Adriano da Cruz Dias

Simulação e Optimização de Sistemas Modelados por Equações Algébrico-Diferenciais

Dissertação de Mestrado Integrado em Engenharia Química,
apresentada ao Departamento de Engenharia Química da
Faculdade de Ciências e Tecnologia da Universidade de Coimbra

Supervisor:
Professor Doutor Nuno Manuel Clemente de Oliveira

Instituição:
Departamento de Engenharia Química
Faculdade de Ciências e Tecnologia da Universidade de Coimbra

UNIVERSIDADE D
COIMBRA

Coimbra 2018

Agradecimentos

Como em dinâmica de sistemas, em que resultado final é a soma dos efeitos de todas as variáveis, e não só da que tem maior contribuição, assim este trabalho, também sofreu influências de um conjunto de pessoas.

Irei aproveitar este pequeno espaço para agradecer às pessoas quem mais me apoiaram neste trabalho.

Comecerei por agradecer ao Professor Doutor Nuno Oliveira por me ter apoiado ao longo deste trabalho.

Gostaria também deixar um especial obrigado ao Professor Doutor Lino Santos, por estar sempre pronto a disponibilizar informações, bem como me ter apoiado no início desta aventura que foi a escrita em LaTeX.

Deixo aqui ainda um grande obrigado à minha colega e amiga Filipa Lopes, por todo o apoio que me deu ao logo desta dissertação, sendo muitas vezes o meu braço direito.

Queria ainda agradecer ao Joel Sansana, Nélson Assunção (Júnior) e a Carolina Curado, pelo apoio que me deram durante este tempo.

Quero aproveitar ainda este espaço para deixar um agradecimento a todos aqueles que fizeram parte desta minha jornada de 5 anos, principalmente a Beatriz Banaco, Eugeniu Strelet, Inês Inocêncio, Joel Sansana, Júnior, e todos os outros que por motivos de espaço não estão aqui referidos.

Por último, queria deixar um especial agradecimento à minha família, pois sem os sacrifícios deles, nada disto seria possível.

Resumo

Nos últimos anos tem se vindo a assistir a um crescimento das capacidades computacionais, o que abriu a possibilidade de resolução de novos problemas e de problemas mais complexos. Um destes problemas são as Sistema de equações algébrico-diferencias (EADs) (DAES).

Sendo o principal objectivo deste trabalho a simulação e a optimização de problemas que recorram a DAES, fez-se uma pesquisa sobre as ferramentas com esta capacidade, chegando a ferramentas como o Pyomo ou o JuMP, sendo ambas ferramentas suportadas por linguagens *open-source*, Python e Julia, respectivamente.

Optou-se por modelar um sistema tipo (Reactor perfeitamente agitado (RPA)) recorrendo ao Wolfram Mathematica® e ao Pyomo, percebendo o comportamento destas ferramentas na resolução de DAES. Na simulação os graus de liberdade do sistema eram zero. Os resultados obtidos no Pyomo foram comparados com os resultados obtidos do Wolfram Mathematica® , ferramenta de simulação com algoritmos sofisticados para a resolução deste tipo de problemas.

Para a análise do modelo tipo, foram feitos vários testes dinâmicos ao sistema, sendo posteriormente programado um controlador Proporcional, Integral (PI) e incorporado neste sistema. Para este último modelo, foi provocada uma perturbação numa variável de entrada e foi feita uma alteração no *set-point*.

Após os vários teste, com a comparação dos resultados entre o Pyomo e o Wolfram Mathematica® , permitiu-se concluir que o Pyomo , mesmo não sendo um programa dedicado à simulação, mas sim à optimização de sistemas modelados com DAES, permite obter tão bons resultados quanto o Wolfram Mathematica® na parte da simulação. O próximo passo será a optimização deste sistema e posteriormente a passagem para sistemas mais complexos e reais como é o caso da coluna de absorção de NO_x.

Palavras-chave: Equações Algébrico-Diferenciais; Métodos Numéricos; Simulação; Optimização; Pyomo.

Abstract

In recent years there has been a growth in computing capacity, which has opened up the possibility for solving new problems and more complex ones. A good example is the System of Differential Algebraic Equations(DAEs).

Since the main objective of this work is the simulation and optimization of problems using DAEs, a research was developed on the tools with this capacity, reaching tools like Pyomo or JuMP, both being supported by open-source languages, Python and Julia, respectively.

It was decided to model a type system (Stirred-Tank Reactor (RPA)) using the Pyomo, understanding the behavior of this tool in solving problems of the sort. In the simulation the degrees of freedom of the system were zero. The results obtained from Pyomo were compared with the results obtained from Wolfram Mathematica® , a simulation tool with sophisticated algorithms to solve this type of problems.

For the analysis of the type model, several dynamic tests were performed on the system, and a Proportional, Integral (PI) controller was programmed and incorporated into the system. For the latter model, a perturbation was caused in an input variable and a change was made in the set-point.

Afterwards, with the comparison of results between Pyomo and Wolfram Mathematica® , it was concluded that Pyomo, normally used as an optimization program, managed to obtain results just as good as Wolfram Mathematica® 's regarding simulation. The next step was the optimization of the system and further transition to more complex and real systems such as the NO_x absorption column.

Keywords: Differential Algebraic Equations; Numerical Methods; Simulation; Optimization; Pyomo.

Conteúdo

Lista de Figuras

Lista de Tabelas

Acrónimos

AMPL *A Mathematical Programming Language*. 27, 28

BLT *Block Lower Triangular*. 15, 20, 22

CBC *COIN-OR Branch and Cut*. 28

COIN-OR *Computational Infrastructure for Operations Research*. 28

DAES Sistema de equações algébrico-diferencias (EADs). iii, 1–3, 5–9, 11–15, 17–20, 22–26, 28, 30, 32, 33, 41, 43, 53, 61, 62

DFC Diferenças Finitas Centradas. 31

DFP Diferenças Finitas Progressivas. 31

DFR Diferenças Finitas Regressivas. 18, 30, 31

GAMS *General Algebraic Modeling System*. 27, 61

GLPK *GNU Linear Programming Kit*. 28

IMC *Internal Model Control*. 51, 52

IPOPT *Interior Point OPTimizer*. 28, 32

LP Problemas Lineares. 31

MILP *Mixed-Integer Linear Problems*. 31

MIMO *Multiple-input, Multiple-output*. 1

MINLP *Mixed-Integer NonLinear Problems*. 32

MIQP *Mixed-Integer Quadratic Programs*. 32

NEOS *Network-Enabled Optimization System*. 28

NLP Problemas Não Lineares. 31–33

ODE Equações Diferencial Ordinária (EDO). 5, 7, 8, 11, 12, 15, 24, 43, 68–70

PI Proporcional, Integral. iii, 51, 52, 54

PID Proporcional, Integral e Diferencial. 1, 50–52

PVI Problema de Valor Inicial. 9–11, 18

QP *Quadratic Programs*. 32

RPA Reactor perfeitamente agitado. iii, 3, 35, 37, 61

SQP Sequential Quadratic Programming. 32

SUNDIALS *SUite of Nonlinear and Differential/ALgebraic Equation Solvers*. 18–20, 22, 32

Simbologia

A Área da secção recta do reactor prependicular ao escoamento $[m^2]$. 38

C_A Concentração do composto A no reactor $[mol/m^3]$. ix, 36, 39, 41–46, 49, 55

C_B Concentração molar do composto B no reactor $[mol/m^3]$. ix, 36, 41–46, 49, 55

C_C Concentração molar do composto C no reactor $[mol/m^3]$. ix, 36, 42–46, 49, 55

C_D Concentração molar do composto D no reactor $[mol/m^3]$. ix, 36, 42–47, 49, 55, 56

C_{A0} Concentração molar do composto A à entrada $[mol/m^3]$. ix, xi, 36, 39, 42, 44, 45, 48, 49

C_{B0} Concentração molar do composto B à entrada $[mol/m^3]$. 36, 42, 44

C_{C0} Concentração molar do composto C à entrada $[mol/m^3]$. 36, 42, 44

C_{D01} Concentração molar do composto D à entrada na corrente 1 $[mol/m^3]$. 40, 44

C_{D02} Concentração molar do composto D à entrada na corrente 2 $[mol/m^3]$. 40, 44

$F_{1_{SS}}$ Caudal volumétrico de 1 em estádo estacionário $[m^3/min]$. 53

F_1 Caudal volumétrico de entrada 1 $[m^3/min]$. ix–xi, 36, 37, 42, 44–49, 51, 54, 55

F_2 Caudal volumétrico de entrada 2 $[m^3/min]$. ix, x, 36, 37, 42, 44, 48, 51, 53, 55

F_3 Caudal volumétrico de saída 3 $[m^3/min]$. ix, 36–38, 42, 44, 45, 47, 48, 53, 55

K_c Ganho Proporcional[min^{-1}]. 51

$K_{F_1,V}$ Ganho do Processo $[min]$. 51

K_{eq} constante de equilíbrio entre a espécie B e a espécie A [−]. 41, 42

V Volume de líquido no reactor $[m^3]$. 39, 42–44, 47, 48, 51

V_{SP} *Set-point* do volume de líquido no reactor $[m^3]$. 52

β_1 coeficiente de descarga do reactor quando F_3 e escrito em função de h. 38

β_2 coeficiente de descarga do reactor quando F_3 e escrito em função de V $[m^2/min]$. 38, 42, 44

$\dot{m}_{1_A}$ Caudal mássico de A em 1 $[kg/min]$. 38

$\dot{m}_{1_D}$ Caudal mássico de D em 1 $[kg/min]$. 40

$\dot{m}_1$ Caudal mássico de 1 $[kg/min]$. 37

$\dot{m}_{2_C}$ Caudal mássico de C em 2 $[kg/min]$. 39

$\dot{m}_{2_D}$ Caudal mássico de D em 2 $[kg/min]$. 40

$\dot{m}_2$ Caudal mássico de 2 $[kg/min]$. 37

$\dot{m}_{3_A}$ Caudal mássico de A em 3 $[kg/min]$. 38

$\dot{m}_{3_C}$ Caudal mássico de C em 3 $[kg/min]$. 39

$\dot{m}_{3_D}$ Caudal mássico de D em 3 $[kg/min]$. 40

$\dot{m}_3$ Caudal mássico de 3 $[kg/min]$. 37

ρ_1 Densidade da corrente 1 $[kg/m^3]$. 37

ρ_2 Densidade da corrente 2 $[kg/m^3]$. 37

ρ_3 Densidade da corrente 3 $[kg/m^3]$. 37

ρ_{global} Densidade global do sistema $[kg/m^3]$. 37

τ Constante de tempo $[min]$. 51

τ_c Constante de tempo em ciclo fechado $[min]$. 51

τ_I Constante de tempo integral $[min]$. 51

θ Atraso do sistema $[min]$. 51

$d(C_A V)/dt$ Variação da quantidade química A ao longo do tempo $[mol/min]$. 39

dC_C/dt Variação da concentração do composto C ao longo do tempo $[mol/m^3.min]$. 40

dC_D/dt Variação da concentração do composto D ao longo do tempo $[mol/m^3.min]$. 40

dV/dt Variação da volume ao longo do tempo $[m^3/min]$. 37

dm/dt Variação da massa ao longo do tempo $[kg/min]$. 37

dm_A/dt Variação da mássica do composto A ao longo do tempo $[kg/min]$. 38

dm_C/dt Variação da massa do composto C ao longo do tempo $[kg/min]$. 39

dm_D/dt Variação da massa do composto D ao longo do tempo $[kg/min]$. 40

h Altura de fluido no reactor $[m]$. 37

k_1 velocidade de reacção $[m^3/(mol.min)]$. 42

NO_x Óxidos de Azoto. iii, v, x, 2, 3, 35, 57, 58, 65

Capítulo 1

Introdução

Nos últimos anos tem-se assistido a um aumento dos recursos computacionais, tendo impacto não só no nosso quotidiano, mas principalmente na pesquisa cientifica e na parte industrial. Uma das principais áreas onde a evolução tecnológica tem um efeito mais acentuado, dentro da área da Engenharia Química, é na área de projecto do processo, do produto e na área da monitorização e controlo de processos.

Se nos dois primeiros tópicos já é comum o recurso a computação avançada, quer no projecto sistemático de um processo, como nas simulações computacionais de moléculas, para o caso do projecto de produto, no controlo de processos ainda existe uma grande resistência por grande parte da indústria. Esta atitude por parte da indústria deve-se à complexidade matemática destes sistemas, e por estes sistemas de controlo não substituírem por completo a necessidade de sistemas mais simples, como o caso de controladores Proporcional, Integral e Diferencial (PID), principalmente por questões de segurança (Aspuru-Guzik et al. 2018).

Apesar da resistência à sua utilização, os sistemas avançados de controlo apresentam vantagens, podendo trabalhar em *Multiple-input, Multiple-output* (MIMO), promovendo melhores condições de operação e melhorar a resposta às perturbações. Estas possibilidades permitem obter mais produto dentro das especificações e equipamentos mais saudáveis.

No entanto, a grande desvantagem destes sistemas consiste, como já referido, na complexidade matemática dos modelos, contendo muitas vezes sistemas de equações diferenciais, e em alguns dos casos DAES.

As DAES aparecem da conjugação da parte dinâmica dos sistemas, representados por equações diferenciais, com a existência de equilíbrios ou restrições de processos, sendo estes representados recorrendo a equações algébricas, como referido no Capítulo 2 (Geletu 2011).

Apesar de todas as resistências oferecidas, cada vez mais assiste-se a uma lógica de informatização e automatização dos processos, tendo por este motivo cada vez mais interesse o aparecimento de métodos e ferramentas informáticas que simplifiquem a resolução de sistemas, mas

que mantenha a eficácia e rapidez pretendida para as operações de controlo.

Um exemplo onde esta tecnologia de controlo pode ser aplicada é no controlo de emissões de NO_x.

Este problema atinge as fábricas de Ácido Nítrico, onde apesar de existir sistemas de tratamento para esta família de compostos, estes só funcionam convenientemente quando a instalação está a operar em "velocidade cruzeiro". Nas fases de arranque e de paragem, estes sistemas, derivado às grandes variações de concentrações de NO_x e às condições de operação, não respondem correctamente, sendo por este motivo emitidas grandes concentrações de NO_x para a atmosfera. Uma alternativa para adaptar estes sistemas de tratamento às situações "anormais", mas comuns, seria aplicar uma estratégia de controlo avançado.

Estas emissões para além de todo o prejuízo ambiental, como chuvas ácidas, ainda acarretam prejuízos para a indústria, pois a emissão destes compostos é controlada, e caso se ultrapasse o limite máximo de emissões atribuído à empresa esta será penalizada economicamente. Estas emissões também implicam uma perda de reagentes, que poderiam ser integrados na produção de ácido nítrico. É ainda importante salientar que a reputação da indústria também sai prejudicada, pois estas são caracterizadas por nuvens de um tom acastanhado.

Derivado à complexidade do sistema e por sua vez do modelo correspondente e como parte do objectivo deste trabalho serão testadas novas ferramentas para aplicar a estratégias de controlo avançado, recorrendo-se a um sistema tipo.

Neste caso será um reactor perfeitamente agitado, como uma reacção genérica de B mais C a dar D, estando B em equilíbrio com A, como representado pela equações (1.1) e (1.2).

$$A \rightleftharpoons B \tag{1.1}$$

$$B + C \rightarrow D \tag{1.2}$$

Este sistema irá originar um sistema de DAES, uma vez que será constituído por uma equação linear, que representa o equilíbrio químico (1.1), e equações diferenciais, que traduzem os fenómenos dinâmicos, neste caso a dinâmica reaccional.

Este sistema será apresentado de uma forma mais detalhada na secção 4.1.

1.1 Estrutura

No Capítulo 2 são apresentados os fundamentos matemáticos das DAES, ou seja, paticularidades, formas canónicas e problemas mais comuns neste tipo de sistemas.

No Capítulo 3, serão apresentadas tanto as ferramentas computacionais com capacidade de resolver DAES como as ferramentas dedicadas à optimização de sistemas.

Serão então apresentados os sistemas em estudo, no Capítulo 4, nomeadamente o caso do RPA (secção 4.1) e o caso da coluna de absorção de NO_x(secção 4.2). Neste capítulo serão ainda apresentados os resultados obtidos para o caso da simulação do RPA.

Por fim, no Capítulo 5 são apresentadas as conclusões retiradas deste estudo, e propostas de trabalhos futuros.

Capítulo 2

Fundamentos Matemáticos

Neste capítulo, seram apresentadass as principais características dos problemas descritos recorrendo a DAES, assim como as suas particularidades. Começa-se por avaliar o motivo do seu recurso para simular em sistemas reais (2.1), explicando a teoria de um índice de um DAES (2.1.1), e apresentando algumas estratégias de resolução.

Os sistemas com DAES requerem algumas atenções relativamente aos sistemas com Equações Diferencial Ordinária (EDO) (ODE), pois estes são mais sensíveis ao passo de integração assim como às condições iniciais, que as ODEs, sendo, por este motivo, necessário ter cuidados especiais na formulação e resolução. Sendo assim a resolução destes tipos de problemas recorrem a métodos numéricos específicos, assim como a *softwares* especializados. Para além destas características, as DAES ainda apresentam outros problemas como os apresentados em 2.1.3.

2.1 DAES

Um DAES, como o próprio nome indica, é composto por equações diferencias e por equações algébricas.

Este sistema de equações são muito comuns para modelar sistemas em Engenharia de Processo, Engenharia Mecânica, Engenharia Electrotécnica, sistemas termodinâmicos, entre outros.

Nestes sistemas, as equações diferenciais costumam estar associadas à dinâmica de sistemas. Já as equações algébricas estão associadas a leis da Física e da Termodinâmica, como a lei da conservação de massa e energia, leis de equilíbrio de compostos e a balanços mássicos, molares ou entálpicos. Outra utilização das equações algébricas é em restrições ao processo, uma vez que estas podem ter uma origem termodinâmica, ou física, como é o caso da solubilidade de um composto, ou ainda, do processo, como é o caso de uma capacidade máxima que um tanque ou reactor têm (Geletu 2011).

A Forma Geral das DAES (*General Form of DAES*) (Ascher & Petzold 1994, Wolfram 2018*a*, Hairer & Wanner 1996) encontra-se apresentada em seguida:

$$\mathrm{F}(t,\mathrm{y},\mathrm{y}') = 0 \tag{2.1}$$

Na equação (2.1), **F** representa o sistema de equações, sendo t a variável independente, **y** o vector de variáveis dependente de t e **y'** a primeira derivada de **y** (Ascher & Petzold 1994).

Contudo, muitas vezes esta não é a forma mais comum de apresentar DAES, existindo outras formas de as representar, adequando-se melhor, tanto aos problemas como os métodos de resolução, sendo um exemplo a forma de *Hessenberg*.

Exemplo de aplicação - Caso do Pêndulo

A análise do comportamento de um pêndulo ao longo do tempo, é um dos casos mais comuns da literatura estando presente em Ascher & Petzold (1994), Wolfram (2018*a*), The MathWorks (2018*c*).

Ir-se-à recorrer a este caso de estudo, de modo a facilitar a compreensão dos diferentes conceitos.

Na Figura 2.1 está apresentado o esquema representativo do sistema.

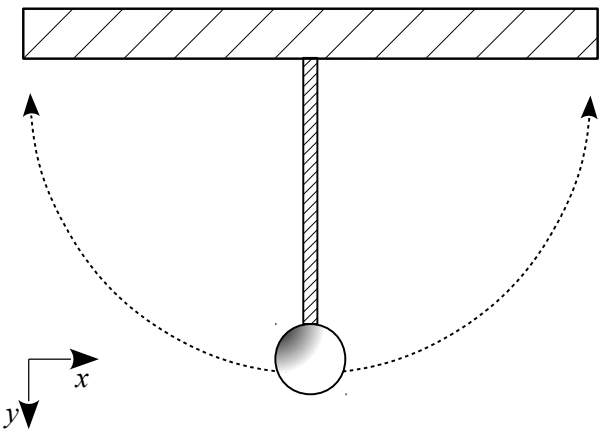


Figura 2.1 Esquema de um pêndulo

Em seguida, será apresentado o conjunto de equações necessárias para a modelação do pêndulo.

$$x'(t) = v_x(t) \tag{2.2}$$

, sendo $x(t)$ uma função que representa a variação da posição ao longo do tempo relativamente à coordenada x, a sua primeira derivada $x'(t)$, representará a variação ao longo do tempo da velocidade, segundo a componente x, sendo representada por $v_x(t)$ (Ascher & Petzold 1994).

$$y'(t) = v_y(t) \tag{2.3}$$

De forma análoga ao que acontece em (2.2), em (2.3), a velocidade segundo a componente y ($v_y(t)$) é igual à primeira derivada da função $y(t)$, função que representa a variação da posição ao longo do tempo segundo a coordenada y (Ascher & Petzold 1994).

$$v_x'(t) = -\lambda(t)x(t) \tag{2.4}$$

A equação (2.4), representa a aceleração do pêndulo segundo a componente x, sendo esta a derivada da velocidade na mesma componente ($v_x'(t)$), ou seja, variação da posição (x) multiplicado pela tensão no cabo ($\lambda(t)$) (Brenan et al. 1996, Ascher & Petzold 1994).

$$v_y'(t) = -\lambda(t)y(t) - g \tag{2.5}$$

Em (2.5), é apresentada a equação que permite obter a variação da aceleração segundo a componente y ($v_y'(t)$), tendo novamente a tensão no cabo $\lambda(t)$, é ainda acrescentada a aceleração gravítica g (Brenan et al. 1996, Ascher & Petzold 1994).

$$0 = x(t) + y(t) - l \tag{2.6}$$

Nesta última equação é apresentada uma restrição linear, garantindo que a distancia (l) entre o centro de massa do pêndulo e o ponto fixo se mantém constante (Ascher & Petzold 1994).

2.1.1 Índice das DAES

Na resolução de DAES, um dos pontos importantes é saber a complexidade destes sistemas, e o quão longe estão das ODEs. Para tal, os DAES, são classificadas pelo número de diferenciações necessárias para as transformar em ODEs. A este índice é chamado de Índice de Diferenciação (*Differentiation Index*) (Hairer & Wanner 1996).

Por exemplo, um DAES que se tenha de diferenciar uma vez terá um Índice 1, uma DAES em que se diferencia duas vezes terá um Índice 2, e assim consecutivamente.

No entanto existem variações na classificação recorrendo ao Índice, como é o caso do Índice na Forma de *Hessenberg*. a metodologia aplicada é idêntica, mas como as equações foram reestruturadas para ficarem na forma de *Hessenberg*, muitas vezes obtém-se índices inferiores aos que eram calculados na forma geral (Ascher & Petzold 1994, Guzel & Bayram 2006).

Caso do Pêndulo - Índice de Diferenciação

Para melhor compreender o funcionamento das classificação ir-se-á pegar no exemplo do pêndulo ((2.2)-(2.6)).

Derivando a equação (2.6) obtém-se:

$$x(t)x(t)' + y(t)y(t)' = 0 \tag{2.7}$$

Substituindo as equações (2.2) e (2.3) em (2.7) obtém-se:

$$x(t)v_x(t) + y(t)v_y(t) = 0 \tag{2.8}$$

No próximo passo ir-se-á diferenciar a equação (2.8), obtendo:

$$x(t)v_x(t)' + y(t)v_y(t)' + v_x(t) + v_y(t) = 0 \tag{2.9}$$

Substituindo e simplificando (2.4) e (2.5) em (2.9) fica-se com:

$$-\lambda(t) - y(t)g + v_x(t) + v_y(t) = 0 \tag{2.10}$$

Da equação (2.10), pode-se evidenciar o $\lambda(t)$, substituindo em (2.4) e (2.5), obtendo:

$$v_x(t)' = (y(t)g - v_x(t) - v_y(t))x(t) \tag{2.11}$$

e

$$v_y(t)' = (y(t)g - v_x(t) - v_y(t))y(t) - g \tag{2.12}$$

Derivando a equação (2.10) obtém-se assim uma equação diferencial em ordem a λ, tendo assim um sistema de ODEs.

$$\lambda(t)' = v_y(t)g - v_x(t)v_x(t)' - v_y(t)v_y(t)' \tag{2.13}$$

Para atingir este ponto, foi necessário diferenciar três vezes, sendo por isto um problema de Índice 3 (Ascher & Petzold 1994).

2.1.2 Forma Canónicas de *Hessenberg*

A maioria dos DAES, podem ser representados por um grupo mais restrito de ODEs e equações algébricas. Este formato é chamado de *Hessenberg* (Guzel & Bayram 2006).

Esta é uma forma canónica, mais comum e útil, principalmente para sistemas semi-explícitos, pois estes sistemas de DAES, são condensados, traduzindo-se muitas vezes numa simplificação dos mesmos.

Esta simplificação deu origem a um novo sistema de DAES, e como tal, este pode ser derivado e classificado de forma idêntica, sendo posteriormente classificados como Índice da Forma de *Hessenberg* (Ascher & Petzold 1994).

Forma Cónica de *Hessenberg* - Índice 1

Consideremos o sistema com a segunte forma,

$$\frac{d\mathrm{x}}{dt} = f(t, \mathrm{x}, \mathrm{z}) \tag{2.14}$$

$$0 = g(t, \mathrm{x}, \mathrm{z}) \tag{2.15}$$

, contendo as seguintes condições iniciais:

$$0 = f(t_0, \mathrm{x}_0, \mathrm{z}_0) \tag{2.16}$$

$$0 = g(t_0, \mathrm{x}_0, \mathrm{z}_0) \tag{2.17}$$

onde estas, têm de ser consistentes com o sistema.

Ao longo desta redução de índice, presente no Anexo A.1 recorreu-se uma vez a diferenciação , logo tratamdo-se por este de um sistema de índice 1, para além desta informação, ainda é possível afirmar, que para Problema de Valor Inicial (PVI), o valor que normalmente é adoptado, por ser comum a ambas as equações será $[\partial g/\partial \mathrm{z}]^{-1}$ (Hairer et al. 1989, Ascher & Petzold 1994).

Forma Cónica de *Hessenberg* - Índice 2

Podemos fazer uma análise idêntica para quando um DAES na forma de Hessenberg apresenta o seguinte formato:

$$\frac{d\mathrm{x}}{dt} = f(t, \mathrm{x}, \mathrm{z}) \tag{2.18}$$
$$0 = g(t, \mathrm{x}) \tag{2.19}$$

, e com as seguintes as condições inicias:

$$0 = f(t_0, \mathrm{x}_0, \mathrm{z}_0) \tag{2.20}$$
$$0 = g(t_0, \mathrm{x}_0) \tag{2.21}$$

onde estas, têm de ser consistentes com o sistema.

O objectivo do processo de diferenciação é, para além de eliminar as equações algébricas, é também obter as derivadas das variáveis dependentes do tempo que estão em falta.

No sistema apresentado pelas equações (A.7 - A.8), a derivada de z em ordem ao tempo é inexistente, logo o objectivo na redução de índice é obter esta derivada.

Na equação algébrica (A.8), a variável z não está presente, ao contrário do que acontecia na equação (A.2).

Este facto será um princípio de que a derivação terá de ser feita mais do que uma vez, neste caso 2 vezes.

Durante o processo de redução de índice, teve de se derivar duas vezes, tratando-se por isso de um problema de Índice na Forma de *Hessenberg* 2. Sendo para um PVI escolhidos, normalmente, para valores iniciais $[\partial g/\partial x]^{-1}$ e $[\partial f/\partial \mathrm{z}]^{-1}$, pelo motivo já apresentado (Hairer et al. 1989, Ascher & Petzold 1994).

A dedução está feita no Anexo A.2.

Forma Cónica de *Hessenberg* - Índice 3

Um sistema com a seguinte estrutura:

$$\frac{d\mathrm{x}}{dt} = f(t, \mathrm{x}, \mathrm{y}, \mathrm{z}) \tag{2.22}$$
$$\frac{d\mathrm{y}}{dt} = g(t, \mathrm{x}, \mathrm{y}) \tag{2.23}$$
$$0 = h(t, \mathrm{y}) \tag{2.24}$$

contendo como condições iniciais,

$$0 = f(t_0, \mathrm{x}_0, \mathrm{y}_0, \mathrm{z}_0) \tag{2.25}$$
$$0 = g(t_0, \mathrm{x}_0, \mathrm{y}_0) \tag{2.26}$$
$$0 = h(t_0, \mathrm{y}_0) \tag{2.27}$$

onde estas, têm de ser consistentes com o sistema.

Como já referido em 2.1.2, o principal objectivo deste método é encontrar as derivadas em falta, no sistema.

Como a equação algébrica (A.18) é escrita em função de y, derivando em ordem ao tempo, obtém-se $d\mathrm{y}/dt$, podendo esta ser substituída por $g(t, \mathrm{x}, \mathrm{y})$, da equação (A.17), mas esta ainda não depende de z. Como tal, terá de ser derivada, novamente, produzindo a derivada de x em função do tempo, sendo depois substituída por $f(t, \mathrm{x}, \mathrm{y}, \mathrm{z})$, pela equação (A.16). Como esta já dependente de z, derivando novamente aemordem a t, obter-se-á a derivada $d\mathrm{z}/dt$.

Durante este processo de manipulação presente em A.3, recorreu-se três vezes à derivação, como tal, o DAES é de Índice na Forma de *Hessenberg* 3. Para um PVI, como já referido, e para minimizar problemas que surgem da resolução dos DAES, costuma-se a usar $[\partial f/\partial z]^{-1}$, $[\partial h/\partial y]^{-1}$ e $[\partial g/\partial x]^{-1}$, como condições iniciais (Hairer et al. 1989, Ascher & Petzold 1994).

Índice quando a Forma de *Hessenberg* é superior a 3

Para calcular índices de ordem n na forma de *Hessenberg*, segue-se uma estratégia semelhante, à apresentada anteriormente, não sendo aqui apresentado, pois os índices mais comuns nesta forma são os índices na forma de *Hessenberg* 1, 2 e 3.

2.1.3 Particularidades na resolução de DAES

Como já referido, a resolução de DAES, requer uma atenção especial relativamente às ODE, existindo varias estratégias de cálculo de modo ao resultado ser o mais fidedigno possível.

Uma destas estratégias será aproximar as DAES a ODEs, transformando a equação algébrica numa ODE, com a derivada em falta. sendo esta derivada multiplicada por uma constante (ε), como um valor que tende para zero (Ascher & Petzold 1994).

De modo a exemplificar, ir-se-á recorrer às equações (A.1) e (A.2).

$$\frac{d\mathrm{x}}{dt} = f(t, \mathrm{x}, \mathrm{z})$$
$$0 = g(t, \mathrm{x}, \mathrm{z})$$

Seguindo a adaptação apresentada anteriormente obtém-se:

$$\frac{dx}{dt} = f(t,x,z) \tag{2.28}$$

$$\varepsilon\frac{dz}{dt} = g(t,x,z), \tag{2.29}$$

$$\text{com } \varepsilon \to 0$$

Esta estratégia de resolução cria um problema *stiff*, derivado ao factos de a constante de tempo (τ_1) associada à função (2.28) ser muito superior à constante de tempo (τ_2) associada à função (2.29).

Sendo assim, quando é feita esta aproximação, deve-se ter em conta este problema, e escolher estratégias de resolução de problemas *stiff* (Ascher & Petzold 1994).

Por outro lado, quando recorremos a algoritmos próprios para a resolução de DAES, é frequente que perante um índice superior a 1 apresentem alguns problemas na sua resolução, quando estas são directamente discretizadas. Por este motivo, existe a necessidade de estabilizar as DAES (Ascher & Petzold 1994).

A forma mais comum de estabilizar os DAES, passa por converter estas equações em ODEs, desta vez recorrendo a derivações das equações algébricas (Ascher & Petzold 1994). Esta estratégia é apelidada de redução de índice.

Os gráficos 2.2 e 2.3 são uma representação dos resultados do problema do pêndulo (2.2-2.6).

O gráfico 2.2 apresenta os resultados obtidos para quando existe redução de índice do DAES, enquanto o gráfico 2.3, traduz os resultados obtidos sem redução do índice (Wolfram 2018*a*, Ascher & Petzold 1994).

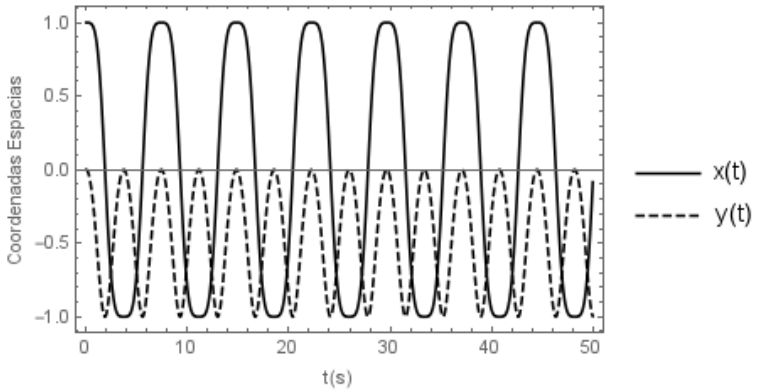


Figura 2.2 Resultados do problema do pêndulo, com redução de índice

Nestes gráficos, está presente um dos problemas comuns à utilização de DAES. Estes sistemas são muito sensíveis às condições iniciais, tornado-se instáveis facilmente, quando estas são mal

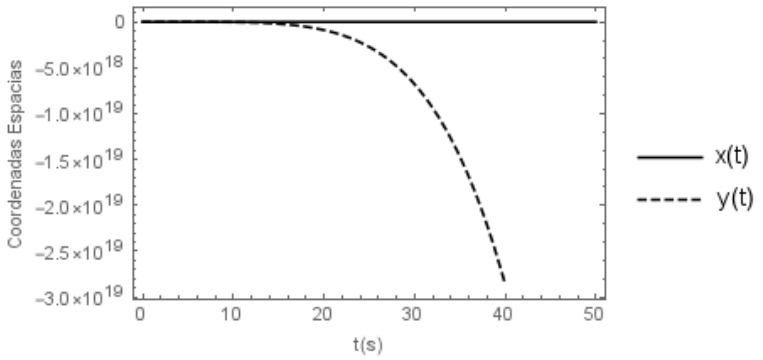


Figura 2.3 Resultados do problema do pêndulo, sem redução de índice

escolhidas. Outros problemas levantados por DAES de índice elevado são as descontinuidades e os "saltos"(Ascher & Petzold 1994, Harney et al. 2013).

Algo que também é comum é o fenómeno do *drift-off*, que é uma consequência da redução de índice e é causado, derivado à acumulação de erros durante a utilização de métodos numéricos (Bendtsen & Thomsen 1999). Este fenómeno é caracterizado por um desvio de valores em relação à realidade, não satisfazendo muitas das restrições lineares inicialmente impostas pelos DAES.

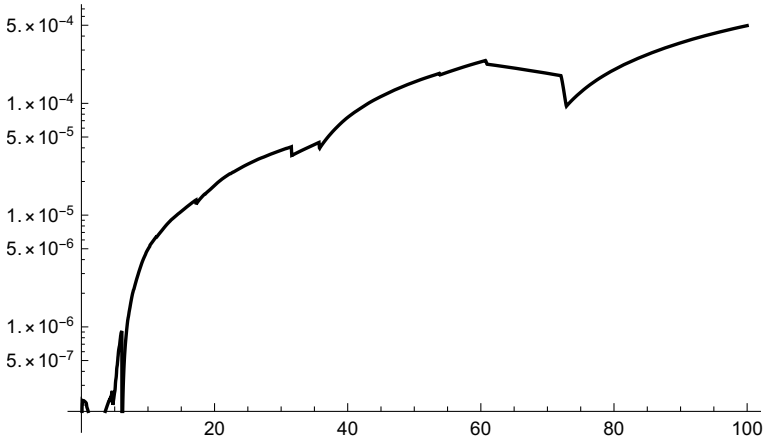


Figura 2.4 Erro absoluto entre o comprimento da corda dado pelo problema (calculado) e o comprimento real(imposto pela restrição linear), com redução de índice

Na Figura 2.4 está presente um caso do fenómeno de *drift-off* para o problema do pêndulo, havendo uma violação da restrição linear, atingindo a ordem de grandeza 10^{-4} (Bendtsen & Thomsen 1999, Wolfram 2018*a*).

2.2 Estratégias computacionais para a resolução de DAES

Em resolução computacional, pode-se seguir duas estratégias distintas de resolução destes problemas, uma estratégia simbólica ou uma estratégia numérica.

Quando optamos pela estratégia simbólica, as equações diferenciais são integradas analiticamente, podendo recorrer a *softwares* com algoritmos sofisticados, como é o caso do Wolfram Mathematica® .

Esta estratégia permite a resolução dos problemas com grande rigor, obtendo a solução analítica. No entanto, torna-se ineficaz para a resolução de certos tipos de problemas, sendo comum recorrer a estratégias numéricas, abrindo a porta para um conjunto de métodos de resolução e por sua vez ferramentas de resolução como os referidos no Capítulo 3.

Por sua vez, a estratégia baseada em métodos numéricos permite obter uma solução, mais ou menos aproximada, de sistemas com soluções analíticas complexas e muitas vezes impossíveis.

Na resolução dos DAES, estes métodos consistem em averiguar o índice DAES e caso este seja superior a 1, aplicar uma redução de índice. Posteriormente é aplicado um algoritmo de discretização, seguido de um *solver*.

A Figura 2.5 representa as etapas que um DAES passa até ser resolvido numericamente. O primeiro passo é a determinação do índice do DAES. Caso o índice seja 1, irá passar para o passo da Metodos de Inicialização dos DAES ou é logo aplicado o *Solver*. Caso o índice seja superior a 1, são aplicadas estratégias de Redução de Índice, podendo ser utilizadas técnicas baseadas no Algoritmo de *Pantelides*, técnicas baseadas na Estrutura da Matriz que representa o DAES, entre outras, dependendo das ferramentas de cálculo utilizadas (Wolfram 2018*a*).

Associado ao passo de redução de índice, costuma também haver uma "limpeza" de modo a eliminar possíveis variáveis redundantes originárias da redução de índice.

Após a eliminação das variáveis redundantes, existe a possibilidade de produzir condições iniciais ou adaptar o problema de modo a garantir que a solução irá ser o mais correcta possível, não existindo singularidades, como as apresentadas na secção 2.1.3. Aqui podemos aplicar estratégias como os métodos QR, que fazem a decomposição da matriz proveniente do sistema de DAES, permitindo calcular as condições iniciais. No entanto métodos como *Block Lower Triangular* (BLT) e a colocação fazem uma adaptação do problema, permitindo uma maior eficácia aquando da aplicação dos *solvers* (Wolfram 2018*a*).

Finalmente, após ter um DAES de índice 1 ou um sistema de ODEs equivalente ao problema inicial, pode-se aplicar os *solvers*

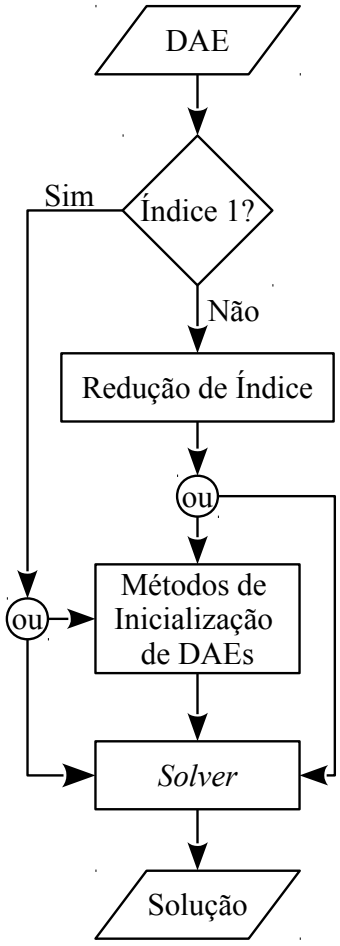


Figura 2.5 Diagrama de fluxo lógico para a resolução de DAES, recorrendo a métodos numéricos (Wolfram 2018*a*, Ascher & Petzold 1994, Bendtsen & Thomsen 1999).

Capítulo 3

Ferramentas Computacionais

Tendo em conta as questões levantadas no Capítulo 2, têm-se desenvolvido bibliotecas numéricas, com algoritmos sofisticados, com capacidade de resolver DAES, sendo apresentadas as mais comuns em 3.1.

Em paralelo a este desenvolvimento, também as linguagens de programação avançadas se têm adaptado para poderem resolver de uma forma rápida, fácil e eficaz estes problemas. Na secção 3.2, são tratadas duas linguagens comuns em Engenharia, o Wolfram Mathematica® e o MATLAB® .

Não esquecendo que o principal objectivo deste estudo é poder aplicar este modelos em estratégias de controlo, nomeadamente controlo óptimo também existe interesse no estudo de linguagens de modelação de sistemas algébricos, presentes na secção 3.3, pois estes têm incorporados optimizadores rápidos e precisos, para vários tipos de equações. No entanto, este tipo de linguagens não têm capacidade de resolução de equações diferenciais.

Com o crescimento de linguagens de código aberto como é o caso do Python ou o Julia, apresentadas em 3.4, e graças à facilidade de integrar outros tipos de linguagens, tem aparecido uma nova estratégia para a resolução de problemas de simulação e optimização para problemas com equações diferenciais. Estas linguagens permitem fazer o cálculo, em simultâneo, de derivadas e do parâmetro optimizado, como será melhor abordado em 3.5.

3.1 Bibliotecas Numéricas

Uma Biblioteca, no contexto informático, é um conjunto de informação ou módulos, (como por exemplo códigos ou rotinas de cálculo), que ajudam o operador/programador a cumprir uma tarefa. (Python 2018)

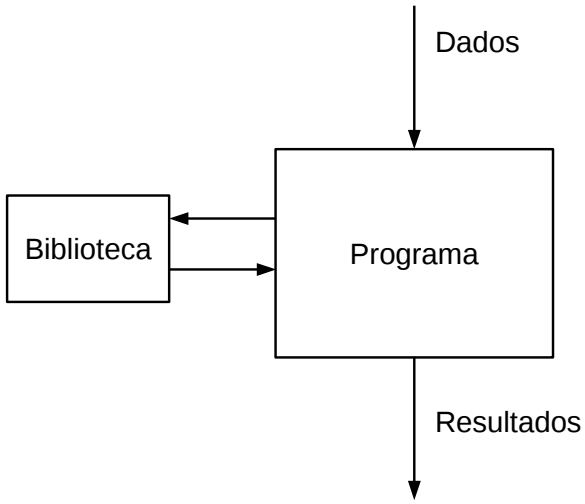


Figura 3.1 Esquema do funcionamento de uma biblioteca

No caso da resolução de DAES, as Bibliotecas, são integradores complexos e robustos que são chamados nas ferramentas computacionais de simulação, o que permite uma resolução do problema com algum grau de precisão.

As duas bibliotecas mais comuns para a resolução deste tipo de problemas, são o DASPK e o *SUite of Nonlinear and Differential/ALgebraic Equation Solvers* (SUNDIALS).

3.1.1 DASPK

O DASPK é uma extensão do DASSL, que foi construído para resolver DAES de PVI, escrita em Fortran (Hindmarsh et al. 2005).

A biblioteca DASSL, já apresentava capacidades de resolver DAES, recorrendo para isso ao método das Diferenças Finitas Regressivas (DFR).

O DASPK1.0 apareceu com a necessidade da resolução de problemas de grande escala. O que distingue esta biblioteca, da DASSL, é o facto de recorrer ao *Generalized Minimal Residual Method* (GMRES), em que é feito um conjunto de interacções para resolver um sistema linear, em cada interacção do método de Newton (StochSS 2018, Petzold et al. n.d., Hindmarsh et al. 2005).

No entanto, no DASPK1.0, assim como no DASSL, é necessário atribuir valores iniciais tanto para todas as variáveis dependentes como para as suas derivadas. No DASPK2.0, já não é necessita da atribuição valores iniciais para as derivadas das variáveis dependentes, uma vez estas são calculadas a partir das condições iniciais, recorrendo ao método de Euler (StochSS 2018, Petzold et al. n.d., Hindmarsh et al. 2005).

Apesar das vantagens, todas a bibliotecas apresentadas anteriormente, só resolviam o problema para DAES de índice 1. Neste sentido, na última versão, o DASPK3.0, o algoritmo já apresenta capacidade para resolver problemas de índice 2 (StochSS 2018, Petzold et al. n.d., Hindmarsh et al. 2005).

Um exemplo de um programa que usa estas bibliotecas é o GNU Octave. (Eaton 2018*b*)

3.1.2 SUNDIALS

O SUNDIALS é um conjunto de bibliotecas, escritas em C, que incluem métodos de resolução para sistemas não lineares de grande escala (*large-scale*) e problemas lineares de passo múltiplo dependente do tempo, entre outros.

Para o caso em apreço, a biblioteca que se destaca é a IDA, uma vez que foi construída com o propósito de resolver DAES.

A IDA é uma implementação em C da biblioteca DASPK sendo também baseada em resolução de problemas de valor inicial (Hindmarsh et al. 2005, Lawrence Livermore 2018).

A garantia de integração, só é verificada quando existe um vector de condições iniciais para todas as variáveis dependentes do sistema, e as respectivas derivadas. Outra situação em que existe garantia de integração, é a caso de problemas em estado *quasi*-estacionário (Lawrence Livermore 2018).

Para outros casos, cabe ao utilizador atribuir valores coerentes, não existindo garantias de integração numérica (Lawrence Livermore 2018).

3.2 Linguagens Avançadas

Nesta secção serão apresentadas diferentes linguagens que podem ser aplicadas para a simulação de sistemas.

Apesar destas linguagens não serem aplicadas para a optimização, estão desenhadas para obter uma solução credível para um determinado modelo (Kallrath 2004). Dentro das linguagens de modelação aqui abordadas, iremos ter linguagens que se baseiam em matemática simbólica, como o caso do Wolfram Mathematica® (Wolfram 2017), e linguagens que se baseiam em matemática numérica, como o caso da linguagem `.m`, linguagem usada pelo MATLAB® ou o GNU Octave.

3.2.1 Wolfram Mathematica®

O Wolfram Mathematica® é uma linguagem e um *software* dedicado a resolver problemas matemáticos recorrendo à linguagem simbólica. Sendo um linguagem comercial, compatível com as plataformas Linux, macOS e Windows (Wikipedia 2018*d*).

Esta ferramenta, mostra uma grande utilidade na resolução de DAES devido ao facto de poder resolver estes sistemas, quando possível, de uma fora analítica. Para além da resolução analítica, o Wolfram Mathematica® ainda tem a possibilidade de resolver DAES com recurso a métodos numéricos.

Para a resolução destes sistemas, o Wolfram Mathematica® , recorre tanto a bibliotecas proprietárias, como a bibliotecas partilhadas, como é o caso do IDAS, uma biblioteca do SUNDIALS. Mas antes de ser aplicada a biblioteca do *solver*, os DAES passam por um conjunto de passos anteriores de modo a reduzir a probabilidade de aparecimento dos problemas característicos das DAES (secção 2.1.3).

Na Figura 3.2, está o diagrama de fluxo lógico, que esquematiza o algoritmo por detrás da resolução numérica de DAES.

O algoritmo começa por calcular o índice dos DAES, se o índice do sistema for 1, caso necessário, este aplica uma estratégia para o cálculo das condições iniciais óptimas, e após o problema possuir todas as condições para a resolução, é aplicado um *solver*.

Caso o índice do DAES seja superior a 1, é aplicada uma estratégia de redução de índice. Esta estratégia pode recorrer a um algoritmo baseado no Algoritmo de *Pantelides* ou um algoritmo baseado na matriz estrutural.

Por vezes existe necessidade de acrescentar variáveis algébricas e de modo a que o problema não fique sobre-determinado, sendo o caso das *Dummy Derivatives*. Uma alternativa às *Dummy Derivatives* é a Projecção. Neste método, o índice dos DAES é reduzido até 1, sendo usada a projecção de modo a garantir as restrições originais.

Outro passo importante para garantir o sucesso desta resolução é escolher as condições iniciais apropriadas. Este é o objectivo do próximo passo.

O Wolfram Mathematica® pode recorrer a três métodos para o cálculo das condições de iniciação, sendo elas o método QR, o método BLT e o método da Colocação.

Os métodos de BLT e os métodos QR são construídos para resolverem de índice 1 e 0.

O método QR baseia-se na decomposição da matriz dos jacobianos do sistema no produto de duas matrizes, uma matriz ortogonal (Q) e numa matriz triangular superior (R). Este processo permite calcular as condições iniciais de uma forma iterativa, o que torna este método bastante eficiente e robusto (Wolfram 2018*a*).

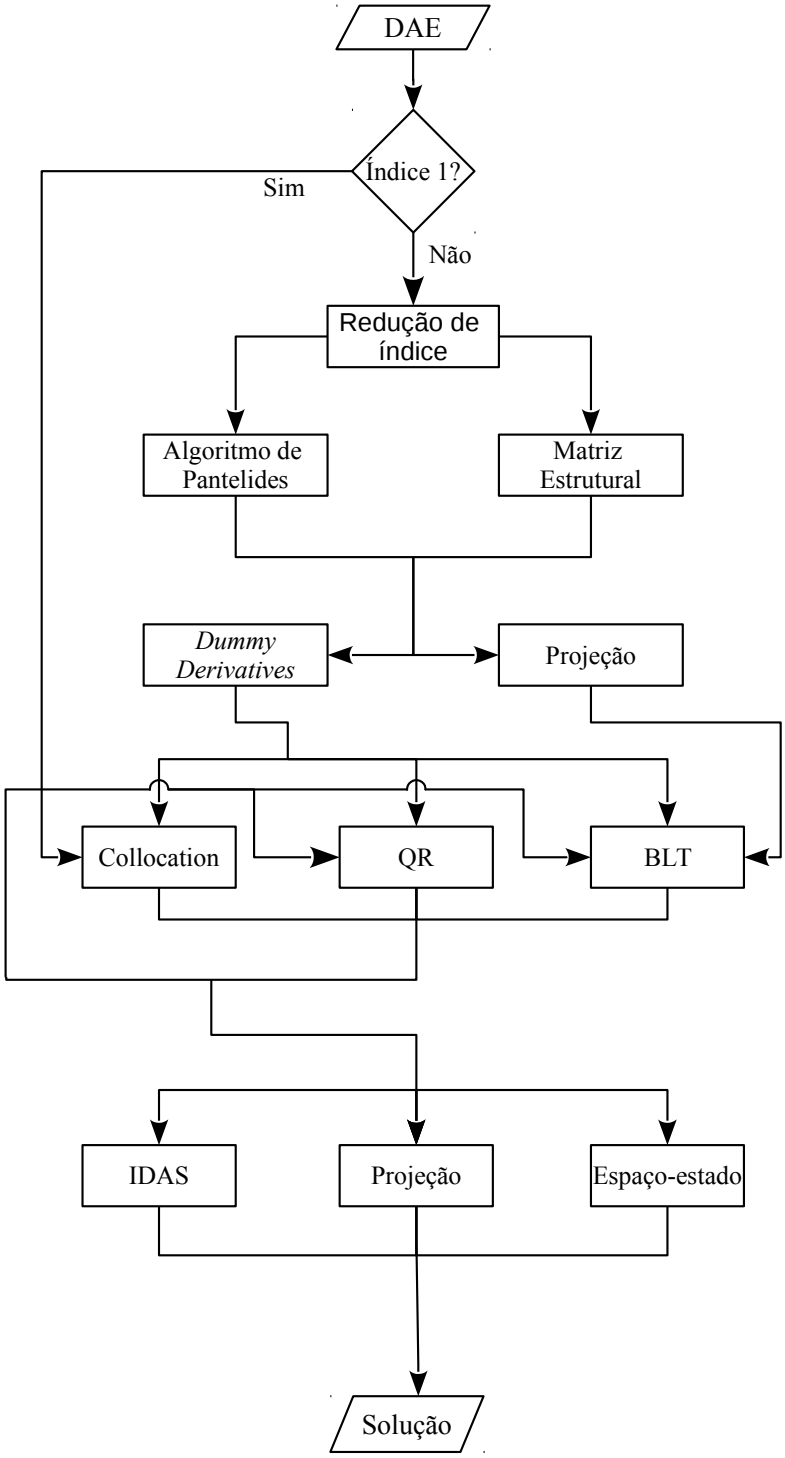


Figura 3.2 Diagrama de fluxo lógico representativo do algoritmo para a resolução numérica de DAES para o Wolfram Mathematica® (Wolfram 2018*a*).

Já no caso do método BLT divide-se o sistema de DAES em vários subsistemas menores. Esta divisão permite que os problemas sejam resolvidos de uma forma mais eficiente. Este método é recomendado para sistemas de grande dimensão (Wolfram 2018*a*).

O método de Colocação, trata o DAES, como se fosse um sistema de caixa preta residual, tentando satisfazer o residual, num intervalo pequeno junto ao ponto de iniciação. Pelo facto deste método tratar o sistema como sendo uma caixa preta, permite manipular o valor de iniciação para DAES de alto índice. No entanto esta estratégia de cálculo, não explora a estrutura do sistema, sendo por isso mais lenta que os métodos supra citados (Wolfram 2018*a*).

Tendo todas as condições reunidas, é aplicado o *solver*. Neste ponto, pode-se optar por uma destas técnicas(Wolfram 2018*a*):

→ Pacote IDA, do SUNDIALS, (secção 3.1.2);

→ Integração temporal "*StateSpace*";

↦ É um método iterativo, que recorre ao método de Newton para o calculo de derivadas.

→ Projecção;

↦ O software recorre a métodos numéricos para a resolução do problema, projectando a solução obtida na formulação inicial do problema.

3.2.2 MATLAB®

O MATLAB® é um *software* caracterizado por recorrer a uma linguagem matricial, com o mesmo nome. Sendo assim, toda a informação é armazenada, sobre a forma de vectores e matrizes (Oliveira 2006).

Sendo o MATLAB® um *software* proprietário, apesar da linguagem não o ser, existem alternativas como o GNU Octave, o FreeMat ou o scilab, de código aberto, destacando-se o GNU Octave, pois este trata a incompatibilidade com o interpretador MATLAB® como um erro, tornando mais fácil adaptar o código entre estas ferramentas (Oliveira 2006, Wikipedia 2018*a*, Eaton 2018*a*, FreeMat 2018, ESI 2017).

Tabela 3.1 Tabela resumo de interpretadores compatíveis da linguagem MATLAB® (The MathWorks 2018*a*, Eaton 2018*a*, FreeMat 2018, ESI 2017)

Software	Licença	Windows	Linux	macOS
MATLAB®	Proprietária	✓	✓	✓
GNU Octave	*Open-Source*	✓	✓	✓
FreeMat	*Open-Source*	✓	✓	✓
scilab	*Open-Source*	✓	✓	✓

A Tabela 3.1 faz um resumo dos *softwares* apresentados, a compatibilidade com as diferentes plataformas, assim como o tipo de licenças.

Apesar do MATLAB® ser um *software* caracterizado pela linguagem matricial, também é capaz de resolver problemas recorrendo a linguagem simbólica.

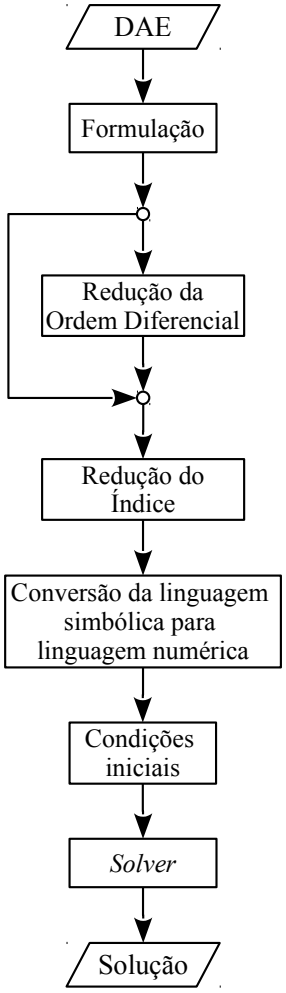


Figura 3.3 Diagrama de fluxo lógico representativo do algoritmo para a resolução numérica de DAES para o MATLAB® (The MathWorks 2018*c*).

No caso da resolução das DAES, a estratégia proposta pela documentação do MATLAB® , presente na Figura 3.3, passa por enunciar o problema recorrendo a variáveis e parâmetros simbólicos, sendo este o primeiro passo (The MathWorks 2018*c*).

Após este passo, poderá-se aplicar uma estratégia de Redução da Ordem Diferencial. Este passo permite eliminar variáveis definidas mas que não estejam presentes no problema formulado (The MathWorks 2018*c*).

Neste passo, é analisada a presença de cada variável na equação, sendo atribuído um 1, para a existência da variável na equação, e um 0, para a ausência da variável na equação. Para esta avaliação recorre-se a uma biblioteca pré-definida do MATLAB® chamada `reduceDifferentialOrder` (The MathWorks 2018*c*).

No passo 3, é avaliado o índice das DAES, recorrendo à biblioteca MATLAB® `isLowIndexDAE`. Esta biblioteca é um avaliador lógico, devolvendo 1 caso que o índice do DAES seja igual ou inferior a 1, ou seja, um DAES de índice 1 ou um sistema de ODEs, e devolve 0 caso que o índice do sistema seja superior a 1.

Em seguida, o DAES é transformado num sistema de ODEs. Para esta função recorre-se a uma biblioteca do MATLAB® chamada `reduceDAEIndex`.

Neste processo é comum que se formem variáveis e equações redundantes. Como tal o último processo do passo 3 é a eliminação de variáveis e equações redundantes. Para este fim é usada a biblioteca `reduceRedundances` (The MathWorks 2018*c*).

Neste ponto temos um sistema de ODE, equivalente ao DAES inicial, que se encontra em linguagem simbólica. No entanto os *solvers* do MATLAB® são baseados em métodos numéricos e como tal existe necessidade de converter as variáveis e parâmetros simbolicos do sistema para variáveis e parâmetros numéricos. Esta transformação é realizada no passo 4. Para este processo recorre-se à biblioteca MATLAB® `symvar`, para converter as variáveis e as equações, e a biblioteca `setdiff` para obter os parâmetros (The MathWorks 2018*c*).

Após esta transformação existe necessidade de criar a função MATLAB® que agrupe as variáveis, equações e parâmetros de modo a que estes possam ser resolvidos, usando a biblioteca MATLAB® `daeFunction`, sendo também definido os valores dos parâmetros neste ponto (The MathWorks 2018*c*).

Para se poder aplicar o *solver* terá de existir condições iniciais tanto para as variáveis como para as derivadas das variáveis, estando-se assim no passo 5.

Para este cálculo recorre-se a uma função MATLAB® chamada `decic`. Esta função recorre a um método interactivo para o cálculo das condições iniciais. Como tal é atribuída uma estimativa inicial tanto para o valor inicial das variáveis assim como para o valor inicial das derivadas dessas variáveis.

É ainda possível definir parâmetros como a tolerância absoluta e relativa, que pretendemos para os valores iniciais.

Tendo as condições iniciais, é agora possível aplicar o *solver*. Este *solver* é a função MATLAB® `ode15i` (The MathWorks 2018*c*).

3.2.3 GNU Octave

Apesar do GNU Octave ser compatível com o MATLAB® em muitas coisas, para casos mais específicos existe diferenças.

O caso dos DAES é um exemplo. O GNU Octave para resolver estes problemas recorre à biblioteca DASPK, (secção 3.1.1).

O funcionamento do DASPK é idêntico aos dos outros *solvers* utilizados pelo GNU Octave. Aqui os DAES, são definidos como uma variável ou função GNU Octave, sendo depois chamado no *solver*, com uma estrutura idêntica à a seguir representada.

```
[x,xdot] = daspk(fcn, x_0, xdot_0, t)
```

Onde, `x` é o vector com os valores das variáveis calculadas para um determinado `t`, `xdot` o vector com o valor das derivadas calculadas para um determinado `t`. Estes são os valores produzidos pelo *solver*.

As condições de entrada da função GNU Octave são a função(`fnc`), seguida das condições iniciais (`x_0, xdot_0`), de acordo com a ordem com que aparece no vector de saída, sendo seguida pelo intervalo temporal(`t`), num vector horizontal de dois elementos o tempo inicial(`ti`) e o tempo final(`tf`) ficando da seguinte forma `[ti tf]`.

3.2.4 gPROMS

O gPROMS é um *software* que permite a criação de modelos de processo a partir de diagramas de correntes. Estes modelos são baseados nos Primeiros Princípios. O gPROMS funciona numa teoria de modelação por blocos em que o modelo do processo é a soma dos vários blocos, estando o utilizador limitado aos modelos presentes nos blocos.

Dentro do gPROMS existe o gPROMS ModelBuilder, que permite a construção de modelos e a validação dos mesmos, assim como fazer optimização dos sistemas (Process Systems Enterprise 2018).

3.2.5 Modelação por Blocos

O sistema ou processo é modelado por um conjunto de blocos. Cada um destes blocos representam equipamentos, sub-processos, ou subsistemas que se relacionam entre si formando o processo. Em cada bloco existe um conjunto de equações, podendo estas ser definidas pelo usuário ou já estarem prédefinidas no sistema.

Temos como exemplo deste tipo de modelação, *softwares* como o Simulink, pertencente ao MATLAB® , o SystemModeler, pertencente ao Wolfram Mathematica® , e a sua alternativa de código aberto, o OpenModelica.

Na Figura 3.4 está exemplificada a modelação por blocos.

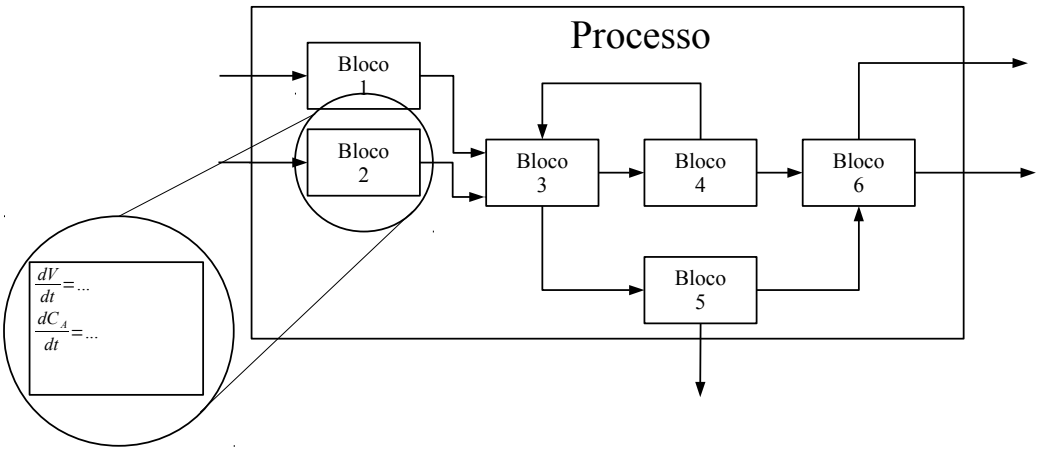


Figura 3.4 Diagrama representativo da modelação por blocos.

Por norma estes *softwares* vêem equipados com *solvers* para DAES, pois é comum aparecer esta classe de equações diferenciais quando fazemos este tipo de modelação, muito relacionada com as interacções entre blocos, ou mesmos aos modelos de certos blocos.

Simulink

O Simulink, é uma ferramenta computacional pertencente ao MATLAB® , que permite a criação de modelos de processo a partir de blocos. Esta ferramenta oferece uma alta integrabilidade com o MATLAB® , o que lhe oferece uma grande capacidade de resolver problemas (The MathWorks 2018*b*, Wikipedia 2018*c*).

O Simulink é utilizado para simular problemas de controlo, processamento de sinais, redes de comunicação via *Wireless*, robótica, entre outros (The MathWorks 2018*b*).

Modelica® (SystemModeler e OpenModelica)

O Modelica® é uma linguagem de modelação por blocos, não-proprietária construída para a resolução de sistemas físicos complexo (Modelica 2018).

No entanto esta linguagem está implementada noutros *softwares* como é o caso do OpenModelica, um programa de código-aberto, e do SystemModeler, pertencente ao Wolfram Mathematica® (Wikipedia 2018*b*, Wolfram 2018*b*, OpenModelica 2018).

3.3 Linguagens de modelação de Sistemas Algébricos

As Linguagens de Modelação de Sistemas Algébricos são uma classe especial de linguagens especificas para a modelação de sistemas algébricos. No entanto, a sua principal característica é a capacidade de resolver problemas de optimização, para este tipo de sistemas.

Este tipo de linguagem não está construída para a resolução de sistemas com equações diferenciais, no entanto, constituem o estado da arte quando se fala de optimização de sistemas.

Normalmente, este tipo de linguagens, apresentam capacidade de resolver problemas com o seguinte formato:

$$min\ f(x) \tag{3.1}$$
$$s.t\ F(x) = 0 \tag{3.2}$$
$$G(x) = 0 \tag{3.3}$$
$$x \in \mathbb{R}^n$$

Tipicamente, estes problemas costumam a recorrer a *sets* (conjuntos), índices, parâmetros e variáveis.

Estas linguagens costumam agrupar entidades(variáveis, parâmetros, restrições) semelhantes em *sets* que depois são devidamente indexadas. Estas entidades podem então ser representadas de forma compacta por expressões algébricas.

Um exemplo de dois programas que recorrem a este tipo de linguagem é o *General Algebraic Modeling System* (GAMS) e o *A Mathematical Programming Language* (AMPL), sendo estes considerados o estado da arte nesta área (Kallrath 2004).

3.3.1 GAMS

O GAMS apareceu para resolver problemas de optimização da vida real no âmbito da ciência e da engenharia, no entanto tem-se desenvolvido mais para a parte da economia mundial, finanças e engenharia química(Kallrath 2004).

Esta linguagem está afastada da linguagem de máquina, permitindo um uso mais amigável ao utilizador.

Outra característica positiva do GAMS é a sua capacidade de cria *inputs* para outras linguagens como o Pyomo (secção 3.4.1) ou o AMPL (secção 3.3.2) (GAMS 2018*a*).

Para além desta versatilidade, que permite recorrer a outras plataformas de optimização, o GAMS ainda tem um largo leque de optimizadores (GAMS 2018*b*).

3.3.2 AMPL

Inicialmente, o AMPL foi construído para resolver problemas lineares, sendo posteriormente expandido para a resolução de problemas não-lineares. Este *software* foi criado na década de 80 do século passado pelo *Computing Science Research Center of Bell Laboratories* (Kallrath 2004).

O AMPL recorre a vários tipos de *solvers*, estando divididos em dois grupos, os *open-source*, ou código aberto, e os *solvers* comerciais.

Dentro dos *solvers* de código aberto temos para problemas lineares o *COIN-OR Branch and Cut* (CBC), desenvolvida pelo *Computational Infrastructure for Operations Research* (COIN-OR), o *GNU Linear Programming Kit* (GLPK), um *solver* pertencente à Fundação GNU, e o *lp_solve*, pertencente ao SourceForge. Para problemas não lineares temos o *Interior Point OPTimizer* (IPOPT), o Bonmin e o Couenne, todos eles *solvers* do projecto COIN-OR (AMPL 2018*a*),

Estes *solvers*, para além de poderem ser usados pelo AMPL, ainda podem ser integrados noutras ferramentas, tal como o Pyomo (secção 3.4.1).

Como *solvers* comerciais, para o caso dos problemas lineares, tem-se o CPLEX, um *solver* da IBM Corporation, o Gurobi, um *solver* da Gurobi Optimization, o Xpress, um *solver* da FICO. Para o caso de problemas não lineares, tem-se *solvers* como o CONOPT da ARKI Consulting & Development, o KNITRO da Ziena Optimization, o LOQO da Princeton University e os MINOS e SOPT da Stanford University (AMPL 2018*b*).

Uma das vantagens do AMPL, para além da linguagem ter um interpretador de código *offline*, é a possibilidade de recorrer aos servidores do projecto para a resolução de problemas, através do projecto *Network-Enabled Optimization System* (NEOS), dando uma resolução mais rápida para problemas mais complexos.

3.4 Novas linguagens de optimização

Em paralelo com o desenvolvimento das linguagens de modelação de sistemas algébricos, também tem havido um crescimento acentuado nas linguagem de alto nível, tais como as referenciadas na secção 3.2, que permitem fazer simulações de problemas complexos, como é o caso de problemas com DAES, mesmo que algumas vezes com algum grau de dificuldade.

Também se tem assistido a um crescimento dos pacotes dedicados a optimização para este tipo de linguagens avançadas, como é o caso do Pyomo (secção 3.4.1) (Hart et al. 2017), um dos pacotes mais utilizados para Python, existindo ainda o PuLP, para problemas lineares. O JuMP (secção 3.4.2) é uma linguagem de modelação que recorre à linguagem Julia. (Dunning et al. 2017)

Estes pacotes permitem criar uma alternativa às técnicas de optimização de equações diferenciais que são usadas convencionalmente, os métodos de optimização sequenciais, recorrendo a métodos de optimização simultânea, que será explicado na secção 3.5 (Hart et al. 2017, Biegler 2000).

Uma característica comum destas duas linguagens, é o facto de serem *open-source*(Hart et al. 2017)(Dunning et al. 2017), tendo por este motivo uma comunidade que vai desenvolvendo estas ferramentas de acordo com as necessidades de cada desenvolvedor. Deste modo, estas ferramentas cobrem um grande leque de problemas, tendo como aspecto negativo, o facto de nem sempre os pacotes acompanharem as mudanças nas linguagens.

Existem também *frameworks*, como é o caso do CasADi, que permitem uma interacção entre plataformas como MATLAB® , Python ou linguagem C++ e diferentes *solvers*.

3.4.1 Pyomo

O Pyomo (*Python Optimization Modeling Objects*) é uma biblioteca Python, tendo como principal objectivo aproximar o Python das linguagens de modelação de sistemas algébricos, mantendo uma proximidade às linguagens mais comuns na resolução deste tipo de problemas (Hart et al. 2017).

Para a resolução de problemas de optimização, o Pyomo recorre a *solvers*,podendo ser proprietários ou de código aberto.

A grande vantagem do Pyomo , relativamente às linguagens convencionais de optimização é o facto de estes poderem discretizar derivadas recorrendo a algoritmos próprios, facilitando assim a optimização de problemas onde ocorrem derivadas, como é o caso dos problemas de simulação de sistemas dinâmicos (Hart et al. 2017), o que se traduz numa poupança de tempo e esforço investido no código.

Na formulação de um problema de optimização em Pyomo , podemos optar por criar um modelo concreto, onde todo o problema está especificado no *script*, ou por criar um problema abstracto, estando presente no *script* todas as restrições e objectivos do problema, sendo depois chamado um ficheiro de dados, independente do ficheiros de optimização, para a resolução do problema.

De modo a fazer a distinção, quando se cria a classe onde irá estar armazenado todo o modelo (`m`), recorre-se a:

```
m = ConcreteModel()
```

,

para formular um modelo concreto, ou a:

```
m = AbstractModel()
```

,

para criar um modelo abstracto (Hart et al. 2017).

O pyomo distribui a informação em 5 classes, sendo elas:

`Var`	Variáveis de optimização do modelo;
`Objective`	Expressão que se pretende maximizar ou minimizar no modelo;
`Constraint`	Restrições do modelo;
`Set`	Conjunto de valores usados para definir um instante do modelo;
`Param`	Parâmetro usado para definir o modelo num determinado instante.

Outro aspecto importante é o facto de se poder usar o Pyomo para optimizar, construindo um modelo e correndo-o com o Pyomo , ou então também é possível criar um *script* em Python, onde são chamados os pacotes do Pyomo, correndo com o Python.

Nesta última hipótese, terá de ser definido o *solver* no próprio *script*, sendo para isso usado o seguinte comando `SolverFactory('nome do solver')`. Caso que se trate de um modelo abstracto, será ainda preciso chamar o ficheiros de dados, sendo para isso usado o seguinte comando `m.create_instance("ficheiro_de_dados.dat")` (Hart et al. 2017).

Cada uma das classes apresenta declarações comuns que ajudam a definir a natureza da informação, que serão especificadas no anexo B.

Como já referido anteriormente, uma das grandes vantagens da utilização do Pyomo é o facto de este ser capaz de discretizar derivadas. Isto é possível graças ao pacote `DAE`, um pacote dedicado à resolução de DAES.

Neste pacote existem duas novas atribuições, o `ContinuousSet` e o `DerivativeVar`. O `ContinuousSet`, serve para identificar a(a) variável(ies) independente(s), ou seja, a(s) variável(ies) pelas quais as outras vão ser derivadas (Hart et al. 2017). ex.:

```
m.t = ContinousSet(bounds = (0, 1))
```

,

Já o `DerivativeVar`, serve para identificar a(s) derivada(s) da(s) variável(ies) dependente(s) que será(ão) derivada(s).

`m.x1 = Var(m.t)`	variável
`m.dx1dt = DerivativeVar(m.x1, wrt=m.t)`	1ª derivada
`m.dx1dt2 = DerivativeVar(m.x1, wrt=(m.t,m.t))`	2ª derivada

As equações diferenciais costumam a ser definidas recorrendo a funções Python, sendo declarada recorrendo ao comando `Constraint`.

Para a resolução das equações diferenciais nestes problemas de optimização, costuma-se a recorrer a técnicas de discretização, nomeadamente as DFR ou a colocação ortogonal.

No Pyomo a estratégia discretização é gerada automaticamente recorrendo ao comando Pyomo `TransformationFactory`. Neste comando, é especificada qual a estratégia utilizada, recor-

rendo a expressão `dae.finite_difference`, para as diferenças finitas, ou `dae.collocation`, para a colocação ortogonal.

No caso das diferenças finitas é necessário definir o modelo onde vai ser aplicado, o número de elementos finitos (`nfe`), a variável que irá ser discretizada (`wrt`) e o tipo de diferenças finitas(`scheme`), podendo estas ser DFR, sendo definida como `BACKWARD`, Diferenças Finitas Centradas (DFC), sendo definida como `CENTRAL`, ou Diferenças Finitas Progressivas (DFP), sendo definidas com o comando `FORWARD`(Hart et al. 2017).

Segue em seguida um exemplo de aplicação para as diferenças finitas.

```
TransformationFactory('dae.finite_difference')
discretizer.apply_to(m, nfe=20, wrt=m.t, scheme='BACKWARD')
```

Para o caso da colocação ortogonal é necessário definir o número de elementos finitos (`nfe`), o número de pontos de colocação em cada elemento(`ncp`), e o tipo de estratégia aplicada(`scheme`), podendo ser usada uma recolocação de raízes Gauss-Radau (`LAGRANGE-RADAU`), ou Gauss-Legendre (`LAGRANGE-LEGENDRE`)(Hart et al. 2017).

Em seguida segue um exemplo da aplicação da colocação ortogonal em Pyomo .

```
TransformationFactory('dae.collocation')
discretizer.apply_to(m, nfe=7, ncp=6, scheme='LAGRANGE-RADAU')
```

3.4.2 JuMP

O JuMP é uma linguagem de modelação dedicada a optimização incorporada em Julia. Esta tem uma capacidade de recorrer a diferentes *solvers*, tanto comerciais como de código aberto.

Estes *software* permite resolver Problemas Lineares (LP), *Mixed-Integer Linear Problems* (MILP), quadráticos, *conic-quadratic*, semi-definidos e Problemas Não Lineares (NLP) (Dunning et al. 2017).

O JuMP, tal como o Pyomo, tem capacidade de discretizar derivadas, sendo uma mais valia quando se trata de sistemas como os que estão em estudo.

Para além deste aspecto, segundo Dunning et al. (2017), o JuMP é capaz de competir com *softwares* comerciais no que se refere ao tempo de cálculo, o que é possível graças ao facto de o JuMP comunicar com os *solver* na memória, não necessitando de gravar em arquivos intermédios.

3.4.3 CasADi

O CasADi é um *framework* simbólico, de código aberto, que tem capacidade de interagir com plataformas como MATLAB® , GNU Octave ou linguagens como o Python e o C++.

Desenhado para poder resolver problemas de optimização não-linear, tem a capacidade de resolver problemas com equações diferenciais, calculando as derivadas pelos métodos de Range-Kutta Implicitos ou Explicitos ou recorrendo a bibliotecas do SUNDIALS (CasADi 2018*a*, Andersson et al. In Press, 2018).

No que se refere aos optimizadores e aos tipos de problemas que pode resolver, este tem a capacidade de resolver NLP, *Mixed-Integer NonLinear Problems* (MINLP), podendo ainda usar uma estrutura em bloco (*block structure*) ou esparsidade geral (*general sparsity*) que é explorada em Sequential Quadratic Programming (SQP), recorrendo a bibliotecas como IPOPT/BONMIN, BlockSQP, WORHP, KNITRO e SNOPT (CasADi 2018*a*, Andersson et al. In Press, 2018).

Este apresenta ainda a capacidade de resolver *Quadratic Programs* (QP) e *Mixed-Integer Quadratic Programs* (MIQP), recorrendo a *solvers* como CPLEX, GUROBI, HPMPC, OOQP ou qpOASES (CasADi 2018*a*, Andersson et al. In Press, 2018).

Dentro do CasADi, existe o `DaeBuilder`*class* dedicada à resolução de DAES. Esta *class* permite construir os modelos passo a passo, ou exportar directamente do Modelica. Como já referido tem a capacidade de resolver simbolicamente estes problemas. Outra grande vantagem é a capacidade de gerar código em C (CasADi 2018*b*).

3.5 Estratégias de optimização de DAES

Na optimização de DAES, pode-se recorrer a duas estratégias distintas, a estratégias sequenciais e as estratégias simultâneas (Biegler 2000).

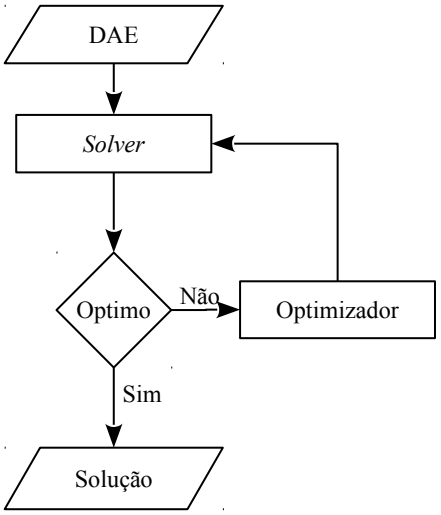
Na Figura 3.5, estão os diagramas de fluxo lógicos associados a cada um dos métodos.

Numa Estratégia Sequencial, ou Métodos de Parametrização Vectorial de Controlo (Biegler 2007), as variáveis de controlo são representadas como polinómios, sendo a optimização feita em relação aos coeficientes do polinómio (Biegler 2007).

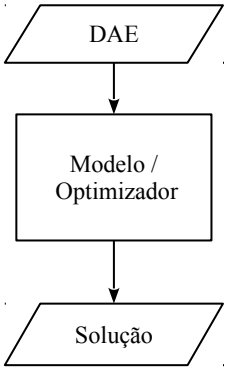
Esta optimização é incorporada num ciclo, juntamente com um *solver* para DAES, como representado pela Figura 3.5a. Este ciclo é quebrado, quando se atinge o valor de coeficientes que satisfaça de uma maneira óptima o modelo (Biegler 2007, Vasantharajan & Biegler 1990).

Este tipo de estratégias apresentam uma grande vantagem, pois para além de serem fáceis de construir e de aplicar, ainda são bastante fidedignas, uma vez que poderemos optar por utilizar *solvers* de DAES e optimizadores robustos. No entanto, estas estratégias são demoradas, não tendo muito interesse para aplicação *on-line* (Biegler 2007).

Por outro lado, temos as Estratégias Simultâneas, ou Transcrição Directa (*Direct Transcription*) (Biegler 2007), que discretiza os perfis das variáveis de estado e de controlo recorrendo à colocação de elementos finitos (Biegler 2007).



(a) Estratégia Sequencial



(b) Estratégia Simultânea

Figura 3.5 Diagrama de fluxo lógico para a optimização de DAES recorrendo a diferentes estratégias (Biegler 2000).

Para problemas com valores de fronteira ou problemas de controlo, esta última técnica acaba por ser uma estratégia menos dispendiosa de obter soluções. Este tipo de abordagem acaba por criar problemas NLP de grande escala, exigindo estratégias de optimização eficientes (Biegler 2007).

Este tipo de estratégia liga directamente a solução das DAES ao problema de optimização, ou seja, o sistema é resolvido uma única vez no ponto inicial, evitando soluções intermediárias, que exigem um esforço computacional maior (Biegler 2007, Diehl et al. 2002, Vasantharajan & Biegler 1990).

Capítulo 4

Casos de Estudo

Neste capítulo ir-se-á apresentar os sistemas em estudo.

Optou-se por fazer um sistema tipo, de poucas variáveis, permitindo assim explorar num ponto inicial as diferentes ferramentas de computação usadas. Para tal optou-se por modelar um RPA.

Também será apresentado o caso real da Coluna de Absorção de NO_x, apesar de nesta fase este sistema ainda não ser explorado.

4.1 Reactor Perfeitamente Agitado

O RPA é uma aproximação da realidade em que se considera que todos os pontos do reactor estão nas mesmas condições, sendo estas as condições de saída (Froment & Bischoff 1990). Por outras palavras, o modelo do sistema é de parâmetros agrupados, sendo a variável independente do sistema o tempo (Bequette 1998).

Optou-se por uma reacção genérica, de segunda ordem, não estando associados nenhuns compostos químicos em especifico. Para além desta característica, um dos reagentes está em equilíbrio com outro composto. As reacções do sistema estão apresentadas em (4.1) e em (4.2).

A equação (4.1) representa o equilíbrio entre a presença da espécie A e da espécie B.

$$A \rightleftharpoons B \tag{4.1}$$

A equação (4.2) representa a reacção de B e C a dar o produto D.

$$B + C \rightarrow D \tag{4.2}$$

Na Figura 4.1 está o esquema do reactor.

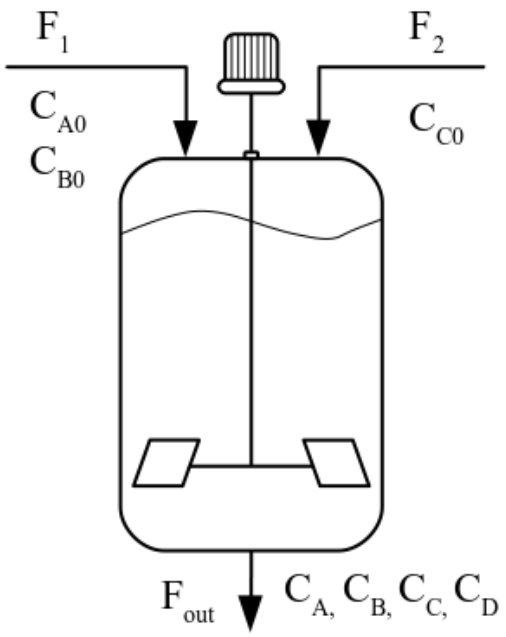


Figura 4.1 Esquema do RPA em estudo

Este reactor é composto por duas alimentações (corrente 1 e 2). A corrente 1, com um caudal de alimentação F_1, introduz no reactor o composto B com uma concentração C_{B0} e, derivado ao equilíbrio, o composto A numa concentração C_{A0}. A segunda corrente de alimentação é a corrente 2, esta alimenta o reactor com um caudal F_2 e uma concentração C_{C0}.

O reactor tem ainda uma saída, a corrente 3. Esta corrente tem um caudal F_3 e uma concentração de A C_A, de B C_B, de C C_C e de D C_D.

Para se poder simular o reactor é necessário criar o modelo matemático e, para tal. recorreu-se aos balanços mássicos do sistema.

4.1.1 Balanços Mássicos

A equação (4.4) traduz a realização de um balanço, sendo este mássico, energético, ou de qualquer outra natureza (Bequette 1998).

$$[\text{acumulação}] = [\text{entrada(s)}] - [\text{saída(s)}] + [\text{produzido}] - [\text{consumido}] \tag{4.3}$$

Tendo em contaque neste sistema a variação foi despresada, não irão ser feitos balanços energéticos ao sistema.

Balanço Global ao Reactor

O objectivo deste ponto é seguir a variação do volume no sistema. Como tal ir-se-á recorrer à equação (4.4) para fazer o balanço mássico global do reactor.

Segundo o Princípio de Lavoisier, quando olhamos num ponto de vista global para o sistema, mesmo quando ocorre reacção química, existe conservação de massa, não havendo consumo nem produção de matéria.

Como tal os termos [produzido] e [consumido] da equação (4.4) não são considerados, ficando:

$$[\text{acumulação}] = [\text{entrada(s)}] - [\text{saída(s)}] \tag{4.4}$$

ou seja,

$$\frac{dm}{dt} = \dot{m}_1 + \dot{m}_2 - \dot{m}_3 \tag{4.5}$$

sendo dm/dt a acumulação mássica e $\dot{m}_1$, $\dot{m}_2$ e $\dot{m}_3$ os caudais mássicos das respectivas correntes.

Tendo em conta que, por norma, na indústria existe um maior interesse em seguir o volume, ir-se-á converter este balanço para um balanço volumétrico, ficando

$$\rho_{global}\frac{dV}{dt} = \rho_1 F_1 + \rho_2 F_2 - \rho_3 F_3 \tag{4.6}$$

sendo dV/dt a acumulação volumétrica, F_1, F_2, F_3, como já referido, os caudais volumétricos das respectivas correntes, ρ_{global} a densidade global do sistema e ρ_1, ρ_2 e ρ_3 as densidades das respectivas correntes.

De acordo com os princípios adjacentes à teoria do modelo do RPA, sabe-se que ρ_3 será igual a ρ_{global}, pois as condições no interior do reactor são iguais às condições de saída.

Para além deste pressuposto relacionado com o tipo de modelo utilizado, recorreu-se a outro pressuposto que defende que ρ_1 e ρ_2, são iguais ao do sistema. Sendo assim, podemos simplificar obtendo:

$$\frac{dV}{dt} = F_1 + F_2 - F_3 \tag{4.7}$$

Considerando que o reactor se comporta como um tanque em descarga livre, o F_3 irá ser dependente da altura de líquido no reactor (h), ficando:

$$F_3 \propto A\beta_1\sqrt{h} \tag{4.8}$$

, onde β_1 é o coeficiente de descarga do reactor e A a área de secção recta perpendicular ao sentido do escoamento.

Neste caso, o reactor irá ser aproximado a um cilindro e, como tal, A irá ser constante ao longo do tempo.

No nosso sistema existe interesse em acompanhar o volume de líquido no reactor ao invés da altura.

Como tal, a equação (4.8) pode ser adaptada, de modo a que o caudal F_3 seja calculado a partir do volume, ao invés da área, obtendo:

$$F_3 = \beta_2 \sqrt{V} \tag{4.9}$$

, onde β_2 é um parâmetro que engloba o coeficiente de descarga, mais as constantes que surgiram da transformação da equação (4.8) para a equação (4.9).

Substituindo a equação (4.9) em (4.7), obtém-se:

$$\frac{dV}{dt} = F_1 + F_2 - \beta_2 \sqrt{V} \tag{4.10}$$

Balanço parcial ao composto A no reactor

Para fazer o balanço parcial de A ir-se-á recorrer à equação (4.4).

Uma vez que o composto A não está envolvido directamente na reacção, os termos $[produzido]$ e $[consumido]$ da equação (4.4) serão desprezados.

Fazendo o balanço molar ao sistema, obtém-se:

$$\frac{dm_A}{dt} = \dot{m}_{1_A} - \dot{m}_{3_A} \tag{4.11}$$

, sendo dm_A/dt a variação mássica do composto A, $\dot{m}_{1_A}$ a massa de composto A presente na corrente 1 e $\dot{m}_{3_A}$ a massa de composto A presente na corrente 3.

Manipulando a equação (4.11) de modo a obter a variação da concentração ao longo do tempo, obtém-se:

$$\frac{d(VC_A)}{dt} = (C_{A0})F_1 - (C_A)F_3 \tag{4.12}$$

, sendo C_{A0} a concentração de entrada do composto A e C_A a concentração de A. Tendo em conta que V pode não ser constante ao longo do tempo, $d(C_A V)/dt$ pode ser decomposta em

$$C_A \frac{dV}{dt} + V \frac{dC_A}{dt}$$

, ficando a equação (4.12), na seguinte forma:

$$V \frac{dC_A}{dt} + C_A \frac{dV}{dt} = (C_{A0})F_1 - (C_A)F_3 \tag{4.13}$$

Substituindo a equação (4.10) e (4.9) em (4.13) obtém-se:

$$V \frac{dC_A}{dt} + C_A \left(F_1 + F_2 - \beta_2 \sqrt{V}\right) = (C_{A0})F_1 - (C_A)\beta_2 \sqrt{V} \tag{4.14}$$

, manipulando a equação (4.14):

$$\begin{aligned}
V \frac{dC_A}{dt} &= C_{A0}F_1 - C_A\beta_2\sqrt{V} - C_A\left(F_1 + F_2 - \beta_2\sqrt{V}\right) \Leftrightarrow \\
V \frac{dC_A}{dt} &= C_{A0}F_1 - C_A\left(F_1 + F_2 - \beta_2\sqrt{V} + \beta_2\sqrt{V}\right) \Leftrightarrow \\
V \frac{dC_A}{dt} &= C_{A0}F_1 - C_A\left(F_1 + F_2\right) \Leftrightarrow \\
V \frac{dC_A}{dt} &= F_1\left(C_{A0} - C_A\right) + F_2C_A
\end{aligned}$$

, se isolarmos o termo $\frac{dC_A}{dt}$ obtém-se:

$$\frac{dC_A}{dt} = \frac{F_1\left(C_{A0} - C_A\right) - F_2C_A}{V} \tag{4.15}$$

Balanço parcial ao composto C do reactor

O composto C é outro reagente na reacção (4.2). Tal como quando aplicámos a equação (4.4), o termo [produzido] é desprezado, ficando:

$$\frac{dm_c}{dt} = \dot{m}_{2_C} - \dot{m}_{3_C} - k_1 \frac{C_B C_C}{M_B M_C} \tag{4.16}$$

Sendo dm_C/dt a acumulação mássica de C ao longo do tempo, $\dot{m}_{2_C}$ a massa de C que entra no reactor pela corrente 2, $\dot{m}_{3_C}$ a massa de C que sai pela corrente 3.

Manipulando a reacção (4.16) de modo a obter a variação da concentração de C ao longo do tempo (dC_C/dt),

$$\frac{d(C_C V)}{dt} = C_{C0}F_2 - C_C F_3 - k_1 C_B C_C V \tag{4.17}$$

, de forma a isolar o termo dC_C/dt, ir-se-à decompor $d(C_C V)/dt$ tendo:

$$V\frac{d(C_C)}{dt} + C_A\frac{d(V)}{dt} = C_{C0}F_2 - C_C F_3 - k_1 C_B C_C \tag{4.18}$$

Aplicando à equação (4.18), uma manipulação idêntica à aplicada à equação (4.14), obtém-se:

$$\frac{dC_C}{dt} = \frac{F_2\,(C_{C0} - C_C) - F_1 C_C - k_1 C_B C_C V}{V} \tag{4.19}$$

Balanço parcial ao composto D do reactor

O composto D é um produto da reacção (4.2). Como tal , este será produzido e, segundo as equações químicas (4.1) e (4.2), não será consumido. Ao aplicar-se a equação (4.4), o termo [consumido] é desprezado, ficando apenas o termo [produzido].

Outro ponto a ter em conta é que, apesar de no sistema não se considerar a entrada de D, este pode ser introduzido em qualquer alimentação. Assim obtém-se:

$$\frac{dm_D}{dt} = \dot{m}_{1_D} + \dot{m}_{2_D} - \dot{m}_{3_D} + k_1\frac{C_B C_C}{M_B M_C} \tag{4.20}$$

, sendo dm_D/dt a acumulação mássica de D ao longo do tempo, $\dot{m}_{1_D}$ a massa de D que entra na corrente 1, $\dot{m}_{2_D}$ a massa de D que entra na corrente 2 e $\dot{m}_{3_D}$ a massa de D que sai pela corrente 3.

Manipulando a reacção (4.20) de modo a obter a variação da concentração de D ao longo do tempo(dC_D/dt), onde C_{D01} é a concentração D que entra no caudal 1 e C_{D02} é a concentração D que entra no caudal 2:

$$\frac{d(C_D V)}{dt} = C_{D01}F_1 + C_{D02}F_2 - C_d F_3 + k_1 C_B C_C V \tag{4.21}$$

De forma a isolar o termo dC_C/dt, ir-se-á decompor $d(C_C V)/dt$, ficando:

$$V\frac{d(C_C)}{dt} + C_A\frac{d(V)}{dt} = C_{D01}F_1 + C_{D02}F_2 - C_d F_3 + k_1 C_B C_C V \tag{4.22}$$

Aplicando à equação (4.22), uma manipulação identica à aplicada à equação (4.14), obtém-se:

$$\frac{dC_C}{dt} = \frac{F_1\left(C_{D01} - C_D\right) - F_2\left(C_{D02} - C_D\right) + k_1 C_B C_C V}{V} \tag{4.23}$$

4.1.2 Equilíbrio Químico

Como já referido anteriormente, neste sistema, a espécie A e a espécie B estão em equilíbrio químico, sendo representado em (4.1).

O equilíbrio químico pode ser traduzido pela equação (4.24).

$$C_A = \frac{C_B}{K_{eq}} \tag{4.24}$$

C_A e C_B, como já referido, são as concentrações dos respectivos compostos e K_{eq} a constante de equilíbrio.

4.1.3 Modelo Completo do Sistema

Para poder simular o sistema terá que se agrupar as diferentes equações vindas dos balanços e da lei de equilíbrio.

Agrupando a equação (4.10) às equações (4.15), (4.19) e (4.23) provenientes dos balanços parciais e a equação (4.24), proveniente do equilíbrio, obtém-se o sistema com as equações (4.25)

$$\frac{dV}{dt} = F_1 + F_2 - \beta_2\sqrt{V} \tag{4.25a}$$

$$\frac{dC_A}{dt} = \frac{F_1\left(C_{A0} - C_A\right) - F_2 C_A}{V} \tag{4.25b}$$

$$\frac{dC_C}{dt} = \frac{F_2\left(C_{C0} - C_C\right) - F_1 C_C - k_1 C_B C_C V}{V} \tag{4.25c}$$

$$\frac{dC_C}{dt} = \frac{F_1\left(C_{D01} - C_D\right) - F_2\left(C_{D02} - C_D\right) + k_1 C_B C_C V}{V} \tag{4.25d}$$

$$C_A = \frac{C_B}{K_{eq}} \tag{4.25e}$$

Este modelo é composto por quarto equações diferenciais ((4.25a), (4.25b), (4.25c) e (4.25d)) e uma equação algébrica (4.25e), tornando-o um DAES.

Este sistema de equações é não-linear, uma vez que a equação (4.25a) é não linear. Estas características podem trazer alguns problemas na integração analítica do sistema, sendo, por

este motivo, muito comum a linearização do sistema, ou seja, a aproximação da equação não-linear a uma equação linear, tendo uma boa aproximação nas redondezas do ponto no qual se baseou a linearização, mas perdendo exactidão à medida que se afasta deste ponto.

Variáveis e Parâmetros

Num modelo existem variáveis, podendo estas ser de estado, de entrada e de saída, e parâmetros.

No sistema apresentado, as varáveis de entrada ($u(t)$) são o caudais F_1, F_2, F_3, as concentrações C_{A0}, C_{B0} e C_{C0} e o coeficiente de descarga β_2. Sendo as variáveis de estado ($x(t)$) o V, C_A, C_B, C_C e C_D, sendo as concentrações dos compostos também variáveis de saída do modelo ($y(t)$).

Quanto aos parâmetros (θ), o sistema tem o k_1 e o K_{eq}.

Na Figura 4.2, estão distribuídas as variáveis e parâmetros de acordo com a sua categoria, sendo apresentados na Tabela 4.1 os valores dos parâmetros assim como as suas unidades e referências.

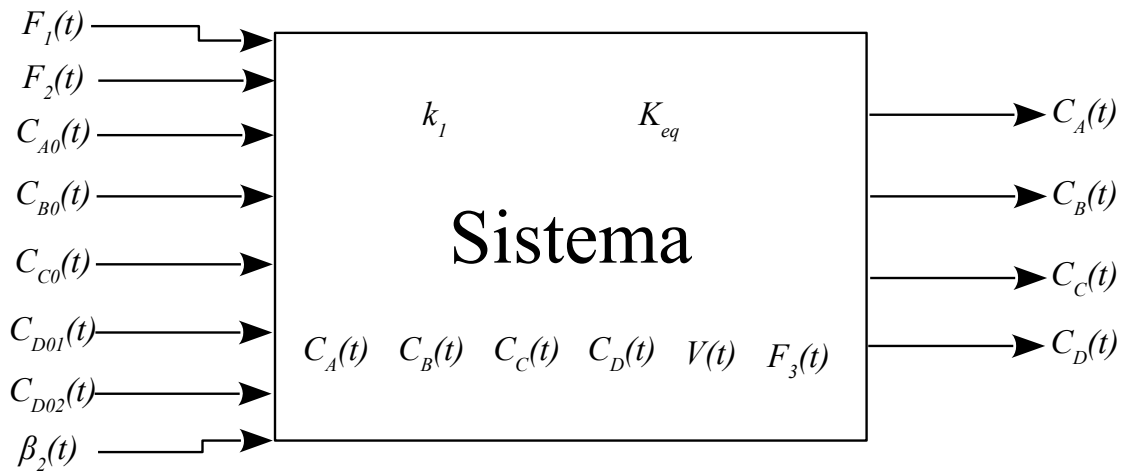


Figura 4.2 Diagrama para o sistema em estudo.

Tabela 4.1 Parâmetros do sistema

	Valor	Unidades
k_1	1.4×10^{-2}	$^{m^3}/_{mol.min}$
K_{eq}	3.0	-

Índice do Modelo

Com o objectivo de perceber a complexidade do modelo, ir-se-á calcular o índice do DAES, procedendo à derivação do sistema em ordem ao tempo. Este processo está explicado em 2.1.1.

Deste modo, começou-se por derivar a equação (4.25e) em função do tempo:

$$\frac{dC_A}{dt} = \frac{1}{K_{eq}} \frac{dC_B}{dt} \tag{4.26}$$

Este processo permitiu-nos obter o seguinte sistema equivalente de ODEs

$$\frac{dV}{dt} = F_1 + F_2 - \beta_2 \sqrt{V} \tag{4.27a}$$
$$\frac{dC_A}{dt} = \frac{F_1 (C_{A0} - C_A) - F_2 C_A}{V} \tag{4.27b}$$
$$\frac{dC_C}{dt} = \frac{F_2 (C_{C0} - C_C) - F_1 C_C - k_1 C_B C_C V}{V} \tag{4.27c}$$
$$\frac{dC_C}{dt} = \frac{F_1 (C_{D01} - C_D) - F_2 (C_{D02} - C_D) + k_1 C_B C_C V}{V} \tag{4.27d}$$
$$\frac{dC_A}{dt} = \frac{1}{K_{eq}} \frac{dC_B}{dt} \tag{4.27e}$$

, como ao longo deste processo, as equações do sistema foram derivadas uma vez, trata-se de um sistema de índice 1.

4.1.4 Caracterização do Sistema

Com o objectivo de conhecer melhor o sistema, procedeu-se à caracterização do mesmo, tendo como ponto de referência para as variáveis entrada os valores apresentado na Tabela .

Tabela 4.2 Valores de referência das Variáveis de Entrada

	Valor	Unidades
C_{A0}	1.0	$^{mol}/_{m^3}$
C_{B0}	3.0	$^{mol}/_{m^3}$
C_{C0}	1	$^{mol}/_{m^3}$
C_{D01}	0	$^{mol}/_{m^3}$
C_{D02}	0	$^{mol}/_{m^3}$
F_1	0.05	$^{m^3}/_{min}$
F_2	0.05	$^{m^3}/_{min}$
β_2	0.031623	$^{m^2}/_{min}$

Procedeu-se a um estudo de modo a conhecer os números de estados estacionários assim como os valores que as variáveis nestes mesmos estados.

Recorrendo ao Wolfram Mathematica® , concluiu-se que o sistema têm um único estado estacionário, tendo os valores apresentados na Tabela

Tabela 4.3 Valores das variáveis de estado, para o Estado Estacionário, do sistema, tendo em conta os valores de referÊncia apresentados na Tabela 4.2.

	Wolfram Mathematica®	Unidades
C_A	0.5	$^{mol}/_{m^3}$
C_B	1.5	$^{mol}/_{m^3}$
C_C	0.161292	$^{mol}/_{m^3}$
C_D	0.338708	$^{mol}/_{m^3}$
V	9.99986	m^3

É de salientar que este sistema só apresenta um estado estacionário, porque conciderou-se a temperatura constante. caso que a temperatura também fosse uma variável, este teria multiplicidade de estados estacionários.

Todas as equações exepto a equação (4.25a), a equação que permite calcular a variação de volume, são lineares.

Sabendo o Estado Estacionário do sistema, foi feito aum estudo da sensibilidade deste a variações das condições de referência. Tendo em conta que a concentração de B está em equilibrio com a concentração de A, não se fez um estudo de sensibilidade a C_B.

Para o estudo de sensibilidade foi feita uma variação de $+/-50$ % para todas as variáveis de entrada em estudo excepto C_D, pois o valor de referência é 0, e não existindo concentrações negativas mão faria sentido fazer uma variação de -50 %, sendo so feita um variação de $+50$ %.

Este estudo permitiu obter os perfis obtidos na Figura 4.3.

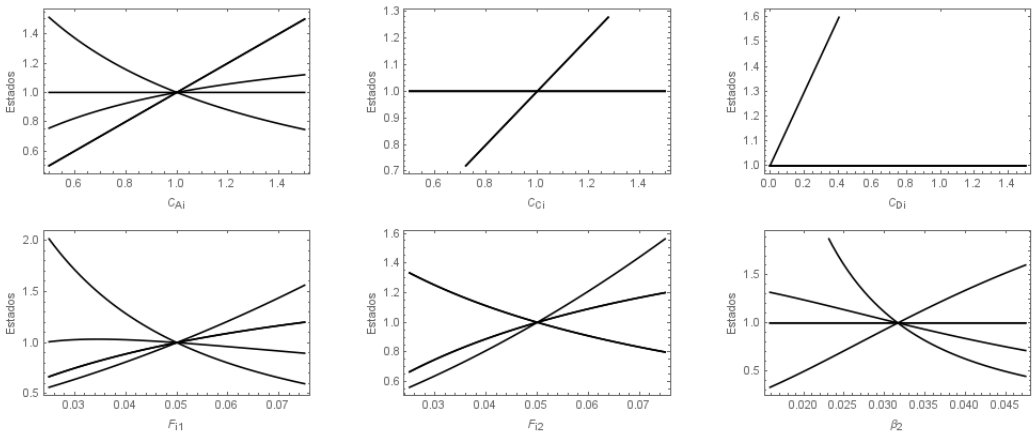


Figura 4.3 Sensibilidade do sistema às variáveis de entrada

Dos perfis apresentados em 4.3, podemos observar a não linearidade do sistema causada pela equação (4.25a), contudo para uma grande parte dos casos, a linearização do sistema não irá introduzir um grande desvio, sendo a excepção as variações de β_2.

4.1.5 Simulação Dinâmicos

Estando o sistema caracterizado, foi feito testes dinâmico de modo estudar o comportamento do sistema, assim como comparar as ferramentas escolhidas para este estudo.

Começou-se por fazer um teste dinâmico, em que as condições iniciais atribuidas às Variáveis de Estado são iguais são iguais às concetrações de entrada, para o caso das concentrações, sendo atribuido um valor para o volume e para F_2, valores arbitarios. Estes valores estão apresentados na Tabela 4.4.

Tabela 4.4 Condições Iniciais das Variáveis de Estado

	Valor	Unidades
C_A	1	$^{mol}/_{m^3}$
C_B	3	$^{mol}/_{m^3}$
C_C	1	$^{mol}/_{m^3}$
C_D	0	$^{mol}/_{m^3}$
F_3	0.1	$^{m^3}/_{min}$
V	10	m^3

Foi feita a simulação recorrendo ao Wolfram Mathematica® e ao Pyomo , obtendo os resultados apresentados na Tabela 4.5. Para ambos os casos a integração ocorreu num tempo inferior a 1 min, não existindo grande diferença a nivel de tempo para as duas ferrametas utilizadas.

Tabela 4.5 Resultados de simulação para as condições apresentadas nas Tabelas 4.2 e 4.4.

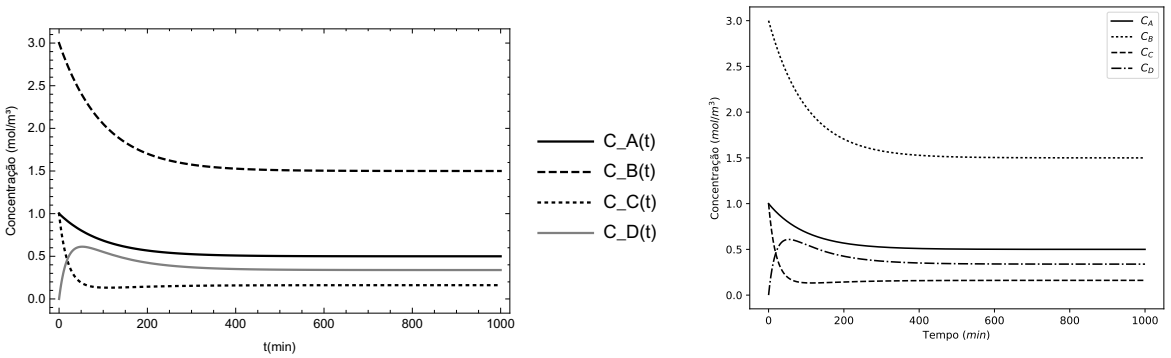
	Wolfram Mathematica®	Pyomo	Unidades
C_A	0.5	0.499999	$^{mol}/_{m^3}$
C_B	1.5	1.499999	$^{mol}/_{m^3}$
C_C	0.161292	0.161291	$^{mol}/_{m^3}$
C_D	0.338708	0.338708	$^{mol}/_{m^3}$
V	9.99986	9.99985	m^3

Comparando os resultados das Tabelas 4.3 e 4.5, verifica-se que o reactor estabilizou nos valores do estado estacionário.

No entanto existe uma pequena diferença entre os valores obtidos pelo Wolfram Mathematica® e os valores obtidos pelo Pyomo . Tendo em conta que em ambas as ferramentas se recorreu a métodos numérios para a resolução dos problemas, e sabendo que a tolerância do método influência os resultados, e ainda que o Wolfram Mathematica® recorre uma tolerância de $0.5 \times$

10^{-17}, segundo Wolfram (2017) e o Pyomo recorre a uma tolerância de 1×10^{-8}, segundo Sandia (2014), justifica a diferença entre as duas ferramentas.

Destas simulações foram ainda retirados os perfis das concentrações dos diferentes compostos, apresentado na Figuras 4.4.



(a) Wolfram Mathematica® **(b)** Pyomo

Figura 4.4 Variação da concentração de C_A, C_B, C_C e C_D ao longo do tempo, para diferentes ferramentas computacionais

Quando comparado os perfis da Figura 4.4a como os perfis da Figura 4.4b, pode-se verificar que os perfis coincidem, começando a estabilizar por volta dos 500 *min*.

Não se verificou uma grande variação no volume de líquido no reactor pois o valor de volume em estado estacionário e a condição inicial atribuida a esta variável eram muito próximas.

Tendo em conta que F_3 é dependente do volume de liquído no reactor, e como este não sofreu variação significativa, F_3 tambem não irá sofrer.

Tanto pela observação dos perfis obtidos, em que os perfis estão muito próximos, como pelos os resultados em estados estacionários obtidos para as condições apresentadas, é possível concluir, que para o caso em que não são provcadas variações das condições de entrada, o Pyomo tem uma resposta proxima do Wolfram Mathematica® . Sendo o Wolfram Mathematica® concederada uma ferramenta computacional de referência, estes resultados permitem validar o Pyomo para este tipo de simulações.

Com o objectivo de analisar o comportamento do sistema a perturbações na variáveis de entrada, procedeu-se a um conjunto de testes dinâmicos.

As variáveis de entrada manipuladas foram o caudal de entrada (F_1) e a concentração de entrada do composto A (C_{A0}).

Perturbação no Caudal de entrada 1 (F_1)

De modo a perceber a influência da variação do caudal no sistema e como iria ser a sua resposta, optou-se por fazer um perturbação em degrau no caudal de entrada 1.

Esta perturbação ocorreu ao minuto 100, onde F_1 passou de 0.05 m^3/min para 0.1 m^3/min.

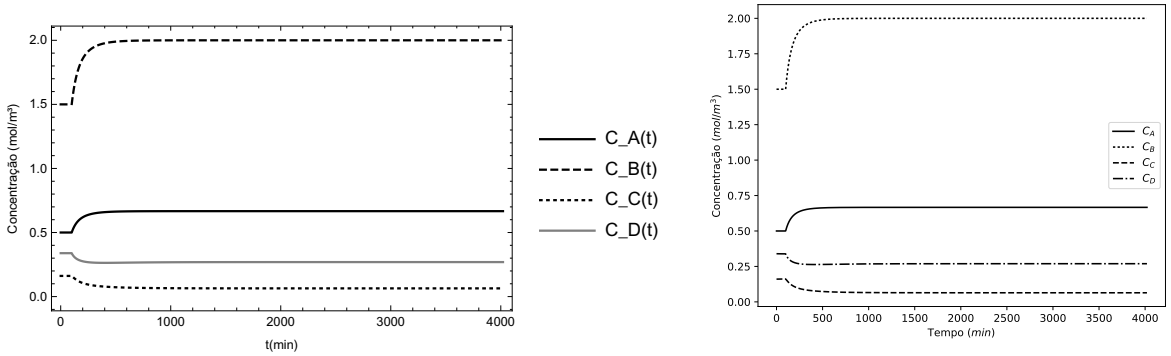
Recorreu-se novamente ao Wolfram Mathematica® e ao Pyomo para fazer a simulação numérica do sistema. Esta simulação permitiu obter os resultados apresentados na Tabela 4.6.

Tabela 4.6 Resultados de simulação para as condições inicais apresentadas na Tabela 4.4 e para a perturbação no caudal F_1.

	Wolfram Mathematica®	Pyomo	Unidades
C_A	0.666666	0.666667	$^{mol}/_{m^3}$
C_B	1.999999	2.000001	$^{mol}/_{m^3}$
C_C	0.064103	0.064103	$^{mol}/_{m^3}$
C_D	0.269230	0.269228	$^{mol}/_{m^3}$
V	22.499682	22.499680	m^3

Mais uma vez verifica-se que os resultados estão muito próximos, apresentando variações nas últimas casas decimais, estando associadas às tolerâncias das ferramnetas computacionais.

Nas Figuras 4.7 estão presentes os perfis das concentrações dos diferentes compostos, para os resultados do Wolfram Mathematica® e do Pyomo .



(a) Wolfram Mathematica® **(b)** Pyomo

Figura 4.5 Variação da concentração de C_A, C_B, C_C e C_D ao longo do tempo, quando aplicada uma perturbação em F_1 aos 100 min, para diferentes ferramentas computacionais.

É possível verificar que um aumento do caudal F_1, provoca um aumento em C_A e C_B. Este fenómeno pode ser justificado pelo facto de o caudal F_1, ser o caudal de alimentação do composto A e B, sendo assim, um aumento desta alimentação irá provocar tambem um aumento da quantidade química destes compostos no sistema, levando ao aumento de C_A e C_B.

Por outro lado, C_C e C_D diminui. Um aumento de F_1 influencia de duas formas a concentração de C. Ao aumentar F_1, está-se a introduzir uma maior quantidade do composto B no sistema, e sendo este juntamente com o composto C reagentes, a reacção será mais extensa, diminuindo a concentração de C. Mas um aumento de F_1, tambem implica um aumento de solvente no sistema, provocando tambem uma diminuição da concentração de C.

Relativamente a C_D, apesar de existir uuma maior quantidade deste composto, pois como ja referido anteriormente a reação tornou-se mais extensa, o aumento do volume do sistema, como é visível pela Figura 4.6, assim como uma mior quantidade de solvente, levou a uma diminuição da concentração.

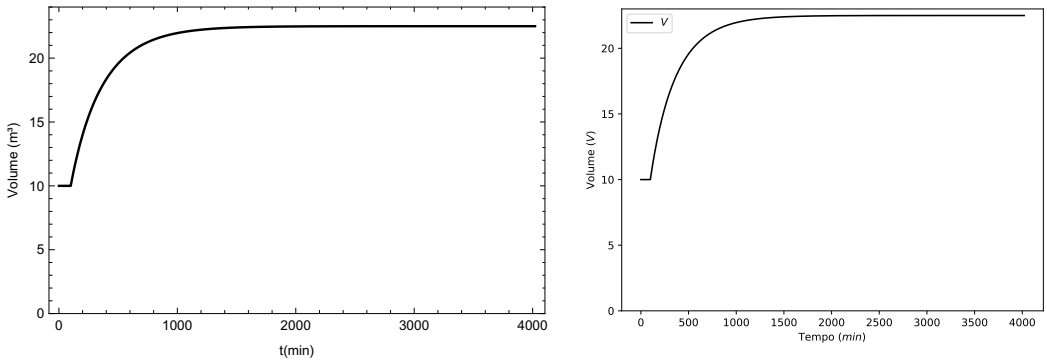
A resposta das variáveis à perturbação em F_1, pode ser aproximada a um comportamento de 1ª ordem sem atraso.

Quanto à dinâmica, estes sistema, para estas variáveis apresenta ter um comportamento aparente de 1ª ordem.

Quando comparados os perfis obtidos pelo Wolfram Mathematica® e pelo Pyomo , precebe-se a proximidade destes, tendo respondido bem a uma variação do caudal de entrada 1.

Em seguida, foi estudado o efeito da variação de F_1 no volume do sistema.

Na Figura 4.6 está presente a resposta do volume à perturbação em degrau em F_1 anteriormente apresentada.



(a) Wolfram Mathematica® **(b)** Pyomo

Figura 4.6 Variação do volume de líquido no sistema ao longo do tempo, quando aplicada uma perturbação em F_1 aos 100 *min*, para diferentes ferramentas computacionais

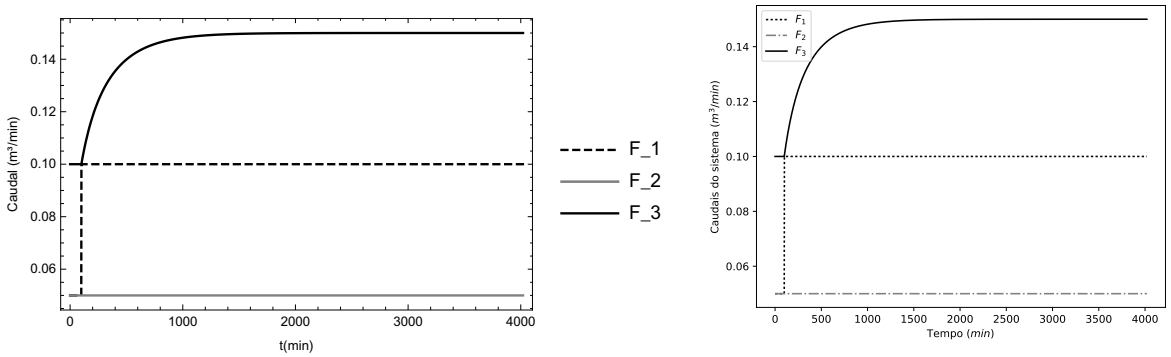
Mais uma vez não é possível distinguir diferenças entre os resultados obtidos pelo Wolfram Mathematica® , Figura 4.6a, e os resultados obtidos pelo Pyomo , Figura 4.6b.

Quanto à influência da variação de F_1 no volume de fluido no reactor, verifica-se, como seria de esperar, que um aumento de F_1 provoca um aumento de V.

O volume do líquido, quando o sistema sofre uma perturbação no caudal F_1, tem um comportamento aparente de 1ª ordem.

Tendo em conta que o caudal F_3 é dependente de V, como representado pela equação (4.9), decidiu-se acompanhar a influência da variação de F_1, nos diferentes caudais. Esta informação está presente nas Figuras 4.7b.

Novamente os perfis obtidos pelo Wolfram Mathematica® , Figura 4.7a, e os perfis obtidos pelo Pyomo , Figura 4.7b, são coincidentes.



(a) Wolfram Mathematica® **(b)** Pyomo

Figura 4.7 Variação dos caudais F_1, F_2 e F_3 ao longo do tempo, quando aplicada uma perturbação em F_1 aos 100 min, para diferentes ferramentas computacionais

O caudal F_2 não apresenta qualquer variação uma vez que este é completamente independente do sistema. Quanto ao caudal F_1 é possível ver a perturbação em degrau provocada neste teste dinâmico

Em relação ao caudal F_3, este teve uma resposta aparente de 1ª ordem, à perturbação ocorrida no sistema, tendo passando de 0.10 m^3/min para 0.15 m^3/min. Esta resposta pode ser justificada pelo facto de F_3 ser dependente de V, como demosntrado pela equação 4.9.

Perturbação na Concentração de entrada de A (C_{A0}))

A perturbação em C_{A0} tem como principal objectivo, ver como uma variação na concentração do composto A na alimentação irá afectar o resto do sistema.

Para tal fez-se uma perturbação em degrau em C_{A0}, ao minuto 100. A alimentação passou de uma concentração de alimentação de A de 1 mol/m^3, para uma concentração de 1.5 mol/m^3.

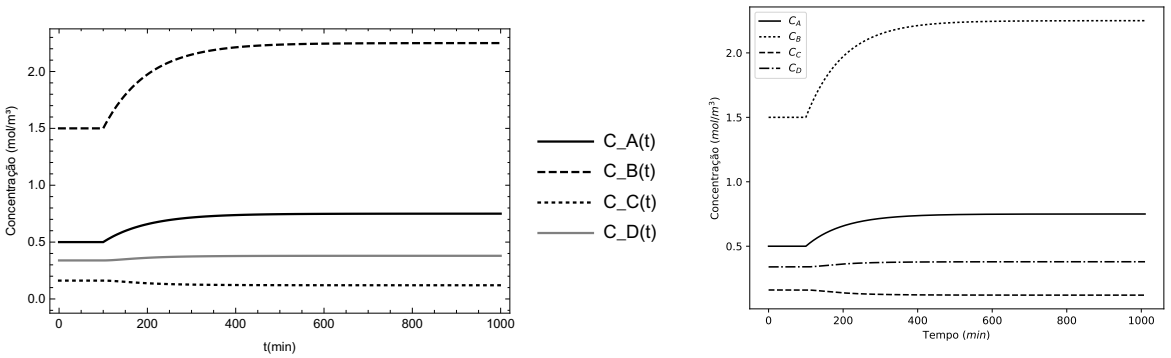
Para as simulações recorreu-se ao Wolfram Mathematica® e ao Pyomo , obtendo os resultados apresentados na Tabela 4.7.

Tabela 4.7 Resultados de simulação para as condições inicias apresentadas na Tabela 4.4 e para a perturbação na concentração C_{A0}.

	Wolfram Mathematica®	Pyomo	Unidades
C_A	0.749969	0.750000	$^{mol}/_{m^3}$
C_B	2.249969	2.250000	$^{mol}/_{m^3}$
C_C	0.120488	0.120483	$^{mol}/_{m^3}$
C_D	0.379511	0.379516	$^{mol}/_{m^3}$
V	9.99985	9.99985	m^3

Também os resultados obtidos para esta simulação são idênticos para as duas ferramentas computacionais usadas, o Wolfram Mathematica® e o Pyomo .

Na Figura 4.8, estão presentes os perfis das concentrações de saída dos diferentes compostos.



(a) Wolfram Mathematica®

(b) Pyomo

Figura 4.8 Variação da concentração de C_A, C_B, C_C e C_D ao longo do tempo, quando aplicada uma perturbação em C_{A0}, para diferentes ferramentas computacionais.

Os perfis obtidos pelo Wolfram Mathematica® , Figura 4.8a, e pelo o Pyomo , Figura 4.8b, são coincidentes.

A perturbação em degrau de C_{A0} provoca um aumento de C_A, uma vez que se está a introduzir mais composto A no sistema. Estando o composto B em equilíbrio com A, um aumento de A provoca também um aumento de B, sendo assim C_B também aumenta.

O aumento de C_B provocou, por sua vez um maior consumo do composto C, uma vez que estes dois elementos são reagentes da reacção. Este consumo provocou uma diminuição de C_C.

Contudo, a reacção química foi mais extensa, o que provocou um aumento de C_D.

Pela análise dos perfis dos gráficos presentes na Figura 4.8, pode-se afirmar que perante uma perturbação em C_{A0}, C_A apresenta uma resposta típica de um sistema de 1ª ordem, assim como C_B. Por outro lado, C_C e C_D, apresentam um comportamento aparente de segunda ordem sobre-amortecido, este comportamento pode ser justificado pelo facto do consumo de C e a produção de D estárem dependentes de C_B.

Em relação ao volume e ao caudal de saída não foram detectadas variações, uma vez que a perturbação foi ao nível da concentração de entrada, mantendo os caudais de entrada constantes.

Após a análise tanto dos perfis obtidos para o caso da perturbação em F_1, como para a perturbação em C_{A0}, assim como os resuldados dos novos estados estacionários, e conciderando que o Wolfram Mathematica® uma ferramenta de referência, pode-se concluir que o Pyomo consegue simular bem perturbações no sistema.

4.1.6 Sistema de controlo

Uma das aplicações da simulação e dos modelos é a capacidade de, a partir destas, poder-se projectar sistemas de controlo, para os casos dos controladores mais simples, como é o caso

do PID ou mesmo incorporar o código de controladores mais complexos, com os *Feedforward* (Stephanopoulos 1984).

Os controladores *Feedback*, apesar de não representarem o estado da arte neste campo, estão muito presentes em controlo, pois, para além de serem de baixo custo, são usados para sistema de redundância e para estabilizar sistemas instáveis quando se recorre a estratégias avançadas de controlo, como é o caso do controlo óptimo.

Neste último exemplo, o modelo que será utilizado para fazer a previsão irá incluir também as equações dos controlador PID.

Com o objectivo de perceber como estas equações influenciariam a simulação do sistema, projectou-se um controlador PID, tendo como objectivo garantir que o volume de líquido no reactor se mantém no valor pretendido e o mais constante possível, mesmo existindo perturbações nas variáveis de entrada.

Projecto do Controlador

Com o objectivo de perceber qual seria a melhor variável manipulada para controlar o nível, optou-se por proceder ao cálculo da Matriz *Relative Gain Array Method* (RGA), sendop necessário primeiro calcular a Matriz dos Ganhos Estacionários, Esta matriz armazena os ganhos estacionários entre uma variável de entrada e uma variável de estado, ou seja

$$\mathrm{K} = \begin{bmatrix} K_{11} & \dots & K_{1j} & \dots & K_{1n} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ K_{i1} & \dots & K_{ij} & \dots & K_{in} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ K_{m1} & \dots & K_{mj} & \dots & K_{mn} \end{bmatrix} \tag{4.28}$$

sendo para o caso do sistema em estudo, $i \in [C_{A0}\ C_{C0}\ C_{D0}\ F_1\ F_2\ \beta_2]$ e $j \in [V\ C_A\ C_B\ C_C\ C_D]$.

Aplicando a matriz (4.28), para o nosso sistema, obtém-se a Matriz dos Ganhos Estacionários (4.29).

$$\begin{bmatrix} 0 & 0 & 0 & 199.997 & 199.997 & -632.442 \\ 0.5 & 0 & 0 & 5 & -5 & 0 \\ 1.5 & 0 & 0 & 15 & -15 & 0 \\ -0.109262 & 0.161292 & 0 & -3.79815 & 1.61262 & 6.91027 \\ 0.109262 & 0.338708 & 0.5 & -1.20185 & 3.38708 & -6.91027 \end{bmatrix} \tag{4.29}$$

Observando a matriz (4.29), é possível verifocar que a linha correspondente a C_A e a linha correspondente a C_B, são linearamente dependentes, o que pode ser justificado pelas relação

de equilibrio entre A e B. Como tal, para efeito de calculos fucturos irá-se eliminar a linha correnpondente a C_B.

Para calcular a matriz RGA recorreu-se ao método de Bristol discrito em Seborg et al. (2004), tendo sido os cálculos feitos no Wolfram Mathematica® , obtendo assim a matriz (4.30), representando as colunas $[C_{A0}\ C_{C0}\ C_{D0}\ F_1\ F_2\ \beta_2]$ respetivamente, e as linhas $[V\ C_A\ C_C\ C_D]$.

$$\begin{bmatrix} 0 & 0 & 0 & 0.989411 & 0.994759 & -0.984171 \\ 0.336979 & 0 & 0 & 0.574609 & 0.088412 & 0 \\ 0 & 0.111104 & 0 & -0.870946 & 0.203154 & 1.55669 \\ 0 & 0.224228 & 0.339638 & 0.29924 & -0.28915 & 0.426044 \end{bmatrix} \tag{4.30}$$

Pela análise da matriz 4.30, pode-se averiguar que as melhores variáveis manipuladas para controlar o volume do tanque são o F_1, o F_2, optando-se por emparelhar F_1 com V, mesmo não sendo a primeira escolha a fazer pela estratégia apresentada por Seborg et al. (2004).

Na Figura 4.9 está esquematizada a configuração de controlo proposta.

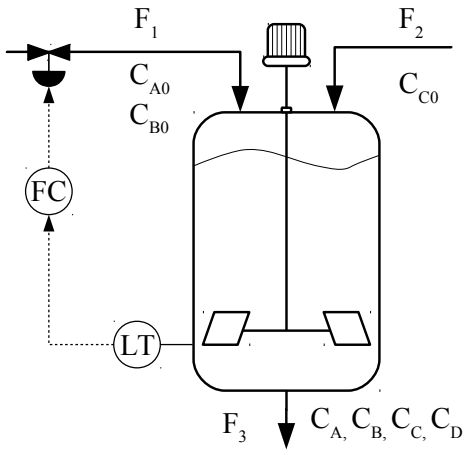


Figura 4.9 Esquema do RPA em estudo, com o controlo do volume recorrendo à corrente de entrada F_1

Como já referido em 4.1.5, a resposta do volume a uma perturbação no caudal F_1 aparenta um comportamento de 1ª ordem, contudo a função de transferência proposta pelo Wolfram Mathematica® é de uma ordem superior. Para o cálculo do PID decidiu-se então aproximar a uma função de transferência de 1ª ordem. Para este cálculo recorreu-se ao teste dinâmico, obtendo um ganho ($K_{F_1,V}$) de 199.997 *min*, vindo da Matriz dos Ganhos Estacionários e uma constante de tempo (τ) de 240 *min*, sendo esta o tempo em que o sistema atinge 63.2 % (Seborg et al. 2004).

Com estes valores, recorrendo às formulas do método *Internal Model Control* (IMC) e tendo em conta que se aproximou a um sistema de 1ª ordem, fez-se os cálculos do Ganho Proporcional (K_c) e da constante de tempo integral (τ_I), obtendo um controlador PI.

Para o cálculo dos parâmetros do controlador é necessário prever a constante de tempo em ciclo fechado desejada (τ_c). Como este valor não é conhecido foi feita uma estimativa recorrendo à estratégia proposta em Chen & Fruehauf (1990), onde propõe que $\tau > \tau_C > \theta$, onde θ é o atraso do sistema. Como se aproximou o sistema, a um sistema de 1ª ordem sem atraso, $\theta = 0$, logo τ_c terá de ter um valor entre 0 min e 240 min. Optou-se por atribuir um valor de τ_c de metade de τ, ficando $\tau_c = 120\ min$.

Segundo o método IMC apresentado por Chen & Fruehauf (1990), para o caso de um sistema de 1ª ordem sem atraso,

$$K_c K = \frac{\tau}{\tau_c} \Leftrightarrow K_c = \frac{\tau}{\tau_c K} \tag{4.31}$$

, e $\tau_I = \tau$. Sendo assim ficamos com $K_c = 0.010\ min^{-1}$, e $\tau_I = 240\ min$.

Implementação do controlador

Tendo os parâmetros dos controladores definidos, ter-se-á agora que incorporar o modelo do controlador no modelo do sistema.

Tipicamente o modelo associado aos controladores PID no domínio tempo tem a estrutura apresentada pela equação (4.32) (Stephanopoulos 1984).

$$u'(t) = K_c \left[e'(t) + \frac{1}{\tau_I} \int_0^t e'(\tau) d\tau + \tau_D \frac{de'(t)}{dt} \right] \tag{4.32}$$

Tendo em conta que no caso de controlo desejado ir-se-á ter um controlador PI, a equação (4.32) fica:

$$u'(t) = K_c \left[e'(t) + \frac{1}{\tau_I} \int_0^t e'(\tau) d\tau \right] \tag{4.33}$$

Aplicando a equação (4.33) para o caso concreto do modelo, obtém-se

$$F_1'(t) = K_c \left[e'(t) + \frac{1}{\tau_I} \int_0^t e'(\tau) d\tau \right] \tag{4.34}$$

, onde $e'(t) = V_{SP} - V(t)$, sendo V_{SP} o *set-point* de volume de líquido no reactor.

Na equação (4.34) foi feita uma substituição de variáveis de modo a eliminar o integral da

equação, sendo este substituído por $\vartheta(t)$, ficando com:

$$\vartheta(t) = \int_0^t e'(\tau)d\tau$$

, ou seja,

$$F_1'(t) = K_c \left[e'(t) + \frac{1}{\tau_I}\vartheta(t) \right] \tag{4.35a}$$

$$\frac{d\vartheta}{dt} = e'(t) \tag{4.35b}$$

$$e'(0) = 0 \tag{4.35c}$$

Sendo $F_1'(t)$ uma variável de desvio, ou seja, $F_1'(t) = F_1(t) - F_{1_{SS}}(t)$, sendo $F_{1_{SS}}$, o caudal 1 em estado estacionário, logo $F_1(t) = F_{1_{SS}}(t) + F_1'(t)$, ficando:

$$F_1(t) = F_{1_{SS}} + K_c \left[e'(t) + \frac{1}{\tau_I}\vartheta(t) \right] \tag{4.36a}$$

$$\frac{d\vartheta}{dt} = e'(t) \tag{4.36b}$$

As equações (4.36) formam um conjunto de DAES de índice 1, pois a equação (4.36a) é uma equação algébrica, e a equação (4.36b) diferencial.

Introduzindo as equações (4.36), no modelo do já apresentado pelas equações (4.25), poderemos simular o comportamento do controlador a perturbações no sistema.

Resultados do Sistema com Controlador

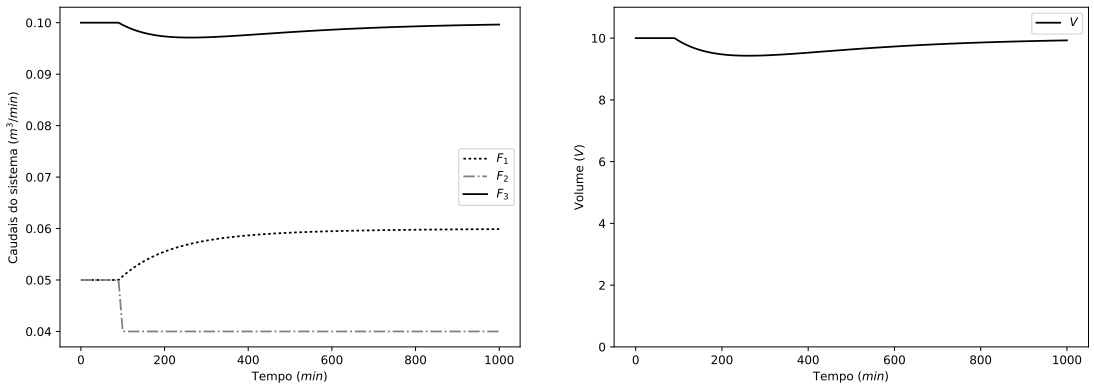
Foram introduzidas algumas perturbações no controlador de modo a perceber como este interagia com o modelo, bem como o sistema Pyomo se comportava quando aumentada a complexidade do sistema simulado.

Com o objectivo de saber como o sistema com controlo se comportava quando aplicada uma variação na alimentação, foi feita uma perturbação em degrau a F_2, passando este de 0.05 m^3/min para 0.04 m^3/min.

Esta perturbação permitiu obter as Figuras 4.10 e 4.11.

Na Figura 4.10a está representada a variação dos caudais ao longo do tempo.

É visível o degrau efectuado em F_2, que provocou uma diminuição do volume como é visível na Figura 4.10b. Esta diminuição do volume justifica a diminuição de F_3.



(a) Variação dos caudais ao longo do tempo. **(b)** Variação dos volumes ao longo do tempo

Figura 4.10 Efeito da perturbação em degrau em F_2

É possível ver que o controlador está a funcionar, pois o volume tem uma tendência crescente, tendendo para 9.99986 m^3, valor de *set-point*. Esta subida ocorre graças à acção do controlador PI, que manipula o caudal F_1, levando este a atingir o valor de 0.06 m^3/min.

Contudo, ainda demora algum tempo para que o volume volte a atinjir o *set-point*, o que pode ser justificado pelos parâmetros do controlador, pois os valores obtidos pela estratégia acima apresentada, servem para fazer uma estimativa inicial dos parâmetros dos controladores, podendo estes ser ajustados à *posteriori*.

Na Figura 4.11, está representada a variação da concentração dos diferentes compostos.

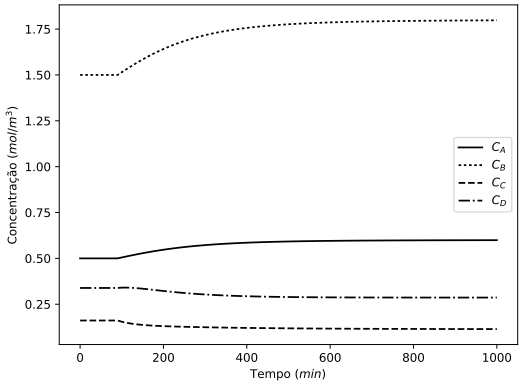


Figura 4.11 Variação das concentrações ao longo do tempo, perante uma perturbação em degrau em F_2.

Como era previsível, ao fazer-se esta perturbação em ciclo fechado, C_C diminui, uma vez que está a ser introduzido em menos quantidade, ao ser alimentado pelo caudal F_2. Por outro lado, existe um aumento de C_A e de C_B, pois estes são alimentados por F_1, caudal que compensou a redução de F_2. Quanto à concentração de C_D, este sofreu uma redução causada pela redução de C_C, um reagente da reacção.

Tendo em conta que o controlador tem de ter a capacidade de levar o sistema a convergir para um novo ponto de operação, fez-se uma variação do *set-point* do volume de líquido no reactor de 9.99986 m^3 para 15 m^3, de forma a ver como o sistema evoluía.

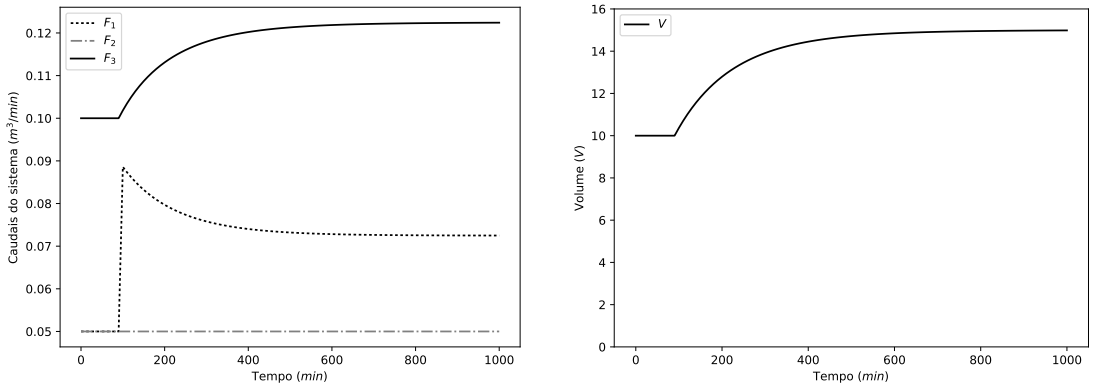
As Figuras 4.12 e 4.13 têm representados os efeitos da variação do *set-point*.

Como é visível pela Figura 4.12b, o sistema respondeu positivamente à mudança de *set-point*, passando de 10 m^3 para 15 m^3. Esta subida provoca um aumento do F_3, pelo facto de este ser dependente do volume de líquido no sistema. O aumento do volume é provocado pelo caudal de entrada F_1 (variável manipulada), que sofre um aumento na fase inicial, de modo a permitir uma resposta mais rápida, estabilizando posteriormente com um caudal de 0.074 m^3/min.

Quanto ao caudal F_2 não sofre qualquer alteração.

Esta variação do *set-point*, também teve consequências em relação às concentrações dos compostos. Novamente se regista um aumento das concentrações C_A e C_B, estando este relacionados com o aumento de F_1.

Por outro lado existe uma diminuição de C_C, pois a quantidade de C que está a ser alimentada ao sistema mantém-se, no entanto o volume de líquido aumenta, o que leva à diminuição de C_C



(a) Variação dos caudais ao longo do tempo **(b)** Variação do volume ao longo do tempo

Figura 4.12 Efeito da mudança de *set-point*.

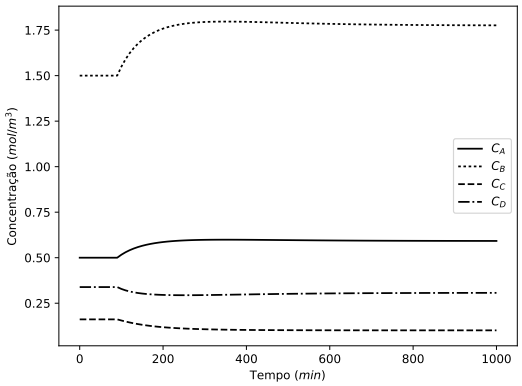


Figura 4.13 Variação das concentrações ao longo do tempo, para uma mudança de *set-point*.

no sistema.

A diminuição de C_D também está associada ao aumento do volume do sistema.

Tendo em conta o comportamento do Pyomo para este sistema mais simples, tendo apresentado um bom desempenho nas várias fases, o próximo ponto seria aplica-lo a um sistema mais composto como é o caso da Coluna de Absorção de NO_x, apresentado na secção 4.2.

4.2 Coluna de Absorção de NO_x

Uma coluna de absorção pode ser constituída por vários andares. Teoricamente em cada um destes andares atinge-se um equilíbrio entre as composições do composto absorvido na fase gasosa e na fase líquida (Azevedo & Alves 2009).

A absorção dos compostos ocorrem na interface entre o líquido e o gás, sendo por este motivo, as concentrações nas camadas filme, tanto no gás, como no líquido importantes.

No caso da Coluna de Absorção de NO_x, em paralelo com o processo de absorção também ocorrem reacções, tanto na fase gasosa como na fase líquida (Vilarinho et al. 2016).

Na Figura 4.14, está esquematizado um andar genérico (n) da Coluna de Absorção de NO_x.

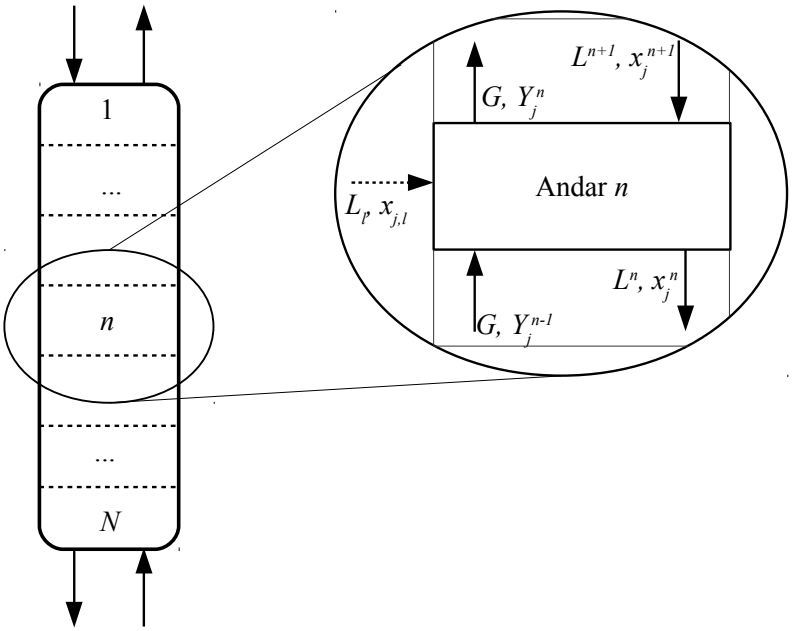


Figura 4.14 Esquema da Coluna de Absorção de NO_x adaptado de Vilarinho et al. (2016)

Cada andar (n) é alimentado por uma corrente de gás de transporte com um caudal G, sendo esta constante ao longo da coluna, e com uma fracção molar Y do composto j, proveniente do andar anterior ($n-1$).

Este andar é ainda alimentado por uma corrente de líquido de transporte, com um caudal L, proveniente do andar seguinte ($n+1$). Esta corrente possui uma fracção molar x do composto j.

Neste caso a corrente de líquido de transporte pode variar de andar para andar, pois pode existir a introdução de uma corrente líquida ao andar, representada na Figura 4.14 com a entrada a tracejado. Esta corrente terá um caudal L_l, tendo uma fracção molar x_l do composto j.

Do andar n sai uma corrente gasosa e o caudal de transporte G, com uma fracção molar Y do composto j, para o andar $n+1$. Sai ainda uma corrente líquida, com um caudal L, sendo este igual ao caudal de alimentação do andar anterior, mais a corrente líquida introduzida na coluna, caso exista, ou seja, $L^n = L^{n+1} + L_l^n$. A saída terá uma fracção molar x do composto j, para o andar n.

Na Figura 4.15, estão esquematizados os mecanismos de transferência de massa que ocorrem na coluna.

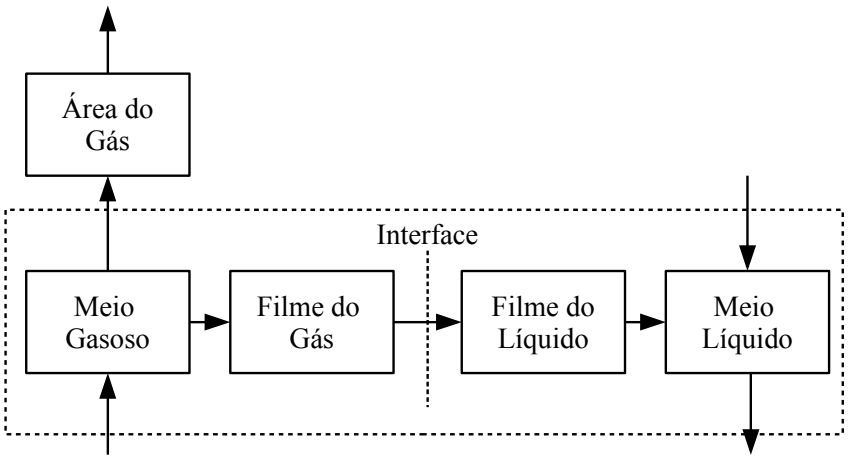


Figura 4.15 Esquema dos Mecanismos de transferência de massa num andar da Absorção de NO_x adaptado de Vilarinho et al. (2016)

Em Vilarinho et al. (2016), considerou-se duas zonas distintas: a área de gás (*Gas Area*), representada na Figura 4.15 pelo bloco com o mesmo nome, e a área de espuma (*Froth Area*).

A área de gás corresponde ao espaço livre entre pratos consecutivos e é onde ocorrem reacções de oxidação (Vilarinho et al. 2016).

Já a área de espuma corresponde à zona onde ocorre o contacto entre a fase gasosa e a fase líquida (Vilarinho et al. 2016).

A transferência de massa, como é visível pela Figura 4.15, ocorre do meio gasoso para o meio líquido. Como tal o compostos irão passar do meio gasoso, meio em que se pode considerar mistura perfeita derivado à agitação do meio, para a camada filme de gás junto à interface. Nesta camada filme, a agitação é reduzida sendo causado um gradiente de concentração entre o meio gasoso e a interface.

Na interface, o composto é transferido para o líquido, mais propriamente para a camada filme do líquido. Esta transferência está dependente das leis de equilíbrio gás-líquido, solubilidade do composto no líquido e no gás, entre outros, sendo representado por equações de estado como a lei de Henry ou a Lei de Raoult.

Tal como na camada filme do gás, a camada filme do líquido apresenta pouca agitação, provocando um gradiente de concentrações ao longo desta camada. Por fim, os composto atingem meio líquido, local onde se pode voltar a considerar agitação perfeita.

Como já referido anteriormente, na coluna de absorção de NO_x, para além do fenómeno de absorção ocorrem em paralelo reacções químicas (Vilarinho et al. 2016).

As equações (4.37), representam as reacções que ocorrem no meio gasoso (Vilarinho et al.

2016).

$$2NO(g) + O_2(g) \rightarrow 2NO_2(g) \tag{4.37a}$$
$$2NO_2(g) \rightleftharpoons N_2O_4(g) \tag{4.37b}$$
$$3NO_2(g) + H_2O \rightleftharpoons 2HNO_3(g) + NO(g) \tag{4.37c}$$
$$NO(g) + NO_2(g) \rightleftharpoons NO_3(g) \tag{4.37d}$$
$$NO(g) + NO_2(g) + H_2O \rightleftharpoons 2HNO_2(g) \tag{4.37e}$$

As equações (4.38), representam as reacções que ocorrem no meio líquido (Vilarinho et al. 2016).

$$N_2O_4(l) + H_2O(l) \rightarrow HNO_2(l) + HNO_3(l) \tag{4.38a}$$
$$N_2O_3(l) + H_2O(l) \rightarrow HNO_2(l) \tag{4.38b}$$
$$2NO_2(l) + H_2O(l) \rightarrow HNO_3(l) + HNO_2(l) \tag{4.38c}$$
$$3HNO_2(l) \rightarrow HNO_3(l) + H_2O(l) + 2NO_2(g) \tag{4.38d}$$

Já a equação (4.39), representa o equilíbrio de fases (Vilarinho et al. 2016).

$$3NO_2(g) + H_2O(l) \rightleftharpoons 2HNO_3(l) + NO(g) \tag{4.39}$$

Tendo em conta as reacções apresentadas e a informação presente em Vilarinho et al. (2016) prevê-se que este sistema tenha um total de 637 equações tendo em conta o número de equações em cada andar, e que estas se repetem ao longo dos andares da coluna de absorção.

O próximo passo seria fazer a modelação e a simulação do sistema, recorrendo ao Pyomo numa primenira fase para fazer a simulação do sistema e depois comparado com resultados vindo de outras plataformas, ou com dados reais, de modo a validar o modelo. Pós o modelo estra validado, poderia-se fazer optimização recorrendo a este. Nesta optimizaçaõ seria possível, por exemplo, calcular o melhor caudal de alimentação da corrente gasosa e/ou líquida, de modo a optimizar o processo de tratamento.

Capítulo 5

Conclusões e Trabalhos Futuros

A resolução de sistemas modelados por DAES apresenta um ponto de interesse devido não só aos modelos que envolvem à partida DAES, mas também para os restantes modelos, uma vez que quando se está a modelar e a resolver problemas usando uma metodologia de blocos, estratégia cada vez mais comum, este tipo de equações costumam a aparecer, como foi o caso de quando foi acrescentado o bloco do controlador ao bloco do modelo.

Nesta dissertação foi possível comparar o desempenho de duas ferramentas computacionais destintas, o Pyomo e o Wolfram Mathematica® , recorrendo a um modelo tipo de um RPA, contendo dois compostos em equilibrio químico.

Quando comparado o Pyomo, uma das ferramentas *open-source*, que permite a resolução e optimização de problemas modelados por DAES, com o Wolfram Mathematica® , ferramenta de simulação com capacidade de resolver DAES, viu-se que os resultados eram idênticos para ambas as ferramentas, apesar de o Pyomo ser construído para problemas de optimização.

Não foi feita uma comparação de tempos computacionais, pois apesar de neste estudo termos recorrido a ambas as ferramentas para fazer simulação, o verdadeiro objectivo do Pyomo seria a optimização, não mostrando grande diferença neste ponto, demorando ambos alguns segundo a correr o código. Contudo é de prever quando se recorrer ao Pyomo para optimizar, uma ou mais, variáveis do processo, este seja mais rápido que o optimizador do Wolfram Mathematica® , pois o Pyomo resolve problemas de optimização recorrendo a métodos simultâneos, enquanto que o Wolfram Mathematica® recorre a métodos sequênciais.

Uma das maiores dificuldades do recurso do Pyomo foi o facto de o desenvolvimento deste ser feito de uma forma independente do Python, não sendo a última versão dop Pyomo completamente compativel com a última versão do Python.

Por outro lado, o facto de ser *open-source*, tráz vantagens, como a capacidade de traduzir o código de optimização do Pyomo para GAMS, discretizando as equações diferenciais, e permitindo assim recorrer ao gams para optimizar, bem como recorrer aos seus *solvers*, mais sufesti-

cados e eficazes que os utilizados pelo Pyomo .

Conhecendo a capacidade do Pyomo para simular DAES, o próximo passo seria optimizar o modelo do reactor recorrendo a esta ferramenta, de forma a testar a capacidade de optimização. Neste ponto poderia-se explorar também a capacide de transcrever o código para outras linguagens, comparando tempos de optimização e resultados.

Poderia-se ainda aplicar uma estratégia de controlo optimo ao sistema

Após explorar de uma forma mais intensa o Pyomo , iria-se passar para a simulação e a optimização de sistemas mais complexos, como o caso do sistema da coluna de absorção de NO_x.

Bibliografia

AMPL, O. i. (2018*a*), 'Open source solvers', https://ampl.com/products/solvers/open-source/.

AMPL, O. i. (2018*b*), 'Solvers we sell', https://ampl.com/products/solvers/solvers-we-sell/.

Andersson, J. A. E., Gillis, J., Horn, G., Rawlings, J. B. & Diehl, M. (In Press, 2018), 'CasADi – A software framework for nonlinear optimization and optimal control', *Mathematical Programming Computation* .

Ascher, U. M. & Petzold, L. R. (1994), *Computer for Ordinary Differential Equantions and Differential-Algebraic Equations*, SIAM, Filadélfia.

Aspuru-Guzik, A., Lindh, R. & Reiher, M. (2018), 'The matter simulation (r)evolution', *ACS Central Science* **4**(2), 144–152.

Azevedo, E. G. d. & Alves, A. M. (2009), *Engenharia de Processos de Separação*, 1st edn, IST Press.

Bendtsen, C. & Thomsen, P. G. (1999), Numerical solution of differential algebraic equations, Technical report, IMM.

Bequette, B. W. (1998), *Process Dynamics - Modeling, Analysis, and Simulation*, Prentice-Hall Inc.

Biegler, L. T. (2000), 'Optimization of differential- algebraic equation systems', https://www.lehigh.edu/~wes1/apci/26may00.pdf.

Biegler, L. T. (2007), 'An overview of simultaneous strategies for dynamic optimization', *Chemical Engineering and Processing* pp. 1043–1053.

Brenan, K. E., Campbell, S. L. & Petzold, L. R. (1996), *Numerical Solution of Initial-Value Problems in Differential-Algebraic Equation*, SIAM.

CasADi (2018*a*), 'CasADi', https://web.casadi.org/.

CasADi (2018*b*), 'Welcome to CasADi's documentation!', https://web.casadi.org/docs/#document-daebuilder.

Chen, I.-L. & Fruehauf, P. S. (1990), 'Consider imc tuning to improve controller performace', *Chem. Eng. Progress* **86**(10), 33.

Diehl, M., Bock, H. G., Schlöder, J. P., Findeisen, R., Nagy, Z. & Allgöwer, F. (2002), 'Real-time optimization and nonlinear model predictive control of processes governed by differential-algebraic equations', *Journal of Process Control* **12**, 577–585.

Dunning, I., Huchette, J. & Lubin, M. (2017), 'JuMP: A Modeling Language for Mathematical Optimization', *SIAM Review* **59**(2), 295–320.

Eaton, J. W. (2018*a*), 'About (GNU Octave)', https://www.gnu.org/software/octave/about.html.

Eaton, J. W. (2018*b*), 'Differential-algebraic equations', https://octave.org/doc/interpreter/Differential_002dAlgebraic-Equations.html.

ESI, G. (2017), 'scilab', http://www.scilab.org/.

FreeMat (2018), 'Freemat', http://freemat.sourceforge.net/.

Froment, G. F. & Bischoff, K. B. (1990), *Chemical Reactor Analysis and Design*, second edn, John Wiley & Sons.

GAMS, D. C. (2018*a*), 'Convert', https://www.gams.com/latest/docs/S_CONVERT.html.

GAMS, D. C. (2018*b*), 'Solver manuals', https://www.gams.com/latest/docs/S_MAIN.html.

Geletu, A. (2011), 'Introduction to differential algebraic equantion', Ilmenau University of Technology.

Guzel, N. & Bayram, M. (2006), 'Numerical solution of differential-algebraic equations whit index-2', *Applied Mathematics and Computation* **174**, 1279–1289.

Hairer, E., Lubich, C. & Roche, M. (1989), *The Numerical Solution of Differential-Algebraic Systems by Runge-Kutta Methods*, Springer-Verlag.

Hairer, E. & Wanner, G. (1996), *Solving Ordinary Differential Equations II*, second edn, Springer.

Harney, D., Mills, T. & Book, N. (2013), 'Numerical evaluation of the stability of stationary points of index-2 differential-algebraic equations: Applications to reactive flash and reactive distillation systems', *Computers & Chemical Engineering* **46**, 61–69.

Hart, W. E., Laird, C. D., Watson, J.-P., Woodruff, D. L., Hackebeil, G. A., Nicholson, B. L. & Siirola, J. D. (2017), *Pyomo–optimization modeling in python*, Vol. 67, second edn, Springer Science & Business Media.

Hindmarsh, A. C., Brown, P. N., Grant, K. E., Lee, S. L., Serban, R., Shumaker, D. E. & Woodward, C. S. (2005), 'Sundials', *ACM Transactions on Mathematical Software* **31**(3), 363–396.

Kallrath, J. (2004), *Modeling Languages in Mathematical Optimization*, Kluwer Academic Publishers.

Lawrence Livermore, N. L. (2018), 'Sundials: Suite of nonlinear and differential/algebraic equation solvers', https://computation.llnl.gov/projects/sundials.

Modelica, A. (2018), 'Modelica and the modelica association', https://www.modelica.org/.

Oliveira, N. M. C. d. (2006), *Práticas de Computação*, Imprensa da Universidade de Coimbra.

OpenModelica (2018), 'OpenModelica', https://www.openmodelica.org/.

Petzold, L., Krintz, C., Lotstedt, P. & Hellander, A. (n.d.), 'Software - stochss'.

Process Systems Enterprise, L. (2018), 'gPROMS', https://www.psenterprise.com/products/gproms.

Python, S. F. (2018), 'The python standard library', https://docs.python.org/2/library/index.html.

Sandia, N. L. L. (2014), 'Pyomo online documentation 3.5', software.sandia.gov/downloads/pub/coopr/CooprGettingStarted.html.

Seborg, D. E., Edgar, T. F. & Mellichamp, D. A. (2004), *Process Dynamics and Control*, 2nd edn, John Wiley & Sons, Inc.

Stephanopoulos, G. (1984), *Chemical Process Control - An Introduction to Theory and Practice*, Prentice-Hall International.

StochSS (2018), 'Stochss - stochastic simulation service', http://www.stochss.org/.

The MathWorks, I. (2018*a*), 'Add-on product requirements & platform availability for R2018a', https://www.mathworks.com/products/availability.html#ML.

The MathWorks, I. (2018*b*), 'Simulink', https://www.mathworks.com/products/simulink.html.

The MathWorks, I. (2018*c*), *Solve Differential Algebraic Equations (DAEs)*.

Vasantharajan, S. & Biegler, L. T. (1990), 'Simultaneous strategies for optimization of differential-algebraic systems with enforcement of error criteria', *Computers Chem. Engng.* **14**(10), 1083–1100.

Vilarinho, I. L., Oliveira, N. M., Duarte, B. P. & Pereira, S. E. (2016), 'Equation-based rigorous modelling of the NO_x absorption process: Model development and process optimization', *Computer Aided Chemical Engineering* **38**, 1479–1484.

Wikipedia (2018*a*), 'Matlab', https://en.wikipedia.org/wiki/MATLAB.

Wikipedia (2018*b*), 'Modelica', https://en.wikipedia.org/wiki/Modelica.

Wikipedia (2018*c*), 'Simulink', https://en.wikipedia.org/wiki/Simulink.

Wikipedia (2018*d*), 'Wolfram mathematica', https://en.wikipedia.org/wiki/Wolfram_Mathematica.

Wolfram (2017), *Wolfram Mathematica 11 Documetation*.

Wolfram (2018*a*), 'Numerical Solution of Differential-Algebraic Equations', https://reference.wolfram.com/language/tutorial/NDSolveDAE.html.

Wolfram (2018*b*), 'Wolfram SystemModeler', http://reference.wolfram.com/

system-modeler/.

Anexo A

Desenvolvimento dos Índices de *Hessenberg*

A.1 Forma de *Hessenberg* - Índice 1

Concideremos os sistema seguinte:

$$\frac{d\mathrm{x}}{dt} = f(t, \mathrm{x}, \mathrm{z}) \tag{A.1}$$

$$0 = g(t, \mathrm{x}, \mathrm{z}) \tag{A.2}$$

Com as seguintes condições iniciais:

$$0 = f(t_0, \mathrm{x}_0, \mathrm{z}_0) \tag{A.3}$$

$$0 = g(t_0, \mathrm{x}_0, \mathrm{z}_0) \tag{A.4}$$

Diferenciando a equação (A.2) em ordem ao tempo, obtem-se:

$$\frac{d}{dt}(0) = \frac{\partial g}{\partial \mathrm{x}}\frac{d\mathrm{x}}{dt} + \frac{\partial g}{\partial \mathrm{z}}\frac{d\mathrm{z}}{dt} \Leftrightarrow \frac{\partial g}{\partial \mathrm{z}}\frac{d\mathrm{z}}{dt} = -\frac{\partial g}{\partial \mathrm{x}}\frac{d\mathrm{x}}{dt} \tag{A.5}$$

Caso que a matriz $\frac{\partial g}{\partial \mathrm{z}}$ seja invertivel, então:

$$\frac{d\mathrm{z}}{dt} = -\left[\frac{\partial g}{\partial \mathrm{z}}\right]^{-1}\frac{\partial g}{\partial \mathrm{x}}\frac{d\mathrm{x}}{dt} \tag{A.6}$$

Das equações (A.1) e (A.6), obteremos o sistema equivalente de ODEs:

$$\frac{d\mathrm{x}}{dt} = f(t,\mathrm{x},\mathrm{z})$$
$$\frac{d\mathrm{z}}{dt} = -\left[\frac{\partial g}{\partial \mathrm{z}}\right]^{-1} \frac{\partial g}{\partial \mathrm{x}} \frac{d\mathrm{x}}{dt}$$

A.2 Forma de *Hessenberg* - Índice 2

Concideranndo o seguinte sistema:

$$\frac{d\mathrm{x}}{dt} = f(t,\mathrm{x},\mathrm{z}) \tag{A.7}$$
$$0 = g(t,\mathrm{x}) \tag{A.8}$$

Com as condições inicias:

$$0 = f(t_0,\mathrm{x}_0,\mathrm{z}_0) \tag{A.9}$$
$$0 = g(t_0,\mathrm{x}_0) \tag{A.10}$$

Diferenciando a equação (A.8), em ordem ao tempo, fica-se:

$$\frac{d}{dt}(0) = \frac{\partial g}{\partial \mathrm{x}} \frac{d\mathrm{x}}{dt} \tag{A.11}$$

Substituindo, (A.7) em (A.11) obtem-se:

$$0 = \frac{\partial g}{\partial \mathrm{x}} f(t,\mathrm{x},\mathrm{z}) \tag{A.12}$$

Diferenciando a equação (A.12) em ordem ao tempo, obtem-se:

$$\frac{d}{dt}(0) = \frac{\partial g}{\partial \mathrm{x}} \frac{\partial f}{\partial \mathrm{x}} \frac{d\mathrm{x}}{dt} + \frac{\partial g}{\partial \mathrm{x}} \frac{\partial f}{\partial \mathrm{z}} \frac{d\mathrm{x}}{dt} \tag{A.13}$$

Manipulando a equação (A.13), tem-se:

$$\frac{\partial g}{\partial \mathrm{x}}\frac{\partial f}{\partial \mathrm{z}}\frac{d\mathrm{x}}{dt} = -\frac{\partial g}{\partial \mathrm{x}}\frac{\partial f}{\partial \mathrm{x}}\frac{d\mathrm{x}}{dt} \tag{A.14}$$

Caso que as matrizes $\frac{\partial g}{\partial \mathrm{x}}$ e $\frac{\partial f}{\partial \mathrm{z}}$ sejam invertiveis, pode-se isolar $\frac{d\mathrm{x}}{dt}$, ficando:

$$\frac{d\mathrm{x}}{dt} = -\left[\frac{\partial g}{\partial \mathrm{x}}\right]^{-1}\left[\frac{\partial f}{\partial \mathrm{z}}\right]^{-1}\frac{\partial g}{\partial \mathrm{x}}\frac{\partial f}{\partial \mathrm{x}}\frac{d\mathrm{x}}{dt} \tag{A.15}$$

Tendo em conta a equação (A.7) e a equação (A.15), obtemos um sistema equivalente de ODEs.

$$\begin{aligned}\frac{d\mathrm{x}}{dt} &= f(t,\mathrm{x},\mathrm{z})\\ \frac{d\mathrm{z}}{dt} &= -\left[\frac{\partial g}{\partial \mathrm{x}}\right]^{-1}\left[\frac{\partial f}{\partial \mathrm{z}}\right]^{-1}\frac{\partial g}{\partial \mathrm{x}}\frac{\partial f}{\partial \mathrm{x}}\frac{d\mathrm{x}}{dt}\end{aligned}$$

A.3 Forma de *Hessenberg* - Índice 3

Partindo do seguinte sistema:

$$\frac{d\mathrm{x}}{dt} = f(t,\mathrm{x},\mathrm{y},\mathrm{z}) \tag{A.16}$$
$$\frac{d\mathrm{y}}{dt} = g(t,\mathrm{x},\mathrm{y}) \tag{A.17}$$
$$0 = h(t,\mathrm{y}) \tag{A.18}$$

Com as seguintes condições iniciais.

$$0 = f(t_0,\mathrm{x}_0,\mathrm{y}_0,\mathrm{z}_0) \tag{A.19}$$
$$0 = g(t_0,\mathrm{x}_0,\mathrm{y}_0) \tag{A.20}$$
$$0 = h(t_0,\mathrm{y}_0) \tag{A.21}$$

onde estas, têm de ser consistentes com o sistema.

Derivando a equação (A.18) em ordem do tempo, obtem-se:

$$\frac{d}{dt}(0) = \frac{\partial h}{\partial \mathrm{y}}\frac{d\mathrm{y}}{dt} \tag{A.22}$$

Substituindo a equação (A.17) em (A.22) fica-se com:

$$0 = \frac{\partial h}{\partial \mathrm{y}} \left(g \left(t, \mathrm{x}, \mathrm{y} \right) \right) \tag{A.23}$$

Diferenciando a equação (A.23) em ordem ao tempo, tem-se:

$$\frac{d}{dt} \left(0 \right) = \frac{\partial h}{\partial \mathrm{y}} \frac{\partial g}{\partial \mathrm{x}} \frac{d\mathrm{x}}{dt} + \frac{\partial h}{\partial \mathrm{y}} \frac{\partial g}{\partial \mathrm{y}} \frac{d\mathrm{y}}{dt} \tag{A.24}$$

Substituindo a equação (A.16) em (A.24), fica-se:

$$0 = \frac{\partial h}{\partial \mathrm{y}} \frac{\partial g}{\partial \mathrm{x}} \left(f \left(t, \mathrm{x}, \mathrm{y}, \mathrm{z} \right) \right) + \frac{\partial h}{\partial \mathrm{y}} \frac{\partial g}{\partial \mathrm{y}} \frac{d\mathrm{y}}{dt} \tag{A.25}$$

Derivando a equação (A.25) em ordem ao tempo, obtem-se:

$$\frac{d}{dt} \left(0 \right) = \frac{\partial h}{\partial \mathrm{y}} \frac{\partial g}{\partial \mathrm{x}} \frac{\partial f}{\partial \mathrm{x}} \frac{d\mathrm{x}}{dt} + \frac{\partial h}{\partial \mathrm{y}} \frac{\partial g}{\partial \mathrm{x}} \frac{\partial f}{\partial \mathrm{y}} \frac{d\mathrm{y}}{dt} + \frac{\partial h}{\partial \mathrm{y}} \frac{\partial g}{\partial \mathrm{x}} \frac{\partial f}{\partial \mathrm{z}} \frac{d\mathrm{z}}{dt} \tag{A.26}$$

Manipulando a equação (A.26), fica-se:

$$\frac{\partial h}{\partial \mathrm{y}} \frac{\partial g}{\partial \mathrm{x}} \frac{\partial f}{\partial \mathrm{z}} \frac{d\mathrm{z}}{dt} = - \left(\frac{\partial h}{\partial \mathrm{y}} \frac{\partial g}{\partial \mathrm{x}} \frac{\partial f}{\partial \mathrm{x}} \frac{d\mathrm{x}}{dt} + \frac{\partial h}{\partial \mathrm{y}} \frac{\partial g}{\partial \mathrm{x}} \frac{\partial f}{\partial \mathrm{y}} \frac{d\mathrm{y}}{dt} \right) \tag{A.27}$$

Caso que as matrizes ${}^{\partial h}/_{\partial \mathrm{y}}$, ${}^{\partial g}/_{\partial \mathrm{x}}$ e ${}^{\partial f}/_{\partial \mathrm{z}}$ sejam invertiveis, podemos isolar ${}^{d\mathrm{z}}/_{dt}$. tendo:

$$\frac{d\mathrm{z}}{dt} = - \left[\frac{\partial h}{\partial \mathrm{y}} \right]^{-1} \left[\frac{\partial g}{\partial \mathrm{x}} \right]^{-1} \left[\frac{\partial f}{\partial \mathrm{z}} \right]^{-1} \left(\frac{\partial h}{\partial \mathrm{y}} \frac{\partial g}{\partial \mathrm{x}} \frac{\partial f}{\partial \mathrm{x}} \frac{d\mathrm{x}}{dt} + \frac{\partial h}{\partial \mathrm{y}} \frac{\partial g}{\partial \mathrm{x}} \frac{\partial f}{\partial \mathrm{y}} \frac{d\mathrm{y}}{dt} \right) \tag{A.28}$$

Assim obtem-se o seguinte sistema equivalente de ODEs:

$$\begin{aligned}
\frac{d\mathrm{x}}{dt} &= f(t, \mathrm{x}, \mathrm{y}, \mathrm{z}) \\
\frac{d\mathrm{y}}{dt} &= g(t, \mathrm{x}, \mathrm{y}) \\
\frac{d\mathrm{z}}{dt} &= - \left[\frac{\partial h}{\partial \mathrm{y}} \right]^{-1} \left[\frac{\partial g}{\partial \mathrm{x}} \right]^{-1} \left[\frac{\partial f}{\partial \mathrm{z}} \right]^{-1} \left(\frac{\partial h}{\partial \mathrm{y}} \frac{\partial g}{\partial \mathrm{x}} \frac{\partial f}{\partial \mathrm{x}} \frac{d\mathrm{x}}{dt} + \frac{\partial h}{\partial \mathrm{y}} \frac{\partial g}{\partial \mathrm{x}} \frac{\partial f}{\partial \mathrm{y}} \frac{d\mathrm{y}}{dt} \right)
\end{aligned}$$

Anexo B

Entradas para as diferentes classes no Pyomo

B.1 `Var`

Na tabela B.1 está representada as declarações para a classe `Var`, assim como as circunstancias onde se aplicam.

Tabela B.1 Declarações comuns para os componente `Var` (Hart et al. 2017).

Palavra-Chave	Descrição	Valores aceites
	reservado para especificar conjunto com índices	qualquer conjunto de dados do Pyomo ou listas do Python
`within` ou `domain`	especifica um domínio especifico para uma variável	qualquer conjunto de dados do Pyomo , listas do Python ou função
`bounds`	fornece a fronteira superior inferior a uma variável	um 2-tuple ou função
`initialize`	fornece o valor inicial da variável	valor escalar ou função

Estes declarações, são muitas vezes conjugadas (Hart et al. 2017).

B.2 `Objective`

A tabela B.2 representa as declarações para a classe `Objective`.

Tabela B.2 Declarações comuns para os componente `Objective` (Hart et al. 2017).

Palavra-Chave	Descrição	Valores aceites
	reservado para especificar conjunto com índices	qualquer conjunto de dados do Pyomo ou listas do Python
`expr`	fornece uma função objectivo	qualquer expressão valida no Pyomo
`rule`	fornece uma função com regra para definir o objectivo	uma função que retorna do Pyomo ou `Objective.Skip`
`sense`	determina se o objectivo e maximizar ou minimizar (por *default* minimiza)	`minimize` ou `maximize`

B.3 `Constraint`

A tabela B.3 representa as declarações para a classe `Constraint`.

Tabela B.3 Declarações comuns para os componente `Constraint` (Hart et al. 2017).

Palavra-Chave	Descrição	Valores aceites
	reservado para especificar conjunto com índices	qualquer conjunto de dados do Pyomo ou listas do Python
`expr`	fornece uma função objectivo	qualquer expressão com operadores lógicos, um 2-tuple, 3-tuple
`rule`	fornece uma função com regra para definir uma restrição	qualquer expressão com operadores lógicos, um 2-tuple, 3-tuple, ou `Constraint.Skip`

B.4 Set

A tabela B.4 representa as declarações para a classe `Set`.

Tabela B.4 Declarações comuns para os componente `Set` (Hart et al. 2017).

Palavra-Chave	Descrição	Valores aceites
	reservado para especificar conjunto com índices	qualquer conjunto de dados do Pyomo ou listas do Python
`initialize`	fornece valores iniciais para as variáveis	Valor escalar ou função
`within` `domain`	especifica o domínio ou valores para uma variável	Pyomo `set`, listas Python função
`ordered`	especifica se a ordem do sistema deve ser preservada ou não	True/False
`virtual`	especifica se o `Set` tem dados explícitos	True/False
`bounds`	fornece um limite superior e inferior ao conjunto de dados	2-tuple ou função

B.5 `Param`

A tabela B.5 representa as declarações para a classe `Param`.

Tabela B.5 Declarações comuns para os componente `Param` (Hart et al. 2017).

Palavra-Chave	Descrição	Valores aceites
	reservado para especificar conjunto com índices	qualquer conjunto de dados do Pyomo ou listas do Python
`initialize`	fornece valores iniciais para as variáveis	Valor escalar ou função
`default`	fornece um valor caso que nenhum tenha sido atribuído	Valor escalar ou função
`validate`	especifica uma função que é chamada para verificar se o valor é valido para o parâmetro	um função que retorne verdadeiro ou falso dando um valor particular
`mutable`	especifica se o parâmetro pode ser alterado pós ter sido corrido o solver	True/False