INSTITUTO POLITÉCNICO DE BRAGANÇA
Escola Superior de Tecnologia e de Gestão

Otimização do movimento de uma partícula passiva induzido por pontos de vórtices

David de Jesus Santos Nº33961

Dissertação apresentada à Escola Superior de Tecnologia e Gestão do Instituto Politécnico de Bragança para obtenção do grau de Mestre em Engenharia Industrial – Especialização em Engenharia Eletrotécnica

Trabalho realizado sob a orientação de:

Prof. Dr. Carlos Jorge da Rocha Balsa

Esta dissertação não inclui as críticas e sugestões feitas pelo Júri.

Novembro 2020

INSTITUTO POLITÉCNICO DE BRAGANÇA
Escola Superior de Tecnologia e de Gestão

Otimização do movimento de uma partícula passiva induzido por pontos de vórtices

David de Jesus Santos Nº33961

Dissertação apresentada à Escola Superior de Tecnologia e Gestão do Instituto Politécnico de Bragança para obtenção do grau de Mestre em Engenharia Industrial – Especialização em Engenharia Eletrotécnica

Trabalho realizado sob a orientação de:

Prof. Dr. Carlos Jorge da Rocha Balsa

Esta dissertação não inclui as críticas e sugestões feitas pelo Júri.

Novembro 2020

Dedicatória

Aos meus pais, António e Fernanda, à minha irmã Marina, e a toda a minha família que, com muito carinho e apoio, não mediram esforços para que eu chegasse até esta etapa da minha vida.

Agradecimentos

Em primeiro lugar gostaria de agradecer ao orientador, Professor Doutor Carlos Balsa , pela paciência, tempo e pelo conhecimento transmitido na elaboração desta dissertação, contribuindo para o meu crescimento tanto a nível profissional como pessoal.

À minha família, em especial aos meus pais e irmã pelo apoio incondicional, por nunca deixarem que me faltasse nada, pelas palavras de carinho e de conforto que fizeram sempre com que tivesse forças para chegar mais longe.

Por fim, aos meus amigos e namorada, que me acompanharam nesta fase de forma incansável.

Resumo

A dinâmica dos vórtices e dos traçadores passivos em fluxos dominados por vórtices forma uma vasta área de investigação que continua a atrair a atenção de numerosos estudos, surgindo recentemente um interesse especial na utilização da teoria de controlo aplicada à dinâmica de vórtices.

Os vórtices pontuais são soluções das equações bidimensionais incompressíveis de Euler que correspondem ao caso limite em que a vorticidade está completamente centralizada num número finito de pontos espaciais, cada um com uma determinada circulação.

Nesta dissertação a grande preocupação é a dinâmica de um traçador passivo advinda de um fluxo de vórtices de pontos bidimensionais, ou seja, pretende-se conduzir uma partícula passiva de um ponto inicial a um ponto final, ambos dados *a priori*, num determinado espaço de tempo finito sendo o fluxo provocado pelo deslocamento de vórtices. Mais precisamente, pretende-se encontrar trajetórias ideais que minimizem a função objetivo que corresponde à energia gasta no controlo das trajetórias, ou seja, no movimento da partícula.

Para o deslocamento da partícula serão testados quatro casos diferentes, em que cada caso será utilizado um dado nº de vórtices, o qual varia de um a quatro. Posteriormente é necessário adotar uma estratégia de modo a encontrar o menor valor de energia gasto no movimento assim como encontrar esse valor o de uma forma mais rápida, permitindo assim otimizar o movimento da partícula.

Palavras-chave: Vórtice, Partícula Passiva, Controlo Ótimo, Otimização Numérica.

Abstract

Vortex dynamics and passive tracers in vortex-dominated flows form a vast area of research that continues to attract the attention of numerous studies, arising in recent times a special interest in the use of control theory applied to vortex dynamics.

Point vortices are singular solutions of the two-dimensional incompressible Euler equations. These solutions correspond to the limiting case where the vorticity is completely concentrated on a finite number of spatial points, each with a prescribed strength/circulation.

In this dissertation the great preoccupation it is the dynamics of a passive tracer advected by two-dimensional point vortex flow, that is, want to drive a passive particle from an initial starting point to a final terminal point, both given a priori, in a given finite time, being The flow is originated by the displacement of N viscous point vórtices. More precisely, we look for the optimal trajectories that minimize the objective function that correspond to the energy expended in the control of the trajectories.

For the same movement of the particle, four different cases will be tested, in which each case a given number of vortexes will be used, which varies from one to four. Subsequently, it is necessary to adopt a strategy in order to find the least amount of energy spent on the movement as well as finding that value as quickly as possible, thus allowing to optimize the movement of the particle.

Keywords: Vortex, Passive Particle, Otimal Control, Numerical Optimization.

Índice

1. Introdução 1

1.1 Considerações gerais 1

1.2 Objetivos 3

1.3 Enquadramento 3

1.4 Estrutura do documento 5

2. Otimização 6

2.1 Considerações gerais 6

2.2 Elementos que constituem o problema de Otimização 8

2.2.1 Variáveis de projeto 8

2.2.2 Função objetivo 8

2.2.3 Restrições 14

2.2.4 Ponto ótimo 15

2.3 Função Fmincon 15

2.4 Algoritmos utilizados no processo de Otimização 19

2.4.1 Algoritmo active-set 19

2.4.2 Algoritmo dos pontos interiores 20

2.4.3 Algoritmo da programação quadrática sequencial 21

3. Descrição do problema de Otimização 23

3.1 Descrição geral do problema 23

3.1.1 Caso 1 vórtice 1 partícula 28

3.1.2 Caso 2 vórtice 1 partícula 29

3.1.3 Caso 3 vórtices 1 partícula 31

3.1.4 Caso 4 vórtices 1 partícula 33

4. Resultados Computacionais 35

4.1 Caso de 1 vórtice e 1 partícula 35

4.2 Caso de 2 vórtices e 1 partícula 38

4.3 Caso de 3 vórtices e 1 partícula 41

4.4 Caso de 4 vórtices e 1 partícula 44

5. Análise de Resultados 47

5.1 Implementação de uma estratégia 48

5.2 Comparação de valores utilizando diferentes tipos de regras de aproximação 50

6. Conclusão 51

Bibliografia 53

Anexo 1 57

Anexo 2 63

Anexo 3 69

Anexo 4 72

Lista de figuras

Figura 1- Aerogerador do tipo "Vortex Bladeless" [3].....1
Figura 2- Fluxograma representativo do processo geral de otimização 7
Figura 3- Regra dos retângulos [23].....11
Figura 4- Regra dos trapézios [24] 12
Figura 5- Regra de Simpson [25] 13
Figura 6- Fluxograma representativo do problema de otimização 26
Figura 7- Implementação das variáveis em Matlab para o caso N=2.....26
Figura 8- Geração de forma aleatória do controlo inicial para cada repetição.....27
Figura 9- Representação da equação da partícula para N=1 28
Figura 10- Representação da equação da partícula para N=2 30
Figura 11- Representação da equação da partícula para N=3 32
Figura 12- Representação da equação da partícula para N=4 34
Figura 13- Evolução do valor de Fval em função do nº de controlos para N=1 35
Figura 14- Trajetória da partícula para menor valor de Fval para N=1.....36
Figura 15- Tempo de computação em função do nº de controlos para N=1 37
Figura 16- Evolução do valor de Fval em função do nº de controlos para N=2 38
Figura 17- Trajetória da partícula para menor valor de Fval para N=2.....39
Figura 18- Tempo de computação em função do nº de controlos para N=2 40
Figura 19- Evolução do valor de Fval em função do nº de controlos para N=3 41
Figura 20- Trajetória da partícula para menor valor de Fval para N=3.....42
Figura 21- Tempo de computação em função do nº de controlos para N=3 43
Figura 22- Evolução do valor de Fval em função do nº de controlos para N=4 44
Figura 23- Trajetória da partícula para menor valor de Fval para N=4.....45
Figura 24- Tempo de computação em função do nº de controlos para N=4 46

Índice de tabelas

Tabela 1- Designação das variáveis de entrada 17
Tabela 2- Designação das variáveis de saída.....17
Tabela 3- Valores possíveis para a variável flag 18
Tabela 4- Comparação do menor valor de Fval referente a cada algoritmo para N=1 ...36
Tabela 5- Comparação do menor valor de Fval referente a cada algoritmo para N=2 ...39
Tabela 6- Comparação do menor valor de Fval referente a cada algoritmo para N=3 ...41
Tabela 7- Comparação do menor valor de Fval referente a cada algoritmo para N=4 ...45
Tabela 8- Comparação do valor de Fval e tempo de computação entre todos os casos testados 48
Tabela 9- Valores obtidos depois de aplicada a estratégia 49
Tabela 10- Comparação entre os valores obtidos por cada uma das regras utilizadas para o caso N=2 50

Siglas

N Nº de vórtices utilizados

T Tempo de deslocamento da partícula

n Nº de subintervalos utilizados

$\boldsymbol{Z_0}$ Ponto de partida

$\boldsymbol{Z_f}$ Ponto final que se pretende atingir

$\boldsymbol{K}$ Circulação de um vórtice

$\boldsymbol{dt}$ Intervalo de tempo

$\boldsymbol{tol}$ Tolerância para se assumir que a partícula atingiu o destino

SQP Programação sequencial quadrática

$\boldsymbol{u_0}$ Controlo inicial

$\boldsymbol{u_r}$ Controlo da função objetivo

1. Introdução

1.1 Considerações gerais

O ser humano, guiado e influenciado pelas limitações naturais, desempenha, quase que instintivamente, todas as funções de um modo que economize energia, sendo que a sua grande motivação é tirar proveito de recursos disponíveis, porém limitados, de maneira a maximizar a produção e diminuir custos. Sendo assim de forma geral, a otimização pode então ser definida como o estudo de um conjunto de técnicas que têm como objetivo a obtenção de um melhor resultado para uma função objetivo e parâmetros pré-especificados dentro de um dado conjunto [1].

A dinâmica dos vórtices e de traçadores passivos em fluxos controlados por vórtices concebem uma vasta área de investigação que continua a atrair cada vez mais a atenção de inúmeros estudos. Os vórtices pontuais são modelos matemáticos empregados para descrever a dinâmica dos fluxos dominados por vórtices, sendo que estes modelos estão baseados numa descrição de pequena dimensão das características dos fluxos [2].

A dinâmica simplificada do vórtice baseada em modelos de vórtice pontual, tem sido empregada em muitas áreas científicas e de engenharia como geofísica, turbulência, superfluidos ou hidrodinâmica. Atualmente existe já um tipo de aerogerador vertical designado por **“**Vortex Bladeless” que é um aerogerador vertical que não contém as tradicionais hélices em movimento, e que transforma a energia mecânica da oscilação provocada pelo efeito da “vorticidade” em energia elétrica [3].

Figura 1- Aerogerador do tipo "Vortex Bladeless" [3]

No entanto em muitos problemas reais encontrar a solução de equações diferenciais parciais requer tempo de computação elevado e um grande armazenamento de memória, por sua vez a solução deste tipo de modelos normalmente é obtida com baixos custos computacionais tornando estes modelos muito atrativos especialmente em problemas de controlo de fluxo [4].

Nos últimos anos, surgiu um interesse especial na utilização da teoria de controlo aplicada à dinâmica de vórtices, sendo que na maioria dos problemas de controlo relativos a fluxos realistas, a solução é obtida por intermédio de modelos simplificados como o vórtice pontual. Em consequência, existe um interesse especial na utilização de algoritmos de controlo aplicados à dinâmica de vórtice, nomeadamente nos campos da dinâmica de fluidos geofísicos, aeronáuticos e hidrodinâmicos [5].

A dinâmica do vórtice pontual é uma área da física matemática que tem servido como um clássico campo de jogos matemáticos. Ao longo dos tempos, muitos algoritmos diferentes da matemática pura e aplicada têm sido utilizados nesta área. Do ponto de vista do controlo ótimo, várias técnicas têm sido também aplicadas [6]. Existem duas grandes classes principais de algoritmos de controlo:

- ✓ Algoritmos diretos;
- ✓ Algoritmo indiretos.

De um modo geral, a abordagem direta consiste em primeiro lugar discretizar o problema e depois em otimizar, enquanto que a abordagem indireta consiste em primeiro lugar otimizar e, depois discretizar. Os algoritmos diretos discretizam o problema relativamente ao tempo, de modo a obter uma Programação Não-Linear (PNL) que pode ser resolvida por um algoritmo de otimização como o algoritmo de Pontos Interiores, entre outros [6].

Neste problema é utilizada uma abordagem direta em que primeiro ocorre a discretização do problema de controlo ótimo e em seguida a otimização através da função Fmincon do Matlab. Os elementos que constituem o problema de otimização, as suas diversas características e classificações serão aqui inicialmente descritos. Posteriormente, a função Fmincon, será introduzida utilizando três algoritmos de otimização diferentes:

- ✓ Pontos interiores;
- ✓ Programação quadrática sequencial;
- ✓ Active-set.

1.2 Objetivos

O objetivo desta investigação é encontrar e testar vários métodos de resolução para um problema de controlo ótimo e determinar qual é o melhor em termos de compromisso entre precisão e eficiência computacional e assim indicar o melhor algoritmo para a sua resolução.

O principal constrangimento é que o problema de otimização que deriva do problema de controlo ótimo não é convexo sendo então mais difícil encontrar um mínimo global para o mesmo. Para isso é testada uma ampla gama de pontos de partida, gerados de forma aleatória, para todos os algoritmos disponíveis na função Fmincon do Matlab pretendendo determinar o mínimo de energia gasta no deslocamento. Consequentemente, devido à existência de muitos mínimos locais é necessário encontrar outros mínimos de forma a convergir para o mínimo global.

1.3 Enquadramento

A otimização pode ser efetuada em diferentes níveis dentro de uma empresa, desde uma combinação de unidades, equipamentos, peças de equipamentos, até entidades menores. Portanto, a base de um problema de otimização pode englobar uma empresa, um processo, um equipamento, uma peça de um equipamento, entre outros [7].

Esta dissertação baseia-se num estudo que permite analisar o movimento num plano ilimitado de uma partícula induzido por vórtices. Um vórtice é um escoamento giratório. Num fluxo induzido por um conjunto de pontos de vórtices as linhas de corrente apresentam um padrão circular ou espiral, tratam-se de movimentos espirais ao redor de um centro de rotação que surgem devido à diferença de pressão entre duas regiões vizinhas [8].

Quando existe essa diferença de pressão o fluído tende a equilibrar o sistema e flui para esta região mudando, eventualmente, a direção original do escoamento e, com isso, gera vorticidade, que é designada como um conceito matemático utilizado na dinâmica dos fluídos. Ela pode ser entendida como a quantidade de circulação ou rotação de um fluído por unidade de área de um ponto no campo de escoamento. Por sua vez os vórtices são encontrados nos mais diversos locais da natureza, como correntes circulares de água vindas de marés conflitantes, furacões e tornados [9].

Esta dissertação diz respeito à dinâmica de uma partícula passiva induzida por um fluxo de vórtice bidimensional, sendo que uma partícula passiva é suficientemente pequena para não perturbar o campo de velocidade, mas também suficientemente grande para não realizar um movimento Browniano. Pelo que as partículas deste tipo são os traçadores utilizados para a visualização do fluxo em experiências de mecânica dos fluidos [6][10].

Neste caso de estudo pretende-se encontrar o mínimo valor de energia gasto no deslocamento de uma partícula de um ponto inicial até um ponto final, em tempo fixo previamente definido, pelo que este problema é analisado a duas dimensões e desenvolve-se no plano complexo, sendo as variáveis de controlo números complexos. De modo a obter uma solução viável e sustentada serão utilizados vários casos em que a diferença de caso para caso será o nº de vórtices utilizados para descrever o deslocamento da partícula, tendo sido testados os seguintes casos:

- ✓ Caso 1 vórtice e 1 partícula;
- ✓ Caso 2 vórtices e 1 partícula;
- ✓ Caso 3 vórtices e 1 partícula;
- ✓ Caso 4 vórtices e 1 partícula;

Para cada um dos casos referidos anteriormente serão também testados diferentes nº de controlos permitindo que seja possível chegar a um ponto ótimo ideal, em que o gasto de energia no deslocamento da partícula seja o mínimo possível.

1.4 Estrutura do documento

Este projeto divide-se em 6 capítulos. O Capítulo 1 descreve o enquadramento do trabalho assim como a sua motivação e objetivos a atingir no decorrer do mesmo. De seguida, no Capítulo 2, foi realizada uma revisão da literatura existente, relativa aos conceitos de Otimização assim como os elementos que constituem todo esse processo. No Capítulo 3 é feita uma descrição do estudo do deslocamento de uma partícula que se move num fluxo bidimensional cuja dinâmica é dada por N vórtices. Por sua vez, o Capítulo 4 descreve os resultados obtidos para o caso de estudo considerado neste trabalho, e em particular o valor mínimo de energia consumido nesse movimento. No Capítulo 5 é realizada uma pequena análise aos resultados obtidos. Por fim, o Capítulo 6 apresenta as conclusões do trabalho. Em anexo encontram-se todos as tabelas com os valores obtidos em Matlab para cada um dos casos.

2. Otimização

2.1 Considerações gerais

A otimização refere-se ao estudo de problemas em que se pretende minimizar ou maximizar uma função através da escolha sistemática dos valores de variáveis reais ou inteiras dentro de um conjunto viável. A otimização é empregada principalmente no ramo da engenharia, no entanto tem um papel preponderante na conceção de qualquer projeto podendo também ser aplicada em resolução de problemas de varias áreas, assumindo cada vez mais importância em resoluções de problemas que envolvam poupança e utilização inteligente dos recursos e matérias-primas, maximização de lucro, poupança de capital e minimização de gastos de energia [11][12].

Qualquer processo de otimização envolve uma análise rigorosa e uma compreensão do problema, sendo posteriormente necessário perceber quais são as restrições impostas, dependendo do tipo de problema. Estas restrições são impostas pelo problema e relacionam-se com os limites que cada variável pode assumir bem como os valores que as variáveis podem tomar ao longo do problema, estas restrições podem ser de dois tipos [12]:

- ✓ Restrições lineares;
- ✓ Restrições não lineares.

Contudo, num problema de otimização deve-se ter em conta os seguintes passos [13]:

- ✓ Análise do processo e as suas variáveis;
- ✓ Determinação do critério para otimização e especificação da função objetivo em termos das variáveis de processo;
- ✓ Desenvolvimento do modelo para o processo, relacionando as suas variáveis através de restrições de igualdade e desigualdade;
- ✓ Se a formulação do problema é de dimensão elevada, então procurar dividir o problema em partes menores ou simplificar a função objetivo;
- ✓ Aplicação de técnicas apropriadas de otimização;
- ✓ Analisar a solução obtida e a sua sensibilidade relativamente a variações de parâmetros.

A otimização aplica-se tanto ao desenvolvimento de um produto, seja ele um simples componente mecânico ou uma complexa peça de engenharia moderna, como a processos e organizações. O seu leque de aplicação é muito alargado e os algoritmos utilizados são constantemente desenvolvidos e melhorados, sendo que das diferentes técnicas de otimização destaca-se a programação não linear e a programação linear [12][14].

A programação linear consiste na otimização de problemas (maximizar ou minimizar uma função) cujas variáveis de decisão e as restrições são todas lineares, ou seja, equações ou inequações de primeiro grau [14].

Na programação não linear é possível modelar as diversas grandezas, no entanto o seu grande problema é o facto de envolverem cálculos muito complexos e com um tempo computacional elevado, o que torna o algoritmo lento e obriga a uma grande disponibilidade ao nível da memória do computador [15].

Sendo o processo de otimização um processo iterativo, são analisadas várias configurações do problema em estudo de um modo sucessivo, até que se encontre uma configuração final que obedeça a todas as restrições impostas, assim encontrada essa configuração será considerada a ideal [16].

No entanto encontrar esta configuração ideal nem sempre é um processo fácil pois muitas das vezes os resultados obtidos não são satisfatórios, quando estes valores não são satisfatórios terá de ser analisado novamente o sistema. O seguinte fluxograma ilustra, de um modo simples, um processo de otimização:

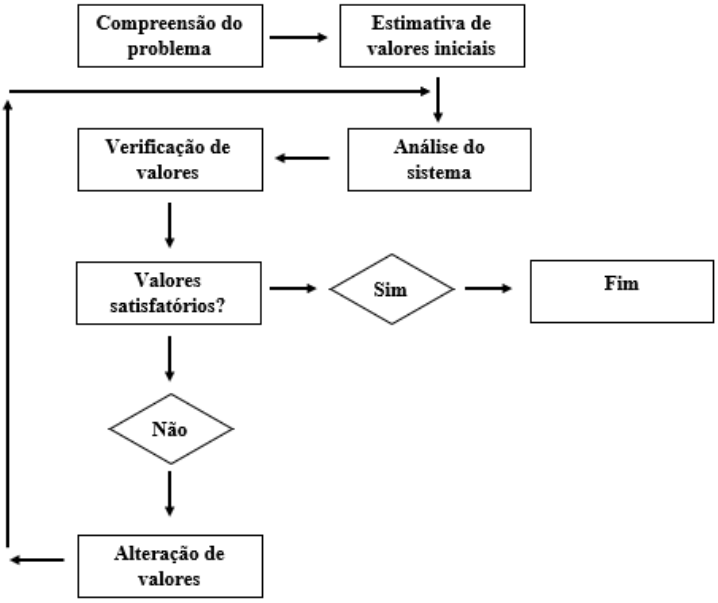

Figura 2- Fluxograma representativo do processo geral de otimização

2.2 Elementos que constituem o problema de Otimização

2.2.1 Variáveis de projeto

A ideia de aperfeiçoar ou otimizar uma estrutura implica alguma liberdade para modificá-la dentro das variações autorizadas num grupo de parâmetros permitindo assim alcançar um melhor desempenho da mesma. Estes parâmetros são geralmente designados de variáveis de projeto e correspondem aos parâmetros variáveis no processo de otimização que definem as características do modelo analisado, ou seja, são estas as variáveis que se pretende determinar dentro de um intervalo pré-definido e para as quais o problema é resolvido [17].

De forma geral, as variáveis podem adotar qualquer valor, mas isso poderia produzir soluções não viáveis no processo de otimização, o que implica que seja necessário introduzir as variáveis do projeto num intervalo cujo limite inferior e cujo limite superior delimitem a variação das mesmas [18].

Por fim é importante definir o tipo de variável, seja ela discreta ou contínua, podendo existir problemas onde possam ser utilizados ambos os tipos de variáveis. Uma variável contínua é aquela que pode assumir qualquer valor dentro de um intervalo de variação, enquanto que uma variável discreta é aquela que pode assumir valores específicos [18].

2.2.2 Função objetivo

A função objetivo é a representação matemática do critério de eficiência adotado num problema de otimização, essa função representa lucro, custo, energia, produção, distância, entre outros, a qual é influenciada pelas variáveis de projeto, também conhecidas como restrições ou variáveis de controlo do problema, podendo esse mesmo problema ser representado por uma única função objetivo ou por várias funções objetivo [19].

Em problemas de engenharia, caso se pretenda minimizar o gasto de energia num determinado deslocamento de uma certa partícula o problema pode ser definido apenas com uma função objetivo [19].

Da mesma forma, quando o critério de eficiência é influenciado por uma única variável, a função objetivo é dita unidimensional, no caso em que várias variáveis tenham influência sobre o critério a função objetivo é denominada multidimensional. Além disso, se a função tem um único mínimo (ou máximo), ela é chamada de função unimodal caso tenha mais que um máximo ou mínimo ela é denominada multimodal [20].

Em muitos problemas de otimização é complicado obter a solução usando técnicas de otimização, isto porque muitos dos problemas apresentam uma função objetivo com algumas complicações, destacando-se as seguintes complicações [13] [20]:

- ✓ Função objetivo e/ou restrições podem apresentar descontinuidades;
- ✓ Função objetivo e/ou restrições não lineares;
- ✓ Função objetivo pode apresentar mínimos locais.

O melhor resultado da função objetivo avaliada num determinado ponto é chamado de solução, podendo essa solução ser classificada como:

- ✓ Mínimo local – melhor solução encontrada numa região específica do espaço de projeto;
- ✓ Mínimo global – melhor solução encontrada em todo o espaço de projeto investigado.

No entanto nem todas as técnicas de otimização garantem que a solução encontrada seja ótima global, pois a grande maioria apenas converge para uma solução local próxima do ponto inicial [21].

A função seguidamente transcrita representa a função objetivo relativa ao gasto de energia no deslocamento de uma partícula de um ponto inicial até um ponto final do plano complexo, pretendendo-se que esse valor gasto seja o mínimo possível.

$$\int_{0}^{T} |\boldsymbol{u}(\boldsymbol{t})|^{2}\, \boldsymbol{dt}$$

T- Tempo de deslocamento permitido;
u - Função de controlo (complexa);
t - Variável tempo.

O valor mínimo de energia gasto no deslocamento da partícula é obtido através de uma integração numérica, a necessidade de usar a integração em muitos problemas deve-se à forma de como é resolvido o problema de controlo ótimo, sendo que na abordagem direta primeiro discretiza-se a função no tempo, ou seja, $u(t)$ é substituída pelas variáveis constantes u_1 , u_2 ,…, u_n em cada um dos n subintervalos em que o intervalo de tempo T está dividido. O valor resultante desta discretização é um valor aproximado do integral que pode ser calculado por intermédio de três diferentes regras, tais como:

- ✓ Regra dos retângulos;
- ✓ Regra dos trapézios;
- ✓ Regra de Simpson.

De uma forma geral todos os algoritmos anteriormente referidos, adotam uma determinada sequência de forma a encontrar o valor final, sendo que inicialmente existe uma decomposição do domínio da função em subintervalos, depois é feita uma integração aproximada da função para cada subintervalo e por último ocorre a soma dos resultados numéricos obtidos chegando assim ao valor final pretendido.

Quando se divide o intervalo de integração e se aplica qualquer uma das regras a cada subintervalo, obtém-se uma aproximação. Essa aproximação depende diretamente do tamanho de cada um dos subintervalos, sendo que quanto menor for esse subintervalo, melhor será o resultado. Em contrapartida, quanto maior for a divisão do intervalo de integração, maior serão os cálculos envolvidos**.**

Apesar da existência de várias regras de aproximação para o cálculo integral, esta dissertação baseia-se na aproximação da função objetivo através da regra dos retângulos para cada um dos casos testados, no entanto para um caso específico haverá uma comparação entre os valores obtidos utilizando cada uma das regras acima referidas e uma consequente verificação de qual das regras permite atingir um valor mínimo de energia gasto no deslocamento da partícula.

- **Regra dos retângulos**

Esta regra baseia-se em subdividir o intervalo de integração em "n" subintervalos iguais que servirão para as bases dos retângulos a serem construídos, a soma das áreas destes retângulos será uma aproximação do integral definido da função f(x) para o intervalo de integração definido. Os retãngulos são criados de modo a que a função f(x) passe no ponto médio de cada um dos retângulos formados como ilustrado a Figura 3 [22] [23].

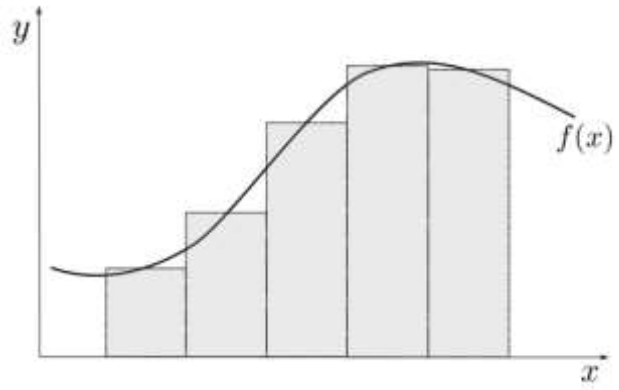

Figura 3- Regra dos retângulos [23]

Considerando um intervalo finito $[x_1; x_{n+1}]$ e em que dx corresponde ao valor da base de cada retângulo, sendo que este valor depende diretamente do nº de subintervalos utilizados no problema. Cada subintervalo tem o mesmo valor de dx e o mesmo é obtido por [22] [23] :

$$dx = \frac{x_{n+1} - x_1}{n} \tag{1}$$

Por sua vez o valor da função f(x) é obtido somando a área de cada um dos retângulos correspondentes a cada subintervalo. Sendo assim o valor do integral da função f(x) obtido pela regra dos retângulos é dado por [22] [23]:

$$A = \sum_{x_1}^{x_{n+1}} f(\overline{x})dx \tag{2}$$

Em que $\overline{x}$ correponde ao valor médio de x para cada subintervalo, e $f(\overline{x})$ correponde ao valor de f(x) para o respetivo valor $\overline{x}$ [22] [23].

- **Regra dos trapézios**

A ideia da regra do trapézio é aproximar o integral definido da função f(x) entre dois pontos por um polinômio de ordem 1, pelo que para cada subintervalo o integral da função f(x) pode ser aproximada pela área de um trapézio, como ilustrado na Figura 4 [22] [24].

Figura 4- Regra dos trapézios [24]

Considerando um intervalo finito $[x_1; x_{n+1}]$ e em que dx corresponde à altura de cada trapézio, sendo que esta altura é a mesma para cada intervalo e é dada por [22] [24]:

$$dx = \frac{x_{n+1} - x_1}{n} \tag{3}$$

Depois de determinado o valor de dx é necessário determinar o integral definido da função f(x), o qual é obtido somando a área de cada um dos trapézios correspondentes a cada subintervalo. Sendo assim o valor aproximado do integral da função f(x) obtido pela regra dos trapézios é dado por [22] [24]:

$$A = \frac{dx}{2}[f(x_1) + 2f(x_2) + \cdots + 2f(x_n) + f(x_{n+1})] \tag{4}$$

- **Regra de Simpson**

Tal como a regra dos trapézios, trata-se de um exemplo da Fórmula de Newton-Cotes fechada, mas, ao invés de se considerar a aproximação em cada subintervalo através de um polinómio interpolador do 1º grau, pode-se pensar numa aproximação um pouco melhor, considerando um polinómio interpolador do 2º grau. Para isso, ao se considerar a regra de integração simples, precisa-se de um ponto adicional, que será o ponto médio, como ilustrado na Figura 5 [22] [25].

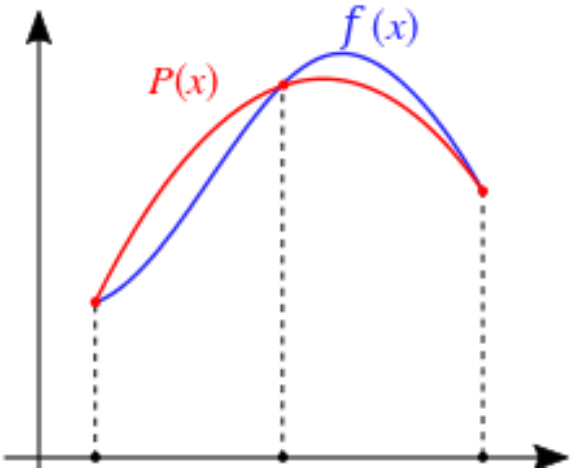

Figura 5- Regra de Simpson [25]

Considerando um intervalo finito $[x_1; x_{2n+1}]$ e em que dx corresponde à diferença entre os extremos de cada subintervalo, sendo que esta altura é a mesma para cada intervalo e é dada por [22] [25]:

$$dx = \frac{x_{2n+1} - x_1}{2n} \tag{5}$$

O valor aproximado do intregal da função f(x) para um dado intervalo de integração,é resultante do somatório da área correspondente a cada subintervalo e pode ser obtido por [22] [25]:

$$A = \frac{dx}{3}[f(x_1) + 4f(x_2) + 2f(x_3) + \cdots + 4f(x_{2n}) + f(x_{2n+1})] \tag{6}$$

2.2.3 Restrições

Em muitos problemas dos mais variados campos da engenharia, especialmente em problemas de otimização, algumas condições são impostas de modo a limitar a escolha de valores, sendo designadas de restrições que dependendo do ponto de vista podem ser de dois tipos:

- ✓ Restrições geométricas ou de comportamento (do ponto de vista físico);
- ✓ Restrições de igualdade ou desigualdade (do ponto de vista matemático).

As restrições geométricas, também chamadas restrições de limite, são as limitações impostas diretamente às variáveis de projeto, já as restrições de comportamento são aquelas que limitam indiretamente as variáveis de projeto. As restrições geométricas são determinadas através de valores que impõem limites inferiores e/ou superiores e são funções de desigualdade por natureza [20] [26].

Por sua vez as restrições de comportamento são determinadas através de especificações de funções que dependem das variáveis de projeto, impondo a limitação das mesmas a um semi-espaço, através de funções de desigualdade ou em uma superfície, através de funções de igualdade. Geralmente, as restrições estão relacionadas com a geometria, os esforços admissíveis, os recursos disponíveis, os custos envolvidos, os gastos de energia, entre outros [27].

Por fim é importante frisar que o número de funções de restrições de igualdade deve ser menor ou igual que o número de variáveis, caso isso não aconteça, obtém-se um sistema de equações sobredeterminado, onde há uma formulação inconsistente ou alguma restrição redundante, ou seja, linearmente dependente de outra. No caso das restrições de desigualdade, não há limitação imposta ao número de funções [27].

2.2.4 Ponto ótimo

O ponto ótimo é o vetor que representa as variáveis de projeto que minimiza ou maximiza uma função objetivo e satisfaz as restrições respetivas ao problema. O valor da função objetivo correspondente a essas variáveis é designado por valor ótimo, sendo que o par formado pelo ponto ótimo e o valor ótimo é chamado de solução ótima, podendo ser local, se for um valor mínimo ou máximo em relação a uma vizinhança desse ponto, ou global, se for um valor mínimo ou máximo em relação a todo o espaço do problema [18] [27].

2.3 Função Fmincon

A função fmincon é uma das funções disponibilizados pela Optimization Toolbox do software Matlab e permite localizar o mínimo de função multivariável não linear restrita. Geralmente, a função fmincon, utiliza os seguintes algoritmos [11][28]:

✓ Active Set;

✓ Pontos Interiores;

✓ Programação quadrática sequencial (SQP).

Este último algoritmo é baseado nos princípios SQP, ou seja, permite transformar um problema em subproblemas mais simples que pode ser estudado e analisado através de um processo iterativo para isso têm de ser respeitado um conjunto de condições [11][28]:

$$c(x) \leq 0 \quad (8)$$

$$ceq(x) = 0 \quad (9)$$

$$A \cdot x \leq b \quad (10)$$

$$Aeq \cdot x = beq \quad (11)$$

$$lb \leq x \leq ub \quad (12)$$

As condições 8 e 9 representam as restrições não lineares de desigualdade e igualdade, respetivamente. Sendo que c e ceq são escalares ou vetores que representam várias restrições, estas restrições permitem restringir a solução a qualquer região que possa ser descrita em termos de funções suaves [28].

A condição 10 representa as restrições lineares de desigualdade, sendo que A é uma matriz k-por-n, onde k é o número de desigualdades e n é o número de variáveis. Por sua vez b é um vetor de comprimento k [28].

A condição 11 representa as restrições lineares de igualdade, em que Aeq é uma matriz m por n, onde m é o número de igualdades e n é o número de variáveis já beq representa um vetor de comprimento m [28].

Por fim a condição 12 corresponde aos limites simples e significa que cada elemento no vetor x deve ser maior que o elemento correspondente de lb e deve ser menor que o elemento correspondente de ub [28].

Na maior parte de problemas relacionados com otimização não implica que estejam presentes num só problema todas as condições acima referidas, no entanto as que existirem terão de ser respeitadas obrigatoriamente.

Quando a função fmincon é implementada em Matlab é necessário primeiramente definir um conjunto específico de variáveis de entrada permitindo assim obter uma solução adequada para um respetivo problema, quando são alteradas essas variáveis ocorre consequentemente a alteração da solução.

- ✓ **Variáveis de entrada**

As variáveis de entrada correspondem a todas as variáveis que necessitam de ser introduzidas para que a função fmincon consiga obter valores finais, ou seja, a solução do problema. Estas variáveis variam de problema para problema e em muitos dos casos existe variáveis que não estão especificadas, no entanto em Matlab é sempre necessário que as mesmas estejam definidas, estas variáveis estão representadas na Tabela 1.

Tabela 1- Designação das variáveis de entrada

Nome da variável	Designação
fun	Função objetivo.
x0	Ponto inicial.
A	Matriz dos coeficientes das restrições lineares de desigualdade.
b	Vetor independente das restrições lineares de desigualdade.
Aeq	Matriz dos coeficientes das restrições lineares de igualdade.
beq	Vetor independente das restrições lineares de igualdade.
lb, ub	*lb, ub* - Vetores dos limites inferiores e superiores de x.
nonlcon	Restrições não lineares (igualdade e desigualdade).
options	Opções criadas com optimset.

- ✓ **Variáveis de saída**

As variáveis de saída variam de acordo com as variáveis de entrada introduzidas e são aquelas que são obtidas depois da execução da função fmincon, as quais correspondem à solução do problema estando representadas na Tabela 2.

Tabela 2- Designação das variáveis de saída

Nome da variável	Designação
x	Vetor solução.
fval	Valor da função objetivo.
flag	Razão pela qual a função fmincon parou.

Depois de alcançada a solução do problema é necessário analisar a respetiva solução porque muitas destas soluções não podem ser tomadas em conta devido à razão pela qual o algoritmo parou, pois existe uma variedade de razões que implicam a paragem do algoritmo.

A variável de saída que indica a razão pela qual o algoritmo interrompe a sua procura é a variável *flag*, esta é uma variável que só pode tomar valores inteiros entre -3 e 5, sendo que a cada valor inteiro dentro deste intervalo corresponde uma razão diferente pela qual o algoritmo parou. Os valores que a variável *flag* pode assumir, assim como a razão associada à interrupção estão transcritos na Tabela 3.

Tabela 3- Valores possíveis para a variável flag

Valor da variável	Designação
-3	A função objetivo na iteração atual foi inferior as opções.
-2	Nenhum ponto viável encontrado.
-1	O algoritmo terminou com problemas na função objetivo.
0	O número de iterações foi excedido.
1	O critério de paragem, relativamente á informação de 1ªordem foi verificado, e a violação máxima dos constrangimentos foi menor que o definido.
2	A alteração nas variáveis de projeto e a violação máxima dos constrangimentos são menores que valores pré-definidos.
3	A mudança no valor da função objetivo foi inferior as opções.
4	A magnitude da direção de busca e a violação máxima dos constrangimentos são menores que valores pré-definidos.
5	A magnitude da derivada direcional na direção de busca e a violação máxima dos constrangimentos são menores que valores pré-definidos.

2.4 Algoritmos utilizados no processo de Otimização

No processo de otimização não existe um único algoritmo que pode ser aplicado eficientemente para todos os problemas, pelo que o algoritmo escolhido para um caso particular depende de certas particularidades do problema a resolver, da natureza das restrições e do número de variáveis do problema.

2.4.1 Algoritmo active-set

Este algoritmo é um algoritmo de média escala pois cria internamente matrizes completas e usa álgebra linear densa no caso em que seja necessário resolver um problema suficientemente grande, as matrizes completas ocupam uma elevada quantidade de memória e a álgebra linear densa pode exigir muito tempo para ser executada [12] [29].

Na otimização matemática, um problema é definido usando uma função objetiva para minimizar ou maximizar e um conjunto de restrições que definem a região praticável, isto é, o conjunto de todos os x para procurar a solução ótima. Dado um ponto x na região viável, uma restrição é chamada ativa em x se g (x) = 0 e inativa em x se g (x)> 0, no caso das restrições de igualdade, as mesmas estão sempre ativas [12] [29].

O active set em x é feito dessas restrições g (x) que estão ativas no ponto atual, este algoritmo é particularmente importante na teoria de otimização, pois determina quais as restrições que influenciarão o resultado final da otimização. Por exemplo, ao resolver o problema de programação linear, o active set fornece os hiperplanos que se cruzam no ponto de solução [12] [29].

Na programação quadrática, como a solução não está necessariamente em uma das arestas do polígono delimitador, uma estimativa do active set fornece um subconjunto de desigualdades a serem observadas durante a pesquisa da solução, o que reduz a complexidade da pesquisa [30].

Por fim, o active-set é usado para otimização com restrições sendo que difere do algoritmo de pontos interiores, pois nenhuma demarcação de barreira é utilizada para garantir que o algoritmo permaneça no interior com relação às restrições de desigualdade.

O algoritmo baseia-se [30]:

- ✓ Numa etapa de projeção de inclinação não-monótona;
- ✓ Numa etapa de otimização irrestrita;
- ✓ Um conjunto de regras para ramificação entre as duas etapas;

O algoritmo active-set explora o algoritmo cíclico para a etapa de projeção de gradiente e por sua vez, o algoritmo de gradiente conjugado, para otimização sem restrições. Um algoritmo com estas melhorias é executado muito mais rápido que o algoritmo básico para problemas definidos positivos e encontra mínimos locais com valores de função mais baixos para problemas indefinidos [30].

2.4.2 Algoritmo dos pontos interiores

O algoritmo de pontos interiores foi desenvolvido para tratar de problemas de dimensões elevadas, ou seja, com um número elevado de vértices. Este algoritmo determina as direções de busca no interior estrito da região viável e baseia-se na aplicação do algoritmo de Newton para a solução de um sistema de equações não lineares obtidas a partir da aplicação das condições de Kuhn-Tucker no problema de otimização. O algoritmo de pontos interiores apresenta duas características bastante importantes que favorecem a sua utilização [13] [20] [31]:

- ✓ Este algoritmo gera pontos ou soluções intermediárias no interior da região viável;
- ✓ Caso por alguma razão, o algoritmo for interrompido antes de alcançar o ponto de ótimo, tem-se um ponto viável e com um valor da função objetivo inferior ao anterior.

Esta característica referida anteriormente resulta devido ao facto de cada ponto intermediário gerar valores decrescentes para a função objetivo, permitindo assim obter um ponto viável para a função objetivo [20].

O algoritmo de pontos interiores é utilizado para problemas de otimização com restrições, ele é baseado na resolução aproximada de uma sequência de minimizações que procuram encontrar a solução de problemas lineares com um número mínimo de iterações, sendo que estas iterações têm elevada complexidade e requerem um esforço computacional maior [20] [31].

Este algoritmo procura encontrar uma solução ótima de um problema de programação linear caminhando pelo interior do cortante positivo.

De acordo com o espaço em que ocorrem as iterações os algoritmos de pontos interiores podem ser divididos em duas categorias [31]:

- ✓ Algoritmos afim escala;
- ✓ Algoritmos de trajetória central.

Este tipo de algoritmo são rápidos e usam pouca memoria, é um algoritmo de larga escala pois utiliza álgebra linear que não precisa de armazenar nem operar matrizes completas [31].

2.4.3 Algoritmo da programação quadrática sequencial

A solução de problemas com rapidez e precisão depende de vários fatores que devem ser levados em consideração: o tamanho do problema, o número de variáveis e o número de restrições pois quanto mais variáveis, mais restrições e mais complexo a relação entre elas, mais tempo de processamento é necessário e menor tende a ser a precisão. Outro fator fundamental consiste na relação entre as variáveis, se é linear ou não, pelo que problemas não lineares são mais complexos de serem resolvidos pois a solução de problemas não lineares normalmente envolve processos iterativos para estabelecer uma direção de procura [32] [33].

A programação quadrática sequencial, ou ainda algoritmos da métrica variável com restrições, são as várias formas de referenciar o algoritmo SQP, este algoritmo baseia-se a resolver as equações de Karush-Kuhn-Tucker (KKT), ou condições necessárias de primeira ordem, sendo que o objetivo do algoritmo é chegar o mais próximo possível do algoritmo de Newton utilizado na solução de problemas sem restrições, que apresenta convergência quadrática [32] [33].

Num algoritmo SQP, uma sequência de subproblemas quadráticos é resolvida onde a função objetivo deste subproblemas é tal que o coeficiente do termo linear é formado pelo gradiente da função objetivo do problema principal enquanto para o termo quadrático, é usada uma aproximação da Hessiana da função Lagrangiana do problema principal [33].

Se um algoritmo de otimização estiver bem implementado, é provável que a solução do problema com restrições seja obtida num número menor de iterações que o respetivo problema sem restrições. Isto geralmente ocorre porque as restrições podem gerar informações adicionais para a determinação da melhor direção de busca e do tamanho do passo mais apropriado [34].

De uma forma resumida pode-se dizer que o algoritmo de programação quadrática sequencial (SQP) é um algoritmo iterativo para otimização de problemas com restrições onde a função objetivo e as restrições são diferenciáveis. Também os algoritmos SQP resolvem uma sequência de subproblemas de otimização, sendo que, cada um dos quais otimiza um modelo quadrático do objetivo sujeito a uma linearização das restrições. Quando o problema não é restrito, o algoritmo reduz-se ao algoritmo de Newton de modo a encontrar um ponto em que o gradiente do objetivo desaparece. Se o problema tiver apenas restrições de igualdade, o algoritmo é equivalente a aplicar o algoritmo de Newton às condições de otimização de primeira ordem, ou condições de Karush-Kuhn-Tucker, do problema [34].

3. Descrição do problema de Otimização

3.1 Descrição geral do problema

Esta dissertação diz respeito à dinâmica de uma partícula passiva inserida num fluxo induzido por vórtices bidimensionais, pelo que uma partícula passiva é suficientemente pequena para não perturbar o campo de velocidade, mas também suficientemente grande para não realizar um movimento Browniano, sendo utilizadas como traçadores para a visualização do fluxo em experiências de mecânica dos fluidos, considerando-se também que a partícula passiva tem a mesma densidade do fluido em que está embutida. Pretende-se conduzir uma partícula passiva de um ponto de partida inicial a um ponto final, situados no plano complexo e dados *a priori*, num determinado tempo finito. O fluxo tem origem no deslocamento de um certo número, digamos N, de vórtices, tendo este problema algumas semelhanças com o problema da locomoção dos peixes sendo que aqui a dinâmica do vórtice é obtida por N vórtices pontuais e pelo controlo [4].

A deslocação da partícula passiva é então transformada num problema de controlo que se resolve através de uma abordagem direta, em que o tempo disponível para o deslocamento é dividido num número fixo n de subintervalos, onde as variáveis de controlo são constantes. Esta deslocação da partícula dá-se no plano complexo, ou seja, as variáveis com ponto de partida e ponto final são números complexos o que implicará que os controlos associados ao movimento da partícula sejam também números complexos.

Como referido em capítulos anteriores uma abordagem direta consiste em primeiro discretizar o problema e em seguida otimizar. Sendo assim uma partícula passiva é, por definição, um vórtice pontual com uma circulação zero, ou seja, $k=0$, em que a dinâmica de um sistema composto por P partículas passivas e N vórtices pontuais é da pelas equações dos vórtices:

$$\frac{dz_\alpha^*}{dt} = \frac{1}{2\pi i} \sum_{\substack{\beta=1 \\ \beta \neq \alpha}}^{N} \frac{k_\beta}{z_\alpha - z_\beta} \ (\alpha = 1,2, \dots, N) \tag{13}$$

Em que z_α é a posição relativa a um determinado vórtice sendo representada na forma $z_\alpha = x_\alpha + y_\alpha i$ enquanto que z_α^* é o seu complexo conjugado. Pelo que as equações para o deslocamento de cada partícula passiva, são:

$$\frac{dz_\alpha^*}{dt} = \frac{1}{2\pi i}\sum_{\beta=1}^{N}\frac{k_\beta}{z_\alpha - z_\beta} \ (\alpha = N+1, N+2, \dots, N+P) \qquad (14)$$

A modelação do sistema completada pelas condições iniciais, correspondentes as posições dos N vórtices e das partículas em $t = 0$. Para todos os casos testados é necessário proceder-se à discretização do controlo da função objetivo, onde a variável $u(t)$ é substituída por *n* variáveis discretas definidas como $u_1, u_2, \cdots, u_n$. A função objetivo é a seguinte:

$$\int_0^T |\boldsymbol{u}(\boldsymbol{t})|^2 \, \boldsymbol{dt}$$

Em que a discretização da variável $u(t)$ ficará da seguinte forma:

$$u(t) = u_0 \text{ quando } t_0 \leq t < t_1$$

$$u(t) = u_1 \text{ quando } t_1 \leq t < t_2$$

$$u(t) = u_2 \text{ quando } t_0 \leq t < t_3$$

$$u(t) = u_{n-1} \text{ quando } t_{n-1} \leq t < t_n$$

Sendo assim cada variável u_i $(i = 0,1,\cdots, n-1)$ corresponde um valor do controlo constante exercido no intervalo $[t_{i-1}, t_i]$. Cada um destes intervalos possui um comprimento constante $\Delta t = \frac{t_n - t_0}{n}$.

Por fim ocorre a discretização da função objetivo de acordo com as várias regras de aproximação, obtendo-se as seguintes funções discretizadas:

- ✓ Pela regra dos retângulos:

$$\int_0^T |u(t)|^2 \, dt \approx \Delta t(|u_0|^2 + |u_1|^2 + |u_2|^2 + \dots + |u_{n-1}|^2)$$

- ✓ Pela regra dos trapézios:

$$\int_0^T |u(t)|^2\, dt \approx \frac{\Delta t}{2}(|u_0|^2 + 2|u_1|^2 + 2|u_2|^2 + \dots + |u_{n-1}|^2)$$

- ✓ Pela regra de Simpson:

$$\int_0^T |u(t)|^2\, dt \approx \frac{\Delta t}{3}(|u_0|^2 + 4|u_1|^2 + 2|u_2|^2 + \dots + 4|u_{n-2}|^2 + |u_{n-1}|^2)$$

Sendo assim para cada um dos casos testados ocorre a discretização do problema, pelo que para o problema de controlo ótimo para um único vórtice, a sua versão discretizada ficará do seguinte modo:

$$\text{Minimizar } \Delta t \sum_{k=0}^{n-1} |u_k|^2$$

Sujeito a:

$\dot{z}^* = \frac{1}{2\pi i} \times \frac{k}{z} + u_0$, $z(0) = z_0$, $|u_0| \leq u_{max}$ quando $t_0 < t < t_1$

$\dot{z}^* = \frac{1}{2\pi i} \times \frac{k}{z} + u_1$, $z(t_1) = z_{t_1}$ e $|u_1| \leq u_{max}$ quando $t_1 < t < t_2$

$\dot{z}^* = \frac{1}{2\pi i} \times \frac{k}{z} + u_{n-1}$, $z(t_{n-1}) = z_{t_{n-1}}$ ⋮ e $|u_{n-1}| \leq u_{max}$ quando $t_{n-1} < t < t_n$

$z(t_n) = z_f$

No entanto para cada um dos outros casos testados o procedimento a adotar é o mesmo, sendo que de caso para casa apenas varia a equação da partícula, isto porque se variam o nº de vórtices de caso para caso implica que a equação da partícula seja diferente de caso para caso, de resto o procedimento a adotar é o mesmo.

Depois de discretizado o problema, é necessário proceder à otimização do mesmo, para isso considera-se que a partícula passiva move se num fluxo bidimensional cuja dinâmica é dada, em qualquer intervalo de tempo, por N vórtices.
Serão testados quatro problemas diferentes, cada um correspondendo a um valor diferente de N vórtices, variando de um a quatro. Para cada um dos problemas testados o movimento da partícula acarretará um gasto de energia durante esse deslocamento, o qual se pretende minimizar, de modo a encontrar esse valor é necessário ter em conta várias etapas, as quais estão descritas na Figura 6.

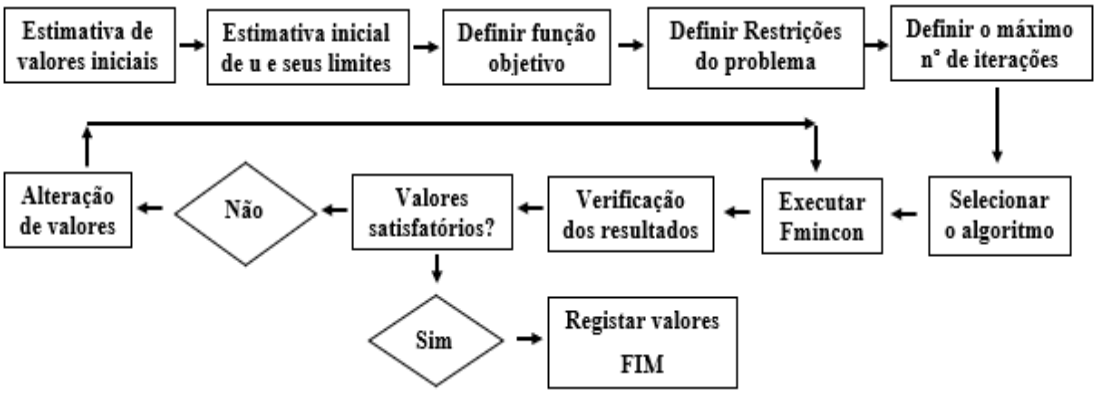

Figura 6- Fluxograma representativo do problema de otimização

A Figura 6 representa o conjunto de etapas necessárias para que seja possível alcançar uma solução final, pelo que primeiramente além de se definir as posições originais de cada um dos vórtices é também é necessário para cada caso definir variáveis iniciais do problema tais como:

- ✓ T, tempo de deslocamento da partícula;
- ✓ n, nº de subintervalos utilizados;
- ✓ Z_0, ponto de partida;
- ✓ Z_f, ponto final que se pretende atingir;
- ✓ K, respetiva circulação de cada vórtice;
- ✓ dt, definido diretamente a partir de n ;
- ✓ tol, tolerância para se assumir que a partícula atingiu o destino.

As variáveis iniciais para o problema em que são utilizados dois vórtices estão representadas na Figura 7.

```
%%  Variáveis do problema
T=10;                           % Tempo de deslocamento
n=1;                            % numero de subintervalos de tempo
zf=2+2*i;                       % ponto final a atingir
z0=-1-i;                        % ponto de partida (tem de ser dif. de zc)
k1=10; k2=10;                   % circulation
z10=0.5+0.5*i;                  % pos. original do vortice 1
z20=1.5-0.5*i;                  % pos. original do vortice 2
d=abs(z20-z10);                 % dist. entre vortices
dt=T/n;                         % passo de tempo do controlo
tol=1e-4;                       % |zf-z(T)|<tol
```

Figura 7- Implementação das variáveis em Matlab para o caso N=2

Estando todas as variáveis definidas é necessário definir o controlo inicias $u_0, u_1, \cdots, u_n$ de forma aleatória, no entanto serão testados um conjunto de diferentes controlos iniciais.

Posteriormente é também necessário definir a função objetivo assim como todas as restrições associadas ao problema de otimização e as quais tem uma grande influência na solução final. Por fim quando todas as variáveis já estejam definidas é necessário selecionar qual dos três algoritmos se pretende utilizar e executar o fmincon.

Quando é selecionado um algoritmo e se executa a função fmincon é gerado de forma aleatória um controlo inicial, sendo assim optou-se que fossem geradas uma serie de repetições em que para cada repetição o controlo inicial fosse gerado de forma aleatória. Para cada repetição eram guardados os valores de fval, flag, o controlos inicias, e os controlos ótimos. Para isso utilizou-se a função *for* do Matlab como mostra a Figura 8.

```
ciclos=input('Escolha o nº de ciclos que quer utilizar:');
options = optimoptions('fmincon','MaxFunEvals',10000,'Algorithm','active-set');
tic
for jj=1:ciclos
u0=rand(2*n,1);     % estimativa inicial de u
% options = optimoptions('fmincon','MaxFunEvals',10000);

[ur,fval,flag,output] = fmincon(@(u)myObj(u,var),u0,[],[],[],[],lb,ub,@(u)myNonlincon(u,var),options);
it = [output.iterations output.funcCount];

Ur(:,jj)=ur; U0(:,jj)=u0; Fval(:,jj)=fval; Flag(:,jj)=flag;
end
toc
tempo = toc ;
minfval=min(Fval); id=find(Fval==minfval); ur=Ur(:,id); u0=U0(:,id); fval=Fval(id);
```

Figura 8- Geração de forma aleatória do controlo inicial para cada repetição

De acordo com a Figura 8 é possível verificar como é gerado o controlo inicial para cada uma das repetições e que cada um dos valores das variáveis que se pretende determinar são guardados. A variável ciclos representa o número de repetições que se pretende fazer, sendo que para cada nº de controlos ou subintervalos utilizados eram testadas diferente número de repetições, este número de repetições variam de um até cem com um incremento de vinte. Depois de efetuadas todas as repetições era exibido o caso em que se obteve menor valor de fval, isto permitiria que se fosse testada uma ampla gama de controlos iniciais e encontrar o mínimo valor de fval para o problema.

3.1.1 Caso 1 vórtice 1 partícula

Neste primeiro caso o movimento da partícula será determinado utilizando apenas um vórtice localizado na origem do plano complexo, consequentemente será apenas necessária a equação que represente o movimento da partícula.

- Equação da partícula

$$\dot{z}^* = \frac{1}{2\pi i} \times \frac{k}{z} + u \tag{15}$$

A variável z corresponde à posição da partícula, por sua vez a variável $\dot{z}^*$ corresponde ao conjugado complexo de z enquanto que a variável k corresponde à circulação do vórtice, a variável u corresponde ao controlo da função.

$$\text{Com } z(0) = z_0 \text{ , } z(t_1) = z_f \text{ e } |u| \leq u_{max} \text{ quando } t_0 < t < t_1$$

As restrições acima referidas definem que para um período de tempo entre o instante t_0 e t_1, $z(0)$ é a posição da partícula para o instante inicial (t_0) e essa posição tem de ser igual a z_0 que é a posição inicial definida, já $z(t_1)$ corresponde à posição da partícula para o instante final (t_1), por sua vez o valor absoluto do controlo da função $(|u|)$ não pode ultrapassar o valor máximo definido (u_{max}).

De forma a aplicar correntemente estas equações em Matlab, é necessário em muitos casos proceder a pequenas modificações de modo que quando aplicadas possam ser processadas de forma correta pelo algoritmo. Neste primeiro caso a equação da partícula inserida no software encontra-se já conjugada como é possível verificar na Figura 9.

```
function dz = myODE(t,z,u,var)
% ODE
dz=(var(12)/conj(z))+u;
end
```

Figura 9- Representação da equação da partícula para N=1

De acordo com a Figura 9 para que seja possível definir corretamente a equação da partícula é necessário definir todas as variáveis que são necessárias para calcular a posição da partícula. As variáveis que é necessário definir é o tempo de deslocamento da partícula (t), a posição da partícula (z), o controlo inicial (u_0) e por fim as variáveis definidas em Matlab (var) como por exemplo a variável 12 que corresponde a $\frac{1}{2\pi i}$.

3.1.2 Caso 2 vórtice 1 partícula

Neste segundo caso a cada vórtice está associada uma equação que retrata a posição desse mesmo vórtice, neste caso como existem 2 vórtices existem duas equações diferentes uma para cada vórtice, respetivamente. No que toca à partícula a mesma também têm uma equação que retrata a sua posição ao longo do tempo, essa posição é determinada em função da posição de cada um dos vórtices, essa posição de cada vórtice influencia o movimento da partícula. As equações que representam o movimento da partícula e dos vórtices são descritas seguidamente.

- Equações da posição dos vórtices

$$z_1(t) = \frac{1}{k_1 + k_2}\left[\left(k_1 z_1(0) + k_2 z_2(0)\right) + \left(z_1(0) - z_2(0)\right)k_2 e^{i\Omega t}\right] \quad (16)$$

$$z_2(t) = \frac{1}{k_1 + k_2}\left[\left(k_1 z_1(0) + k_2 z_2(0)\right) + \left(z_2(0) - z_1(0)\right)k_1 e^{i\Omega t}\right] \quad (17)$$

As variáveis k_1 e k_2 correspondem à circulação do vórtice 1 e 2 respetivamente, enquanto que as variáveis $z_1(0)$ e $z_2(0)$ correspondem à posição inicial do vórtice 1 e 2 respetivamente. Por fim as variáveis $z_1(t)$ e $z_2(t)$ correspondem à posição dos vórtices 1 e 2 para um determinado instante t.

Em que $\Omega = \frac{k_1 + k_2}{2\pi D^2}$ e $D = |z_2(0) - z_1(0)|$.

- Equação da posição da partícula

$$\dot{z}^* = \frac{1}{2\pi i}\left(\frac{k_1}{z - z_1(t)} + \frac{k_2}{z - z_2(t)}\right) + u \tag{18}$$

As variáveis k_1 e k_2 correspondem à circulação do vórtice 1 e 2 respetivamente, enquanto as variáveis $z_1(t)$ e $z_2(t)$ correspondem à posição dos vórtices 1 e 2 para um determinado instante t.

Esta equação é uma equação diferencial ordinária não linear devido ao facto de as variáveis z_1 e z_2 não são constantes pois depende do tempo.

Neste segundo caso as equações representativas quer do deslocamento da partícula que da posição de cada um dos vórtices, quando implementadas em Matlab são definidas no mesmo ficheiro e dentro de uma mesma função criada como ilustrado na Figura 10.

```
function dz = myODE(t,z,u,var)
% ODE
z1=myZ1(t,var);
z2=myZ2(t,var);
dz=conj(var(10)*(var(8)/(z-z1)+var(9)/(z-z2))+u);
end
```

Figura 10- Representação da equação da partícula para N=2

De acordo com a Figura 10 para que seja possível definir corretamente a equação da partícula é necessário definir todas as variáveis que sejam necessárias para calcular a posição da partícula. No entanto as variáveis z_1 e z_2 correspondem à posição de cada um dos vórtices e as quais estão definidas noutro ficheiro, enquanto que a variável 10 corresponde a $\frac{1}{2\pi i}$, já a variável 8 e 9 corresponde à circulação dos vórtices 1 e 2 respetivamente, estas variáveis foram criadas de modo a simplificar a equação correspondente ao movimento da partícula.

3.1.3 Caso 3 vórtices 1 partícula

Neste terceiro caso tem-se o movimento de uma partícula tendo em conta três vórtices, como no caso anterior existe uma equação para o movimento da partícula e existe uma equação que retrata a posição de cada um dos três vórtices. É de realçar também que cada uma das equações que representa a posição de um dado vórtice é em função da circulação e da posição dos outros dois vórtices. As equações estão transcritas seguidamente.

- Equações da posição dos vórtices

$$\dot{z}_1^* = \frac{1}{2\pi i}\left(\frac{k_2}{z_1 - z_2} + \frac{k_3}{z_1 - z_3}\right) \tag{19}$$

$$\dot{z}_2^* = \frac{1}{2\pi i}\left(\frac{k_1}{z_2 - z_1} + \frac{k_3}{z_2 - z_3}\right) \tag{20}$$

$$\dot{z}_3^* = \frac{1}{2\pi i}\left(\frac{k_1}{z_3 - z_1} + \frac{k_2}{z_3 - z_2}\right) \tag{21}$$

As variáveis k_1 , k_2 e k_3 correspondem à circulação dos vórtices 1 , 2 e 3 respetivamente, enquanto as variáveis z_1 , z_2 e z_3 correspondem à posição dos vórtices 1 , 2 e 3, enquanto que $\dot{z}_1^*$, $\dot{z}_2^*$ e $\dot{z}_3^*$ correspondem ao conjugado complexo respetivamente.

Com as condições iniciais $z_1(0) = z_{10}$, $z_2(0) = z_{20}$ e $z_3(0) = z_{30}$.

- Equação da partícula

$$\dot{z}^* = \frac{1}{2\pi i}\left(\frac{k_1}{z - z_1} + \frac{k_2}{z - z_2} + \frac{k_3}{z - z_3}\right) + u \tag{22}$$

As variáveis k_1 , k_2 e k_3 correspondem à circulação dos vórtices 1 , 2 e 3 respetivamente, enquanto as variáveis z_1 , z_2 e z_3 correspondem à posição dos vórtices 1 , 2 e 3.

Com a condição inicial $z(0) = z_0$.

As equações referidas anteriormente quando implementadas em Matlab, ficam transcritas de acordo como ilustrado na Figura 11.

```
function dz = myODE(t,z,u,var)
% ODE
dz= conj(var(12)*[var(10)/(z(1)-z(2))+var(11)/(z(1)-z(3));
    var(9)/(z(2)-z(1))+var(11)/(z(2)-z(3));
    var(9)/(z(3)-z(1))+var(10)/(z(3)-z(2));
    var(9)/(z(4)-z(1))+var(10)/(z(4)-z(2))+var(11)/(z(4)-z(3))+u]);
end
```

Figura 11- Representação da equação da partícula para N=3

De acordo com a Figura 11 é possível verificar que existe um conjunto de variáveis, as variáveis 9, 10 e 11 correspondem à circulação dos vórtices 1, 2 e 3 respetivamente, a variável 12 corresponde a $\frac{1}{2\pi i}$, por fim $z(1)$, $z(2)$ e $z(3)$ correspondem à posição dos vórtices 1, 2 e 3 respetivamente, já $z(4)$ corresponde à posição da partícula.

3.1.4 Caso 4 vórtices 1 partícula

Neste último o movimento da uma partícula tem em conta quatro vórtices, como no caso anterior existe uma equação para o movimento da partícula e existe uma equação que retrata a posição de cada um dos quatro vórtices. Neste caso cada uma das equações que representa a posição de um dado vórtice é em função da circulação e da posição dos outros vórtices. As equações estão transcritas seguidamente.

- Equações da posição dos vórtices

$$\dot{z}_1^* = \frac{1}{2\pi i}\left(\frac{k_2}{z_1 - z_2} + \frac{k_3}{z_1 - z_3} + \frac{k_4}{z_1 - z_4}\right) \quad (23)$$

$$\dot{z}_2^* = \frac{1}{2\pi i}\left(\frac{k_1}{z_2 - z_1} + \frac{k_3}{z_2 - z_3} + \frac{k_4}{z_2 - z_4}\right) \quad (24)$$

$$\dot{z}_3^* = \frac{1}{2\pi i}\left(\frac{k_1}{z_3 - z_1} + \frac{k_2}{z_3 - z_2} + \frac{k_4}{z_3 - z_4}\right) \quad (25)$$

$$\dot{z}_4^* = \frac{1}{2\pi i}\left(\frac{k_1}{z_4 - z_1} + \frac{k_2}{z_4 - z_2} + \frac{k_3}{z_4 - z_3}\right) \quad (26)$$

As variáveis k_1, k_2, k_3 e k_4 e correspondem à circulação dos vórtices 1 , 2 , 3 e 4 respetivamente, enquanto as variáveis z_1 , z_2 , z_3 e z_4 correspondem à posição dos vórtices 1 , 2 , 3 e 4.

Com as condições iniciais $z_1(0) = z_{10}$, $z_2(0) = z_{20}$, $z_3(0) = z_{30}$ e $z_4(0) = z_{40}$.

- Equação da partícula

$$\dot{z}^* = \frac{1}{2\pi i}\left(\frac{k_1}{z - z_1} + \frac{k_2}{z - z_2} + \frac{k_3}{z - z_3} + \frac{k_4}{z - z_4}\right) + u \tag{27}$$

As variáveis k_1, k_2, k_3 e k_4 e correspondem à circulação dos vórtices 1 , 2 , 3 e 4 respetivamente, enquanto as variáveis z_1 , z_2 , z_3 e z_4 correspondem à posição dos vórtices 1 , 2 , 3 e 4.

Com a condição inicial $z(0) = z_0$.

As equações transcritas anteriormente, e como nos casos anteriores sofrem pequenas modificações possibilitando assim que sejam utilizadas corretamente, estando ilustrado na Figura 12 as respetivas equações para este caso específico.

```
function dz = myODE(t,z,u,var)
% ODE
dz= conj(var(14)*[var(11)/(z(1)-z(2))+var(12)/(z(1)-z(3))+var(13)/(z(1)-z(4));
    var(10)/(z(2)-z(1))+var(12)/(z(2)-z(3)+var(13)/(z(2)-z(4)));
    var(10)/(z(3)-z(1))+var(11)/(z(3)-z(2))+var(13)/(z(3)-z(4));
    var(10)/(z(4)-z(1))+var(11)/(z(4)-z(2))+var(12)/(z(4)-z(3));
    var(10)/(z(5)-z(1))+var(11)/(z(5)-z(2))+var(12)/(z(5)-z(3))+var(13)/(z(5)-z(4))+u]);
end
```

Figura 12- Representação da equação da partícula para N=4

De acordo com a Figura 12 é possível verificar que existe um conjunto de variáveis, as variáveis 10, 11, 12 e 13 correspondem à circulação dos vórtices 1,2,3 e 4 respetivamente, a variável 14 corresponde a $\frac{1}{2\pi i}$, por fim $z(1)$, $z(2)$, $z(3)$ e $z(4)$ correspondem à posição dos vórtices 1, 2 , 3 e 4 respetivamente, já $z(5)$ corresponde à posição da partícula.

4. Resultados Computacionais

Neste capítulo irão ser apresentados os valores mínimos de energia consumida no movimento da partícula obtidos para cada um dos quatro casos testados assim como o tempo de computação utilizando três algoritmos diferentes para cada um dos casos , assim como a sua evolução à medida que são alterados alguns parâmetros como o caso do nº de controlos e dentro de cada controlo é alterado o nº de ciclos, esta variável corresponde ao nº de repetições efetuadas para gerar o controlo inicial de forma aleatória e se obter um valor de fval em cada repetição.

Para cada caso modificou-se o nº de subintervalos, que corresponde ao nº de controlos, de 1 até 10, sendo que para cada subintervalos variou-se o nº de ciclos utilizados, esta última variável continha valores que variavam de 1 a 100 com espaçamento de 20 em 20 entre eles. É de salientar que o menor valor obtido para cada um dos subintervalos corresponde a um nº variável de ciclos, ou seja, para cada um os dos subintervalos foi selecionado o menor valor independentemente do nº de ciclos utilizado para obtenção do menor valor de energia consumida pela partícula. Posteriormente cada caso foi testado utilizando cada um dos três algoritmos.

4.1 Caso de 1 vórtice e 1 partícula

Primeiramente serão apresentados os resultados obtidos para o caso em que o movimento da partícula ocorre considerando apenas um vórtice, estando na Figura 13 representados os valores mínimos de energia consumida obtidos em função do nº respetivo de subintervalos para cada um dos algoritmos utilizados.

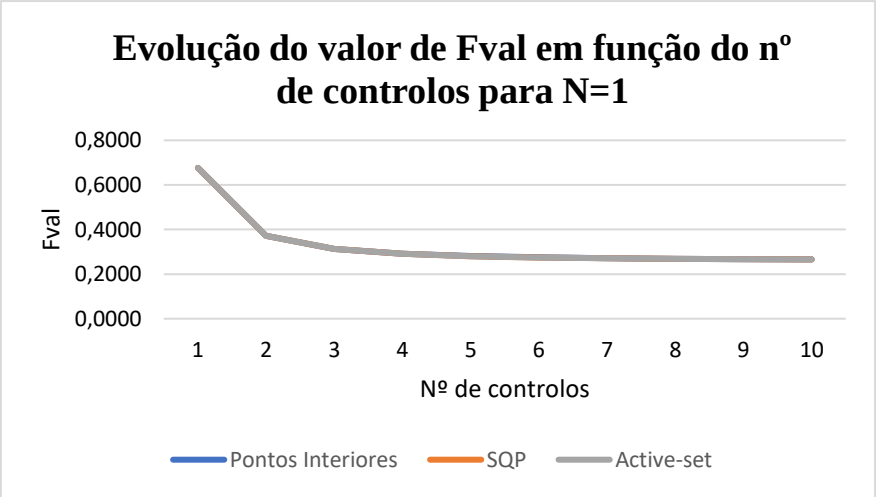

Figura 13- Evolução do valor de Fval em função do nº de controlos para N=1

Da Figura 13 pode-se retirar que o valor de fval para cada um dos algoritmos tende a diminuir á medida que aumenta o nº de subintervalos, quando se passa de um controlo para dois controlos verifica-se que existe uma grande diminuição no valor de fval, sendo que a partir de dois controlos verifica-se que o valor de fval tende a estabilizar para valores perto de 0,26. Embora os valores obtidos para cada um dos algoritmos sejam idênticos, de forma a comparar qual dos algoritmos conseguiu menor de fval foi registado o melhor valor para cada um dos algoritmos. O menor valor de fval para cada um dos algoritmos foi obtido utilizando dez controlos, no entanto o nº de ciclos necessários para chegar a essa solução variam de algoritmo para algoritmo. O menor valor para cada um dos algoritmos, o tempo de computação e outros parâmetros estão especificados na Tabela 4.

Tabela 4- Comparação do menor valor de Fval referente a cada algoritmo para N=1

Algoritmo	Nº de controlos	Nº de ciclos	Fval	Flag	Tempo
Active-set	10	20	0,2659	0	4316,751
Pontos Interiores	10	40	0,2661	2	1065,72
SQP	10	1	0,2659	2	3,529

De acordo com a Tabela 4 podemos verificar que embora a diferença seja mínima os algoritmos active-set e SQP apresentam menor resultado face ao algoritmo de pontos interiores. No entanto como o algoritmo SQP foi o mais rápido a obter a solução considera-se que foi o algoritmo SQP que obteve melhor resultado, ou seja, menor valor de fval, sendo que na Figura 14 está representada a trajetória do movimento da partícula.

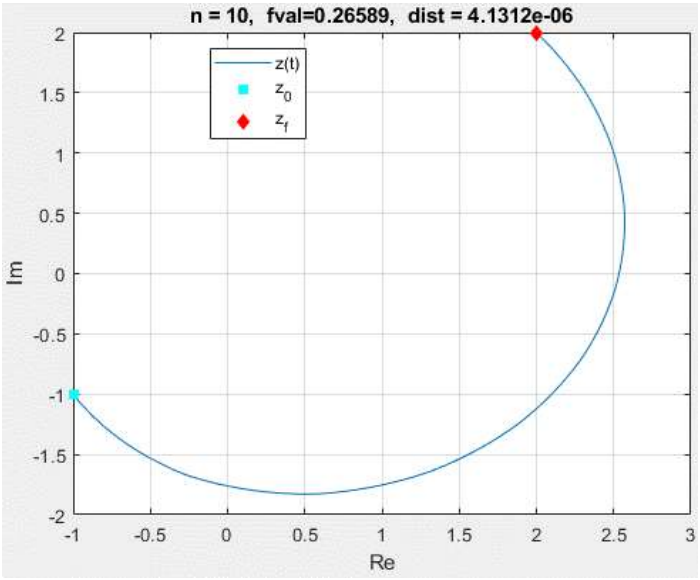

Figura 14- Trajetória da partícula para menor valor de Fval para N=1

De maneira a testar qual dos algoritmos é o mais rápido a obter uma solução viável, registou- o tempo de computação de cada um dos algoritmos para cada um dos valores de controlo utilizados. Neste caso e para se obter uma comparação justa em termos de rapidez utilizou-se um nº de ciclos constante que neste caso foi 60 ciclos para cada valor de n. Na Figura 15 está representada a evolução do tempo de computação para cada um dos algoritmos utilizados.

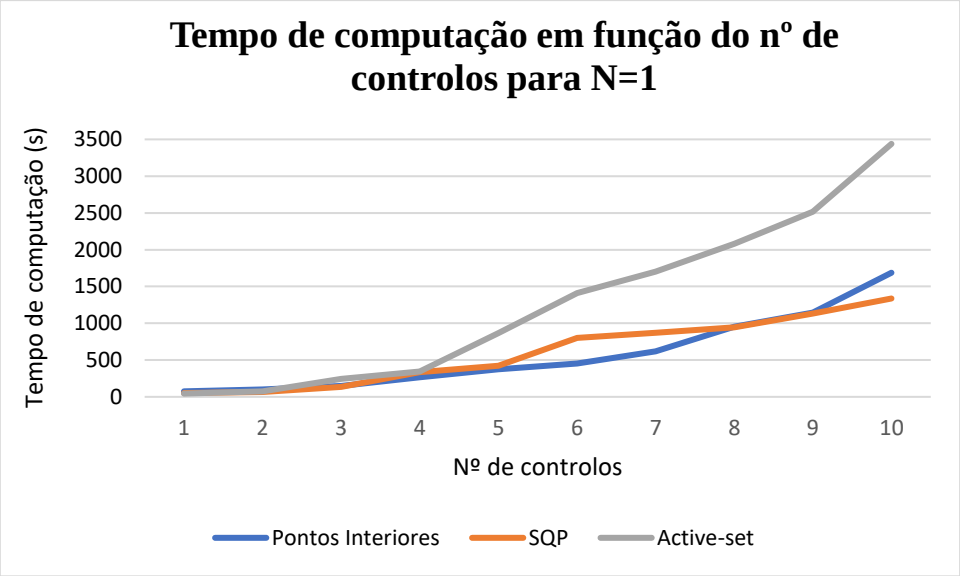

Figura 15- Tempo de computação em função do nº de controlos para N=1

Tendo em conta a Figura 15 pode-se verificar que quando o número de controlo é inferior a três os algoritmos apresentam tempo de computação, sendo o Active set o algoritmo mais rápido.

A partir do caso em que o numero de controlos é superior a três verifica-se que o Active-set começa a apresentar tempos de computação superiores em relação ao outros algoritmos em estudo, que para alguns casos apresentam tempos de computação muitos idênticos sendo que em outros casos pode-se verificar que estão numa disputa em relação a qual deles o mais rápido. Para se verificar qual deles o mais rápido teria de se amplia a gama de controlo e verificar o comportamento de cada algoritmo.

4.2 Caso de 2 vórtices e 1 partícula

Como referido em capítulos anteriores neste caso o movimento da partícula é determinado utilizando dois vórtices, para encontrar o valor mínimo de energia gasta foram testados os três algoritmos diferentes. A evolução de fval para os três algoritmos ao longo de cada número de controlos utilizados está representado na Figura 16.

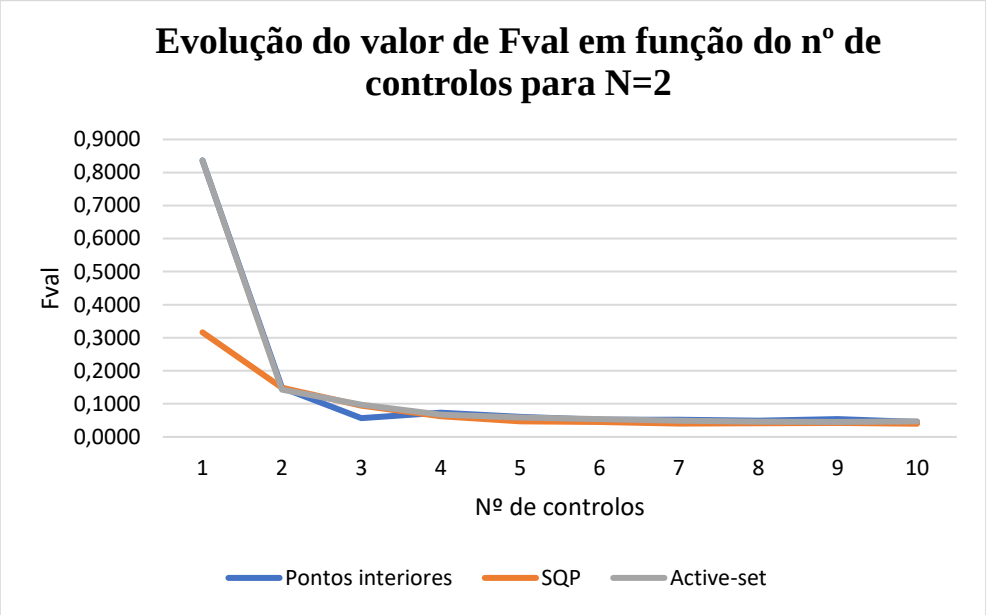

Figura 16- Evolução do valor de Fval em função do nº de controlos para N=2

Da Figura 16 pode-se retirar que fval diminui drasticamente quando se passa de um controlo para dois controlos, a partir de dois controlos tende a estabilizar como aconteceu quando foi apenas utilizado um vórtice. No entanto quando considerado um vórtice á medida que se aumentava o nº de controlos o valor de fval diminuía, neste caso em que são utilizados dois vórtices não é possível afirmar que o valor de fval diminui sempre que é aumentado o nº de controlos, pois verifica-se ligeiros aumentos no valor de fval para alguns controlos face ao controlo utilizado anteriormente.

Quando considerados pelos menos dois vórtices verifica-se que aumentado o nº de ciclos não implica um menor valor de fval como no caso de N=1 pois em alguns casos utilizando um nº de ciclos maior obtém-se um valor de fval maior face ao valor de fval utilizando um nº de ciclos menor.

Ainda referente à Figura 16 pode-se verificar que apesar da diferença no valor de fval quando utilizado um controlo, verifica-se que menor valor de fval para cada algoritmo tendem a aproximar-se á medida que o nº de controlos aumenta. A Tabela 5 mostra a comparação entre o menor valor de fval para cada algoritmo.

Tabela 5- Comparação do menor valor de Fval referente a cada algoritmo para N=2

Algoritmo	Nº de controlos	Nº de ciclos	Fval	Flag	Tempo
Active-set	9	100	0,0453	5	1566,08
Pontos Interiores	10	100	0,0454	2	2434,348
SQP	10	100	0,0400	2	933,482

Em relação à tabela anterior pode-se verificar que o algoritmo que conseguiu obter um menor valor de fval foi o algoritmo SQP conseguindo atingir um valor de 0,04 com um tempo de computação inferior face aos outros algoritmos, ou seja, para além de atingir o menor valor atingiu esse valor muito mais rápido que os outros dois algoritmos. Por sua vez o algoritmo active-set e pontos interiores apresentam valores de fval idênticos, no entanto o algoritmo active-set atingiu esse valor mais rapidamente face ao algoritmo de pontos interiores. A trajetória do movimento da partícula referente ao algoritmo que apresentou menor valor de fval esta representado na Figura 17.

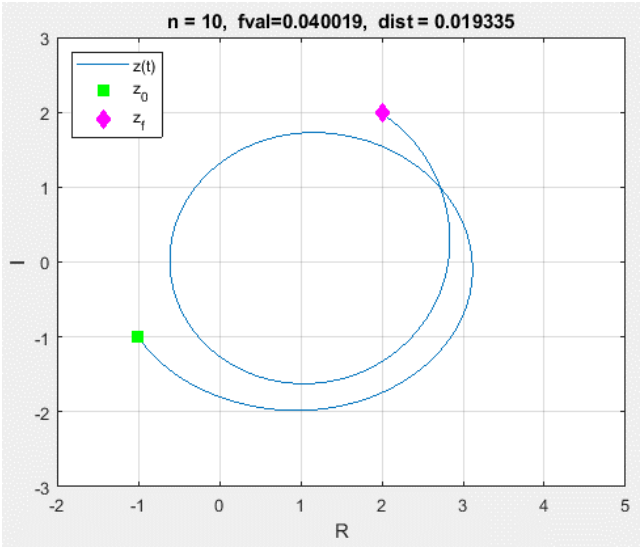

Figura 17- Trajetória da partícula para menor valor de Fval para N=2

Tal como para o caso de N=1 decidiu-se verificar o tempo de computação de cada um dos algoritmos a atingir uma solução viável para os diferentes valores de controlos utilizados, para isso e de modo a ter uma comparação justa considerou-se para cada número de controlos utilizados um nº de ciclos constante, considerando então 40 ciclos. A Figura 18 mostra a evolução do tempo de computação para cada um dos algoritmos utilizados.

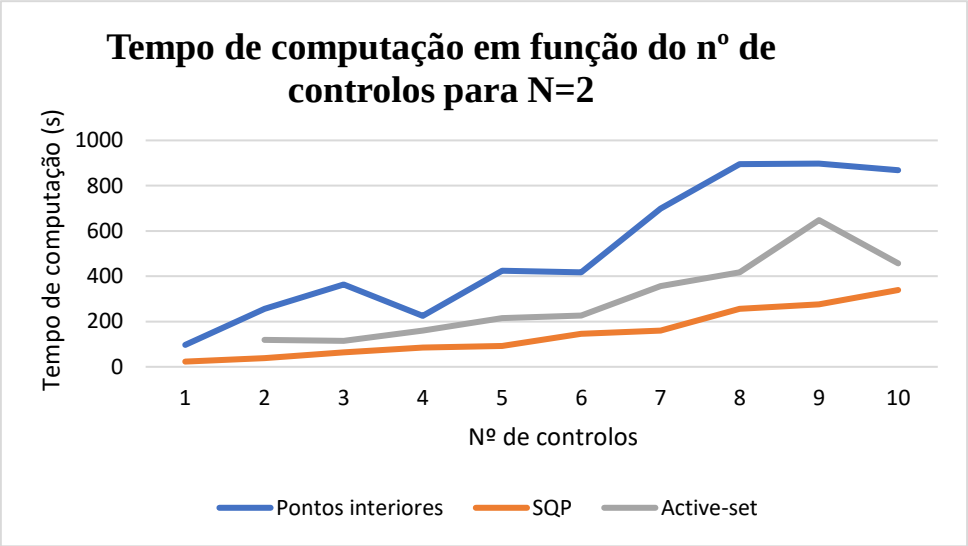

Figura 18- Tempo de computação em função do nº de controlos para N=2

Tendo em conta a Figura 18 pode-se verificar que variando o nº de controlos de 1 até 10 o algoritmo SQP é o algoritmo que sempre obteve resultados viáveis mais rápido, seguido do algoritmo de pontos interiores e por fim o algoritmo active-set.

Para n=1 o algoritmo active-set não possui valor devido ao facto que quando utilizados 40 ciclos o algoritmo não convergia, ou seja, não atingia o ponto final logo não poderia ser considerada uma solução viável.

O tempo de computação de cada algoritmo aumenta á medida que o nº de controlos aumento, no entanto existe uma queda para o algoritmo active set para n=4 e para o algoritmo de pontos interiores quando o n=10, isto está diretamente relacionado com uma maior trajetória da partícula e consequentemente um maior tempo até atingir o ponto final.

4.3 Caso de 3 vórtices e 1 partícula

Neste caso o movimento da partícula é dado tendo em conta três vórtices de modo a encontrar o menor valor de energia gasto no deslocamento foram testados três algoritmos diferentes de modo a verificar qual dos algoritmos apresenta menor valor de fval variando o nº de controlos utilizados. A evolução do valor de fval em função do nº de controlos está especificada na Figura 19.

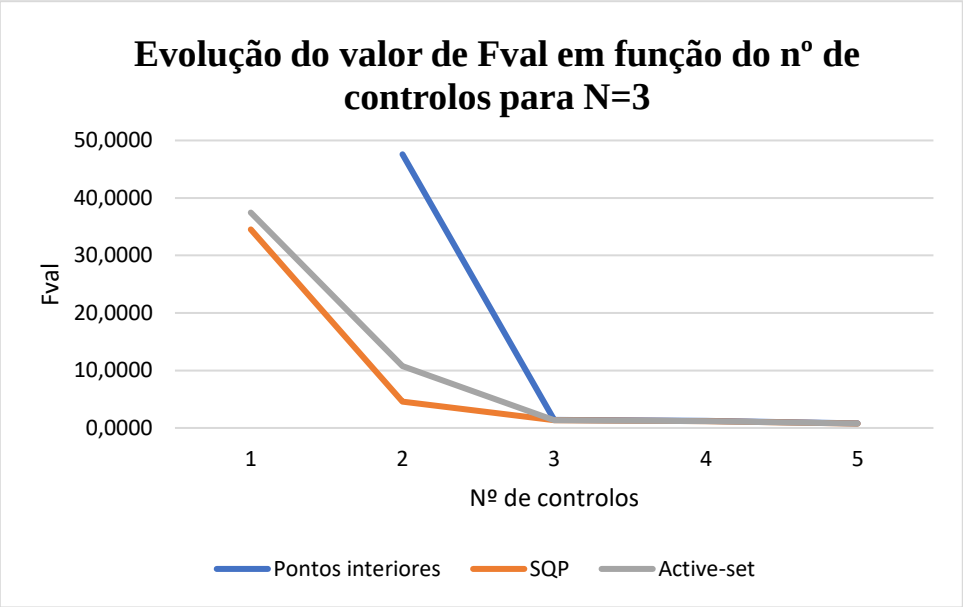

Figura 19- Evolução do valor de Fval em função do nº de controlos para N=3

De acordo com a Figura 19 pode-se salientar que o algoritmo de pontos interiores não converge quando é apenas usado um controlo, por sua vez os outros algoritmos convergem tendo obtido um valor de fval elevado, estes dois algoritmos apresentam grande quebra para dois controlos face a um controlo. No entanto quando utilizado um nº de controlos superior a três verifica-se que os três algoritmos tendem para um valor na ordem da unidade. Devido ao facto de apresentarem valores de fval idênticos é necessário apresentar esses valores, sendo apresentados todos os valores na Tabela 6.

Tabela 6- Comparação do menor valor de Fval referente a cada algoritmo para N=3

Algoritmo	Nº de controlos	Nº de ciclos	Fval	Flag	Tempo
Active-set	5	60	0,7635	5	1799,322
Pontos Interiores	5	80	0,8024	0	6126,065
SQP	5	60	0,7624	2	1640,409

No que diz respeito á Tabela 6 é possível verificar que o algoritmo que apresentam menor valor de fval é o algoritmo SQP apresentando também um tempo de computação menor em relação aos outros algoritmos.

O algoritmo active-set apresenta também um valor de fval muito próximo do algoritmo SQP, por outro lado o algoritmo que apresenta maior valor de fval é o algoritmo de pontos interiores tendo também um tempo de computação muito superior em relação aos outros dois algoritmos. A trajetória do movimento da partícula referente ao melhor valor de fval está representado na Figura 20.

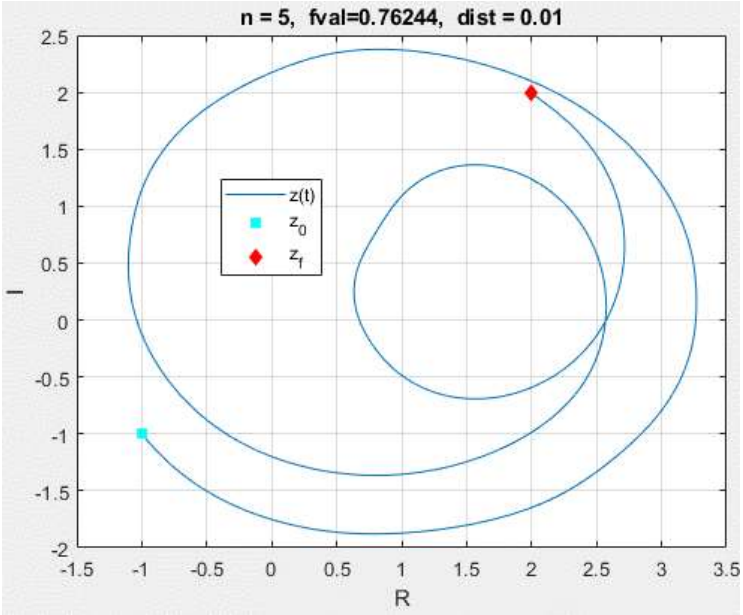

Figura 20- Trajetória da partícula para menor valor de Fval para N=3

Por fim foi testado também o tempo de computação de modo a comparar cada um dos algoritmos para cada nº de controlos utilizados, estando representados na Figura 21 os tempos para cada um dos algoritmos utilizando 40 ciclos.

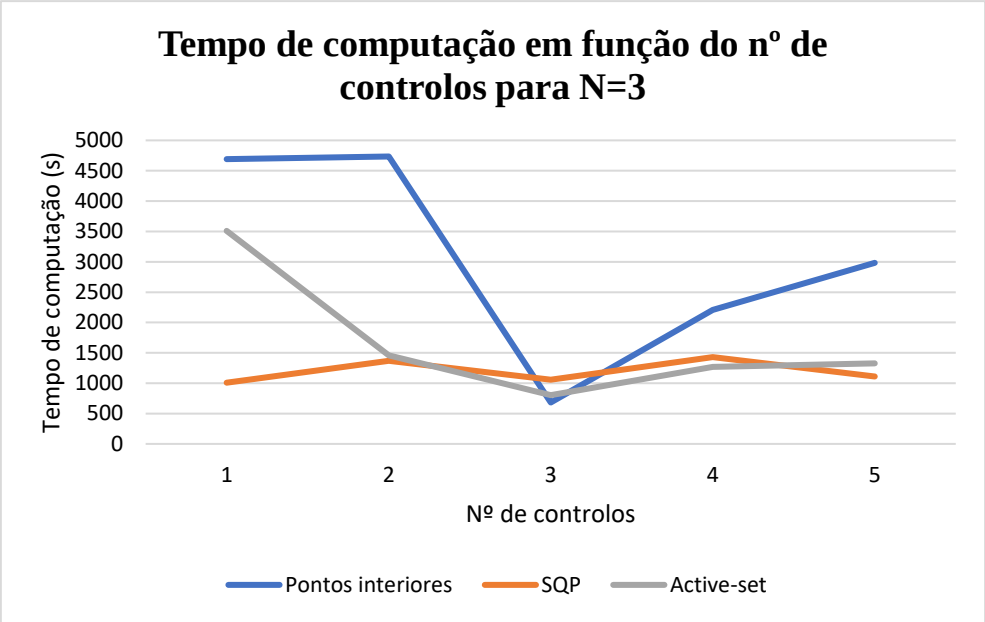

Figura 21- Tempo de computação em função do nº de controlos para N=3

Da Figura 21 pode-se retirar que os algoritmos SQP e active set apresentam tempos de computação muito próximos quando são utilizados pelo menos dois controlos, enquanto que o algoritmo de pontos interiores apresenta valores muito elevados quando utilizados menos de três controlos, isto deve-se ao facto de o valor atingindo pelo algoritmo não convergir e assim consequentemente levar a um maior tempo de computação. Apesar de os pontos interiores apresentar um menor valor quando utilizados três controlos, verifica-se que a partir daí apresenta tempos de computação bem superiores aos outros algoritmos.

4.4 Caso de 4 vórtices e 1 partícula

Neste ultimo caso o movimento da partícula será feito tendo em conta quatro vórtices utilizando como nos casos anteriores três algoritmos diferentes de modo a verificar qual dos algoritmos apresenta menor valor de fval variando o nº de controlos utilizados.

Dado que estão a ser considerados quatro vórtices isso implica um tempo de computação elevado para cada um dos valores, por isso para este caso apenas se fez variar o nº de controlos de um a cinco, estando a evolução do valor de fval em função do nº de controlos especificada na Figura 22.

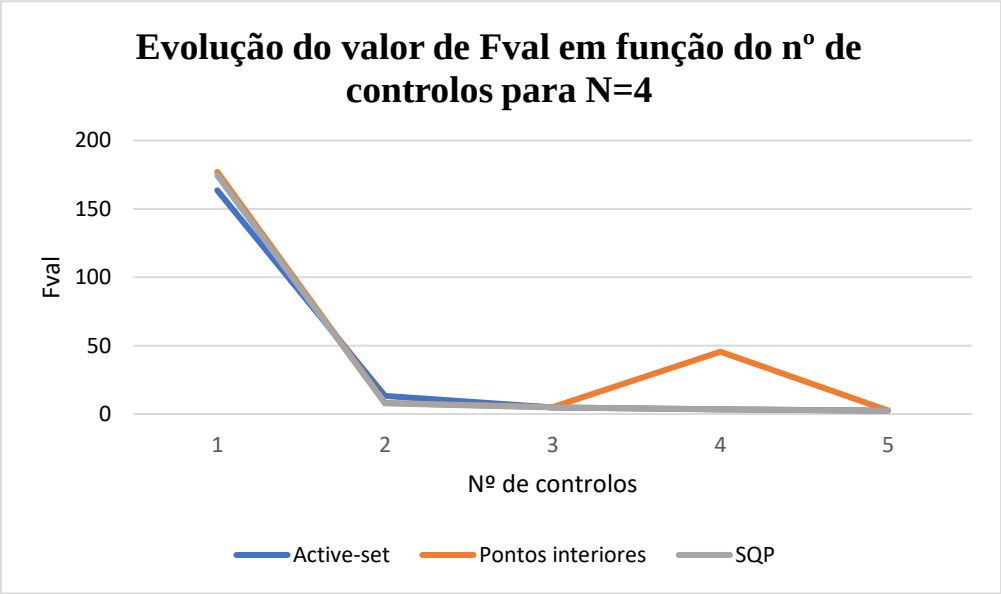

Figura 22- Evolução do valor de Fval em função do nº de controlos para N=4

Em relação à Figura 22 pode-se salientar que todos os algoritmos possuem uma evolução idêntica em função do nº de controlos utilizados exceto no caso em que foram utilizados quatro controlos pois verifica-se que o algoritmo de pontos interiores apresenta para esse caso especifico um valor de fval muito mais elevado face ao valor apresentado pelos outros dois algoritmos.

Ainda que apresentem uma evolução idêntica e apesar de cada um dos algoritmos atingir o seu melhor valor, ou seja, o gasto de energia no movimento foi mínimo quando utilizados cinco controlos. Contudo o algoritmo active-set e SQP apresentam o mesmo valor de fval enquanto que o algoritmo de pontos interiores apresenta um valor superior face aos outros dois algoritmos utilizados.

Na Tabela 7 está representado o menor valor de fval para cada algoritmo assim como os parâmetros associados a esse valor.

Tabela 7- Comparação do menor valor de Fval referente a cada algoritmo para N=4

Algoritmo	Nº de controlos	Nº de ciclos	Fval	Flag	Tempo
Active-set	5	60	2,4588	0	6285,245
Pontos Interiores	5	80	2,5854	2	13604,25
SQP	5	100	2,4588	2	5300,113

Da Tabela 7 pode-se verificar que apesar de o algoritmo active-set e SQP apresentarem valores de fval iguais, o algoritmo SQP mesmo utilizando um nº de ciclos maior para encontrar o valor mínimo de fval apresenta um tempo de computação inferior ao outro algoritmo, o que permite afirmar que o algoritmo SQP é o algoritmo que apresenta melhor valor, estando a trajetória do movimento da partícula representada na Figura 23.

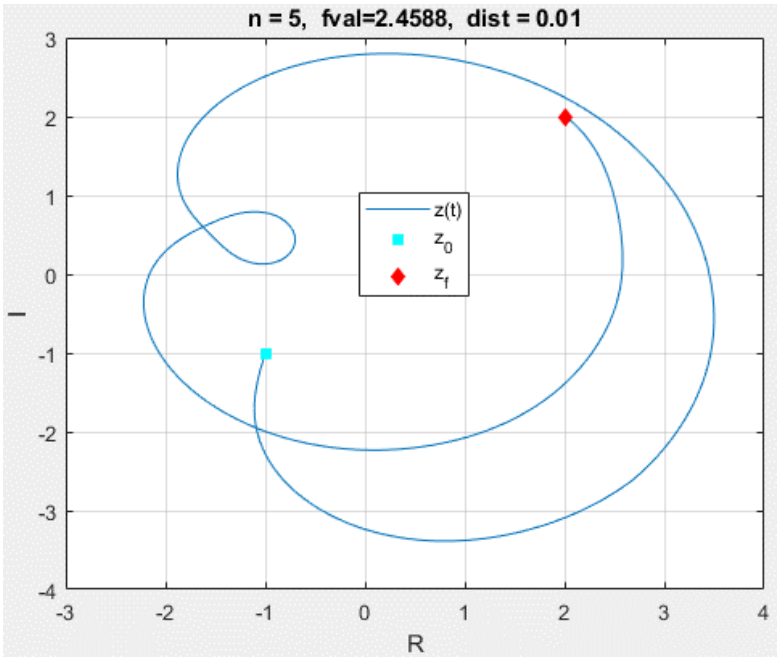

Figura 23- Trajetória da partícula para menor valor de Fval para N=4

Como nos casos anteriores foi testado também o tempo de computação de modo a comparar cada um dos algoritmos para cada nº de controlos utilizados, estando representados na Figura 24 os tempos para cada um dos algoritmos utilizando 40 ciclos.

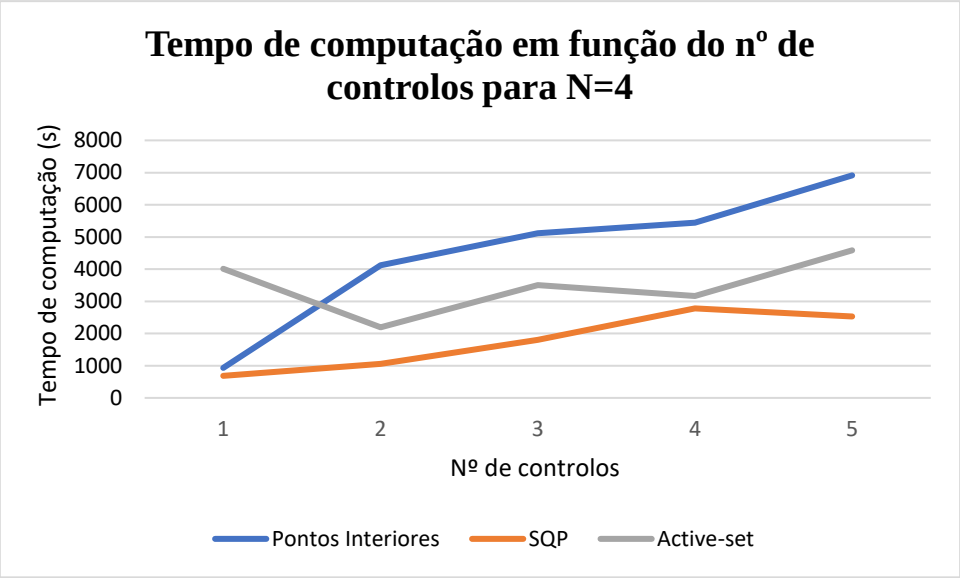

Figura 24- Tempo de computação em função do nº de controlos para N=4

Tendo em conta a Figura 24 é possível afirmar que quando utilizado um controlo o algoritmo mais demorado é o algoritmo active-set enquanto que para um nº de controlos igual ou superior a 2 o algoritmo SQP é o algoritmo mais rápido a obter uma solução, seguidamente o active-set sendo que o algoritmo mais lento a obter uma solução é o algoritmo de pontos interiores.

No entanto dentro de cada algoritmo ocorre que para um nº de controlos superiores o algoritmo apresenta uma maior rapidez face ao nº de controlos anterior o que estará relacionado com a razão pela qual o algoritmo parou.

5. Análise de Resultados

Apesar de que no capítulo anterior tenha sido elaborada uma pequena análise à medida que eram apresentados os resultados para cada um dos casos, achou-se por necessário elaborar uma análise sintetizada caso a caso. Primeiramente será analisado qual o algoritmo que atingiu o valor mínimo de fval e posteriormente será feita uma análise ao algoritmo que atingiu um menor tempo de computação para um nº de ciclos constante ao longo dos controlos utilizados.

No caso em que apenas foi utilizado um vórtice o menor valor de energia consumida no deslocamento da partícula foi de 0,2659 utilizando o algoritmo SQP, por sua vez o algoritmo active-set apresentou o mesmo valor mais com um tempo de computação superior. No que diz respeito ao tempo de computação para um nº de ciclos constante os algoritmos que apresentaram mais rapidez em chegar a solução é o algoritmo de pontos interiores alternando com o algoritmo SQP tendo em conta o nº de controlos utilizados.

No caso em que foram utilizados dois vórtices foi o caso em que se obteve o valor mínimo de energia consumida face a todos casos testados, obtendo-se assim um valor de fval de 0,04 por intermédio do algoritmo SQP. Por sua vez o algoritmo que atingiu um tempo de computação menor a atingir uma solução para um dado nº de ciclos foi o algoritmo de pontos interiores.

No caso em que foram utilizados três vórtices o menor valor atingido foi de 0,7624 através do algoritmo SQP, enquanto que no que diz respeito ao menor tempo de computação os algoritmos que atingiram melhores valores foi o SQP e o active-set dependo dos controlos utilizados.

No caso em que foram utilizados quatro vórtices o menor valor obtido foi de 2,4548 uma vez mais por intermédio do algoritmo SQP, o mesmo algoritmo obteve também um menor tempo de computação para um nº de ciclos constante face aos outros algoritmos.

Em síntese é possível afirmar que o algoritmo SQP é o algoritmo que apresenta os valores mínimos de energia consumida para todos os casos testados, e tendo obtido o valor mínimo de 0,04 para o caso em que foram utilizados dois vórtices.

De forma a facilitar uma melhor análise e comparação entre cada caso testado, todos os valores referidos anteriormente estão sistematizados na Tabela 8.

Tabela 8- Comparação do valor de Fval e tempo de computação entre todos os casos testados

Método	1 Vórtice		2 Vórtices		3 Vórtices		4 Vórtices	
	Fval	Tempo de computação	Fval	Tempo de computação	Fval	Tempo de computação	Fval	Tempo de computação
Active-set	0,2659	4316,751	0,0453	1566,080	0,7635	1799,322	2,4588	6285,245
Pontos Inteiores	0,2661	1065,720	0,0454	2434,348	0,8024	6126,065	2,5854	13604,250
SQP	0,2659	3,529	0,0400	933,482	0,7624	1640,409	2,4588	5300,113

5.1 Implementação de uma estratégia

Face a todos os resultados analisados anteriormente pretendeu-se adotar e implementar determinadas estratégias de modo a tentar alcançar o valor obtido para cada um dos casos ou obter um valor ainda menor, ou seja, com a implementação destas estratégias o objetivo era otimizar o problema tentando atingir um menor valor de energia gasto no movimento da partícula assim como um tempo de computação menor, permitindo assim aumentar a eficiência deste problema de controlo ótimo.

Uma das estratégias aplicadas baseou-se na utilização de uma estimativa inicial obtida através dos controlos anteriores, em que foi utlizada uma média ponderada de modo a dar diferentes pesos a cada um dos controlos anteriores. Esta estratégia foi implementada para o caso de dois vórtices e quando são utilizados 3 controlos, ou seja, neste caso utilizou-se os controlos dos casos de n=1 e n=2 para obter a estimativa do controlo inicial para o caso em que são utilizados 3 controlos.

De forma a obter a estimativa inicial dos controlos, neste caso 3 controlos, é necessário utilizar a parte real e a parte imaginária para se efetuar a média dos controlos (u) para os casos n=1 e n=2. Como estes controlos são números complexos é necessário transformar estes controlos em números reais para que seja possível utilizar os mesmos como controlo inicial, pois um controlo inicial é um vetor com apenas números reais. Isto deve-se ao facto de a função Fmincon só trabalhar com números reais sendo necessário posteriormente transformar o controlo obtido através da execução da função num controlo complexo pois o deslocamento dá-se no plano complexo.

Depois de conhecidos os controlos para n=1 e n=2 transformam-se esses controlos que estão na forma complexa para números reais e efetua-se a média para cada um dos casos.

Após calculada a respetiva média dos controlos para cada um dos casos, é necessário atribuir diferentes pesos a cada uma dessas médias, de modo a efetuar uma média ponderada, cada valor da média ponderada obtido servirá de estimativa inicial do controlo para o caso em que eram utilizados 3 controlos, sendo registados todos os valores de fval assim como o tempo de computação associado a cada estimativa de controlo inicial. Depois de implementada a estratégia foram obtidos os resultados representados na Tabela 9.

Tabela 9- Valores obtidos depois de aplicada a estratégia

U médio para n=1	U médio para n=2	Peso n=1	Peso n=2	Média Final	Fval
-0,02275	-0,0252	0,1	0,9	-0,024955	0,525837
		0,2	0,8	-0,02471	0,7319
		0,3	0,7	-0,024465	0,5257
		0,4	0,6	-0,02422	0,5257
		0,5	0,5	-0,023975	0,1321
		0,6	0,4	-0,02373	0,1617
		0,7	0,3	-0,023485	0,1311
		0,8	0,2	-0,02324	0,5257
		0,9	0,1	-0,022995	0,1380
		1	0	-0,02275	0,5262

É de salientar que o menor valor de fval obtido para este caso antes de ter sido aplicada a estratégia foi de 0,0951, valor que é inferior a qualquer um dos valores obtidos após a implementação da estratégia que de acordo com a Tabela 9 o menor valor obtido através da implementação da estratégia é de 0,1311, valor que foi obtido aquando de atribuído um peso de 70% aos controlos de n=1 e um peso de 30% aos controlos de n=2. É ainda possível afirmar que de acordo com o peso atribuído a cada um dos controlos obtém-se valores de fval diferentes entre si, no entanto existem casos que os valores de fval são iguais ou muito próximos.

5.2 Comparação de valores utilizando diferentes tipos de regras de aproximação

Para este problema de controlo ótimo a energia mínima gasta no deslocamento é obtida pelo integral da função objetivo, para se efetuar a integração no código Matlab é necessário recorrer a regras de aproximação. Existem uma variedade de regras de aproximação para uma função, pelo que para cada problema de controlo existe uma regra que melhor se adequa e consequentemente permite atingir menores valores da função objetivo no caso de se tratar de um problema que se pretende encontrar o mínimo da função.

A regra de aproximação utilizada para encontrar o valor de energia mínima gasta no deslocamento da partícula foi a regra dos retângulos, pelo que todos os resultados apresentados anteriormente foram obtidos a partir da utilização da mesma.

Como já referido anteriormente para um mesmo problema pode-se utilizar várias regras de aproximação, consequentemente e de modo a obter uma pequena comparação entre os valores atingidos decidiu-se testar também a regra dos trapézios e a regra de Simpson. Para esta comparação utilizou-se o caso para o qual se tinha obtido o menor valor de energia consumida, caso N=2, utilizando a regra dos retângulos. A comparação entre os valores obtidos por cada uma das regras está representado na Tabela 10.

Tabela 10- Comparação entre os valores obtidos por cada uma das regras utilizadas para o caso N=2

Nº de subintervalos	Regra de aproximação	Valor de Fval	Tempo de computação
3	Regra dos Retângulos	0,0400	1,9422
	Regra dos Trapézios	0,4567	2,3203
	Regra de Simpson	0,5840	10,6770

De acordo com a tabela é possível verificar que a regra dos retângulos é a que apresenta menor valor de fval assim como um menor tempo de computação, por sua vez a regra dos trapézios é aquela que apresenta o segundo menor valor de fval e de tempo de computação. A regra de Simpson é aquela que apresenta um maior valor de fval maior assim como um maior tempo de computação.

6. Conclusão

O processo de otimização pode ter um papel preponderante na conceção de qualquer projeto podendo ser aplicada em resolução de problemas de várias áreas e até pode ser efetuada em diferentes níveis dentro de uma empresa, desde uma combinação de unidades, equipamentos, até entidades menores. Dependendo do problema em análise, a otimização permite a obtenção de um melhor resultado para uma função objetivo, um valor mínimo no caso de se pretender minimizar a função objetivo ou um valor máximo no caso de se pretender maximizar a função objetivo.

Embora que a teoria de controlo aplicada à dinâmica de vórtices apenas despertado um interesse especial nos últimos anos, atualmente já existe um grande interesse na utilização desta teoria, nomeadamente nos campos da dinâmica de fluidos geofísicos, aeronáuticos e hidrodinâmicos, existindo já um aerogerador em que o seu funcionamento se baseia nesta teoria.

No que toca ao problema de controlo primeiramente pode-se concluir que fazendo aumentar o numero de ciclos de repetição do problema de otimização, aumentam também as possibilidades de chegar à solução ótima. Isso verificou-se sobretudo no problema com um único vórtice. Contudo, não garante que a solução obtida seja ótima.

Por outro lado, considerando cada um dos casos é possível afirmar que de uma forma geral à medida que é aumentado o nº de controlos utilizados dentro do mesmo caso o valor da função objetivo vai diminuindo à medida que aumenta o nº de controlos utilizados.

Apesar de terem sido testados três algoritmos diferentes é de salientar que para todos os quatro casos utilizados o algoritmo SQP foi o algoritmo que atingiu sempre melhores resultados, embora em alguns casos o algoritmo active-set tenha apresentado valores de fval iguais ao algoritmo SQP apresentou um tempo de computação superior, o que implica que o algoritmo SQP tenha sido aquele que apresentou melhor resultado.

Por sua vez para um determinado nº de ciclos utilizado verifica-se que dependendo do caso em estudo, o algoritmo a obter uma solução de uma forma mais rápida, podendo esta ter convergido ou não, varia de caso para caso tendo todos os algoritmos alcançado melhores tempo de computação para casos específicos.

Tendo em consideração todos os casos testados é de fácil verificação que o valor mínimo de energia consumida pelo deslocamento da partícula é de 0,04, o qual foi alcançado no caso em que foram utilizados apenas dois vórtices. O caso que apresentou pior valor, ou seja, um valor mais elevado de energia gasto foi o caso em que foram utilizados quatro vórtices apresentado um valor de 2,4588.

Perante todos os valores obtidos foi implementada uma estratégia baseada na utilização controlos obtidos em casos anteriores, no entanto essa estratégia não permitiu obter menores valores de energia gasta no deslocamento da partícula.

Por fim é possível concluir que apesar de ser possível utilizar varias regras de aproximação para a integração de uma função objetivo, para este caso de controlo ótimo a regra que apresenta menor valor de fval é a regra dos retângulos face a todas as outras regras utilizadas.

Bibliografia

[1] Paredes, B. J. B., Santana, G. A., & de Albuquerque Fell, A. F. (2014). "Um estudo de aplicação do radar da inovação: o grau de inovação organizacional em uma empresa de pequeno porte do setor metalo-mecânico". Navus-Revista de Gestão e Tecnologia, 4(1), 76-88.

[2] Koumoutsakos, P. D., & Mezic, I. (2006). "Control of fluid flow". Springer.

[3] Reis P. (2015). "Aerogerador sem hélices pode revolucionar energia eólica". Endereço: https://www.portal-energia.com, consultado em 10/04/2020.

[4] Protas, B. (2008). Vortex dynamics models in flow control problems. Nonlinearity, 21(9), R203.

[5] Barbosa, L. Z. (2012) "Técnicas de otimização baseadas no paradigma de enxames de partículas e sua aplicação ao projeto de equipamentos eletromagnéticos". (Dissertação de Doutoramento, Universidade de São Paulo).

[6] Balsa, C., Gama, S., & Braz-César, M. T. (2019). "Control problem in passive tracer advection by point vortex flow: a case study." In 7th International Conference on Computational Methods in Structural Dynamics and Earthquake Engineering (pp. 3495-3509). ECCOMAS.

[7] Miranda, J. L. (2007). "Optimização em sistemas de processos químicos: generalização de modelos com planeamento e sequenciamento". (Dissertação de Doutoramento, Universidade Técnica de Lisboa).

[8] Tian, S., Fu, H., Xia, J., & Yang, Y. (2020). "A vortex identification method based on local fluid rotation. Physics of Fluids", 32(1), 015104.

[9] Ribeiro, P. A. R. (2002). "Desprendimento de vórtices e controle em esteira de cilindros por simulação numérica direta". (Dissertação de Pós-Graduação, Universidade Federal do Rio Grande do Sul).

[10] Babiano, A., Boffetta, G., Provenzale, A., & Vulpiani, A. (1994). "Chaotic advection in point vortex models and two-dimensional turbulence. Physics of fluids", 6(7), 2465-2474.

[11] Nocedal, J., & Wright, S. (2006). “Numerical Optimization”. Springer Science & Business Media.

[12] Bragança S. M. T. C. (2012), “Optimização da configuração de reforços numa classe de painéis planos considerando efeitos de plasticidade”. (Dissertação de Mestrado, Universidade Técnica de Lisboa).

[13] Figueiredo, J. R. (2020). “Reprocessamento de Rejeitados: uma Análise de Otimização Multicritério como Ferramenta de Gestão Integrada”. (Dissertação de Doutoramento, Faculdade de Engenharia da Universidade do Porto).

[14] Neto, J. H. S. (2017). “Aplicação de programação linear e simulação de eventos discretos para o problema de balanceamento de linhas de montagem de larga escala”. (Dissertação de Doutoramento, Universidade Federal do Pará).

[15] Feliciano, N. M. (2016). “Otimização da exploração de sistemas hídricos reversíveis em cascata”. (Dissertação de Doutoramento, Instituto Superior de Engenharia de Lisboa).

[16] Martinez, J. M., & Santos, S. A. (1995). “Algoritmos computacionais de otimização”. Colóquio Brasileiro de Matemática, Apostilas, 20.

[17] Carvalho, E. C. R. (2014). Solução de problemas de otimização com restrições usando estratégias de penalização adaptativa e um algoritmo do tipo PSO. (Mestrado, PPGMC, Programa de Pós-graduação em Modelagem Computacional, UFJF).

[18] Morais, C. (2002). “Descrição, análise e interpretação de informação quantitativa. Escalas de medida, estatística descritiva e inferência estatística”. Escola Superior de Educação-Instituto Politécnico de Bragança.

[19] De Sousa, A. C. P. (2016). “Metodologia de Otimização Energética da Operação de Sistemas de Abastecimento de Água com Base num Caso de Estudo”. (Dissertação de Mestrado, Faculdade de Engenharia da Universidade do Porto).

[20] Barboza, C. B., & Oliveira, A. R. L. D. (2006). “Planejamento do tratamento por radioterapia através de algoritmos de pontos interiores. Pesquisa Operacional”, 26(1), 1-24.

[21] Ferreira da Silva, M. (2009). “Estratégias de aproximação para a otimização estrutural”. (Dissertação de Mestrado, Universidade Federal de Pernambuco).

[22] Arenales, S. H. D. V., & Salvador, J. A. (2017). “Cálculo numérico”.

[23] Barroso, L. C., Barroso, M. D. A., Campos-Filho, F. F., Carvalho, M. L. B. D., & Maia, M. L. (1987). “Cálculo numérico”. Harbra Ltd., São Paulo.

[24] Santos, F. C. (2002). “Fundamentos de Análise Numérica”. Edições Sílabo, Lisboa.

[25] Ourique, L. E., & Nardi, R. M. (2006). “Técnicas de integração numérica”.

[26] Vargas, D. E. C., Lemonge, A. C. C., Barbosa, H. J. C., & Bernardino, H. S. (2016). “Um algoritmo baseado em evolução diferencial para problemas de otimização estrutural multiobjectivo com restrições”. Revista Internacional de Algoritmos Numéricos para Cálculo Y Diseño En Ingeniería, 32(2), 91-99.

[27] Melo, T. M. (2015). “Otimização com restrições de complementaridade: algoritmos e aplicações”. (Dissertação de Doutoramento, Universidade do Minho).

[28] Coleman, T., Branch, M. A., & Grace, A. (1999). “Optimization toolbox”. For Use with MATLAB. User’s Guide for MATLAB 5, Version 2, Relaese II.

[29] Schittkowski, K. (2009). “An active set strategy for solving optimization problems with up to 200,000,000 nonlinear constraints. Applied Numerical Mathematics”, 59(12), 2999-3007.

[30] Gentil, J. M. P. (2010). “Estudo e implementação de um algoritmo de restrições ativas para problemas de otimização em caixas”. (Dissertação de Doutoramento, Universidade de São Paulo).

[31] De Oliveira, A. R. L. (1991). “Implementação de um algoritmo de pontos interiores para programação linear”. (Dissertação de Mestrado, Universidade Estadual de Campinas).

[32] Faria, C. A. (2019). “Determinação de estados estacionários usando programação quadrática sequencial”. (Trabalho de Conclusão de Curso, Universidade Federal de Uberlândia).

[33] Carmo, A. L. D. (2010). “Técnicas de otimização por aproximação sequencial aplicadas a ajuste de histórico na simulação de reservatórios”. (Dissertação de Mestrado, Universidade Federal de Pernambuco).

[34] Oliveira, P. C. M. (2007). “Modelagem e otimização de fermentadores para obtenção de etanol”. (Dissertação de Doutoramento, Universidade Estadual de Campinas).

Anexo 1

- Tabelas de valores obtidos para o caso N=1 utilizando o algoritmo Pontos interiores.

n=1				n=2			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**	**Nº de ciclos**	**Fval**	**Flag**	**Tempo de execução**
1	0,6766	2	2,10	1	0,3716	2	2,51
20	0,1773	2	26,72	20	0,3715	2	34,17
40	0,1773	2	42,01	40	0,3715	2	70,00
60	0,1773	2	75,25	60	0,3715	2	100,90
80	0,1773	2	103,36	80	0,3715	2	132,23
100	0,1773	2	133,34	100	0,3715	2	179,55

n=3				n=4			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**	**Nº de ciclos**	**Fval**	**Flag**	**Tempo de execução**
1	0,3142	2	3,14	1	0,2926	2	4,8074
20	0,3137	2	51,69	20	0,2914	2	82,059
40	0,3135	2	108,07	40	0,2918	2	156,59
60	0,3135	2	148,95	60	0,2917	2	266,74
80	0,3135	2	225,26	80	0,2914	2	321,615
100	0,3134	2	259,93	100	0,2915	2	427,1716

n=5				n=6			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**	**Nº de ciclos**	**Fval**	**Flag**	**Tempo de execução**
1	0,2823	2	5,483	1	0,2901	2	6,4
20	0,2812	2	152,866	20	0,2752	2	157,697
40	0,2809	2	269,267	40	0,2752	2	233,379
60	0,2808	2	309,1	60	0,2748	2	672,214
80	0,2810	2	585,66	80	0,2750	2	550,988
100	0,2809	2	619,31	100	0,2749	2	731,747

n=7				n=8			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**	**Nº de ciclos**	**Fval**	**Flag**	**Tempo de execução**
1	0,2726	2	5,508	1	0,2784	2	5,674204
20	0,2720	2	214,53	20	0,2690	2	232,1453
40	0,2715	2	498,871	40	0,2692	2	553,915
60	0,2713	2	728,171	60	0,2688	2	736,452
80	0,2713	2	1042,341	80	0,2689	2	1231,158
100	0,2714	2	1338,834	100	0,2689	2	1512,133

n=9				n=10			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**	**Nº de ciclos**	**Fval**	**Flag**	**Tempo de execução**
1	0,6949	2	23,02	1	0,8968	0	82,47995
20	0,2677	2	234,84	20	0,2666	2	894,855
40	0,2676	2	941,22	40	0,2661	2	1065,72
60	0,2671	2	1145,368	60	0,2661	2	1363,258
80	0,2671	2	1423,314	80	0,2661	2	1687,24
100	0,2673	2	1602,578	100	0,2661	2	1987,014

- Tabelas de valores obtidos para o caso N=1 utilizando o algoritmo SQP.

n=1				n=2			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**	**Nº de ciclos**	**Fval**	**Flag**	**Tempo de execução**
1	0,6766	2	0,8852	1	0,3715	-2	1,02
20	0,1771	2	23,737	20	0,3715	-2	26,1289
40	0,1771	2	32,765	40	0,3715	-2	45,923
60	0,1771	2	52,075	60	0,3715	-2	65,457
80	0,1773	-2	81,184	80	0,3715	2	83,484
100	0,1769	-2	87,385	100	0,3715	-2	117,709

n=3				n=4			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**	**Nº de ciclos**	**Fval**	**Flag**	**Tempo de execução**
1	0,3133	-2	1,444	1	0,8176	-2	5,999
20	0,3133	2	57,203	20	0,2913	-2	116,623
40	0,3133	2	136,144	40	0,2913	-2	230,911
60	0,3133	2	134,28	60	0,2913	2	334,204
80	0,3133	-2	213,315	80	0,2913	2	435,255
100	0,3133	-2	317,774	100	0,2913	2	545,748

n=5				n=6			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**	**Nº de ciclos**	**Fval**	**Flag**	**Tempo de execução**
1	0,7754	-2	5,811	1	0,7348	-2	12,524
20	0,2807	-2	108,6964	20	0,2747	-2	251,903
40	0,2807	-2	226,812	40	0,2747	2	503,771
60	0,2807	2	423,742	60	0,2747	-2	801,042
80	0,2807	-2	427,69	80	0,2747	-2	1077,394
100	0,2807	-2	689,05	100	0,2747	2	1404,032

n=7				n=8			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**	**Nº de ciclos**	**Fval**	**Flag**	**Tempo de execução**
1	0,7423	-2	26,791	1	0,7350	-2	24,468
20	0,2711	-2	215,188	20	0,2687	-2	349,949
40	0,2711	-2	454,414	40	0,2687	-2	576,95
60	0,2711	2	872,485	60	0,2687	-2	943,114
80	0,2711	2	1232,269	80	0,2687	-2	1389,098
100	0,2711	2	1237,875	100	0,2687	2	1581,116

n=9				n=10			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**	**Nº de ciclos**	**Fval**	**Flag**	**Tempo de execução**
1	0,7044	-2	28,086	1	0,2659	2	3,529
20	0,2671	2	533,617	20	0,2659	2	813,807
40	0,2671	2	949,874	40	0,2659	2	733,74
60	0,2671	-2	1133,498	60	0,2659	2	1336,88
80	0,2671	-2	2142,560	80	0,2659	-2	1337,71
100	0,2671	-2	2475,480	100	0,2659	2	2586,73

- Tabelas de valores obtidos para o caso N=1 utilizando o algoritmo Active-set.

n=1				n=2			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**	**Nº de ciclos**	**Fval**	**Flag**	**Tempo de execução**
1	0,6766	4	3,417	1	0,3715	5	1,584
20	0,6766	4	14,743	20	0,3715	5	26,208
40	0,6766	4	31,855	40	0,3715	5	61,944
60	0,6766	4	44,551	60	0,3715	5	74,090
80	0,6766	4	54,873	80	0,3715	5	125,420
100	0,6766	4	62,172	100	0,3715	5	148,860

n=3				n=4			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**	**Nº de ciclos**	**Fval**	**Flag**	**Tempo de execução**
1	0,9771	5	7,699	1	0,2913	5	2,991
20	0,3133	4	91,215	20	0,2913	5	98,054
40	0,3133	5	177,766	40	0,2913	5	226,295
60	0,3133	5	246,219	60	0,2913	5	345,669
80	0,3133	5	364,028	80	0,2913	5	501,678
100	0,3133	5	443,030	100	0,2913	5	693,269

n=5				n=6			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**	**Nº de ciclos**	**Fval**	**Flag**	**Tempo de execução**
1	0,7784	5	21,963	1	0,7612	0	42,981
20	0,2807	5	291,082	20	0,2747	5	515,687
40	0,2807	5	586,961	40	0,2747	5	1211,567
60	0,2807	5	867,165	60	0,2547	5	1410,058
80	0,2807	5	1087,643	80	0,2747	5	2043,765
100	0,2807	5	1430,425	100	0,2747	5	3116,900

n=7				n=8			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**	**Nº de ciclos**	**Fval**	**Flag**	**Tempo de execução**
1	0,2711	5	5,855	1	0,6969	0	71,168
20	0,2711	5	849,240	20	0,2687	0	1044,850
40	0,2711	5	1208,950	40	0,2687	5	1239,690
60	0,2711	5	1702,140	60	0,2687	5	2081,255
80	0,2711	0	3760,268	80	0,2687	5	2753,350
100	0,2711	5	3830,503	100	0,2687	5	3956,206

n=9				n=10			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**	**Nº de ciclos**	**Fval**	**Flag**	**Tempo de execução**
1	0,7278	5	44,059	1	0,7085	0	72,115
20	0,2671	5	772,645	20	0,2659	5	1496,521
40	0,2671	0	2108,343	40	0,2659	0	2574,125
60	0,2671	5	2935,039	60	0,2671	5	3438,781
80	0,2671	2	3663,123	80	0,2659	0	4316,751
100	0,2671	0	4215,755	100	0,2659	5	6119,723

Anexo 2

- Tabelas de valores obtidos para o caso N=2 utilizando o algoritmo Pontos Interiores.

n=1				n=2			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**	**Nº de ciclos**	**Fval**	**Flag**	**Tempo de execução**
1	0,8365	2	1,3845	1	2,2140	2	2,818
20	0,0845	2	58,95	20	0,8354	2	48,59
40	0,0791	2	96,94	40	0,2327	2	256,879
60	não converge			60	0,5165	2	365,991
80	não converge			80	0,1725	2	306,19
100	não converge			100	0,1492	2	291,882

n=3				n=4			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**	**Nº de ciclos**	**Fval**	**Flag**	**Tempo de execução**
1	0,5290	2	4,611	1	0,6833	2	5,569
20	0,6362	2	81,873	20	0,1175	2	110,8
40	0,0569	2	363,48	40	0,0733	2	225,21
60	0,1052	2	371,025	60	0,1037	2	357,46
80	0,1497	2	567,148	80	0,0808	2	549,085
100	0,1279	2	647,228	100	0,0907	2	583,297

n=5				n=6			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**	**Nº de ciclos**	**Fval**	**Flag**	**Tempo de execução**
1	0,6609	2	6,443	1	0,6361	2	5,639
20	0,3619	2	131,68	20	0,0847	2	262,301
40	0,0924	2	424,394	40	0,0519	2	417,045
60	0,0970	0	551,566	60	0,0519	-2	795,18
80	0,0612	2	866,099	80	0,0863	2	1000,024
100	0,0945	2	1138,1675	100	0,0597	2	1344,756

n=7				n=8			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**	**Nº de ciclos**	**Fval**	**Flag**	**Tempo de execução**
1	0,6308	2	6,075023	1	0,1805	-2	4,449
20	0,0769	2	261,96	20	0,0642	2	370,82
40	0,0630	2	699,286	40	0,0509	2	895,792
60	0,0525	2	846,28	60	0,0572	2	1321,973
80	0,0542	-2	1049,02	80	0,0494	2	1455,441
100	0,0689	2	2305,146	100	0,0658	2	1358,974

n=9				n=10			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**	**Nº de ciclos**	**Fval**	**Flag**	**Tempo de execução**
1	0,6157	2	23,004	1	0,6145	2	18,9
20	0,0851	2	439,872026	20	0,1121	2	384,735
40	0,0842	2	897,633	40	0,0654	2	868,044
60	0,0545	2	1748,408	60	0,0495	2	1528,191
80	0,0757	2	1785,625	80	0,0623	0	1906,663
100	0,0718	-2	2033,079	100	0,0454	2	2434,348

- Tabelas de valores obtidos para o caso N=2 utilizando o algoritmo SQP.

n=1				n=2			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**	**Nº de ciclos**	**Fval**	**Flag**	**Tempo de execução**
1	0,8364	2	0,710935	1	0,8369	-2	1,685
20	0,3164	2	13,718	20	0,1883	-2	30,689
40	0,5422	2	23,11	40	0,1543	-2	38,809
60	Não converge			60	0,1486	-2	64,717
80	Não converge			80	0,1495	-2	96,178
100	Não converge			100	0,1508	-2	102,485

n=3				n=4			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**	**Nº de ciclos**	**Fval**	**Flag**	**Tempo de execução**
1	0,7309	2	2,615	1	0,6816	2	2,332
20	0,0999	-2	37,387	20	0,0917	2	38,72
40	0,0971	2	63,552	40	0,0676	2	85,011
60	0,0971	-2	85,467	60	0,0626	2	111,902
80	0,0951	2	117,552	80	0,0642	2	161,953
100	0,0953	2	154,529	100	0,0638	2	190,22

n=5				n=6			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**	**Nº de ciclos**	**Fval**	**Flag**	**Tempo de execução**
1	0,6508	2	2,576	1	0,3075	2	6,307
20	0,0509	-2	56,924	20	0,0493	-2	102,333
40	0,0498	-2	92,68	40	0,0465	2	145,7
60	0,0487	2	188,167	60	0,0475	-2	236,585
80	0,0492	-2	245,88	80	0,0450	2	274,707
100	0,0471	2	332,21	100	0,0479	-2	379,525

n=7				n=8			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**	**Nº de ciclos**	**Fval**	**Flag**	**Tempo de execução**
1	0,2888	2	9,297	1	0,6159	2	3,654
20	0,0438	-2	101,928	20	0,0452	2	133,899
40	0,0496	-2	160,091	40	0,0436	2	255,721
60	0,0453	-2	309,682	60	0,0446	2	369,037
80	0,0401	2	457,772	80	0,0433	2	460,691
100	0,0433	2	437,776	100	0,0418	2	549,176

n=9				n=10			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**	**Nº de ciclos**	**Fval**	**Flag**	**Tempo de execução**
1	0,6116	2	9,114	1	0,2676	2	18,032
20	0,0597	2	158,981	20	0,0398	-2	156,053
40	0,0427	2	275,95	40	0,0431	2	339,252
60	0,0412	-2	453,493	60	0,0418	-2	544,543
80	0,0475	2	587,257	80	0,0411	2	726,379
100	0,0420	2	795,594	100	0,0400	2	933,482

- Tabelas de valores obtidos para o caso N=2 utilizando o algoritmo Active-set.

n=1				n=2			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**	**Nº de ciclos**	**Fval**	**Flag**	**Tempo de execução**
1	0,8364	4	1,234	1	1,6538	0	32,815
20	Não converge			20	0,3317	4	73,216
40	Não converge			40	0,1426	4	119,13
60	Não converge			60	0,1448	0	159,121
80	Não converge			80	0,1829	4	194,079
100	Não converge			100	0,1499	4	396,935

n=3				n=4			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**	**Nº de ciclos**	**Fval**	**Flag**	**Tempo de execução**
1	0,7311	5	2,488	1	0,6816	5	2,5224
20	0,1459	5	102,578	20	0,0701	5	68,831
40	0,5261	5	114,717	40	0,0825	5	159,965
60	0,1485	0	143,035	60	0,0680	5	543,28
80	0,0982	5	392,832	80	0,0692	5	496,116
100	0,0976	5	513,927	100	0,0697	5	472,107

n=5				n=6			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**	**Nº de ciclos**	**Fval**	**Flag**	**Tempo de execução**
1	0,0666	0	39,108	1	0,6330	5	5,334
20	0,0589	5	123,253	20	0,0601	5	126,512
40	0,0649	5	214,569	40	0,0549	5	226,818
60	0,0694	5	301,957	60	0,0537	4	500,503
80	0,0585	5	530,134	80	0,0554	5	499,882
100	0,0587	0	667,719	100	0,0555	5	586,757

n=7				n=8			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**	**Nº de ciclos**	**Fval**	**Flag**	**Tempo de execução**
1	0,6225	5	5,802	1	0,0579	0	47,012
20	0,0495	5	153,826	20	0,0509	5	303,862
40	0,0556	5	357,073	40	0,0489	5	416,981
60	0,0520	5	452,071	60	0,0494	5	468,898
80	0,0515	4	571,659	80	0,0458	5	726,204
100	0,0533	5	656,991	100	0,0496	5	1204,347

n=9				n=10			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**	**Nº de ciclos**	**Fval**	**Flag**	**Tempo de execução**
1	0,5712	5	14,879	1	0,6086	5	5,684
20	0,0505	5	172,707	20	0,0503	5	231,124
40	0,0512	5	438,023	40	0,0564	5	456,405
60	0,0579	5	832,874	60	0,0539	5	694,109
80	0,0496	5	893,136	80	0,0520	5	878,246
100	0,0453	5	1566,08	100	0,0472	5	1433,665

Anexo 3

- Tabelas de valores obtidos para o caso N=3 utilizando o algoritmo Pontos Interiores.

n=1				n=2			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**	**Nº de ciclos**	**Fval**	**Flag**	**Tempo de execução**
1	16,966	-2	3,67	1	47,5911	2	17,845
20	4,816	-2	1822,509	20	13,1482	2	2045,645
40	7,492	0	4691,369	40	13,6491	2	4734,836
60	Não Converge			60	Não Converge		
80	Não Converge			80	Não Converge		
100	Não Converge			100	Não Converge		

n=3				n=4			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**	**Nº de ciclos**	**Fval**	**Flag**	**Tempo de execução**
1	16,5637	2	177,303	1	10,6119	2	66,16
20	16,4880	2	432,114	20	10,2026	2	975,996
40	1,4138	2	682,217	40	10,17	2	2207,47
60	16,4802	2	1234,795	60	10,1872	-2	3431,567
80	1,4039	2	2219,815	80	1,24167	2	3874,623
100	16,4751	2	2164,187	100	1,24178	2	5641,667

n=5			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**
1	6,5603	2	21,306
20	6,4924	2	1763,015
40	6,4924	2	2982,283
60	0,8155	0	3930,307
80	0,8024	0	6126,065
100	0,8060	2	3511,724

- Tabelas de valores obtidos para o caso N=3 utilizando o algoritmo SQP.

n=1				n=2			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**	**Nº de ciclos**	**Fval**	**Flag**	**Tempo de execução**
1	76,788	-2	37,337	1	41,586	2	42,374
20	34,527	-2	547,192	20	41,5857	0	636,588
40	46,489	-2	1008,267	40	18,7393	2	1367,433
60	51,353	-2	1750,975	60	4,57307	2	2155,877
80	50,759	-2	2102,001	80	41,5856	0	2861,844
100	40,277	-2	2643,76	100	41,5857	0	3601,53

n=3				n=4			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**	**Nº de ciclos**	**Fval**	**Flag**	**Tempo de execução**
1	15,383	2	50,285	1	9,42707	2	31,91
20	15,383	2	687,999	20	2,14594	2	670,333
40	15,383	2	1055,754	40	1,18373	2	1429,865
60	15,383	2	1257,469	60	9,42703	2	2195,844
80	15,383	2	1870,483	80	1,18373	2	2925,044
100	1,3302	2	3701,873	100	2,0644	2	3234,976

n=5			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**
1	6,059	2	27,372
20	6,059	2	553,822
40	6,059	2	1107,787
60	0,7624	2	1640,409
80	6,059	2	2108,106
100	6,059	2	2741,357

- Tabelas de valores obtidos para o caso N=3 utilizando o algoritmo Active-set.

n=1				n=2			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**	**Nº de ciclos**	**Fval**	**Flag**	**Tempo de execução**
1	298,345	4	31,469	1	175,008	5	13,767
20	45,7453	-2	2085,314	20	35,5841	0	690,859
40	37,457	4	3509,339	40	41,586	4	1459,231
60	43,9335	0	5323,295	60	28,16	5	1869,534
80	43,9123	0	6617,587	80	10,7447	0	2979,773
100	43,9094	0	8627,334	100	29,869	5	3310,805

n=3				n=4			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**	**Nº de ciclos**	**Fval**	**Flag**	**Tempo de execução**
1	15,4028	5	25,437	1	9,4271	5	25,888
20	15,3831	5	602,433	20	9,4271	5	641,915
40	15,3831	5	803,377	40	9,4271	5	1270,503
60	15,3831	5	1141,355	60	1,1837	5	1851,596
80	1,33016	5	1455,191	80	9,4270	5	2369,072
100	1,33016	5	1950,325	100	9,4270	5	2808,443

n=5			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**
1	6,0591	5	15,088
20	6,0590	5	627,607
40	6,0590	5	1329,116
60	0,7635	5	1799,322
80	6,0590	5	2758,576
100	6,0590	5	3250,614

Anexo 4

- Tabelas de valores obtidos para o caso N=4 utilizando o algoritmo Pontos Interiores.

n=1				n=2			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**	**Nº de ciclos**	**Fval**	**Flag**	**Tempo de execução**
1	177,039	0	114,316	1	277,979	2	127,190
20	6,444	-2	367,053	20	29,176	-2	1789,603
40	6,444	-2	932,450	40	11,359	0	4119,174
60	6,444	-2	978,872	60	11,378	-2	5776,759
80	6,444	1	2397,663	80	11,386	2	14858,224
100	6,444	0	2309,245	100	8,037	0	20976,341

n=3				n=4			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**	**Nº de ciclos**	**Fval**	**Flag**	**Tempo de execução**
1	338,842	0	171,505	1	331,534	0	211,139
20	54,393	0	2715,200	20	44,643	0	2515,532
40	5,039	0	5114,002	40	48,731	0	5447,546
60	53,660	0	7723,613	60	9,093	0	8024,732
80	17,946	0	8480,939	80	45,516	0	10446,008
100	16,375	-2	30657,617	100	3,898	0	13578,112

n=5			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**
1	301,958	2	78,784
20	9,337	0	3071,523
40	11,551	0	6912,612
60	5,016	0	8471,084
80	2,585	2	13604,246
100	26,886	0	15893,793

- Tabelas de valores obtidos para o caso N=4 utilizando o algoritmo SQP

n=1				n=2			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**	**Nº de ciclos**	**Fval**	**Flag**	**Tempo de execução**
1	174,422	2	26,543	1	78,273	0	58,222
20	3,977	-2	261,613	20	7,923	2	505,183
40	4,469	-2	685,503	40	5,047	2	1053,584
60	4,463	-2	1232,641	60	3,924	2	1445,430
80	6,444	-2	2374,179	80	7,923	0	2900,162
100	4,401	-2	1796,594	100	7,923	2	5469,919

n=3				n=4			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**	**Nº de ciclos**	**Fval**	**Flag**	**Tempo de execução**
1	8,918	2	42,394	1	59,203	0	99,335
20	4,910	2	826,746	20	8,166	0	1092,731
40	4,910	2	1804,817	40	9,177	0	2780,240
60	5,392	2	2546,387	60	3,319	0	3210,494
80	44,882	-2	3654,200	80	3,319	0	3770,976
100	4,910	-2	3870,344	100	3,319	-2	6272,575

n=5			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**
1	68,777	0	137,842
20	37,296	-2	1289,596
40	4,928	-2	2532,140
60	7,026	0	3174,102
80	2,535	2	5356,562
100	2,459	2	5300,113

- Tabelas de valores obtidos para o caso N=4 utilizando o algoritmo Active-set

n=1				n=2			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**	**Nº de ciclos**	**Fval**	**Flag**	**Tempo de execução**
1	163,567	0	146,749	1	244,145	0	131,581
20	61,743	0	1750,249	20	65,226	5	779,447
40	25,156	-2	4011,078	40	65,240	4	2192,725
60	26,041	0	6009,393	60	27,772	0	3407,547
80	40,510	-2	8274,427	80	13,176	4	5389,120
100	15,951	-2	11984,393	100	7,923	-2	7201,007

n=3				n=4			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**	**Nº de ciclos**	**Fval**	**Flag**	**Tempo de execução**
1	184,907	0	176,509	1	51,056	0	89,055
20	52,134	0	1145,065	20	48,301	-2	1077,013
40	52,948	0	3507,197	40	34,702	0	3166,774
60	4,910	0	4447,328	60	3,319	4	4749,591
80	4,910	0	5295,080	80	14,151	0	6995,177
100	4,910	0	8272,867	100	9,671	5	8055,259

n=5			
Nº de ciclos	**Fval**	**Flag**	**Tempo de execução**
1	80,755	0	128,160
20	42,756	0	2114,227
40	25,187	4	4584,867
60	2,459	0	6285,245
80	34,975	0	8862,025
100	2,553	0	10080,028