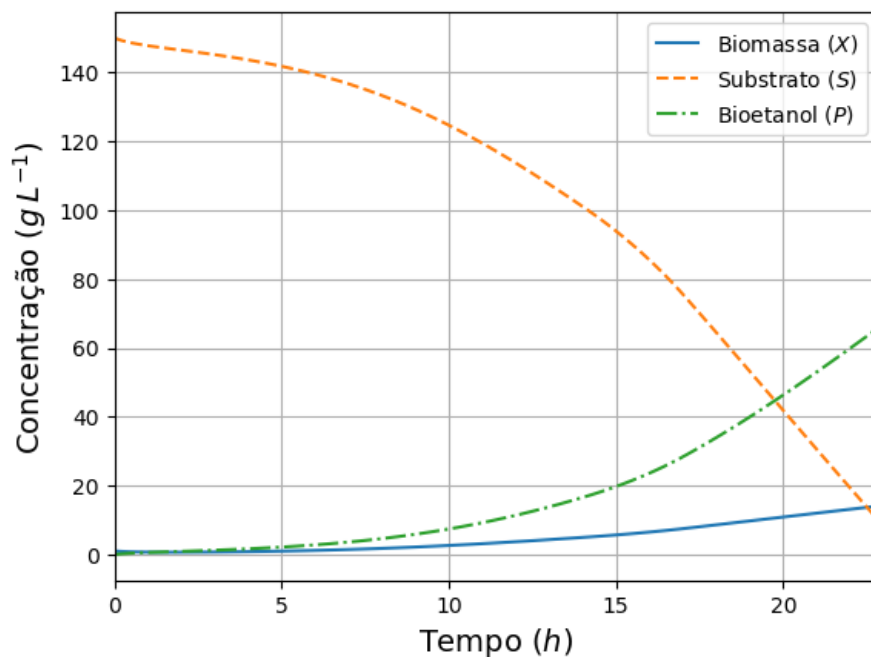


Figura 44 – Perfil temporal das concentrações de biomassa, substrato e bioetanol, obtido a partir da rotina GA, para a expressão paramétrica polinomial.

Fonte – Próprio autor.

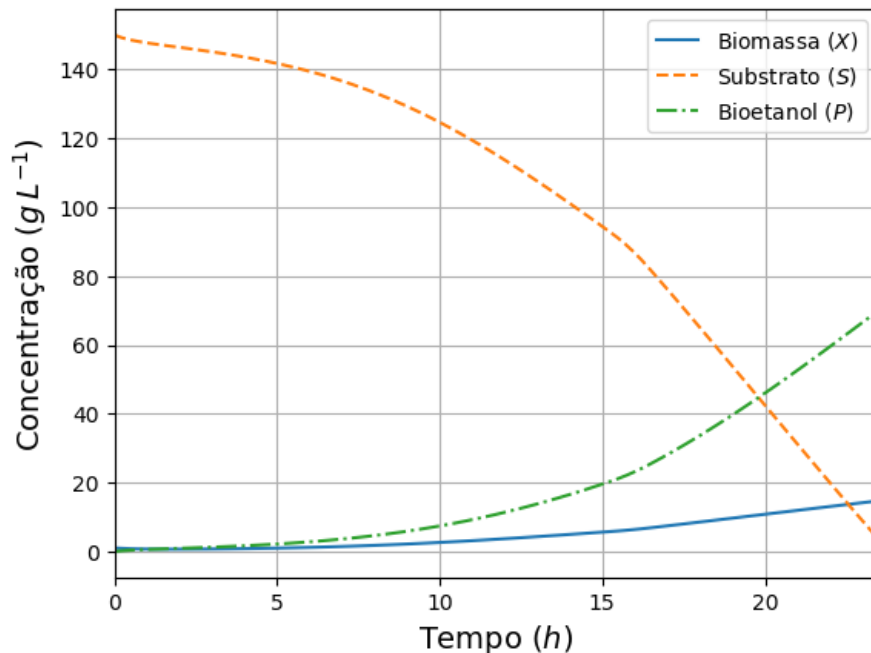


Figura 45 – Perfil temporal das concentrações de biomassa, substrato e bioetanol, obtido a partir da rotina GA, para a expressão paramétrica cossenoidal.

Fonte – Próprio autor.

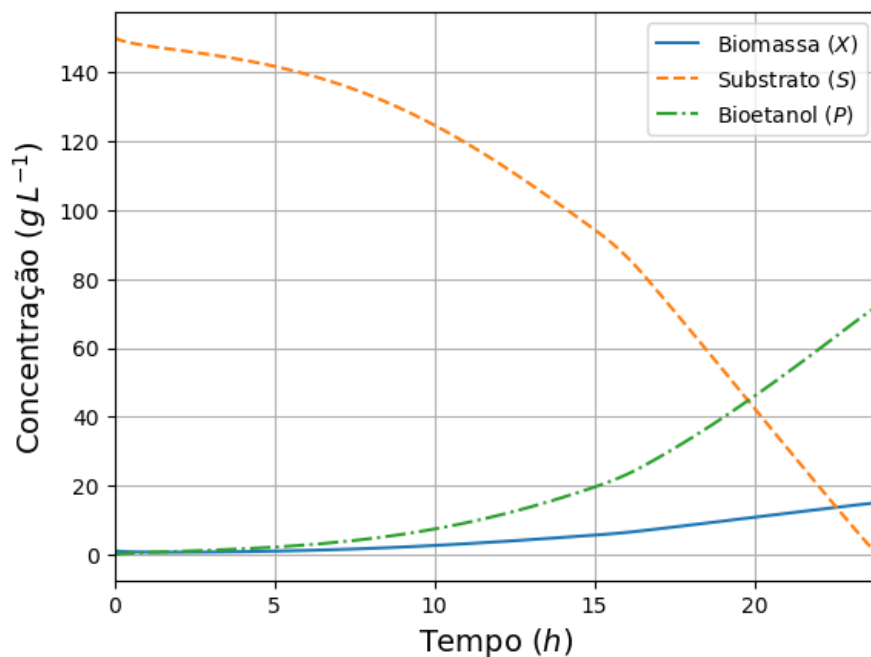


Figura 46 – Perfil temporal das concentrações de biomassa, substrato e bioetanol, obtido a partir da rotina DE, para a expressão paramétrica polinomial.

Fonte – Próprio autor.

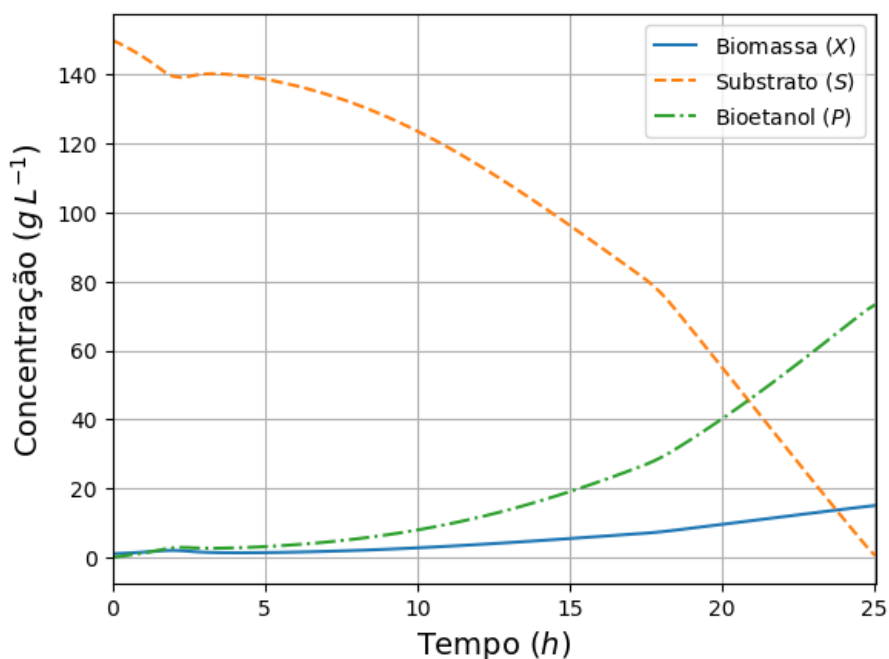


Figura 47 – Perfil temporal das concentrações de biomassa, substrato e bioetanol, obtido a partir da rotina DE, para a expressão paramétrica cossenoidal.

Fonte – Próprio autor.

A partir dos resultados apresentados, é possível notar que à despeito de alguma variação, o valor da função objetivo é aproximadamente o mesmo para os

quatro casos, o que é refletido também na semelhança entre os perfis dinâmicos de concentração de biomassa, substrato e bioetanol no interior do fermentador (Figuras 44-47). O tempo final da fermentação é praticamente o mesmo para ambas as parametrizações, variando discretamente entre os algoritmos de otimização. Os resultados indicam que os algoritmos de otimização conseguem modular satisfatoriamente a taxa de alimentação, com vistas a controlar a taxa de produção de etanol, que atua como um inibidor do metabolismo celular, e em última instância, da própria produção de bioetanol. Convém notar que a redução na vazão de alimentação ocorre justamente no período em que há um significativo aumento na taxa de produção de etanol, conforme pode ser visto comparando as Figuras 42 e 43 e as Figuras 44-47.

A semelhança no desempenho entre os algoritmos GA e DE distoa do resultado obtido por Rocha et al. (2014), no qual os autores empregaram o algoritmo genético (GA) com codificação real, comparando-o ao DE. Os resultados obtidos pelo referido trabalho mostram que a rotina EA convergiu prematuramente quando comparado a outros algoritmos, fato que não foi observado pela rotina GA no presente trabalho, o que pode ser justificada pela capacidade de exploração do espaço de busca deste algoritmo por meio do operador de mutação genética, que permite que as soluções candidatas tenham seus valores ligeiramente alterados, o que propicia a descoberta de potenciais soluções melhoradas para o problema de otimização em questão.

Em última análise, o estudo de caso apresentado no presente capítulo mostra que a ferramenta `sloth` mostra-se adequada a estudos de simulação dinâmica e otimização de bioprocessos, ainda que a presente análise tenha se limitado a um estudo em malha aberta. Contudo, é importante notar que o tempo para obtenção da solução mostrou-se muito alto (cerca de 12 h em um computador com um processador *Intel i5* de 3 Ghz e 8 Gb de RAM), o que pode ser justificado pelo fato de que os cálculos ocorrem majoritariamente em tempo de execução compatível com o esperado da linguagem *Python*. Assim, é importante analisar a possibilidade de atribuir essa tarefa a linguagens computacionais de mais baixo nível, o que implicaria em um significativo ganho de performance quanto ao tempo necessário para obter-se uma resposta com a ferramenta.

CAPÍTULO 6

Caso de estudo II: simulação e controle de
processo de produção de bioetanol por
fermentação contínua

No presente capítulo, será apresentado um caso de estudo para aplicação da ferramenta `sloth`, descrita neste trabalho, referente a simulação e controle de um processo de produção de bioetanol por meio de fermentação contínua utilizando o micro-organismo *Zymomonas mobilis*, apresentada no trabalho de Mirlekar et al. (2017). Por meio da simulação dinâmica do modelo, serão avaliadas as influências de alguns parâmetros na resposta obtida a partir do mesmo, conforme será descrito posteriormente. Em termos do controle do processo de produção de bioetanol *in-silico*, a técnica de otimização dinâmica em tempo real (DRTO, do inglês *Dynamic Real-Time Optimization*) será empregada. O código-fonte utilizado para a resolução de ambas as tarefas referentes ao presente caso de estudo pode ser encontrado no Apêndice B.

6.1 Apresentação da problemática

O presente estudo de caso tem como objetivo a modelagem, simulação e controle de um processo produtivo de bioetanol operado em um regime contínuo, em um reator do tipo CSTR (*Continuous Stirred Tank Reactor*, ou tanque reator agitado contínuo), por meio do cultivo do micro-organismo fermentador *Zymomonas mobilis*. A complexidade inerente ao processo produtivo reside no fato de a biomassa ter um papel análogo ao de um catalisador para a conversão do substrato, e ao mesmo tempo representa um produto do processo fermentativo, ao passo que o substrato consiste em uma solução de glicose para manutenção do metabolismo microbiano. Por sua vez, o bioetanol representa o produto de interesse, e simultaneamente, ocupa o papel de inibidor dos processos enzimáticos dos micro-organismos de trabalho (MIRLEKAR et al., 2017).

Embora o processo de produção de bioetanol em escala industrial utilize-se tipicamente da ação da *Saccharomyces cerevisiae*, o micro-organismo *Z. mobilis* apresenta diversas características de interesse que o tornam apto a este processo biotecnológico, dentre os quais podem ser citados a grande superfície celular específica, bem como uma rota metabólica diferenciada para a produção etanologênica, a rota Entner-Doudoroff (ENTNER; DOUDOROFF, 1952), que a torna parcialmente desacoplada do crescimento celular. Esse fato por consequência implica que mais substrato pode ser utilizado na produção do bioetanol e então, repercutindo em um aumento no rendimento observado na produção do biocombustível (XIA et al., 2019; YANG et al., 2016). Na Figura 48, uma representação simplificada do metabolismo etanologênico dos referidos micro-organismos é apresentada esquematicamente. Na referida imagem, em destaque, no metabolismo da *Z. mobilis*, é possível observar o parcial desacoplamento da produção de bioetanol da produção de biomassa

celular, aspecto não observado no metabolismo da *S. cerevisiae*. A sigla KDPG representa o composto 2-ceto-3-desoxi-6-fosfogluconato, um metabólito central para a rota Entner-Doudoroff.

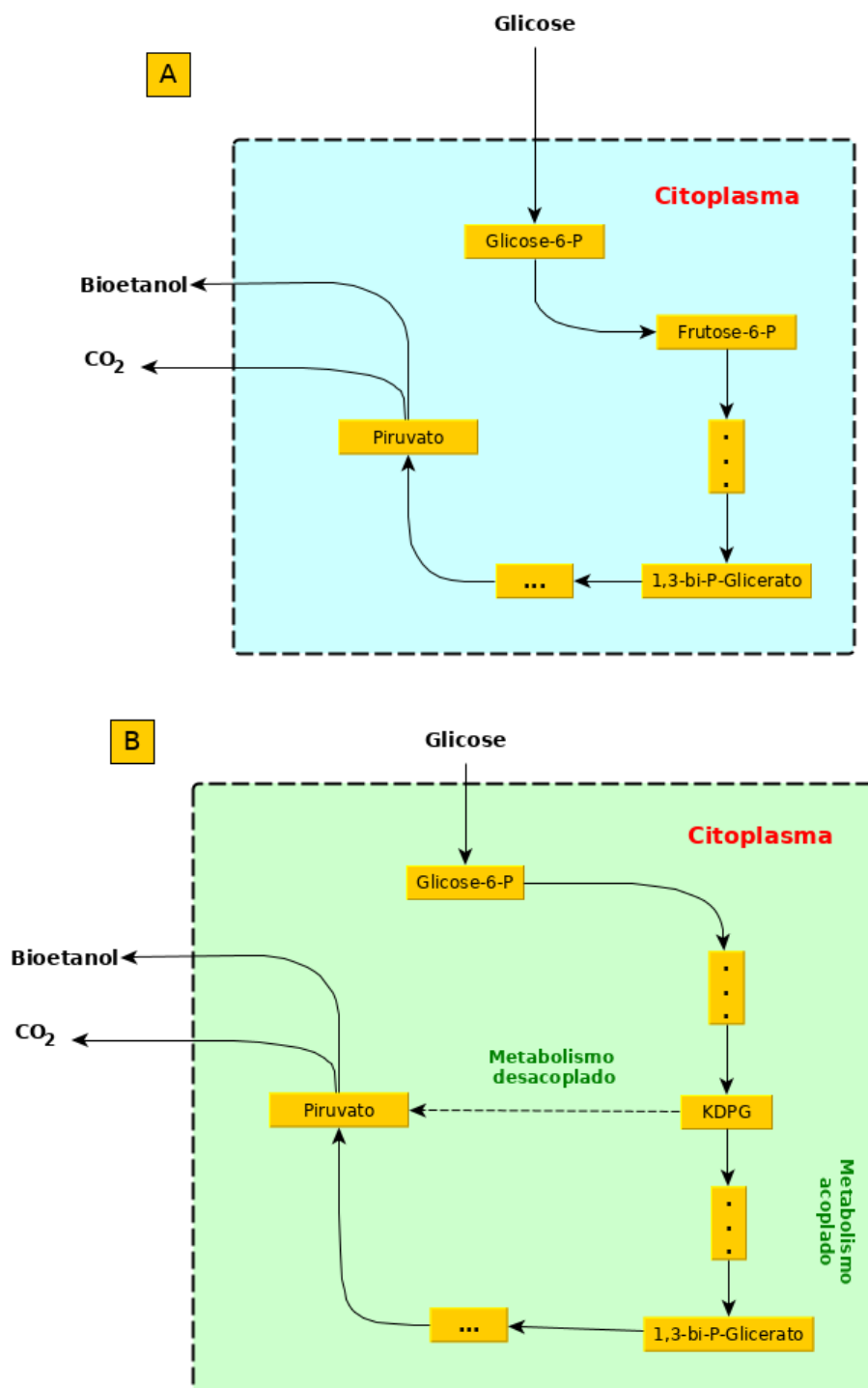


Figura 48 – Comparação de rotas metabólicas (simplificadas) envolvidas no metabolismo da produção do bioetanol, para os micro-organismos *Saccharomyces cerevisiae* (A) e *Zymomonas mobilis* (B).

Fonte – Adaptado de Xia et al. (2019).

De modo geral, em se tratando dos sistemas de produção de bioetanol, a inibição pelo produto de interesse da fermentação envolve dois aspectos: o histórico da concentração de produto e a taxa com que ocorre essa mudança (BAI et al., 2008). A configuração operacional de fermentador contínuo, ou seu substituto frequentemente encontrado na indústria de um conjunto de tanques em série, mostra-se como uma alternativa fiável para a produção de biocombustíveis em larga escala para elevada produtividade (BAI et al., 2008; XIA et al., 2019). Nesse sentido, fica clara a importância do estudo de caso da implementação de um sistema de controle na produção de bioetanol por meio de fermentação contínua, conforme será apresentado a seguir.

6.1.1 Modelagem do problema

O modelo empregado no presente caso de estudo é apresentado no trabalho de Mirlekar et al. Mirlekar et al. (2017), contemplando um reator continuamente agitado, a partir do qual será produzido bioetanol. O modelo é descrito por meio das Equações 22- 25, a seguir.

$$\frac{dS}{dt} = \frac{-Y_x}{Y_{sx}} \frac{SE}{(K_s + S)} - m_s X + D_{in} S_0 - D_{out} S \quad (22)$$

$$\frac{dX}{dt} = Y_x \frac{SE}{(K_s + S)} + D_{in} X_0 - D_{out} X \quad (23)$$

$$\frac{dE}{dt} = (k_1 - k_2 P + k_3 P^2) \frac{SE}{(K_s + S)} + D_{in} E_0 - D_{out} E \quad (24)$$

$$\frac{dP}{dt} = \frac{Y_x}{Y_{px}} \frac{SE}{(K_s + S)} + m_p X + D_{in} P_0 - D_{out} P \quad (25)$$

Nas Equações 22-25, os termos X , P , E , S representam respectivamente a concentração de biomassa, do produto de interesse (bioetanol), componentes-chave para o metabolismo do micro-organismo de trabalho – referido como RNA e proteínas no trabalho de Sridhar (2011) – e substrato, todos com unidade de $kg\ m^{-3}$. O termo D_{in} representa a taxa de diluição da alimentação do biorreator, com unidades de h^{-1} , o qual pode ser definido como a relação entre a vazão de alimentação do fermentador e o seu volume operacional. Os demais parâmetros do modelo e suas condições iniciais são apresentados na Tabela 11. Durante as simulações dinâmicas do problema, assume-se que as condições iniciais do sistema correspondem aos parâmetros da corrente de alimentação.

Tabela 11 – Parâmetros e condições iniciais do modelo para a produção de bioetanol, por meio de fermentação contínua, utilizando a *Zymomonas mobilis* como micro-organismo de trabalho (MIRLEKAR et al., 2017).

Parâmetro	Unidade	Definição	Valor
Y_{sx}	$kg\ kg^{-1}$	Fator de rendimento baseado em substrato	0,0244
Y_{px}	$kg\ kg^{-1}$	Fator de rendimento baseado em produto	0,0526
K_s	$kg\ m^{-3}$	Constante de Monod	0,5
m_s	h^{-1}	Fator de manutenção baseado em substrato	2,16
m_p	h^{-1}	Fator de manutenção baseado em produto	1,1
D_{out}	h^{-1}	Taxa de diluição na saída do fermentador	0,5
k_1	h^{-1}	Constante empírica	16
k_2	$m^3\ kg^{-1}\ h^{-1}$	Constante empírica	0,497
k_3	$m^6\ kg^{-2}\ h^{-1}$	Constante empírica	0,0038
S_0	$kg\ m^{-3}$	Concentração de substrato na alimentação	150,3
P_0	$kg\ m^{-3}$	Concentração de produto na alimentação	0
E_0	$kg\ m^{-3}$	Concentração de componentes-chave na alimentação	0,02
X_0	$kg\ m^{-3}$	Concentração de biomassa na alimentação	0,08
Y_x	h^{-1}	Taxa específica de crescimento máxima	1

6.2 Métodos utilizados

6.2.1 Simulação dinâmica do problema

A simulação dinâmica do sistema de fermentação em tela no presente caso de estudo foi realizada a partir da escolha de diversos valores para o fator de diluição na alimentação do biorreator (D_{in}) e concentração de substrato na alimentação (S_0), analisando os impactos dos diferentes valores para este parâmetro na resposta do processo, conforme apresentado no trabalho de Mirlekar et al. (2017). Em seu trabalho, os autores esclarecem que essa análise é de suma importância, haja vista que podem ocorrer fatores indesejáveis em termos do desempenho e estabilidade do sistema, tais como oscilações e multiplicidade de estados estacionários, os quais são suscitados também no trabalho de Sridhar (2011). Assim, foram escolhidos quatro cenários diferentes para esses parâmetros: valores de $D_{in} \in [0,05; 0,1; 0,2; 0,5]$, e valores de $S_0 \in [140,3; 145,3; 150,3; 160,3]$, determinados de forma empírica. Todos os cenários empregaram as condições iniciais para as variáveis do modelo descritas na Tabela 12. Convém notar que foram utilizadas os mesmos valores daqueles defi-

nidos para a concentração de biomassa, bioetanol, substrato e componentes-chave para o metabolismo celular presentes na corrente de alimentação do biorreator, apresentadas na Tabela 11.

Tabela 12 – Concentração inicial para as variáveis do modelo de produção de bioetanol por meio de fermentação contínua, utilizando a cultura de *Zyomonas mobilis* (MIRLEKAR et al., 2017).

Parâmetro	Unidade	Definição	Valor
X_0	$kg\ m^{-3}$	Concentração inicial de biomassa celular no interior do fermentador	0,08
S_0	$kg\ m^{-3}$	Concentração inicial de substrato no interior do fermentador	150,3
P_0	$kg\ m^{-3}$	Concentração inicial de bioetanol no interior do fermentador	0
E_0	$kg\ m^{-3}$	Concentração inicial de componentes-chave no interior do fermentador	0,02

A partir das configurações definidas anteriormente, foi realizado o estudo de simulação dinâmica do problema, utilizando a ferramenta `sloth`. Para tanto, o *solver* do sistema diferencial empregado foi o *odeint*, da biblioteca `scipy` (JONES et al., 2001). Conforme já foi mencionado, esse *solver* realiza uma interface com o algoritmo LSODA da biblioteca *ODEPACK* (HINDMARSH, 1983). Convém mencionar que também foi avaliado o outro integrador disponível na versão atual da ferramenta `sloth`, o *CVODE* da biblioteca `assimulo` (ANDERSSON et al., 2015), contudo, os resultados obtidos foram bastante similares, como será discutido na seção 6.3.

6.2.2 Problema de controle

O controle ótimo da trajetória de produção do processo previsto em termos de um modelo matemático descritivo constitui o núcleo das abordagens de controle baseadas em modelos, dos quais podemos citar a otimização dinâmica em tempo real (DRTO, do inglês *Dynamic Real-Time Optimization*), também referida como controle preditivo baseado em modelo não-linear orientado economicamente (ENLMPC, do inglês *Economically-oriented Non-Linear Model Predictive Control*), que consiste em uma estratégia eficiente para o controle de sistemas com natureza intrinsecamente dinâmica, a exemplo dos processos biotecnológicos (ŞENDRESCU, 2011; ŞENDRESCU et al., 2011). A otimização dinâmica em tempo real (DRTO) é um membro de uma ampla gama de ferramentas avançadas de controle de processos, que contabilizam características intrínsecas transitórias (preços de mercado voláteis, distúrbios de entrada, etc.) na maximização (ou minimização) de um função

objetiva, geralmente orientada em torno do desempenho de processo em termos de um produto de interesse (FREITAS et al., 2017). Assim, o problema de controle pode ser representado a partir das Equações 26- 30.

$$\min J(x,u,t) \quad (26)$$

sujeito a:

$$\dot{x} = f(x(t),u(t)) \quad (27)$$

$$u_{min} \leq u \leq u_{max} \quad (28)$$

$$t \in [t_k, t_{k+1}] \quad \forall k \in [1,2,\dots,n] \quad (29)$$

$$x_{min} \leq x \leq x_{max} \quad (30)$$

Na Figura 49, o fluxograma do operação para o problema de controle em do bioprocesso referido no presente caso de estudo é apresentado, no qual deseja-se fazer com que a resposta do sistema aproxime-se de um *set-point* desejado.

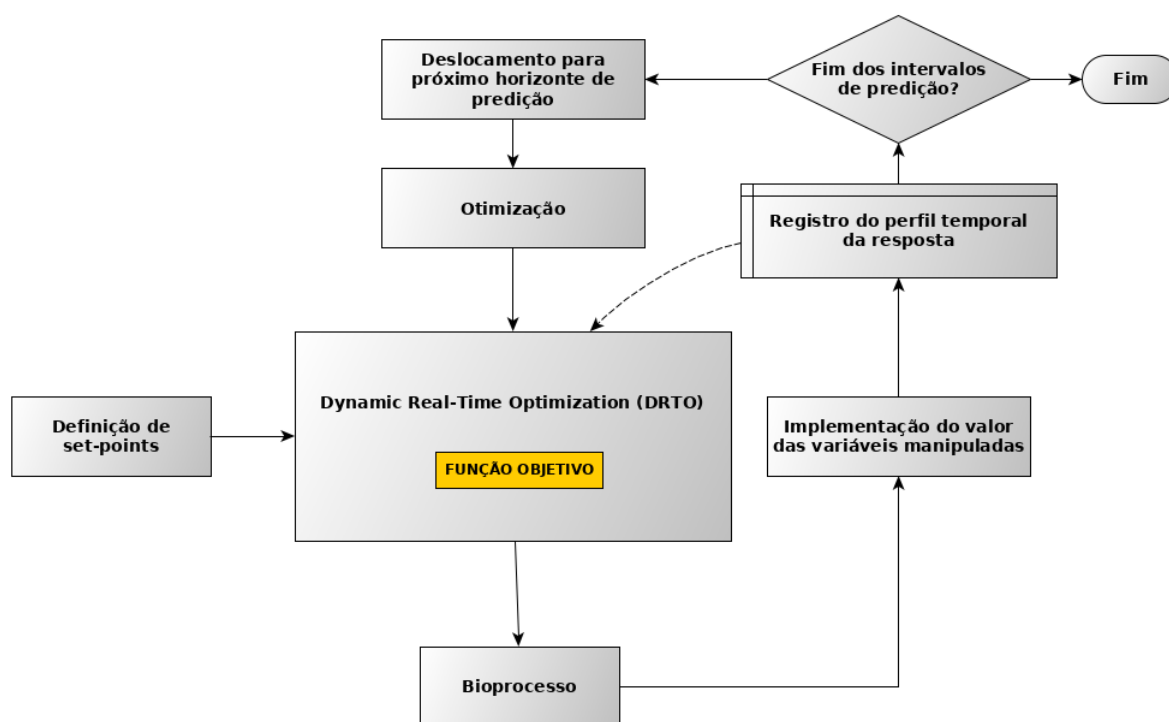


Figura 49 – Fluxograma de operação para o problema de controle de um bioprocesso por meio de DRTO, para o problema de busca por *set-point*.

O termo J , o qual figura na Equação 26, representa uma função objetivo a ser minimizada para cada um dos n intervalos de tempo em que o tempo de processo será dividido. No presente estudo, será adotada uma função que objetiva levar a concentração de bioetanol a um patamar desejado a partir da alteração da taxa de diluição na alimentação do biorreator, um problema referido como busca por *set-point* (ou, no inglês, *set-point tracking*). Portanto, a concentração de bioetanol no fermentador (P) figura como variável controlada, e a taxa de diluição na alimentação (D_{in}) como variável manipulada. Nesse sentido, a referida expressão matemática pode ser representada pela Equação 31 (MIRLEKAR et al., 2017), na qual o termo $P(t)$ representa a concentração de produto em um dado instante de tempo, e P_{sp} o *set-point* desejado. Essa função corresponde ao critério ISE (*Integral Square Error*, ou integral do erro quadrático), uma métrica frequentemente utilizada em estudos de desempenho de sistemas de controle (HOWITT; LUUS, 1990; KEALY; O'DWYER, 2003).

$$J = \min \int_0^t [P(t) - P_{sp}]^2 dt \quad (31)$$

Para o referido caso, em que desconsidera-se a existência de uma fonte de distúrbio, o problema de DRTO reduz-se a um problema de otimização da variável manipulada para cada um dos n intervalos em que o tempo total de batelada foi dividido, duração esta que corresponde a $20 h$ para o presente estudo. Foram avaliados três diferentes valores para o número de intervalos: 20, 40 e 80. Esses valores implicam que o valor da variável manipulada será reavaliado pelo algoritmo de otimização com diferentes frequências, respectivamente: $1 h$, $0,5 h$ e $0,25 h$. Na realização do estudo de controle descrito, foi definido um problema na ferramenta `sloth`, o qual iniciava uma instância de uma das rotinas de otimizações inclusas no *software* escolhidas, para cada um dos intervalos de tempo. Foi utilizado o algoritmo de *evolução diferencial*, o qual utiliza as funções definidas na biblioteca `pagmo` para seu funcionamento. As configurações definidas para esta rotina são apresentadas na Tabela 13, as quais foram determinadas de maneira empírica.

Tabela 13 – Configuração empregada no algoritmo DE, para a resolução dos problemas de otimização oriundos do DRTO.

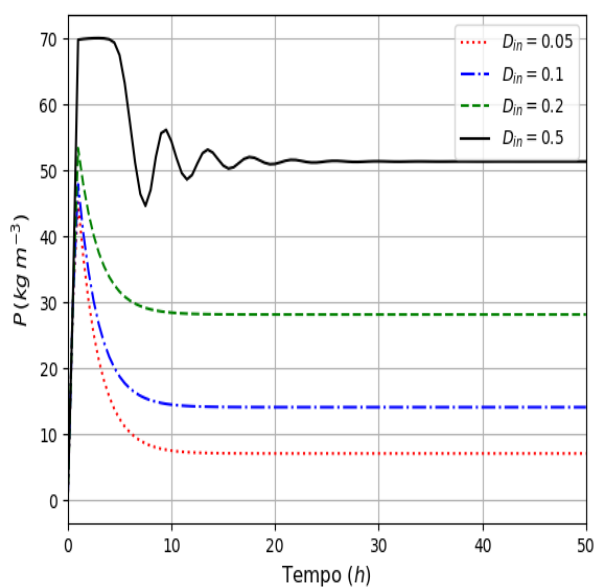
Parâmetro	Valor
Probabilidade de <i>crossover</i>	80%
Fator de escala	0.5 (constante)
Variação do algoritmo DE	DE/best/2/bin
Número de soluções candidatas	20
Critério de parada	Número de gerações (100)

6.3 Resultados e discussões

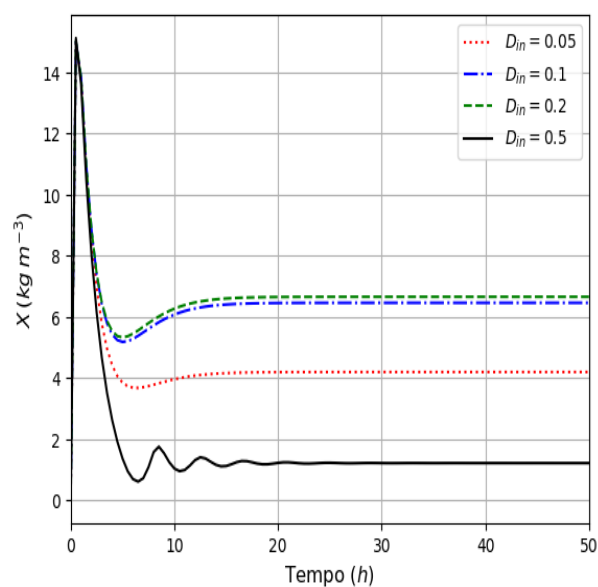
Na presente seção, serão apresentados os resultados obtidos a partir da simulação dinâmica do modelo do processo de produção de bioetanol *in-silico* por meio de fermentação contínua a partir da cultura de *Zymomonas mobilis*. Serão apresentados também os resultados obtidos a partir de um estudo de controle do processo, utilizando a abordagem de DRTO.

6.3.1 Simulação dinâmica do problema

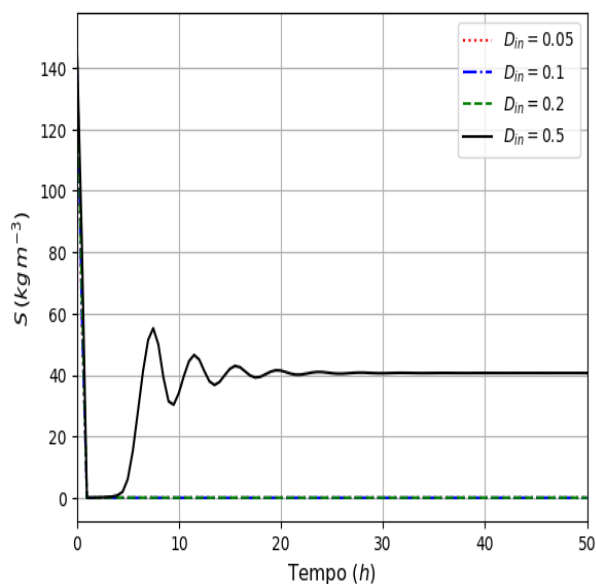
Os resultados obtidos para o estudo de simulação dinâmica para os cenários relativos a variação nos parâmetros de taxa de diluição da alimentação do biorreator (D_{in}) e concentração de substrato na alimentação do biorreator (S_0) são apresentados nas Figuras 50 e 51, respectivamente.



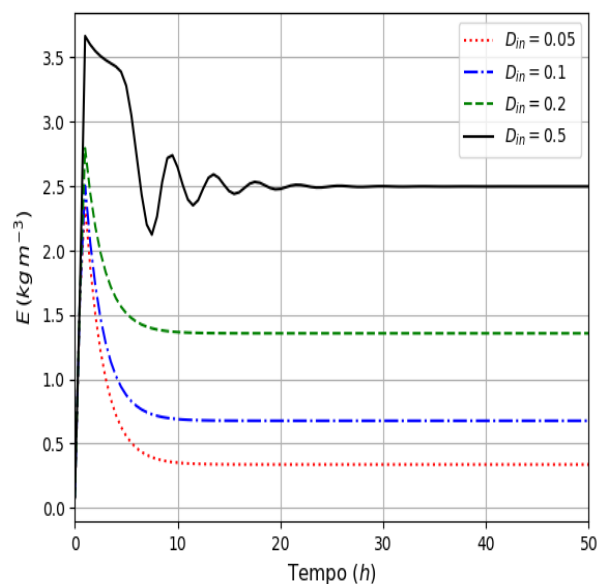
(a)



(b)



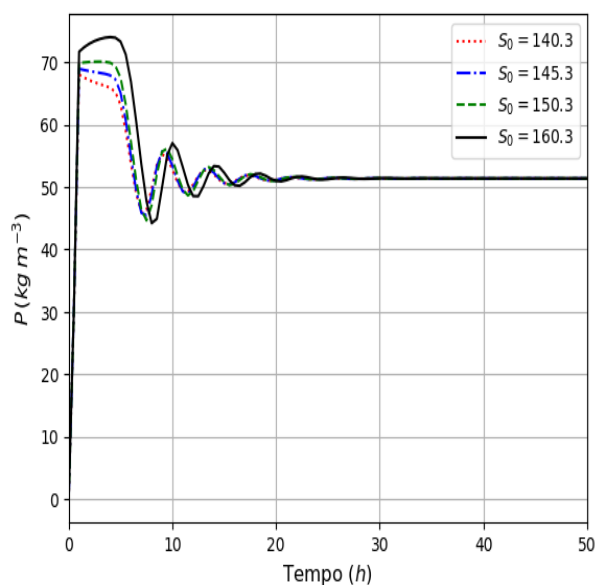
(c)



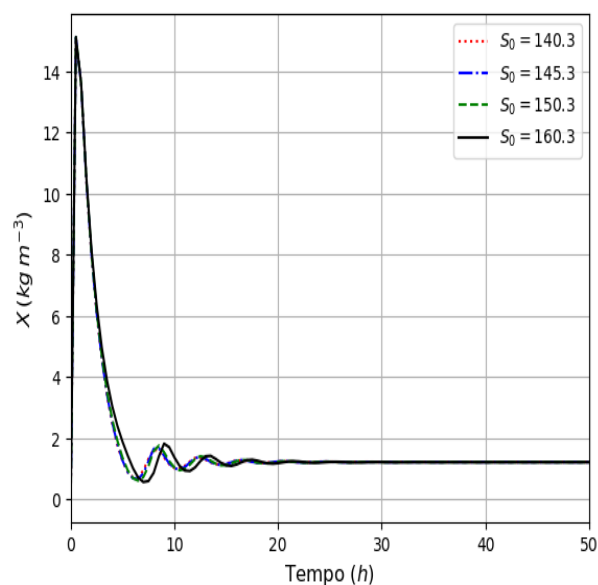
(d)

Figura 50 – Resultado obtidos para a simulação dinâmica do modelo de produção de bioetanol pela em fermentação contínua, em termos do perfil temporal da concentração do produto (50a), concentração de biomassa (50b), concentração de substrato (50c) e concentração de componentes-chave (50d). Todos os resultados foram obtidos para um valor de concentração de substrato na alimentação do biorreator (S_0) equivalente a $150,3 \text{ kg m}^{-3}$.

Fonte – Próprio autor.



(a)



(b)

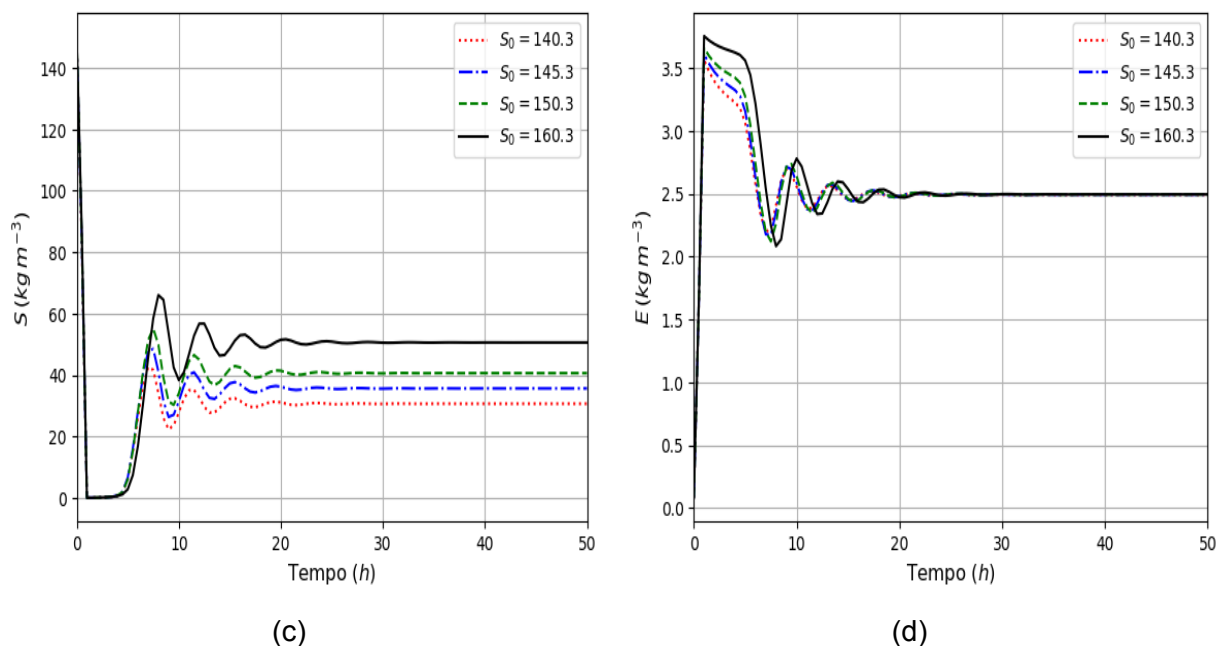


Figura 51 – Resultado obtidos para a simulação dinâmica do modelo de produção de bioetanol pela em fermentação contínua, em termos do perfil temporal da concentração do produto (51a), concentração de biomassa (51b), concentração de substrato (51c) e concentração de componentes-chave (51d). Todos os resultados foram obtidos para um valor de taxa de diluição na alimentação do biorreator (D_{in}) equivalente a $0,5\ h^{-1}$.

Fonte – Próprio autor.

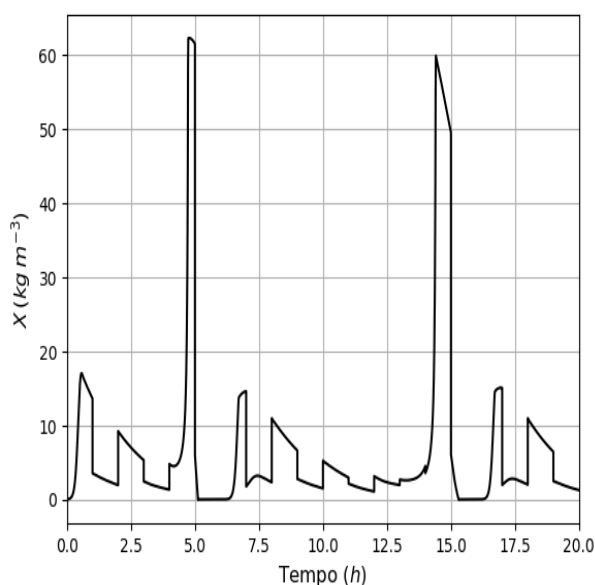
A partir da análise dos resultados apresentados na Figura 50, é possível notar que o parâmetro taxa de diluição na alimentação do biorreator (D_{in}) afeta sensivelmente a estabilidade da resposta, especialmente no que tange a concentração de bioetanol no interior do biorreator (P). Convém mencionar que os resultados divergem significativamente daqueles apresentados no trabalho de Mirlekar et al. (2017) para os valores de D_{in} inferiores a 0,5. Esse pode ser creditado à performance da rotina computacional utilizada para a resolução do modelo diferencial por meio da ferramenta `sloth`, a qual corresponde à `LSODA` no presente trabalho (por meio da interface provida pela biblioteca `scipy`, como já mencionado). Resultados semelhantes foram observados com o emprego da rotina `CVODE` (por meio da biblioteca `assimulo`, como já mencionado). Os dois `solvers` retornaram diversos erros de falha de convergência, fato endossado pela natureza altamente não-linear do modelo em estudo. Assim, sugere-se a necessidade de se estudar o emprego de algoritmos mais robustos para a resolução de problemas diferenciais na ferramenta `sloth` em suas vindouras versões futuras. Contudo, convém mencionar que os resultados obtidos para um valor de D_{in} igual a 0,5 mostraram-se bastante acurados quando comparados aos do trabalho já referido. A existência de bifurcações, oscilações e múltiplos estados estacionários desse modelo em particular é discutida nos traba-

Ihos de Sridhar (2011) e Mahecha-Botero et al. (2005), entre outros.

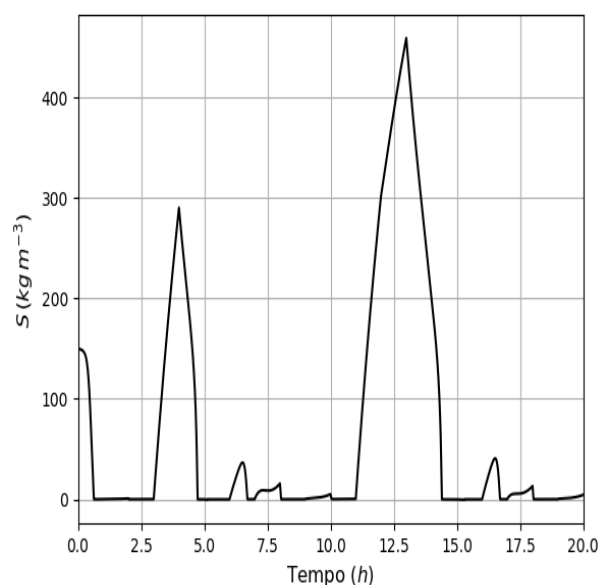
Quanto ao parâmetro concentração de substrato na corrente de alimentação do biorreator (S_0), a partir da análise dos resultados apresentados na Figura 51, é possível notar que a variação de S_0 pouco influencia nos resultados, sendo observada apenas uma pequena variação as respostas quanto ao perfil temporal de substrato (Figura 51c), mas para todas as demais variáveis de resposta do modelo (X, P, E), nota-se que os ensaios para diferentes valores de S_0 convergem praticamente para o mesmo estado estacionário.

6.3.2 Controle da produção de bioetanol

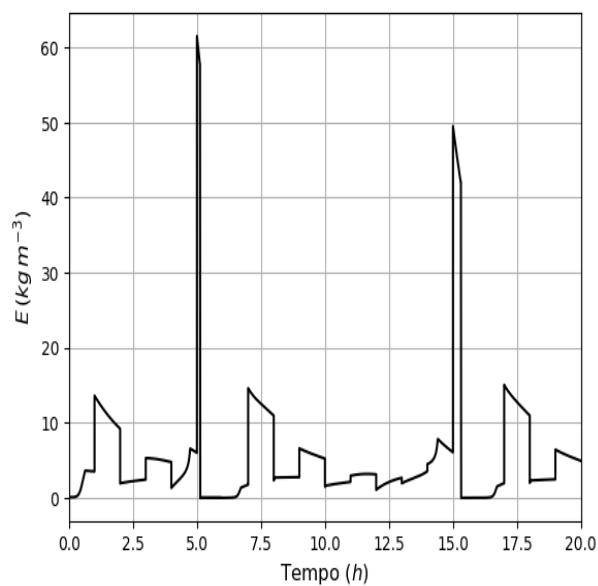
Os resultados obtidos para o estudo de controle de produção do bioetanol por meio de DRTO são apresentados nas Figuras 52-54, nas quais são apresentados os resultados obtidos para as variáveis de resposta do modelo (X, S e E) excetuando-se a variável controlada, para os três valores de frequência de ativação da otimização no algoritmo de DRTO, a saber: $1 h$, $0,5 h$ e $0,25 h$. Na Figura 55, o perfil temporal da variável controlada (P) é comparado para os três valores de frequência de ativação da otimização no controle do processo *in-silico* já mencionados, com destaque para o valor do *set-point* utilizado no presente estudo de controle, conforme apresentado na Equação 31.



(a)



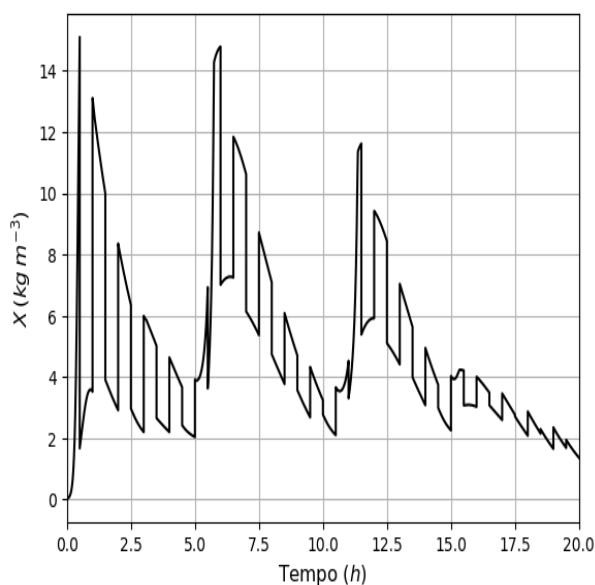
(b)



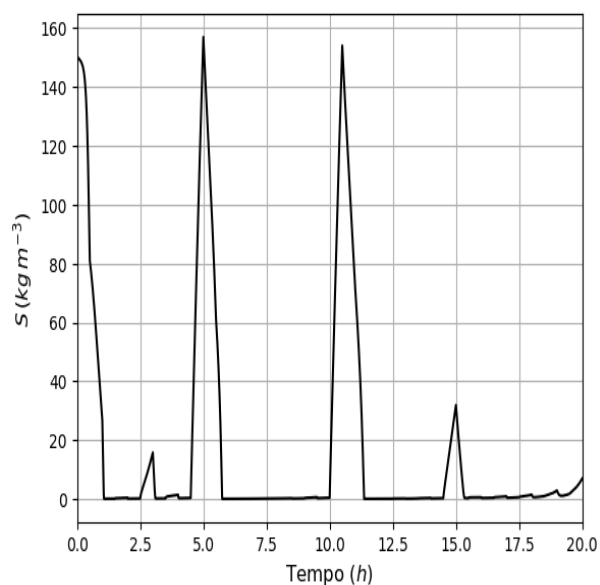
(c)

Figura 52 – Resultados obtidos para o estudo de controle do processo de produção de bioetanol *in-silico* por meio de fermentação contínua, utilizando a abordagem DRTO. Foram empregados 20 intervalos para a otimização da variável manipulada ao longo dos horizontes de predição, correspondendo a uma duração para o horizonte de predição de 1 h. Nas Figuras 52a, 52b e 52c são retratados respectivamente os perfis temporais de concentração de biomassa celular (X), substrato (S) e componentes-chave para o metabolismo celular (E).

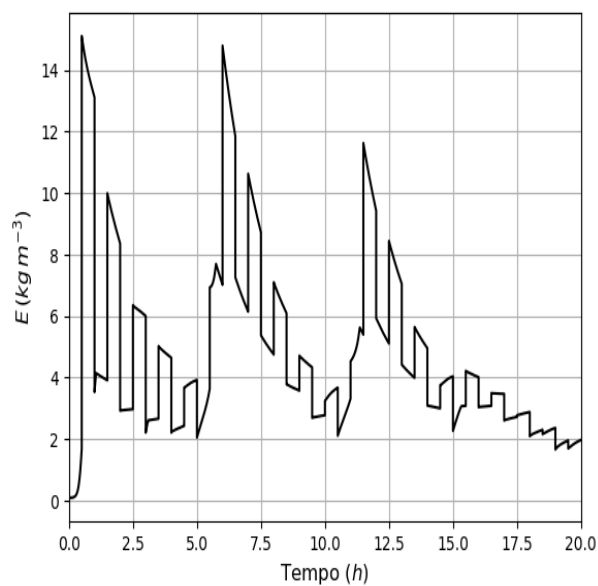
Fonte – Próprio autor.



(a)



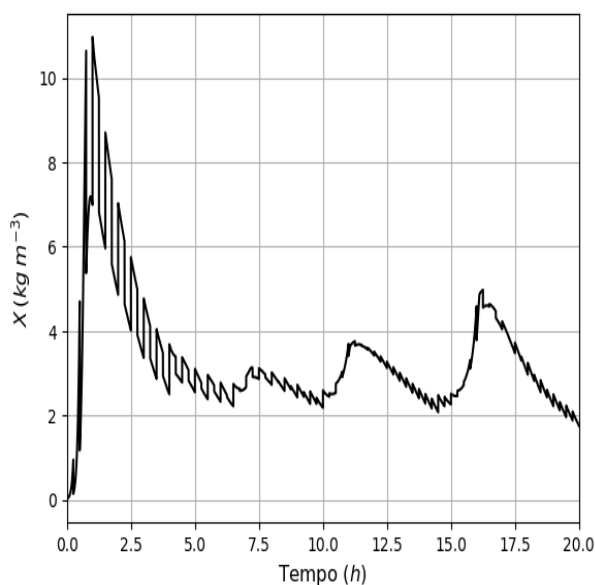
(b)



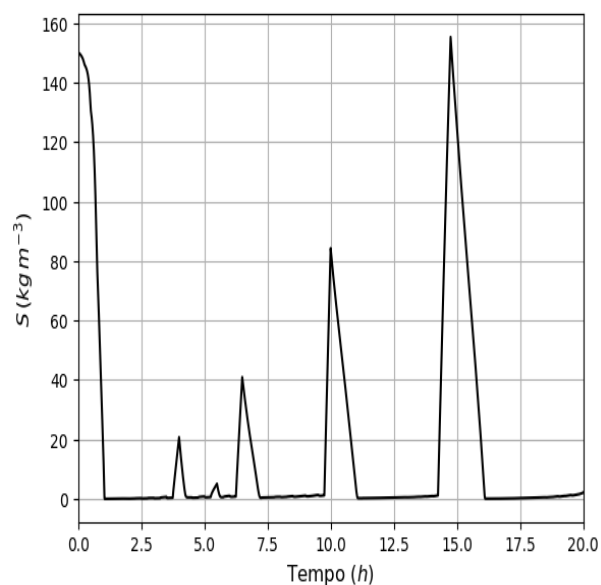
(c)

Figura 53 – Resultados obtidos para o estudo de controle do processo de produção de bioetanol *in-silico* por meio de fermentação contínua, utilizando a abordagem DRTO. Foram empregados 40 intervalos para a otimização da variável manipulada ao longo dos horizontes de predição, correspondendo a uma duração para o horizonte de predição de 0,5 h (trinta minutos). Nas Figuras 53a, 53b e 53c são retratados respectivamente os perfis temporais de concentração de biomassa celular (X), substrato (S) e componentes-chave para o metabolismo celular (E).

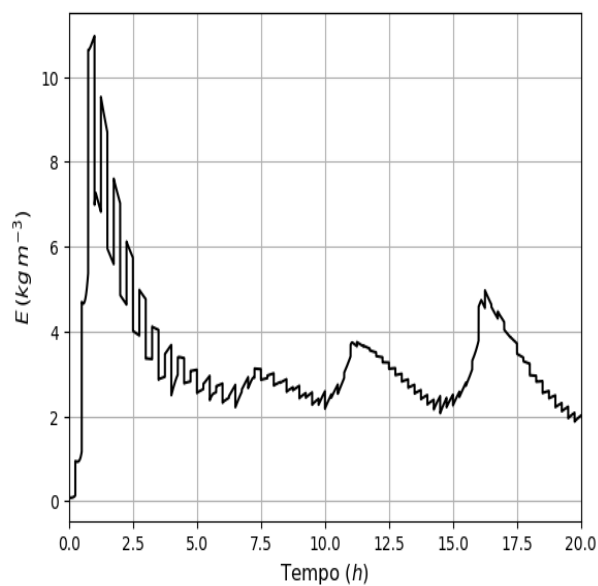
Fonte – Próprio autor.



(a)



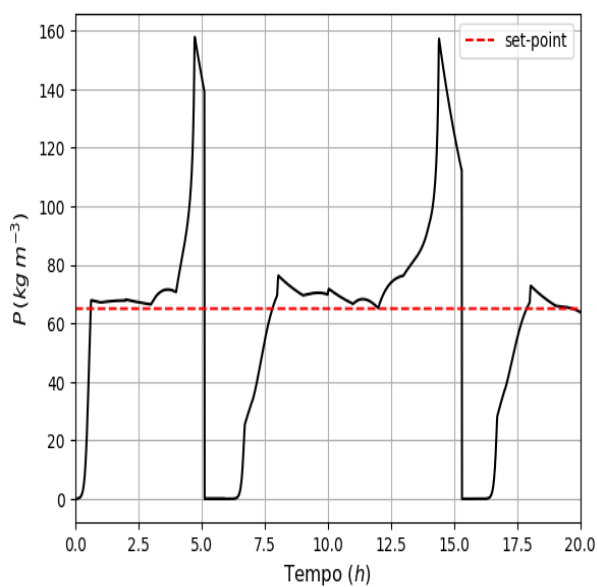
(b)



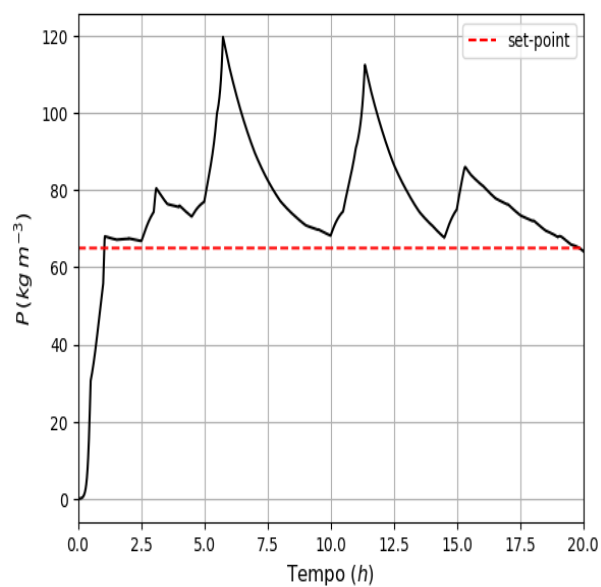
(c)

Figura 54 – Resultados obtidos para o estudo de controle do processo de produção de bioetanol *in-silico* por meio de fermentação contínua, utilizando a abordagem DRTO. Foram empregados 80 intervalos para a otimização da variável manipulada ao longo dos horizontes de previsão, correspondendo a uma duração para o horizonte de previsão de 0,25 h (15 minutos). Nas Figuras 54a, 54b e 54c são retratados respectivamente os perfis temporais de concentração de biomassa celular (X), substrato (S) e componentes-chave para o metabolismo celular (E).

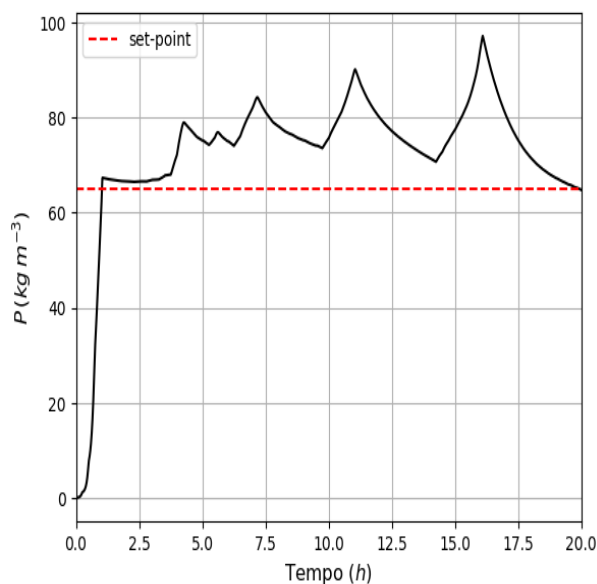
Fonte – Próprio autor.



(a)



(b)



(c)

Figura 55 – Resultados obtidos para os perfis temporais da variável controlada, a concentração de bioetanol (P), utilizando a abordagem DRTO, com destaque para o valor do *set-point* estabelecido para a mesma ($P_{sp} = 65 \text{ kg m}^{-3}$). Foram empregados 20, 40 e 80 intervalos para a otimização da variável manipulada ao longo dos horizontes de predição, correspondendo a uma duração para o horizonte de predição de 1 h (60 minutos), 0,5 h (30 minutos) e 0,25 h (15 minutos), retratados respectivamente nas Figuras 55a, 55b e 55c.

Fonte – Próprio autor.

A partir dos resultados apresentados na Figura 55, é possível notar a influência que o número de intervalos exerce sobre a resposta do sistema simulado em termos da concentração de bioetanol, mostrando que o ensaio de DRTO com uma frequência de ativação do algoritmo de otimização da variável manipulada (D_{in}) correspondente a 0,25 h (ou em outras palavras, o tempo de processo dividido em 80 horizontes de predição), a qual pode ser visualizada na Figura 55c, exibe uma maior estabilidade do perfil da variável controlada (P) em torno do *set-point* desejado ($P = 65 \text{ kg m}^{-3}$). Contudo, ainda que a variável controlada seja mantida acima do patamar desejado da concentração do produto de interesse no interior do biorreator, observa-se significativa oscilação na resposta do sistema. Esse fato corrobora com o comportamento sugerido nos trabalhos de diversos autores, os quais enfatizam a ocorrência sistemática de severas oscilações na resposta do sistema, em razão de sua natureza numérica intrínseca (SRIDHAR, 2011; MAHECHA-BOTERO et al., 2005; MIRLEKAR et al., 2017).

Entretanto, convém mencionar que as condições iniciais para o problema de controle foram as mesmas daquelas utilizadas na simulação dinâmica, apre-

sentadas na Tabela 12. No trabalho de Mirlekar et al. (2017), os autores utilizaram uma concentração inicial diferenciada em termos da concentração de bioetanol e de componentes-chave no interior do fermentador, bem como a sua concentração na corrente de alimentação do equipamento. De acordo com Mirlekar et al., essa configuração diferenciada contribui para a redução nas oscilações observadas para a resposta. Não obstante, é necessário mencionar que o algoritmo de DRTO mostrou-se demasiadamente lento em obter o resultado do ensaio de controle, atingindo cerca de 24 h de tempo computacional (em um computador com um processador *Intel i5* de 3 Ghz e 8 Gb de RAM) para o cenário em que foi utilizado uma duração do horizonte de predição de 0,25 h (ou 80 intervalos). A utilização de algoritmos de otimização mais robustos em vindouras versões futuras da ferramenta `sloth`, que explorem recursos como computação paralela, podem vir a reduzir significativamente esse tempo, especialmente quando tratar-se de problemas com maior dimensionalidade.

Os resultados apresentados nas Figuras 52-54 indicam que para o estudo de controle por meio de DRTO realizado, com uma duração do horizonte de predição de 0,25 h, as oscilações observadas para a resposta das variáveis X e E são significativamente reduzidas. Apesar desse fato, aumentam o número de oscilações na resposta de S , ocorrendo justamente nos intervalos de tempo em que notam-se as oscilações agressivas na resposta de P , conforme pode ser visto na Figura 55. Esse fato endossa a conclusão já estabelecida de que o algoritmo de integração do problema diferencial não foi robusto o suficiente para esse problema em questão, sendo importante analisar a adoção de outras rotinas nas versões futuras da ferramenta `sloth`.

CAPÍTULO 7

Conclusão

No presente capítulo deste trabalho, serão apresentadas as conclusões gerais do trabalho, bem como serão apresentados perspectivas para trabalhos futuros.

7.1 Conclusão geral

A presente proposta abordou o desenvolvimento de uma ferramenta orientada a equações para modelagem e simulação de processos, a qual foi empregada no desenvolvimento de estudos de otimização e controle aplicados a processos biotecnológicos, dentre os quais esse trabalho voltou-se especificamente aos processos fermentativos para produção de bioetanol. Conforme apresentado, a ferramenta `sloth` foi capaz de fornecer a estrutura básica para o equacionamento dos modelos, sua simulação e estudos de controle.

A metodologia utilizada no desenvolvimento da ferramenta foi apresentada, contribuindo com o seu propósito que não é o de rivalizar com *softwares* livres análogos a sua proposta, entre os quais podemos citar o EMSO, GAMS, DAETOOLS, ASCEND. No entanto, a proposta do presente trabalho é o desenvolvimento de uma ferramenta que tenha o cunho educacional e/ou sirva como fundamentação para o desenvolvimento de outras ferramentas orientadas a equações para modelagem e simulação de processos, sendo o seu código disponibilizado na plataforma *online* Github. A escolha da linguagem computacional empregada no desenvolvimento da ferramenta `sloth`, o Python, corrobora com o já referido propósito, sendo de fácil compreensão, possuindo um grande arsenal de bibliotecas para os mais diferentes propósitos, e podendo ser associada a outras linguagens computacionais quando necessário.

Os resultados obtidos para os casos de estudo I e II – respectivamente, a otimização em malha aberta de um processo de produção de bioetanol em batelada alimentada por *S. cerevisiae*, e o controle do processo de produção de bioetanol por meio de fermentação contínua mediante o cultivo de *Z. mobilis*, utilizando a abordagem DRTO – apontam que a ferramenta mostrou-se adequada ao seu propósito, ainda que sejam apontados alguns aspectos que carecem de melhora no *software* em suas versões futuras. A partir dos casos de estudo realizados, foi possível perceber que robustez dos algoritmos de integração utilizada na solução dos problemas diferenciais precisa ser melhorada, a partir da adoção de rotinas mais estáveis e eficientes, bem como a importância da utilização de computação paralela para a redução do tempo necessário para a resposta da ferramenta, especialmente para os estudos de controle.

7.2 Perspectivas para trabalhos futuros

Serão apontadas a seguir as perspectivas para trabalhos futuros, delineadas a partir do que foi exposto no presente trabalho.

- Desenvolvimento de uma interface gráfica para a ferramenta `sloth`, facilitando o equacionamento de modelos, bem como a entrada de dados
- Adoção de rotinas termodinâmicas a serem disponibilizadas na ferramenta, de modo que operações unitárias de interesse da engenharia de processos que façam uso de princípios termodinâmicos, como a destilação, sejam disponibilizadas na forma de modelos pré-definidos para a utilização do usuário
- Reestruturação das rotinas de integração utilizadas para a resolução de problemas diferenciais, de modo que estas apresentem maior robustez no cálculo de sistemas altamente não-lineares, a exemplo dos processos biotecnológicos em geral
- Ampliação do rol de rotinas de otimização disponíveis para o usuário para além do pequeno conjunto de rotinas meta-heurísticas disponibilizado na versão da ferramenta apresentada neste trabalho
- Desenvolvimento de uma interface efetiva do núcleo da ferramenta com linguagens de médio baixo nível, como *Fortran*, *C* ou *C++*, com o intuito de acelerar o processamento da ferramenta, implicando em um menor tempo para obtenção das respostas
- Adoção do paradigma de computação paralela com o intuito de acelerar o processamento da ferramenta, implicando em um menor tempo para obtenção das respostas, especialmente no que tange os estudos de otimização e controle.

Referências

- ADITIYA, H. et al. Second generation bioethanol production: A critical review. *Renewable and sustainable energy reviews*, Elsevier, v. 66, p. 631–653, 2016.
- ANDERSSON, C.; FÜHRER, C.; ÅKESSON, J. Assimulo: A unified framework for ODE solvers. *Mathematics and Computers in Simulation*, v. 116, n. 0, p. 26 – 43, 2015. ISSN 0378-4754.
- APEL, A. C.; WEUSTER-BOTZ, D. Engineering solutions for open microalgae mass cultivation and realistic indoor simulation of outdoor environments. *Bioprocess and biosystems engineering*, Springer, v. 38, n. 6, p. 995–1008, 2015.
- ARNELL, M. et al. Multi-objective performance assessment of wastewater treatment plants combining plant-wide process models and life cycle assessment. *Journal of Water and Climate Change*, IWA Publishing, v. 8, n. 4, p. 715–729, 2017.
- AWG-ADENI, D. S. et al. Recovery of glucose from residual starch of sago ham-pas for bioethanol production. *BioMed research international*, Hindawi, v. 2013, 2013.
- AZHAR, S. H. M. et al. Yeasts in sustainable bioethanol production: a review. *Biochemistry and Biophysics Reports*, Elsevier, v. 10, p. 52–61, 2017.
- BAEYENS, J. et al. Challenges and opportunities in improving the production of bio-ethanol. *Progress in Energy and Combustion Science*, Elsevier, v. 47, p. 60–88, 2015.
- BAI, F.; ANDERSON, W.; MOO-YOUNG, M. Ethanol fermentation technologies from sugar and starch feedstocks. *Biotechnology advances*, Elsevier, v. 26, n. 1, p. 89–105, 2008.
- BALAT, M. Production of bioethanol from lignocellulosic materials via the biochemical pathway: a review. *Energy conversion and management*, Elsevier, v. 52, n. 2, p. 858–875, 2011.
- BALAT, M.; BALAT, H. Recent trends in global production and utilization of bio-ethanol fuel. *Applied energy*, Elsevier, v. 86, n. 11, p. 2273–2282, 2009.
- BANGA, J. R. Optimization in computational systems biology. *BMC systems biology*, BioMed Central, v. 2, n. 1, p. 1, 2008.
- BANGA, J. R. et al. Dynamic optimization of bioprocesses: Efficient and robust numerical strategies. *Journal of Biotechnology*, Elsevier, v. 117, n. 4, p. 407–419, 2005.
- BANGA, J. R.; SINGH, R. P. Optimization of air drying of foods. *Journal of Food Engineering*, Elsevier, v. 23, n. 2, p. 189–211, 1994.
- BAUGHMAN, D. R.; LIU, Y. A. *Neural networks in bioprocessing and chemical engineering*. Orlando, Estados Unidos da América: Academic press, 1995.
- BELLMAN, R. *Dynamic programming*. Nova Jersey: Princeton University Press, 1957.

- BERGH, F. van den; ENGELBRECHT, A. P. A cooperative approach to particle swarm optimization. *IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION*, IEEE, v. 8, n. 3, p. 225–239, 2004.
- BISCANI, F. et al. *esa/pagmo2: pagmo 2.10*. 2019. Disponível em: <<https://doi.org/10.5281/zenodo.2529931>>.
- BOOCH, G.; RUMBAUGH, J.; JACOBSON, I. *UML: guia do usuário*. Rio de Janeiro: Elsevier Brasil, 2006.
- BORZANI, W. et al. *Biotecnologia Industrial*. São Paulo: Blucher, 2001. v. 1.
- CHANG, W.-D. A modified particle swarm optimization with multiple subpopulations for multimodal function optimization problems. *Applied Soft Computing*, Elsevier, v. 33, p. 170–182, 2015.
- CHEN, C. T.; HWANG, C. Optimal on-off control for fed-batch fermentation processes. *Industrial & engineering chemistry research*, ACS Publications, v. 29, n. 9, p. 1869–1875, 1990.
- CHEN, H.; FU, X. Industrial technologies for bioethanol production from lignocellulosic biomass. *Renewable and Sustainable Energy Reviews*, Elsevier, v. 57, p. 468–478, 2016.
- CHEN, H. et al. Macroalgae for biofuels production: Progress and perspectives. *Renewable and Sustainable Energy Reviews*, Elsevier, v. 47, p. 427–437, 2015.
- CHEN, S.; MONTGOMERY, J.; BOLUFÉ-RÖHLER, A. Measuring the curse of dimensionality and its effects on particle swarm optimization and differential evolution. *Applied Intelligence*, Springer, v. 42, n. 3, p. 514–526, 2015.
- CHOWDHURY, A.; ZOMORRODI, A. R.; MARANAS, C. D. k-optforce: integrating kinetics with flux balance analysis for strain design. *PLoS computational biology*, Public Library of Science, v. 10, n. 2, p. e1003487, 2014.
- CHRISTIAN, D.; BENOÎT, A. et al. Method development and validation of an inline process analytical technology method for blend monitoring in the tablet feed frame using raman spectroscopy. *Analytical chemistry*, American Chemical Society, 2018.
- CLEMENTSCHITSCH, F.; BAYER, K. Improvement of bioprocess monitoring: development of novel concepts. *Microbial cell factories*, BioMed Central, v. 5, n. 1, p. 1, 2006.
- CRAVEN, S. et al. Process model comparison and transferability across bioreactor scales and modes of operation for a mammalian cell bioprocess. *Biotechnology progress*, Wiley Online Library, v. 29, n. 1, p. 186–196, 2013.
- CROUGHAN, M. S.; KONSTANTINOV, K. B.; COONEY, C. The future of industrial bioprocessing: Batch or continuous? *Biotechnology and bioengineering*, Wiley Online Library, v. 112, n. 4, p. 648–651, 2015.
- DAVIDSON, V. Fuzzy control for food processes. In: *Computerized control systems in the food industry*. [S.l.]: Routledge, 2018. p. 179–205.

- DEMIRBAS, A. Biofuels sources, biofuel policy, biofuel economy and global biofuel projections. *Energy conversion and management*, Elsevier, v. 49, n. 8, p. 2106–2116, 2008.
- DIETZSCH, C.; SPADIUT, O.; HERWIG, C. On-line multiple component analysis for efficient quantitative bioprocess development. *Journal of biotechnology*, Elsevier, v. 163, n. 4, p. 362–370, 2013.
- DOCHAIN, D. What are the challenges for the control of bioprocesses? In: DOCHAIN, D. (Ed.). *Automatic control of bioprocesses*. Estados Unidos: John Wiley & Sons, 2008. p. 11–16.
- EGEA, J. A.; MARTÍ, R.; BANGA, J. R. An evolutionary method for complex-process optimization. *Computers & Operations Research*, Elsevier, v. 37, n. 2, p. 315–324, 2010.
- EGEA, J. A. et al. Scatter search for chemical and bio-process optimization. *Journal of Global Optimization*, Springer, v. 37, n. 3, p. 481–503, 2007.
- EIJCK, J. van; BATIDZIRAI, B.; FAAIJ, A. Current and future economic performance of first and second generation biofuels in developing countries. *Applied Energy*, Elsevier, v. 135, p. 115–141, 2014.
- ELSHAGHABEE, F. M. et al. Ethanol production by selected intestinal microorganisms and lactic acid bacteria growing under different nutritional conditions. *Frontiers in microbiology*, Frontiers, v. 7, p. 47, 2016.
- ENITAN, A. M. et al. Optimization of biogas generation using anaerobic digestion models and computational intelligence approaches. *Reviews in Chemical Engineering*, De Gruyter, v. 33, n. 3, p. 309–335, 2017.
- ENTNER, N.; DOUDOROFF, M. Glucose and gluconic acid oxidation of *Pseudomonas saccharophila*. *Journal of Biological Chemistry*, ASBMB, v. 196, n. 2, p. 853–862, 1952.
- FERNANDES, R. L. et al. Applying mechanistic models in bioprocess development. In: MANDENIUS, C.-F.; TITCHENER-HOOKER, N. J. (Ed.). *Measurement, Monitoring, Modelling and Control of Bioprocesses*. Berlim, Alemanha: Springer, 2013. p. 137–166.
- FERNANDES, R. L. et al. Experimental methods and modeling techniques for description of cell population heterogeneity. *Biotechnology advances*, Elsevier, v. 29, n. 6, p. 575–599, 2011.
- FESTEL, G. Industrial biotechnology: Market size, company types, business models, and growth strategies. *Industrial Biotechnology*, Mary Ann Liebert, Inc., v. 6, n. 2, p. 88–94, 2010.
- FESTEL, G.; DETZEL, C.; MAAS, R. Industrial biotechnology-markets and industry structure. *Journal of Commercial Biotechnology*, thinkBiotech LLC, v. 18, n. 1, 2012.
- FOWLER, M.; SCOTT, K. *UML Essencial: Um breve guia para a linguagem-padrão de modelagem de objetos*. 3^a ed. São Paulo: Bookmann, 2007.

- FREDRICKSON, A. Formulation of structured growth models. *Biotechnology and bioengineering*, Wiley Online Library, v. 18, n. 10, p. 1481–1486, 1976.
- FREITAS, H.; OLIVO, J.; ANDRADE, C. Optimization of bioethanol in silico production process in a fed-batch bioreactor using non-linear model predictive control and evolutionary computation techniques. *Energies*, Multidisciplinary Digital Publishing Institute, v. 10, n. 11, p. 1763, 2017.
- FREITAS, H. F. S.; ANDRADE, C. M. G. A brief review on biotechnological process sensing. In: IEEE. *Electronic Measurement & Instruments (ICEMI), 2015 12th IEEE International Conference on*. [S.l.], 2015. v. 3, p. 1622–1627.
- FREITAS, H. F. S. de. *Sloth - A noobish, humble and unpretentious tool for process simulation using simultaneous equation solving*. 2019. Disponível em: <<https://doi.org/10.5281/zenodo.2567445>>.
- FRITZSON, P. et al. The open source modelica project. In: *Proceedings of The 2th International Modelica Conference*. [S.l.: s.n.], 2002. p. 18–19.
- GADKAR, K. G.; MEHRA, S.; GOMES, J. On-line adaptation of neural networks for bioprocess control. *Computers & chemical engineering*, Elsevier, v. 29, n. 5, p. 1047–1057, 2005.
- GANDOMI, A. H. et al. Metaheuristic algorithms in modeling and optimization. In: *Metaheuristic applications in structures and infrastructures*. [S.l.]: Elsevier, 2013. p. 1–24.
- GERNAEY, K. V. et al. Application of mechanistic models to fermentation and biocatalysis for next-generation processes. *Trends in biotechnology*, Elsevier, v. 28, n. 7, p. 346–354, 2010.
- GNOTH, S. et al. Process analytical technology (pat): batch-to-batch reproducibility of fermentation processes by robust process operational design and control. *Journal of biotechnology*, Elsevier, v. 132, n. 2, p. 180–186, 2007.
- GODA, T. On the separability of multivariate functions. *arXiv preprint arXiv:1301.5962*, 2013.
- GOLDEMBERG, J.; COELHO, S. T.; GUARDABASSI, P. The sustainability of ethanol production from sugarcane. *Energy policy*, Elsevier, v. 36, n. 6, p. 2086–2097, 2008.
- GUEDES, G. T. A. *UML – Uma abordagem prática*. 2^a. ed. São Paulo: Novatec, 2011.
- GUPTA, A.; VERMA, J. P. Sustainable bio-ethanol production from agro-residues: a review. *Renewable and Sustainable Energy Reviews*, Elsevier, v. 41, p. 550–567, 2015.
- HAN, W. et al. Utilization of waste cake for fermentative ethanol production. *Science of The Total Environment*, Elsevier, v. 673, p. 378–383, 2019.
- HANSEN, N.; OSTERMEIER, A. Completely derandomized self-adaptation in evolution strategies. *Evolutionary computation*, MIT Press, v. 9, n. 2, p. 159–195, 2001.

- HARMS, P.; KOSTOV, Y.; RAO, G. Bioprocess monitoring. *Current opinion in biotechnology*, Elsevier, v. 13, n. 2, p. 124–127, 2002.
- HAVLIK, I. et al. On-line monitoring of large cultivations of microalgae and cyanobacteria. *Trends in biotechnology*, Elsevier, v. 31, n. 7, p. 406–414, 2013.
- HEGERTY, B.; HUNG, C.-C.; KASPRAK, K. A comparative study on differential evolution and genetic algorithms for some combinatorial problems. In: SMAI. *Proceedings of 8th Mexican International Conference on Artificial Intelligence*. [S.I.], 2009. p. 9–13.
- HENES, B.; SONNLEITNER, B. Controlled fed-batch by tracking the maximal culture capacity. *Journal of biotechnology*, Elsevier, v. 132, n. 2, p. 118–126, 2007.
- HERRERA, F.; LOZANO, M. Workshop for evolutionary algorithms and other metaheuristics for continuous optimization problems—a scalability test. In: *Proceedings of International Conference on Intelligent System Design and Applications*. Pisa, Itália: [s.n.], 2009.
- HERRERA, F.; LOZANO, M.; MOLINA, D. *Test suite for the special issue of soft computing on scalability of evolutionary algorithms and other metaheuristics for large scale continuous optimization problems*. Granda, Espanha, 2010. Disponível em: <<https://sci2s.ugr.es/EAMHCO>>.
- HINDMARSH, A. C. Odepack, a systematized collection of ode solvers. *Scientific computing*, North-Holland, p. 55–64, 1983.
- HOLLAND, J. H. et al. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. [S.I.]: MIT press, 1992.
- HONG, J. Optimal substrate feeding policy for a fed batch fermentation with substrate and product inhibition kinetics. *Biotechnology and bioengineering*, Wiley Online Library, v. 28, n. 9, p. 1421–1431, 1986.
- HOWITT, G. D.; LUUS, R. Model reduction by minimization of integral square error performance indices. *Journal of the Franklin Institute*, Elsevier, v. 327, n. 3, p. 343–357, 1990.
- HUANG, T.; MOHAN, A. S. Micro-particle swarm optimizer for solving high dimensional optimization problems (μ pso for high dimensional optimization problems). *Applied Mathematics and Computation*, Elsevier, v. 181, n. 2, p. 1148–1154, 2006.
- HUNTER, J. D. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, IEEE COMPUTER SOC, v. 9, n. 3, p. 90–95, 2007.
- INGHAM, J. et al. *Chemical engineering dynamics: an introduction to modelling and computer simulation*. [S.I.]: John Wiley & Sons, 2008. v. 3.
- JABARIVELISDEH, B.; WALDHERR, S. Improving bioprocess productivity using constraint-based models in a dynamic optimization scheme. *IFAC-PapersOnLine*, Elsevier, v. 49, n. 26, p. 245–251, 2016.

- JACOB, S.; BANERJEE, R. Modeling and optimization of anaerobic codigestion of potato waste and aquatic weed by response surface methodology and artificial neural network coupled genetic algorithm. *Bioresource technology*, Elsevier, v. 214, p. 386–395, 2016.
- JAMIL, M.; YANG, X.-S. A literature survey of benchmark functions for global optimisation problems. *International Journal of Mathematical Modelling and Numerical Optimisation*, Inderscience Publishers Ltd, v. 4, n. 2, p. 150–194, 2013.
- JAYARAMAN, V. et al. Dynamic optimization of fed-batch bioreactors using the ant algorithm. *Biotechnology Progress*, Wiley Online Library, v. 17, n. 1, p. 81–88, 2001.
- JONES, E. et al. *SciPy: Open source scientific tools for Python*. 2001. Disponível em: <<http://www.scipy.org/>>. Acesso em: 23 jul. 2018.
- KAMATH, R. S.; BIEGLER, L. T.; GROSSMANN, I. E. An equation-oriented approach for handling thermodynamics based on cubic equation of state in process optimization. *Computers & Chemical Engineering*, Elsevier, v. 34, n. 12, p. 2085–2096, 2010.
- KANA, E. G. et al. Modeling and optimization of biogas production on saw dust and other co-substrates using artificial neural network and genetic algorithm. *Renewable energy*, Elsevier, v. 46, p. 276–281, 2012.
- KARABOGA, D.; BASTURK, B. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (abc) algorithm. *Journal of global optimization*, Springer, v. 39, n. 3, p. 459–471, 2007.
- KAWOHL, M.; HEINE, T.; KING, R. Model based estimation and optimal control of fed-batch fermentation processes for the production of antibiotics. *Chemical Engineering and Processing: Process Intensification*, Elsevier, v. 46, n. 11, p. 1223–1241, 2007.
- KEALY, T.; O'DWYER, A. Analytical isse calculation and optimum control system design. Dublin Institute of Technology, 2003.
- KENNEDY, J.; EBERHART, R. Particle swarm optimization (pso). In: *Proc. IEEE International Conference on Neural Networks, Perth, Australia*. [S.l.: s.n.], 1995. p. 1942–1948.
- KEOGH, E.; MUEEN, A. *Curse of dimensionality*. 2017. Disponível em: <https://doi.org/10.1007/978-1-4899-7687-1_192>.
- KHATIWADA, D. et al. Optimizing ethanol and bioelectricity production in sugarcane biorefineries in brazil. *Renewable energy*, Elsevier, v. 85, p. 371–386, 2016.
- KIRAN, E. U.; LIU, Y. Bioethanol production from mixed food waste by an effective enzymatic pretreatment. *Fuel*, Elsevier, v. 159, p. 463–469, 2015.
- KIRKPATRICK, S.; GELATT, C. D.; VECCHI, M. P. Optimization by simulated annealing. *science*, American Association for the Advancement of Science, v. 220, n. 4598, p. 671–680, 1983.

- KOPPRAM, R. et al. Lignocellulosic ethanol production at high-gravity: challenges and perspectives. *Trends in biotechnology*, Elsevier, v. 32, n. 1, p. 46–53, 2014.
- KRAUSE, D.; HUSSEIN, M.; BECKER, T. Online monitoring of bioprocesses via multivariate sensor prediction within swarm intelligence decision making. *Chemo-metrics and Intelligent Laboratory Systems*, Elsevier, v. 145, p. 48–59, 2015.
- LAPIN, A.; SCHMID, J.; REUSS, M. Modeling the dynamics of e. coli populations in the three-dimensional turbulent field of a stirred-tank bioreactor—a structured–segregated approach. *Chemical engineering science*, Elsevier, v. 61, n. 14, p. 4783–4797, 2006.
- LATORRE, A.; MUELAS, S.; PEÑA, J.-M. A comprehensive comparison of large scale global optimizers. *Information Sciences*, Elsevier, v. 316, p. 517–549, 2015.
- LEE, J.; RAMIREZ, W. F. Optimal fed-batch control of induced foreign protein production by recombinant bacteria. *AIChE Journal*, Wiley Online Library, v. 40, n. 5, p. 899–907, 1994.
- LI, X. et al. *Benchmark Functions for the CEC'2013 Special Session and Competition on Large-Scale Global Optimization*. Hefei, China, 2013. 1–23 p.
- LIAO, J. C. et al. Fuelling the future: microbial engineering for the production of sustainable biofuels. *Nature Reviews Microbiology*, Nature Publishing Group, v. 14, n. 5, p. 288, 2016.
- LIU, R.; SHEN, F. Impacts of main factors on bioethanol fermentation from stalk juice of sweet sorghum by immobilized *saccharomyces cerevisiae* (cicc 1308). *Bioresource technology*, Elsevier, v. 99, n. 4, p. 847–854, 2008.
- LOIZIDOU, M. et al. Pilot scale system of two horizontal rotating bioreactors for bioethanol production from household food waste at high solid concentrations. *Waste and biomass valorization*, Springer, v. 8, n. 5, p. 1709–1719, 2017.
- LUTTMANN, R. et al. Soft sensors in bioprocessing: a status report and recommendations. *Biotechnology journal*, Wiley Online Library, v. 7, n. 8, p. 1040–1048, 2012.
- LUYBEN, W. L. *Process modeling, simulation and control for chemical engineers*. [S.l.]: McGraw-Hill Higher Education, 1989.
- MAHDAVI, S.; SHIRI, M. E.; RAHNAMAYAN, S. Metaheuristics in large-scale global continues optimization: A survey. *Information Sciences*, v. 295, n. 20, p. 407–428, 2015.
- MAHECHA-BOTERO, A.; GARHYAN, P.; ELNASHAIE, S. S. Bifurcation, stabilization, and ethanol productivity enhancement for a membrane fermentor. *Mathematical and computer modelling*, Elsevier, v. 41, n. 4-5, p. 391–406, 2005.
- MALAJOVICH, M. A. *Biotechnologia 2011*. Rio de Janeiro: Edições da Biblioteca Max Feffer do Instituto de Tecnologia ORT, 2012.

- MÁRQUEZ-VERA, M.; RAMOS-VELASCO, L.; BALDERRAMA-HERNÁNDEZ, B. Stable fuzzy control and observer via lmis in a fermentation process. *Journal of Computational Science*, Elsevier, 2018.
- MAURER, M. et al. Versatile modeling and optimization of fed batch processes for the production of secreted heterologous proteins with pichia pastoris. *Microbial Cell Factories*, BioMed Central Ltd, v. 5, n. 1, p. 37, 2006.
- MAYER, F. D. et al. Why small-scale fuel ethanol production in brazil does not take off? *Renewable and Sustainable Energy Reviews*, Elsevier, v. 43, p. 687–701, 2015.
- MEARS, L. et al. Mechanistic fermentation models for process design, monitoring, and control. *Trends in biotechnology*, Elsevier, 2017.
- MENDESA, Á. J. B.; VALDMANA, B.; JR, M. B. de S. Uma revisão de modelagem matemática em bioprocessos. Parte I: Fundamentos básicos e classificação. *Revista Militar de Ciência e Tecnologia*, XXVIII, p. 40–59, 2011.
- MENDESA, Á. J. B.; VALDMANA, B.; JR, M. B. de S. Uma revisão de modelagem matemática em bioprocessos. parte ii: Modelos mecanicistas e redes neurais artificiais. *Revista Militar de Ciência e Tecnologia*, XXVIII, p. 60–89, 2011.
- METROPOLIS, N. et al. Equation of state calculations by fast computing machines. *The journal of chemical physics*, AIP, v. 21, n. 6, p. 1087–1092, 1953.
- MEURER, A. et al. Sympy: symbolic computing in python. *PeerJ Computer Science*, v. 3, p. e103, 2017. ISSN 2376-5992. Disponível em: <<https://doi.org/10.7717/peerj-cs.103>>.
- MIELENZ, J. R. Ethanol production from biomass: technology and commercialization status. *Current opinion in microbiology*, Elsevier, v. 4, n. 3, p. 324–329, 2001.
- MILES, R.; HAMILTON, K. *Learning UML 2.0*. Estados Unidos: "O'Reilly Media, Inc.", 2006.
- MIRLEKAR, G.; LI, S.; LIMA, F. V. Design and implementation of a biologically inspired optimal control strategy for chemical process control. *Industrial & Engineering Chemistry Research*, ACS Publications, v. 56, n. 22, p. 6468–6479, 2017.
- MITCHELL, M. *An introduction to genetic algorithms*. [S.I.]: MIT press, 1998.
- MOLES, C. G. et al. Integrated process design and control via global optimization: a wastewater treatment plant case study. *Chemical Engineering Research and Design*, Elsevier, v. 81, n. 5, p. 507–517, 2003.
- MOLES, C. G.; MENDES, P.; BANGA, J. R. Parameter estimation in biochemical pathways: a comparison of global optimization methods. *Genome research*, Cold Spring Harbor Lab, v. 13, n. 11, p. 2467–2474, 2003.
- MONTALVO, S. et al. Kinetic evaluation and performance of pilot-scale fed-batch aerated lagoons treating winery wastewaters. *Bioresource technology*, Elsevier, v. 101, n. 10, p. 3452–3456, 2010.

- MUKHOPADHYAY, S.; DAS, S. A system on chip development of customizable ga architecture for real parameter optimization problem. In: *Handbook of Research on Natural Computing for Optimization Problems*. [S.I.]: IGI Global, 2016. p. 66–102.
- NAJAFPOUR, G. D. *Biochemical Engineering and Biotechnology*. Amsterdã: Elsevier, 2007.
- NIKDEL, A.; BRAATZ, R. D.; BUDMAN, H. M. A systematic approach for finding the objective function and active constraints for dynamic flux balance analysis. *Bioprocess and biosystems engineering*, Springer, v. 41, n. 5, p. 641–655, 2018.
- NIKOLIĆ, D. D. Dae tools: equation-based object-oriented modelling, simulation and optimisation software. *PeerJ Computer Science*, PeerJ Inc., v. 2, p. e54, 2016.
- OCHOA, S. A new approach for finding smooth optimal feeding profiles in fed-batch fermentations. *Biochemical Engineering Journal*, Elsevier, v. 105, p. 177–188, 2016.
- OCHOA, S.; REPKE, J.-U.; WOZNY, G. A new parallel tempering algorithm for global optimization: Applications to bioprocess optimization. In: *19th European Symposium on Computer Aided Process Engineering – ESCAPE19*. [S.I.]: Elsevier, 2009. v. 26, p. 513–518.
- OCHOA, S.; REPKE, J.-U.; WOZNY, G. Integrating real-time optimization and control for optimal operation: Application to the bio-ethanol process. *Biochemical Engineering Journal*, Elsevier, v. 53, n. 1, p. 18–25, 2010.
- OGUNNAIKE, B. A.; RAY, W. H. *Process dynamics, modeling, and control*. [S.I.]: Oxford University Press New York, 1994. v. 1.
- OHNO, H.; NAKANISHI, E.; TAKAMATSU, T. Optimal control of a semibatch fermentation. *Biotechnology and bioengineering*, Wiley Online Library, v. 18, n. 6, p. 847–864, 1976.
- OLGUIN-CARBAJAL, M.; ALBA, E.; ARELLANO-VERDEJO, J. Micro-differential evolution with local search for high dimensional problems. In: *IEEE. Evolutionary Computation (CEC), 2013 IEEE Congress on*. [S.I.], 2013. p. 48–54.
- OMG. *Unified Modeling Language Specification, version 2.5.1*. 2000. Disponível em: <www.omg.org>.
- OMIDVAR, M. N. et al. Cooperative co-evolution with differential grouping for large scale optimization. *IEEE Transactions on evolutionary computation*, IEEE, v. 18, n. 3, p. 378–393, 2014.
- PANTANO, M. N. et al. Multivariable control for tracking optimal profiles in a nonlinear fed-batch bioprocess integrated with state estimation. *Industrial & Engineering Chemistry Research*, ACS Publications, v. 56, n. 20, p. 6043–6056, 2017.
- PATTISON, R. C.; BALDEA, M. Equation-oriented flowsheet simulation and optimization using pseudo-transient models. *AIChE Journal*, Wiley Online Library, v. 60, n. 12, p. 4104–4123, 2014.

- PILLIS, L. G. D.; RADUNSKAYA, A. The dynamics of an optimally controlled tumor model: A case study. *Mathematical and computer modelling*, Elsevier, v. 37, n. 11, p. 1221–1244, 2003.
- PIMENTEL, G. A. et al. An observer-based robust control strategy for overflow metabolism cultures in fed-batch bioreactors. *IFAC-PapersOnLine*, Elsevier, v. 48, n. 8, p. 1081–1086, 2015.
- POPP, J. et al. The effect of bioenergy expansion: food, energy, and environment. *Renewable and Sustainable Energy Reviews*, Elsevier, v. 32, p. 559–578, 2014.
- PROVOST, A.; BASTIN, G. Dynamic metabolic modelling under the balanced growth condition. *Journal of Process Control*, Elsevier, v. 14, n. 7, p. 717–728, 2004.
- RANI, K. Y.; RAO, V. R. Control of fermenters—a review. *Bioprocess Engineering*, Springer, v. 21, n. 1, p. 77–88, 1999.
- RENEWABLE FUELS ASSOCIATION. *World Fuel Ethanol Production*. 2017. Disponível em: <<http://www.ethanolrfa.org/resources/industry/statistics/#1454099103927-61e598f7-7643>>. Acesso em: 14 jun. 2017.
- RIO-CHANONA, E. A. del; ZHANG, D.; VASSILIADIS, V. S. Model-based real-time optimisation of a fed-batch cyanobacterial hydrogen production process using economic model predictive control strategy. *Chemical Engineering Science*, Elsevier, v. 142, p. 289–298, 2016.
- ROCHA, I.; FERREIRA, E. On-line simultaneous monitoring of glucose and acetate with fia during high cell density fermentation of recombinant e. coli. *Analytica Chimica Acta*, Elsevier, v. 462, n. 2, p. 293–304, 2002.
- ROCHA, M. et al. Optimization of fed-batch fermentation processes with bio-inspired algorithms. *Expert Systems with Applications*, Elsevier, v. 41, n. 5, p. 2186–2195, 2014.
- ROCHA, M. et al. Evolutionary algorithms for optimal control in fed-batch fermentation processes. In: SPRINGER. *Workshops on Applications of Evolutionary Computation*. [S.l.], 2004. p. 84–93.
- ROCHA, M. et al. Evaluating evolutionary algorithms and differential evolution for the online optimization of fermentation processes. In: SPRINGER. *European Conference on Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics*. [S.l.], 2007. p. 236–246.
- ROUBOS, J.; STRATEN, G. V.; BOXTEL, A. V. An evolutionary strategy for fed-batch bioreactor optimization; concepts and performance. *Journal of Biotechnology*, Elsevier, v. 67, n. 2-3, p. 173–187, 1999.
- ROYLE, K. E.; VAL, I. J. del; KONTORAVDI, C. Integration of models and experimentation to optimise the production of potential biotherapeutics. *Drug discovery today*, Elsevier, v. 18, n. 23, p. 1250–1255, 2013.
- SALLES-FILHO, S. L. M. et al. Perspectives for the brazilian bioethanol sector: The innovation driver. *Energy Policy*, Elsevier, v. 108, p. 70–77, 2017.

- SANTO, G. E. et al. Development of fed-batch profiles for efficient biosynthesis of catechol-o-methyltransferase. *Biotechnology Reports*, Elsevier, v. 3, p. 34–41, 2014.
- ŞENDRESCU, D. Nonlinear model predictive control of a depollution bioprocess. In: IEEE. *Circuits, Communications and System (PACCS), 2011 Third Pacific-Asia Conference on*. [S.I.], 2011. p. 1–4.
- ŞENDRESCU, D. et al. Nonlinear model predictive control of a lipase production bioprocess. In: IEEE. *Carpathian Control Conference (ICCC), 2011 12th International*. [S.I.], 2011. p. 337–341.
- SHACHAM, M. et al. Equation oriented approach to process flowsheeting. *Computers & Chemical Engineering*, Pergamon, v. 6, n. 2, p. 79–95, 1982.
- SHANG, Y.-W.; QIU, Y.-H. A note on the extended rosenbrock function. *Evolutionary Computation*, v. 1, n. 14, p. 119–126, 2006.
- SIMON, L. L. et al. Assessment of recent process analytical technology (pat) trends: a multiauthor review. *Organic Process Research & Development*, ACS Publications, v. 19, n. 1, p. 3–62, 2015.
- SIMUTIS, R.; LÜBBERT, A. Bioreactor control improves bioprocess performance. *Biotechnology journal*, Wiley Online Library, v. 10, n. 8, p. 1115–1130, 2015.
- SINDHU, R.; BINOD, P.; PANDEY, A. Biological pretreatment of lignocellulosic biomass—an overview. *Bioresource technology*, Elsevier, v. 199, p. 76–82, 2016.
- SMEETS, E. M. et al. A bottom-up assessment and review of global bio-energy potentials to 2050. *Progress in Energy and combustion science*, Elsevier, v. 33, n. 1, p. 56–106, 2007.
- SOARES, R. d. P. *Desenvolvimento de um simulador genérico de processos dinâmicos*. Mestrado em engenharia química — Universidade Federal do Rio Grande do Sul, Porto Alegre, 2003.
- SOARES, R. d. P.; SECCHI, A. Emso: A new environment for modelling, simulation and optimisation. In: *Computer Aided Chemical Engineering*. [S.I.]: Elsevier, 2003. v. 14, p. 947–952.
- SOMMEREGGER, W. et al. Quality by control: Towards model predictive control of mammalian cell culture bioprocesses. *Biotechnology journal*, Wiley Online Library, v. 12, n. 7, p. 1600546, 2017.
- SOSNOWSKI, P. et al. Kinetic investigations of methane co-fermentation of sewage sludge and organic fraction of municipal solid wastes. *Bioresource technology*, Elsevier, v. 99, n. 13, p. 5731–5737, 2008.
- SPADIUT, O.; HERWIG, C. Dynamics in bioprocess development for pichia pastoris. *Bioengineered*, Taylor & Francis, v. 5, n. 6, p. 401–404, 2014.
- SPADIUT, O. et al. Dynamic process conditions in bioprocess development. *Engineering in Life Sciences*, Wiley Online Library, v. 13, n. 1, p. 88–101, 2013.

- SRIDEVI, K.; SIVARAMAN, E.; MULLAI, P. Back propagation neural network modeling of biodegradation and fermentative biohydrogen production using distillery wastewater in a hybrid upflow anaerobic sludge blanket reactor. *Bioresource technology*, Elsevier, v. 165, p. 233–240, 2014.
- SRIDHAR, L. N. Elimination of oscillations in fermentation processes. *AIChE Journal*, Wiley Online Library, v. 57, n. 9, p. 2397–2405, 2011.
- STORN, R.; PRICE, K. Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization*, Springer, v. 11, n. 4, p. 341–359, 1997.
- SUGANTHAN, P. N. et al. *Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization*. [S.l.], 2005.
- SURJHANOVIC, S.; BINGHAM, D. *Virtual Library of Simulation Experiments: Test Functions and Datasets*. 2013. Disponível em: <<https://www.sfu.ca/~ssurjano/optimization.html>>. Acesso em: 23 jul. 2018.
- TAKORS, R. Scale-up of microbial processes: impacts, tools and open questions. *Journal of biotechnology*, Elsevier, v. 160, n. 1, p. 3–9, 2012.
- TANG, K.; YAO, X.; SUGANTHAN, P. *Benchmark functions for the CEC'2010 special session and competition on large scale global optimization*. Henfei, China, 2010. 1–23 p.
- TANG, K. et al. *Benchmark functions for the CEC'2008 special session and competition on large scale global optimization*. Hefei, China, 2007.
- TIAN, H. et al. Optimization of auto-induction medium for g-csf production by escherichia coli using artificial neural networks coupled with genetic algorithm. *World Journal of Microbiology and Biotechnology*, Springer, v. 29, n. 3, p. 505–513, 2013.
- UYMAZ, S. A.; TEZEL, G.; YEL, E. Artificial algae algorithm (aaa) for nonlinear global optimization. *Applied Soft Computing*, Elsevier, v. 31, p. 153–171, 2015.
- VILLAVERDE, A. F. et al. Biopredyn-bench: a suite of benchmark problems for dynamic modelling in systems biology. *BMC Systems Biology*, v. 9, n. 1, 2015.
- VITALIY, F. *Differential evolution—in search of solutions*. Nova Iorque: Springer, 2006.
- VOHRA, M. et al. Bioethanol production: Feedstock and current technologies. *Journal of Environmental Chemical Engineering*, Elsevier, v. 2, n. 1, p. 573–584, 2014.
- WALT, S. V. D.; COLBERT, S. C.; VAROQUAUX, G. The numpy array: a structure for efficient numerical computation. *Computing in Science & Engineering*, IEEE Computer Society, v. 13, n. 2, p. 22, 2011.
- WANG, F.-S.; WU, W.-H.; HSU, K.-C. Fuzzy optimization in metabolic systems. *International Journal of Biological, Food, Veterinary and Agricultural Engineering*, v. 8, n. 7, p. 661–665, 2014.

- WANG, H.; RAHNAMAYAN, S.; WU, Z. Parallel differential evolution with self-adapting control parameters and generalized opposition-based learning for solving high-dimensional optimization problems. *Journal of Parallel and Distributed Computing*, Elsevier, v. 73, n. 1, p. 62–73, 2013.
- WANG, H.; WU, Z.; RAHNAMAYAN, S. Enhanced opposition-based differential evolution for solving high-dimensional continuous optimization problems. *Soft Computing*, Springer, v. 15, n. 11, p. 2127–2140, 2011.
- WANG, S. et al. Optimization and modeling of biohydrogen production by mixed bacterial cultures from raw cassava starch. *Frontiers of Chemical Science and Engineering*, Springer, v. 11, n. 1, p. 100–106, 2017.
- WANG, Y. et al. Industrial bioprocess control and optimization in the context of systems biotechnology. *Biotechnology advances*, Elsevier, v. 27, n. 6, p. 989–995, 2009.
- WECHSELBERGER, P. et al. Efficient feeding profile optimization for recombinant protein production using physiological information. *Bioprocess and biosystems engineering*, Springer, v. 35, n. 9, p. 1637–1649, 2012.
- WERNER, R. G. Economic aspects of commercial manufacture of biopharmaceuticals. *Journal of biotechnology*, Elsevier, v. 113, n. 1, p. 171–182, 2004.
- WESTERBERG, A. W. et al. *Plans for ASCEND IV: Our next generation equational-based modeling environment*. [S.l.]: Carnegie Mellon University, Engineering Design Research Center, 1994.
- WICKE, B. et al. The current bioenergy production potential of semi-arid and arid regions in sub-saharan africa. *Biomass and bioenergy*, Elsevier, v. 35, n. 7, p. 2773–2786, 2011.
- WÜRTH, L.; RAWLINGS, J. B.; MARQUARDT, W. Economic dynamic real-time optimization and nonlinear model-predictive control on infinite horizons. *IFAC Proceedings Volumes*, Elsevier, v. 42, n. 11, p. 219–224, 2009.
- XIA, J. et al. Engineering zymomonas mobilis for robust cellulosic ethanol production. *Trends in biotechnology*, Elsevier, 2019.
- YANG, S. et al. Zymomonas mobilis as a model system for production of biofuels and biochemicals. *Microbial biotechnology*, Wiley Online Library, v. 9, n. 6, p. 699–717, 2016.
- YANG, Z.; TANG, K.; YAO, X. Differential evolution for high-dimensional function optimization. In: IEEE. *Evolutionary Computation, 2007. CEC 2007. IEEE Congress on*. [S.l.], 2007. p. 3523–3530.
- YE, J. et al. Optimization of a fed-batch bioreactor for 1, 3-propanediol production using hybrid nonlinear optimal control. *Journal of Process Control*, Elsevier, v. 24, n. 10, p. 1556–1569, 2014.
- YÜZGEÇ, U.; TÜRKER, M.; HOCALAR, A. On-line evolutionary optimization of an industrial fed-batch yeast fermentation process. *ISA transactions*, Elsevier, v. 48, n. 1, p. 79–92, 2009.

- YWORKS. *yED Graph Editor*. 2018. Disponível em: <<https://www.yworks.com/products/yed>>. Acesso em: 14 jun. 2017.
- ZABED, H. et al. Bioethanol production from fermentable sugar juice. *The Scientific World Journal*, Hindawi, v. 2014, 2014.
- ZABED, H. et al. Bioethanol production from renewable sources: Current perspectives and technological progress. *Renewable and Sustainable Energy Reviews*, Elsevier, v. 71, p. 475–501, 2017.
- ZAIN, M. Z. bin M. et al. A multi-objective particle swarm optimization algorithm based on dynamic boundary search for constrained optimization. *Applied Soft Computing*, Elsevier, 2018.
- ZAIN, M. Z. bin M. et al. Optimization of fed-batch fermentation processes using the backtracking search algorithm. *Expert Systems with Applications*, Elsevier, v. 91, p. 286–297, 2018.
- ZHANG, Q. et al. Substrate and product inhibition on yeast performance in ethanol fermentation. *Energy & Fuels*, ACS Publications, v. 29, n. 2, p. 1019–1027, 2015.
- ZHANG, Q. et al. *Multiobjective optimization test instances for the CEC 2009 special session and competition*. Reino Unido e Singapura, 2008. Special session on performance assessment of multi-objective optimization algorithms.
- ZHAO, L. et al. Advances in process monitoring tools for cell culture bioprocesses. *Engineering in Life Sciences*, Wiley Online Library, v. 15, n. 5, p. 459–468, 2015.
- ZYDNEY, A. L. Perspectives on integrated continuous bioprocessing—opportunities and challenges. *Current Opinion in Chemical Engineering*, Elsevier, v. 10, p. 8–13, 2015.

APÊNDICE A

Código fonte utilizado no caso de estudo I

No presente capítulo será apresentado o código-fonte utilizado para o estudo de otimização em malha aberta realizado no Capítulo 5, utilizando a ferramenta sloth.

A.1 Otimização em malha aberta

```

1  #Importando bibliotecas da ferramenta SLOTH
2  from sloth.model import *
3  from sloth.problem import Problem
4  from sloth.simulation import Simulation
5  from sloth.optimization import Optimization, OptimizationProblem
6  from sloth.core.domain import Domain
7
8  #Bibliotecas adicionais (numpy, matplotlib)
9  import numpy as np
10 import matplotlib.pyplot as plt
11
12 #Modelo 1 - Função de alimentação polinomial
13 class ethanol_opt_1(Model):
14
15     def __init__(self):
16
17         super().__init__(name="E0", description="Ethanol optimization
18         ↪ problem")
19
20         self.t = self.createVariable("t", dimless, "t")
21         self.dom = Domain("domain", dimless, self.t, "generic domain")
22
23         self.V = self.createVariable("V", dimless, "V")
24         self.X = self.createVariable("X", dimless, "X")
25         self.S = self.createVariable("S", dimless, "S")
26         self.P = self.createVariable("P", dimless, "P")
27
28         self.V.distributeOnDomain(self.dom)
29         self.X.distributeOnDomain(self.dom)
30         self.S.distributeOnDomain(self.dom)
31         self.P.distributeOnDomain(self.dom)
32
33         self.mu0 = self.createParameter("mu0", dimless, "")

```

```

33     self.mu0.setValue(0.408)
34     self.q0 = self.createParameter("q0", dimless, "")
35     self.q0.setValue(1)
36     self.ks = self.createParameter("ks", dimless, "")
37     self.ks.setValue(0.22)
38     self.ksI = self.createParameter("ksI", dimless, "")
39     self.ksI.setValue(0.44)
40     self.kp = self.createParameter("kp", dimless, "")
41     self.kp.setValue(16.)
42     self.kpI = self.createParameter("kpI", dimless, "")
43     self.kpI.setValue(71.5)
44     self.y = self.createParameter("y", dimless, "")
45     self.y.setValue(0.1)
46     self.S_0 = self.createParameter("S_0", dimless, "")
47     self.S_0.setValue(150.)
48
49     self.tf = self.createParameter("tf", dimless, "tf")
50     self.a = self.createParameter("a", dimless, "a")
51     self.b = self.createParameter("b", dimless, "b")
52     self.c = self.createParameter("c", dimless, "c")
53     self.d = self.createParameter("d", dimless, "d")
54
55     def DeclareEquations(self):
56
57         #Artifício usado para evitar longas linhas de texto
58         mu_ = (self.mu0()/(1.+self.P()/self.kp()))
59         mu = mu_*(self.S()/(self.ks()+self.S()))
60
61         q_ = (self.q0()/(1.+self.P()/self.kpI()))
62         q = q_*(self.S()/(self.ksI()+self.S()))
63
64         F_exp = self.a()*(self.t()/self.tf())**3. +
65         ↪ self.b()*(self.t()/self.tf())**2. +
66         ↪ self.c()*(self.t()/self.tf()) + self.d()
67
68         #Restrição da variável manipulada (0<=F<=12)
69         u = Max(Min(F_exp, 12.),0.)
70
71         eqx1 = self.V.Diff(self.t) == u

```

```

70     self.eqx1 = self.createEquation("eq_V", "", eqx1)
71
72     eqx2 = self.X.Diff(self.t) == mu*self.X() -
73     ↪ u*(self.X()/self.V())
74     self.eqx2 = self.createEquation("eq_X", "", eqx2)
75
76     eqx3 = self.S.Diff(self.t) == -1*mu*(self.X()/self.y()) +
77     ↪ u*(self.S_0()-self.S())/self.V()
78     self.eqx3 = self.createEquation("eq_S", "", eqx3)
79
80     eqx4 = self.P.Diff(self.t) == q*self.X() -
81     ↪ u*(self.P()/self.V())
82     self.eqx4 = self.createEquation("eq_P", "", eqx4)
83
84     #Modelo 2 - Função de alimentação cosenoidal
85     class ethanol_opt_2(Model):
86
87         def __init__(self):
88
89             super().__init__(name="E0", description="Ethanol optimization
90             ↪ problem")
91
92             self.t = self.createVariable("t", dimless, "t")
93             self.dom = Domain("domain", dimless, self.t, "generic domain")
94
95             self.V = self.createVariable("V", dimless, "V")
96             self.X = self.createVariable("X", dimless, "X")
97             self.S = self.createVariable("S", dimless, "S")
98             self.P = self.createVariable("P", dimless, "P")
99
100             self.V.distributeOnDomain(self.dom)
101             self.X.distributeOnDomain(self.dom)
102             self.S.distributeOnDomain(self.dom)
103             self.P.distributeOnDomain(self.dom)
104
105             self.mu0 = self.createParameter("mu0", dimless, "")
106             self.mu0.setValue(0.408)
107             self.q0 = self.createParameter("q0", dimless, "")
108             self.q0.setValue(1)

```

```

105     self.ks = self.createParameter("ks", dimless, "")
106     self.ks.setValue(0.22)
107     self.ksI = self.createParameter("ksI", dimless, "")
108     self.ksI.setValue(0.44)
109     self.kp = self.createParameter("kp", dimless, "")
110     self.kp.setValue(16.)
111     self.kpI = self.createParameter("kpI", dimless, "")
112     self.kpI.setValue(71.5)
113     self.y = self.createParameter("y", dimless, "")
114     self.y.setValue(0.1)
115     self.S_0 = self.createParameter("x2_0", dimless, "")
116     self.S_0.setValue(150.)
117
118     self.tf = self.createParameter("tf", dimless, "tf")
119     self.a = self.createParameter("a", dimless, "a")
120     self.b = self.createParameter("b", dimless, "b")
121     self.c = self.createParameter("c", dimless, "c")
122     self.d = self.createParameter("d", dimless, "d")
123     self.e = self.createParameter("e", dimless, "e")
124     self.f = self.createParameter("f", dimless, "f")
125     self.g = self.createParameter("g", dimless, "g")
126
127     def DeclareEquations(self):
128
129         #Artifício usado para evitar longas linhas de texto
130         mu_ = (self.mu0()/(1.+self.x4()/self.kp()))
131         mu = mu_*(self.x3()/(self.ks()+self.x3()))
132
133         q_ = (self.q0()/(1.+self.x4()/self.kpI()))
134         q = q_*(self.x3()/(self.ksI()+self.x3()))
135
136         F_exp = self.a()*Cos(self.b()*(self.t()/self.tf())) + self.c()
137         ↪ + self.d()
138
139         #Restrição da variável manipulada (0<=F<=12)
140         u = Max(Min(F_exp, 12.),0.)
141
142         eqx1 = self.V.Diff(self.t) == u
143         self.eqx1 = self.createEquation("eq_V", "", eqx1)

```

```

143
144     eqx2 = self.X.Diff(self.t) == mu*self.X() -
        ↪ u*(self.X()/self.V())
145     self.eqx2 = self.createEquation("eq_X", "", eqx2)
146
147     eqx3 = self.S.Diff(self.t) == -1*mu*(self.X()/self.y()) +
        ↪ u*(self.S_0()-self.S())/self.V()
148     self.eqx3 = self.createEquation("eq_S", "", eqx3)
149
150     eqx4 = self.P.Diff(self.t) == q*self.X() -
        ↪ u*(self.P()/self.V())
151     self.eqx4 = self.createEquation("eq_P", "", eqx4)
152
153     #Definindo um problema de otimização da produção de bioetanol (classe
        ↪ derivada de OptimizationProblem)
154     class ethanol_max_prob(OptimizationProblem):
155
156         def __init__(self, number_of_dimensions, mod):
157
158             super().__init__(number_of_dimensions)
159
160             self.mod = mod
161
162         def DeclareObjectiveFunction(self, x):
163
164             #Inserindo os valores dos parâmetros na simulação
165
166             a = x[0]
167             b = x[1]
168             c = x[2]
169             d = x[3]
170             tf = x[4]
171
172             self.simulation_instance[self.mod.a].setValue(a)
173             self.simulation_instance[self.mod.b].setValue(b)
174             self.simulation_instance[self.mod.c].setValue(c)
175             self.simulation_instance[self.mod.d].setValue(d)
176             self.simulation_instance[self.mod.tf].setValue(tf)
177

```

```
178     self.simulation_instance.problem.setTimeVariableName(['t_E0'])
179
180     self.simulation_instance.problem.resolve()
181
182     self.simulation_configuration['end_time'] = tf
183
184     self.simulation_instance.setConfigurations(definition_dict =
185         self.simulation_configuration)
186
187     self.simulation_instance.runSimulation()
188
189     result = self.simulation_instance.getResults('dict')
190
191     #O problem de otimização deve ser sempre convertido a um
192     ↪ problema de minimização. Assim, usa-se o
193     # recurso de inserir um sinal negativo no cálculo da função
194     ↪ objetivo
195
196     Pfinal = result['t_E0']['Ethanol(P)'][-1]
197     Vfinal = result['t_E0']['Volume(V)'][-1]
198     Tfinal = result['t_E0']['Time(t)'][-1]
199
200     J = -1*(Pfinal*Vfinal)/Tfinal
201
202     #Reseta a simulação, removendo os valores calculados e os
203     ↪ parâmetros inseridos
204
205     self.simulation_instance.reset()
206
207     #Penalizando soluções indesejáveis
208
209     if result['t_E0']['Volume(V)'][-1] <= 11. :
210
211         J = 1e100
212
213     if J > 0. or result['t_E0']['Ethanol(P)'][-1] < 0. :
214
215         J = 1e100
216
217     if result['t_E0']['Volume(V)'][-1] < 0. :
```



```

214
215         J = 1e100
216
217         if any(V_i >= 200. for V_i in result['t_E0']['Volume(V)'][:]):
218
219             J = 1e100
220
221         return[J]
222
223     def DeclareSetBounds(self):
224
225         return(tuple(self.bounds))
226
227     #Retorna um objeto Problem
228     def problem_pse():
229
230         return Problem("problem_PSE", "Ethanol maximization problem")
231
232     #Retorna um objeto Simulation
233     def simulation_pse():
234
235         return Simulation("problem_PSE", "Ethanol maximization problem")
236
237     #Realiza um estudo de otimização em malha aberta
238     def run_optimization_study(param, compile_equations=False,
239     ↪ optimizer_name='de'):
240
241         sim = simulation_pse()
242
243         #Escolhendo o modelo adequado (1- parametrização polinomial 2-
244         ↪ cossenoidal)
245         if param==1:
246             mod = ethanol_opt_1()
247         else:
248             mod = ethanol_opt_2()
249
250         #Processando o modelo
251         mod()

```

```

251     prob = problem_pse()
252
253     prob_opt = ethanol_max_prob(5, mod)
254
255     #Processando o problema de otimização
256     prob_opt()
257
258     #Criando uma classe para o estudo de otimização (derivado de
259     ↪ Optimization)
260
261     class opt_study(Optimization):
262
263         def __init__(self, simulation, optimization_problem,
264             ↪ simulation_configuration, optimization_parameters,
265             ↪ constraints, optimizer, optimization_configuration=None):
266
267             super().__init__(simulation=sim,
268                 optimization_problem=optimization_problem,
269                 simulation_configuration=simulation_configuration,
270                 optimization_parameters=optimization_parameters,
271                 constraints=constraints,
272                 optimization_configuration=optimization_configuration,
273                 optimizer=optimizer)
274
275     sim.setConfigurations(initial_time=0., end_time=mod.tf.value,
276     ↪ domain=mod.dom,
277     time_variable_name='t_EO', is_dynamic=True, print_output=True,
278     compile_equations=compile_equations,
279     output_headers=["Time",
280         "Volume(V)",
281         "Biomass(X)",
282         "Substrate(S)",
283         "Ethanol(P)"],
284     variable_name_map={"t_EO": "Time(t)",
285         "V_EO": "Volume(V)",
286         "X_EO": "Biomass(X)",
287         "S_EO": "Substrate(S)",
288         "P_EO": "Ethanol(P)"},
289     differential_solver='ODEINT')

```

```

286     #Repassando o modelo ao objeto Problem
287     prob.addModels(mod)
288
289     #Definindo a variável tempo
290     prob.setTimeVariableName(['t_E0'])
291
292     #Processando o problema
293     prob.resolve()
294
295     #Definindo as condições iniciais
296     prob.setInitialConditions({'t_E0':0., 'V_E0':10., 'X_E0':1.,
297     ↪ 'S_E0':150., 'P_E0':0.})
298
299     #Definindo o problema o qual será simulado por meio do objeto
300     ↪ Simulation
301     sim.setProblem(prob)
302
303     #Criando um objeto a partir da classe derivada de Optimization
304     # Obs: as restrições são passadas como uma tupla contendo duas
305     ↪ listas, uma para o limite inferior, e outra para o limite
306     ↪ superior
307     # das variáveis manipuladas
308     opt = opt_study(simulation=sim,
309     ↪ optimization_problem=prob_opt,
310     ↪ simulation_configuration=None,
311     ↪ optimization_parameters=[mod.a, mod.b, mod.c,
312     ↪ mod.d,mod.tf],
313     ↪ constraints=[[-100,-100,-100,-100,4.],
314     ↪ [100.,100.,100.,100,96.]],
315     ↪ optimizer=optimizer_name)
316
317     #Processando o problema de otimização já definido no objeto
318     ↪ Optimization
319     opt.optimization_problem()
320
321     #Executando a otimização
322     opt.runOptimization()
323
324     #Retorno de informações ao usuário pelo console

```

```

319     with open("test_log.backup", "w") as openfile:
320
321         openfile.write("=>Results: {}".format(opt.getResults()))
322
323     print("Sucessful run? ", opt.run_sucessful)
324
325     print("\n\n--->RESULT: ",opt.getResults())
326
327     #Realiza um estudo de simulação a partir dos parâmetros previamente
    ↪ determinados
328     def run_simulation_study(a, b, c, d, tf, compile_equations=False,
    ↪     with_plots=False, param=1):
329
330         sim = simulation_pse()
331
332         #Escolhendo o modelo adequado (1- parametrização polinomial 2-
    ↪ cossenoidal)
333         if param==1:
334             mod = ethanol_opt_1()
335         else:
336             mod = ethanol_opt_2()
337
338         #Definindo os parâmetros no modelo
339         mod.a.setValue(a)
340         mod.b.setValue(b)
341         mod.c.setValue(c)
342         mod.d.setValue(d)
343         mod.tf.setValue(tf)
344         mod()
345
346         prob = problem_pse()
347
348         prob.addModels(mod)
349
350         prob.setTimeVariableName(['t_EO'])
351
352         prob.resolve()
353
354         prob.setInitialConditions({'t_EO':0., 'V_EO':10., 'X_EO':1.,
    ↪     'S_EO':150., 'P_EO':0.})

```

```

355
356     sim.setProblem(prob)
357
358     sim.setConfigurations(initial_time=0., end_time=mod.tf.value,
↪     domain=mod.dom,
359         time_variable_name='t_E0', is_dynamic=True, print_output=True,
360         compile_equations=compile_equations,
361         output_headers=["Time",
362                         "Volume(V)",
363                         "Biomass(X)",
364                         "Substrate(S)",
365                         "Ethanol(P)"],
366         variable_name_map={"t_E0": "Time(t)",
367                             "x1_E0": "Volume(V)",
368                             "x2_E0": "Biomass(X)",
369                             "x3_E0": "Substrate(S)",
370                             "x4_E0": "Ethanol(P)"},
371         differential_solver='ODEINT')
372
373     sim.runSimulation()
374
375     sim.showResults()
376
377     print("\n\n\n RESULTS:\n")
378
379     print("\n\n", sim.getResults()[-1][-1])
380
381     #Para a produção de gráficos específicos do caso de estudo
382     if with_plots is True:
383
384         Tf=tf
385
386         #Funções para o cálculo da vazão de alimentação a partir dos
↪ parâmetros determinados
387         def calc_F1(t):
388
389             t = np.array(t)
390
391             return a*(t/Tf)**3 + b*(t/Tf)**2 + c*(t/Tf) + d

```

```

392
393     def calc_F2(t):
394
395         t = np.array(t)
396
397         return a*np.cos(b*(t/Tf)+c) + d
398
399     results = sim.getResults('dict')
400
401     time = results['t_E0']['Time(t)']
402     X = results['t_E0']['Biomass(X)']
403     S = results['t_E0']['Substrate(S)']
404     P = results['t_E0']['Ethanol(P)']
405
406     plt.plot(time, X, ls='-', label=r'Biomassa $(X)$')
407     plt.plot(time, S, ls='--', label=r'Substrato $(S)$')
408     plt.plot(time, P, ls='-.', label=r'Bioetanol $(P)$')
409     plt.xlabel(r'Tempo $(h)$', fontsize=14)
410     plt.ylabel(r'Concentração $(g\,L^{-1})$', fontsize=14)
411     plt.grid()
412     plt.legend()
413     plt.savefig('plot_concentrations_'+str(param)+'.png',
414               ↪ bbox_inches='tight')
415
416     plt.clf()
417
418     time = np.linspace(0., Tf, 1000)
419
420     if param == 1:
421
422         F = calc_F1(time)
423
424     else:
425
426         F = calc_F2(time)
427
428     #Filtrando os resultados e adequando-os às restrições do
429     ↪ problema

```

```
429
430     F[F<0.] = 0.
431     F[F>12.] = 12.
432
433     plt.step(time, F, where='mid',label=r'$u(t)$')
434     plt.xlabel(r'Tempo $(h)$', fontsize=14)
435     plt.ylabel(r'Vazão de alimentação $(L\,h^{-1})$')
436     plt.grid()
437     plt.legend()
438     plt.savefig('plot_feed_'+str(param)+'.png',
    ↪     bbox_inches='tight')
```

APÊNDICE B

Código fonte utilizado no caso de estudo II

No presente capítulo serão apresentados os códigos-fonte utilizados para o estudo de simulação dinâmica realizado no Capítulo 6, bem como o estudo de controle do processo, utilizando a ferramenta `sloth`.

B.1 Estudo de simulação dinâmica da produção de bioetanol

```

1  #Importando bibliotecas da ferramenta SLOTH
2  from sloth.model import *
3  from sloth.problem import Problem
4  from sloth.simulation import Simulation
5  from sloth.optimization import Optimization, OptimizationProblem
6  from sloth.core.domain import Domain
7
8  #Bibliotecas adicionais (numpy, matplotlib)
9  import numpy as np
10 import matplotlib.pyplot as plt
11
12 #Modelo de produção de bioetanol
13 class ethanol_model(Model):
14
15     def __init__(self):
16
17         super().__init__(name="EO", description="Ethanol optimization
18         ↪ problem")
19
20         self.t = self.createVariable("t", dimless, "t")
21         self.dom = Domain("domain", dimless, self.t, "generic domain")
22
23         self.S = self.createVariable("S", dimless, "S")
24         self.E = self.createVariable("E", dimless, "E")
25         self.X = self.createVariable("X", dimless, "X")
26         self.P = self.createVariable("P", dimless, "P")
27
28         self.S.distributeOnDomain(self.dom)
29         self.E.distributeOnDomain(self.dom)
30         self.X.distributeOnDomain(self.dom)
31         self.P.distributeOnDomain(self.dom)
32
33         #Grau de liberdade: variável manipulada

```

```

33     self.Din = self.createParameter("Din", dimless, "")
34     #self.Din.setValue(1.)
35     self.S_0 = self.createParameter("S_0", dimless, "")
36     self.S_0.setValue(150.3)
37
38     self.K_P = self.createParameter("K_P", dimless, "")
39     self.K_P.setValue(1.)
40     self.Y_sx = self.createParameter("Y_sx", dimless, "")
41     self.Y_sx.setValue(.0244)
42     self.m_s = self.createParameter("m_s", dimless, "")
43     self.m_s.setValue(2.16)
44     self.m_p = self.createParameter("m_p", dimless, "")
45     self.m_p.setValue(1.1)
46     self.Dout = self.createParameter("Dout", dimless, "")
47     self.Dout.setValue(.5)
48     self.K_s = self.createParameter("K_s", dimless, "")
49     self.K_s.setValue(.5)
50     self.X_0 = self.createParameter("X_0", dimless, "")
51     self.X_0.setValue(0.08)
52     self.k1 = self.createParameter("k1", dimless, "")
53     self.k1.setValue(16.)
54     self.k2 = self.createParameter("k2", dimless, "")
55     self.k2.setValue(.497)
56     self.k3 = self.createParameter("k3", dimless, "")
57     self.k3.setValue(.0038)
58     self.E_0 = self.createParameter("E_0", dimless, "")
59     self.E_0.setValue(.02)
60     self.P_0 = self.createParameter("P_0", dimless, "")
61     self.P_0.setValue(0.)
62     self.Y_px = self.createParameter("Y_px", dimless, "")
63     self.Y_px.setValue(.0526)
64
65     def DeclareEquations(self):
66
67         eq1 = self.S.Diff(self.t) ==
68             ↪ (-1./self.Y_sx()*(self.S()*self.E()/(self.K_s()+self.S())
69             ↪ - self.m_s()*self.X() + self.Din()*self.S_0() -
70             ↪ self.Dout()*self.S())
71         self.eq1 = self.createEquation("eq1", "", eq1)

```

```

69
70     eq2 = self.X.Diff(self.t) ==
    ↪ (self.K_P()*(self.S()*self.E()/(self.K_s()+self.S())) +
    ↪ self.Din()*self.X_0() - self.Dout()*self.X())
71     self.eq2 = self.createEquation("eq2", "", eq2)
72
73     eq3 = self.E.Diff(self.t) == (self.k1()-self.k2()*self.P() +
    ↪ self.k3()*(self.P()**2.))*self.S()*self.E()/(self.K_s()+self.S())
    ↪ + self.Din()*self.E_0() - self.Dout()*self.E()
74     self.eq3 = self.createEquation("eq3", "", eq3)
75
76     eq4 = self.P.Diff(self.t) ==
    ↪ (1./self.Y_px()*(self.S()*self.E()/(self.K_s()+self.S()))
    ↪ + self.m_p()*self.X() + self.Din()*self.P_0() -
    ↪ self.Dout()*self.P())
77     self.eq4 = self.createEquation("eq4", "", eq4)
78
79     #Definindo um problema de otimização da produção de bioetanol (classe
    ↪ derivada de OptimizationProblem)
80     class ethanol_max_prob(OptimizationProblem):
81
82         def __init__(self, number_of_dimensions, mod, initial_time,
    ↪ end_time, initial_conditions):
83
84             super().__init__(number_of_dimensions)
85
86             self.mod = mod
87
88             self.initial_time = initial_time
89
90             self.end_time = end_time
91
92             self.initial_conditions = initial_conditions
93
94         def DeclareObjectiveFunction(self, x):
95
96             Din = x[0]
97
98             self.simulation_instance[self.mod.Din].setValue(Din)

```

```

99
100     self.simulation_instance.problem.setTimeVariableName(['t_EO'])
101
102     self.simulation_instance.problem.resolve()
103
104     self.simulation_configuration['initial_time'] =
105     ↪ self.initial_time
106
107     self.simulation_configuration['end_time'] = self.end_time
108
109     self.simulation_instance.setConfigurations(
110         definition_dict=self.simulation_configuration)
111
112     self.simulation_instance.runSimulation()
113
114     result = self.simulation_instance.getResults('dict')
115
116     ethanol = np.array(result['t_EO']['Ethanol(P)'][:])
117     keys = np.array(result['t_EO']['KeyComponents(E)'][:])
118     biomass = np.array(result['t_EO']['Biomass(X)'][:])
119     substrate = np.array(result['t_EO']['Substrate(S)'][:])
120     time = np.array(result['t_EO']['Time(t)'][:])
121
122     f =.simps((ethanol-ethanol_sp)**2., time)
123
124     self.simulation_instance.reset()
125
126     if result['t_EO']['Ethanol(P)'][-1] < 0. :
127
128         f = 1e100
129
130     #if result['t_EO'][:][-1] < 0. :
131     #f = 1e100
132
133     #print("\n\n==>OBJ = ", f)
134     #print("\n\n==>x = ", x)
135
136     return[f]

```

```

137     def DeclareSetBounds(self):
138
139         return(tuple(self.bounds))
140
141     #Retorna um objeto Problem
142     def problem_pse():
143
144         return Problem("problem_PSE", "Ethanol maximization problem")
145
146     #Retorna um objeto Simulation
147     def simulation_pse():
148
149         return Simulation("problem_PSE", "Ethanol maximization problem")
150
151     #Realiza um estudo de simulação dinâmica do processo, variando os
152     ↪ parâmetros Din e SO
153     def run_simulation_study(compile_equations=False, Din_study=False,
154     ↪ SO_study=False):
155
156     #Variar entre os valores de Din
157     if Din_study is True:
158
159         for var in ['P', 'X', 'S', 'E']:
160
161             Dins = [0.05, 0.1, 0.2, 0.5] #Valores utilizados
162
163             lss = [':', '-.', '---', '-']
164
165             cs = ['r', 'b', 'g', 'k']
166
167             for i in range(len(Dins)):
168
169                 Din = Dins[i]
170
171                 sim = simulation_pse()
172
173                 mod = ethanol_model()
174
175                 mod.Din.setValue(Din)

```

```

174     mod.S_0.setValue(150.3)
175     mod()
176
177     prob = problem_pse()
178
179     prob.addModels(mod)
180
181     prob.setTimeVariableName(['t_EO'])
182
183     prob.resolve()
184
185     prob.setInitialConditions({'t_EO':0., 'S_EO':150.3,
186     ↪ 'X_EO':.08, 'E_EO':.02, 'P_EO':0.})
187
188     sim.setProblem(prob)
189
190     sim.setConfigurations(initial_time=0., end_time=50.,
191     ↪ domain=mod.dom, time_variable_name='t_EO',
192     ↪ is_dynamic=True, print_output=True,
193     ↪ compile_equations=True,
194     ↪ output_headers=["Time", "Substrate(S)", "Biomass(X)",
195     ↪ "KeyComponents(E)", "Ethanol(P)"],
196     ↪ variable_name_map={"t_EO": "Time(t)",
197     ↪ "S_EO": "Substrate(S)", "X_EO": "Biomass(X)",
198     ↪ "E_EO": "KeyComponents(E)", "P_EO": "Ethanol(P)"},
199     ↪ differential_solver='ODEINT')
200
201     sim.runSimulation()
202
203     print("\nDrawing for {} (Din = {} S_0 =
204     ↪ {}".format(var, Din, 150.3))
205
206     results = sim.getResults('dict')
207
208     time = results['t_EO']['Time(t)']
209     X = results['t_EO']['Biomass(X)']
210     S = results['t_EO']['Substrate(S)']

```

```

205         E = results['t_E0']['KeyComponents(E)']
206         P = results['t_E0']['Ethanol(P)']
207
208         if var == 'X':
209
210             y = X
211             ylabel = r'$X$, (kg, m^{-3})$'
212
213         if var == 'S':
214
215             y = S
216             ylabel = r'$S$, (kg, m^{-3})$'
217
218         if var == 'P':
219
220             y = P
221             ylabel = r'$P$, (kg, m^{-3})$'
222
223         if var == 'E':
224
225             y = E
226             ylabel = r'$E$, (kg, m^{-3})$'
227
228         plt.plot(time, y, linestyle=lss[i], color=cs[i],
229                 ↪ label=r'$D_{in}$'+str(Din)+'$')
230
231
232         xlabel = r'Tempo $(h)$'
233
234         #Produção dos gráficos
235         plt.xlim(0,50.)
236         plt.xlabel(xlabel, fontsize=12)
237         plt.ylabel(ylabel, fontsize=12)
238         plt.legend()
239         plt.grid()
240         plt.savefig('zym_openloop_'+str(var)+'_Din.png',
241                 ↪ bbox_inches='tight')
242
243         plt.clf()
244
245         #Variar entre os valores de S0

```



```

242     if S0_study is True:
243
244         for var in ['P', 'X', 'S', 'E']:
245
246             S0s = [140.3,145.3,150.3,160.3] #Valores utilizados
247
248             lss = [':', '-.', '---', '-']
249
250             cs = ['r', 'b', 'g', 'k']
251
252             for i in range(len(S0s)):
253
254                 Din = .5
255
256                 sim = simulation_pse()
257
258                 mod = ethanol_model()
259
260                 mod.S_0.setValue(S0s[i])
261                 mod.Din.setValue(Din)
262                 mod()
263
264                 prob = problem_pse()
265
266                 prob.addModels(mod)
267
268                 prob.setTimeVariableName(['t_E0'])
269
270                 prob.resolve()
271
272                 prob.setInitialConditions({'t_E0':0., 'S_E0':150.3,
273                 ↪ 'X_E0':.08, 'E_E0':.02, 'P_E0':0.})
274
275                 sim.setProblem(prob)
276
277                 sim.setConfigurations(initial_time=0., end_time=50.,
278                 ↪ domain=mod.dom, time_variable_name='t_E0',
279                 ↪ is_dynamic=True, print_output=True,
280                 ↪ compile_equations=True,

```

```

277         output_headers=["Time", "Substrate(S)", "Biomass(X)",
278             "KeyComponents(E)", "Ethanol(P)"],
279         variable_name_map={"t_E0": "Time(t)",
        ↪   "S_E0": "Substrate(S)", "X_E0": "Biomass(X)",
        ↪   "E_E0": "KeyComponents(E)", "P_E0": "Ethanol(P)"},
        ↪   differential_solver='ODEINT')

280
281     sim.runSimulation()
282
283
284     print("\nDrawing for {} (Din = {} S_0 =
        ↪   {})".format(var, Din, S0s[i]))
285
286
287     results = sim.getResults('dict')
288
289     time = results['t_E0']['Time(t)']
290     X = results['t_E0']['Biomass(X)']
291     S = results['t_E0']['Substrate(S)']
292     E = results['t_E0']['KeyComponents(E)']
293     P = results['t_E0']['Ethanol(P)']
294
295     if var == 'X':
296
297         y = X
298         ylabel = r'$X\,(kg\,m^{-3})$'
299
300     if var == 'S':
301
302         y = S
303         ylabel = r'$S\,(kg\,m^{-3})$'
304
305     if var == 'P':
306
307         y = P
308         ylabel = r'$P\,(kg\,m^{-3})$'
309
310     if var == 'E':
311

```

```

312         y = E
313         ylabel = r'$E\,(kg\,m^{-3})$'
314
315         plt.plot(time, y, linestyle=lss[i], color=cs[i],
316                 ↪ label=r'$S_{0}$'+str(S0s[i])+'$')
317
318         xlabel = r'Tempo $(h)$'
319
320         #Produção dos gráficos
321         plt.xlim(0,50.)
322         plt.xlabel(xlabel, fontsize=12)
323         plt.ylabel(ylabel, fontsize=12)
324         plt.legend()
325         plt.grid()
326         plt.savefig('zym_openloop_'+str(var)+'_S0.png',
327                 ↪ bbox_inches='tight')
328         plt.clf()

```

B.2 Estudo de controle do processo de produção de bioetanol

```

1  #Importando bibliotecas da ferramenta SLOTH
2  from sloth.model import *
3  from sloth.problem import Problem
4  from sloth.simulation import Simulation
5  from sloth.optimization import Optimization, OptimizationProblem
6  from sloth.core.domain import Domain
7
8  #Bibliotecas adicionais (numpy, matplotlib)
9  import numpy as np
10 import matplotlib.pyplot as plt
11 from scipy.integrate import simps
12
13 #=====
14 #Configurações importantes para o estudo de controle
15 ethanol_sp = 65. #Set-point de concentração de etanol
16 process_end_time = 20. #Duração da batelada
17 #=====
18

```

```

19 #Modelo de produção de bioetanol
20 class ethanol_model(Model):
21
22     def __init__(self):
23
24         super().__init__(name="E0", description="Ethanol optimization
25         ↪ problem")
26
27         self.t = self.createVariable("t", dimless, "t")
28         self.dom = Domain("domain", dimless, self.t, "generic domain")
29
30         self.S = self.createVariable("S", dimless, "S")
31         self.E = self.createVariable("E", dimless, "E")
32         self.X = self.createVariable("X", dimless, "X")
33         self.P = self.createVariable("P", dimless, "P")
34
35         self.S.distributeOnDomain(self.dom)
36         self.E.distributeOnDomain(self.dom)
37         self.X.distributeOnDomain(self.dom)
38         self.P.distributeOnDomain(self.dom)
39
40         #Grau de liberdade: variável manipulada
41         self.Din = self.createParameter("Din", dimless, "")
42         #self.Din.setValue(1.)
43         self.S_0 = self.createParameter("S_0", dimless, "")
44         self.S_0.setValue(150.3)
45
46         self.K_P = self.createParameter("K_P", dimless, "")
47         self.K_P.setValue(1.)
48         self.Y_sx = self.createParameter("Y_sx", dimless, "")
49         self.Y_sx.setValue(.0244)
50         self.m_s = self.createParameter("m_s", dimless, "")
51         self.m_s.setValue(2.16)
52         self.m_p = self.createParameter("m_p", dimless, "")
53         self.m_p.setValue(1.1)
54         self.Dout = self.createParameter("Dout", dimless, "")
55         self.Dout.setValue(.5)
56         self.K_s = self.createParameter("K_s", dimless, "")
57         self.K_s.setValue(.5)

```

```

57     self.X_0 = self.createParameter("X_0", dimless, "")
58     self.X_0.setValue(0.08)
59     self.k1 = self.createParameter("k1", dimless, "")
60     self.k1.setValue(16.)
61     self.k2 = self.createParameter("k2", dimless, "")
62     self.k2.setValue(.497)
63     self.k3 = self.createParameter("k3", dimless, "")
64     self.k3.setValue(.0038)
65     self.E_0 = self.createParameter("E_0", dimless, "")
66     self.E_0.setValue(.02)
67     self.P_0 = self.createParameter("P_0", dimless, "")
68     self.P_0.setValue(0.)
69     self.Y_px = self.createParameter("Y_px", dimless, "")
70     self.Y_px.setValue(.0526)
71
72     def DeclareEquations(self):
73
74         eq1 = self.S.Diff(self.t) ==
75         ↪ (-1./self.Y_sx()*(self.S()*self.E()/(self.K_s()+self.S()))
76         ↪ - self.m_s()*self.X() + self.Din()*self.S_0() -
77         ↪ self.Dout()*self.S()
78     self.eq1 = self.createEquation("eq1", "", eq1)
79
80         eq2 = self.X.Diff(self.t) ==
81         ↪ (self.K_P()*(self.S()*self.E()/(self.K_s()+self.S())) +
82         ↪ self.Din()*self.X_0() - self.Dout()*self.X()
83     self.eq2 = self.createEquation("eq2", "", eq2)
84
85         eq3 = self.E.Diff(self.t) == (self.k1()-self.k2()*self.P() +
86         ↪ self.k3()*(self.P()**2.))*self.S()*self.E()/(self.K_s()+self.S())
87         ↪ + self.Din()*self.E_0() - self.Dout()*self.E()
88     self.eq3 = self.createEquation("eq3", "", eq3)
89
90         eq4 = self.P.Diff(self.t) ==
91         ↪ (1./self.Y_px()*(self.S()*self.E()/(self.K_s()+self.S()))
92         ↪ + self.m_p()*self.X() + self.Din()*self.P_0() -
93         ↪ self.Dout()*self.P()
94     self.eq4 = self.createEquation("eq4", "", eq4)
95

```

```

86 #Definindo um problema de otimização da produção de bioetanol (classe
    ↪ derivada de OptimizationProblem)
87 class ethanol_max_prob(OptimizationProblem):
88
89     def __init__(self, number_of_dimensions, mod, initial_time,
    ↪ end_time, initial_conditions):
90
91         super().__init__(number_of_dimensions)
92
93         self.mod = mod
94
95         self.initial_time = initial_time
96
97         self.end_time = end_time
98
99         self.initial_conditions = initial_conditions
100
101     def DeclareObjectiveFunction(self, x):
102
103         Din = x[0]
104
105         self.simulation_instance[self.mod.Din].setValue(Din)
106
107         self.simulation_instance.problem.setTimeVariableName(['t_EO'])
108
109         self.simulation_instance.problem.resolve()
110
111         self.simulation_configuration['initial_time'] =
    ↪ self.initial_time
112
113         self.simulation_configuration['end_time'] = self.end_time
114
115         self.simulation_instance.setConfigurations(
116             definition_dict=self.simulation_configuration)
117
118         self.simulation_instance.runSimulation()
119
120         result = self.simulation_instance.getResults('dict')
121

```

```

122     ethanol = np.array(result['t_E0']['Ethanol(P)'][:])
123     keys = np.array(result['t_E0']['KeyComponents(E)'][:])
124     biomass = np.array(result['t_E0']['Biomass(X)'][:])
125     substrate = np.array(result['t_E0']['Substrate(S)'][:])
126     time = np.array(result['t_E0']['Time(t)'][:])
127
128     f = simps((ethanol-ethanol_sp)**2., time)
129
130     self.simulation_instance.reset()
131
132     if result['t_E0']['Ethanol(P)'][-1] < 0. :
133
134         f = 1e100
135
136         #if resultt['t_E0'][:][-1] < 0. :
137             #f = 1e100
138
139         #print("\n\n==>OBJ = ",f)
140         #print("\n\n==>x = ", x)
141
142     return [f]
143
144     def DeclareSetBounds(self):
145
146         return(tuple(self.bounds))
147
148     #Retorna um objeto Problem
149     def problem_pse():
150
151         return Problem("problem_PSE", "Ethanol maximization problem")
152
153     #Retorna um objeto Simulation
154     def simulation_pse():
155
156         return Simulation("problem_PSE", "Ethanol maximization problem")
157
158
159     #Realiza uma otimização da variável manipulada
160     def run_optimization(initial_time, end_time, initial_conditions):

```

```

161
162     sim = simulation_pse()
163
164     mod = ethanol_model()
165
166     mod()
167
168     prob = problem_pse()
169
170     prob_opt = ethanol_max_prob(1, mod, initial_time, end_time,
171     ↪ initial_conditions)
172
173     class opt_study(Optimization):
174
175         def __init__(self, simulation, optimization_problem,
176         ↪ simulation_configuration, optimization_parameters,
177         ↪ constraints, optimizer, optimization_configuration=None):
178
179             super().__init__(simulation=sim,
180             ↪ optimization_problem=optimization_problem,
181             ↪ simulation_configuration=simulation_configuration,
182             ↪ optimization_parameters=optimization_parameters,
183             ↪ constraints=constraints,
184             ↪ optimization_configuration=optimization_configuration,
185             ↪ optimizer=optimizer)
186
187     sim.setConfigurations(initial_time=initial_time, end_time=end_time,
188     ↪ domain=mod.dom, time_variable_name='t_E0', is_dynamic=True,
189     ↪ print_output=False, compile_equations=True,
190     ↪ output_headers=["Time", "Substrate(S)", "Biomass(X)",
191     ↪ "KeyComponents(E)", "Ethanol(P)"],
192     ↪ variable_name_map={"t_E0": "Time(t)", "S_E0": "Substrate(S)",
193     ↪ "X_E0": "Biomass(X)",
194     ↪ "E_E0": "KeyComponents(E)", "P_E0": "Ethanol(P)"},
195     ↪ differential_solver='ODEINT')
196
197     prob.addModels(mod)
198
199     prob.setTimeVariableName(['t_E0'])

```



```
191
192     prob.resolve()
193
194     prob.setInitialConditions(initial_conditions)
195
196     sim.setProblem(prob)
197
198     prob_opt()
199
200     opt = opt_study(simulation=sim, optimization_problem=prob_opt,
201     ↪ simulation_configuration=None,
202     ↪ optimization_parameters=[mod.Din], constraints=([0.],[3.]),
203     ↪ optimizer='de')
204
205
206     opt.optimization_problem()
207
208     opt.runOptimization(report_frequency=1)
209
210     Din = opt.best_parameters[-1]
211
212     J = opt.best_fitness[-1]
213
214     return(Din, J)
215
216
217     #Faz a simulação do modelo para um dado intervalo, a partir de
218     ↪ condições iniciais e
219     # do valor da variável manipulada
220     def run_simulation(initial_time, end_time, initial_conditions, Din):
221
222         sim = simulation_pse()
223
224         mod = ethanol_model()
225
226         mod.Din.setValue(Din)
227         mod()
228
229         prob = problem_pse()
230
231         prob.addModels(mod)
```

```

226
227     prob.setTimeVariableName(['t_E0'])
228
229     prob.resolve()
230
231     #{'t_E0':0., 'S_E0':150.3, 'X_E0':.08, 'E_E0':.02, 'P_E0':0.}
232     prob.setInitialConditions(initial_conditions)
233
234     sim.setProblem(prob)
235
236     sim.setConfigurations(initial_time=initial_time, end_time=end_time,
237     ↪ domain=mod.dom, time_variable_name='t_E0', is_dynamic=True,
238     ↪ print_output=False, compile_equations=True,
239     ↪ output_headers=["Time(t)", "Substrate(S)", "Biomass(X)",
240     ↪ "KeyComponents(E)", "Ethanol(P)"],
241     ↪ variable_name_map={"t_E0": "Time(t)", "S_E0": "Substrate(S)",
242     ↪ "X_E0": "Biomass(X)",
243     ↪ "E_E0": "KeyComponents(E)", "P_E0": "Ethanol(P)"},
244     ↪ differential_solver='ODEINT')
245
246     sim.runSimulation()
247
248     results = sim.getResults('dict')
249
250     t_hist = np.array(results['t_E0']['Time(t)'])
251
252     X_hist = np.array(results['t_E0']['Biomass(X)'])
253
254     S_hist = np.array(results['t_E0']['Substrate(S)'])
255
256     E_hist = np.array(results['t_E0']['KeyComponents(E)'])
257
258     P_hist = np.array(results['t_E0']['Ethanol(P)'])
259
260     return (t_hist, X_hist, S_hist, E_hist, P_hist)
261
262
263     #Realiza um estudo de DRTO com set-point tracking
264     def run_drto(initial_time, end_time, number_of_intervals,
265     ↪ compile_equations=True, draw_plots=False):

```

```

260
261     sim = simulation_pse()
262
263     mod = ethanol_model()
264
265     dt = (end_time - initial_time)/number_of_intervals
266
267     best_Din = np.zeros(number_of_intervals)
268
269     initial_conditions = {'t_E0':0., 'S_E0':150.3, 'X_E0':.08,
    ↪ 'E_E0':.02, 'P_E0':0.}
270
271     start = initial_time
272
273     t_hist, X_hist, S_hist, E_hist, P_hist = np.array([]),
    ↪ np.array([]), np.array([]), np.array([]), np.array([])
274
275     for k in range(number_of_intervals):
276
277         end = start + dt
278
279         print("\n\n===== Running instance #{} / {} [{}~{}]"
    ↪ =====".format(k+1, number_of_intervals, start, end))
280
281         #Obter valor para Din(variável manipulada) e J(função
    ↪ objetivo)
282
283         Din, J = run_optimization(start, end_time, initial_conditions)
284
285         best_Din[k] = Din
286
287         print("\n\t\t\t---> D_in = {}      J = {}".format(Din, J))
288
289         #Realiza a simulação para um novo estado
290
291         t_, X_, S_, E_, P_ = run_simulation(start, end,
    ↪ initial_conditions, Din)
292
293         initial_conditions = {'t_E0':end, 'S_E0':S_[-1], 'X_E0':X_[-1],
    ↪ 'E_E0':E_[-1], 'P_E0':P_[-1]}

```

```

294
295     start += dt
296
297     #Store the results
298
299     t_hist = np.append(t_hist, t_)
300     X_hist = np.append(X_hist, X_)
301     S_hist = np.append(S_hist, S_)
302     E_hist = np.append(E_hist, E_)
303     P_hist = np.append(P_hist, P_)
304
305     final_J =.simps((P_hist - ethanol_sp)**2., t_hist)
306     final_J2 =.simps((P_hist - ethanol_sp)**2.)
307
308     print("\n\n====> Final results: Din = {}      J = {} or
309     ↪ {}".format(best_Din, final_J, final_J2))
310
311     #Produção de gráficos
312     if draw_plots is True:
313
314         vars = ['X', 'S', 'P', 'E']
315
316         for i in range(len(vars)):
317
318             var = vars[i]
319
320             if var == 'X':
321
322                 y = X_hist
323                 ylabel = r'$X\,(kg\,m^{-3})$'
324
325             if var == 'S':
326
327                 y = S_hist
328                 ylabel = r'$S\,(kg\,m^{-3})$'
329
330             if var == 'P':
331
332                 y = P_hist

```

```
332         ylabel = r'$P\,(kg\,m^{-3})$'
333
334     if var == 'E':
335
336         y = E_hist
337         ylabel = r'$E\,(kg\,m^{-3})$'
338
339     plt.plot(t_hist, y, color='k')
340
341     if var == 'P':
342
343         sp_ = np.zeros(t_hist.shape)
344
345         sp_.fill(ethanol_sp)
346
347         plt.plot(t_hist, sp_, linestyle='--', color='r',
348                 ↪ label=r'set-point')
349
350     xlabel = r'Tempo $(h)$'
351
352     plt.xlim(0,t_hist[-1])
353     plt.xlabel(xlabel, fontsize=12)
354     plt.ylabel(ylabel, fontsize=12)
355
356     if var == 'P':
357         plt.legend()
358
359     plt.grid()
360     plt.savefig('zym_drto_' + var +
361                 ↪ '_DE_'+str(number_of_intervals) + '.png',
362                 ↪ bbox_inches='tight')
363     plt.clf()
```