Mariana Almeida Loreto

Funções de Otimização e Computação em Paralelo

UNIVERSIDADE D COIMBRA 1 2 9 0

1 2 9 0

UNIVERSIDADE D
COIMBRA

Mariana Almeida Loreto

Funções de Otimização e Computação em Paralelo

Dissertação no âmbito do Mestrado em Engenharia Informática, especialização em Engenharia de Software, orientada pelo Professor Doutor Fernando Penousal Machado e Engenheiro Hugo Amaro apresentada ao Departamento de Engenharia Informática da Faculdade de Ciências e Tecnologia da Universidade de Coimbra.

Setembro de 2023

FACULDADE DE
CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE D
COIMBRA

DEPARTMENT OF INFORMATICS ENGINEERING

Mariana Almeida Loreto

Optimisation Functions and Parallel Computing

Dissertation in the context of the Master in Informatics Engineering, specialization in Software Engineering, advised by Prof. Fernando Penousal Machado and Hugo Amaro and presented to the Department of Informatics Engineering of the Faculty of Sciences and Technology of the University of Coimbra.

Setembro 2023

FACULDADE DE
CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE D
COIMBRA

DEPARTAMENTO DE ENGENHARIA INFORMÁTICA

Mariana Almeida Loreto

Funções de Otimização e Computação em Paralelo

Dissertação no âmbito do Mestrado em Engenharia Informática, especialização em Engenharia de Software, orientada pelo Professor Doutor Fernando Penousal Machado e Engenheiro Hugo Amaro apresentada ao Departamento de Engenharia Informática da Faculdade de Ciências e Tecnologia da Universidade de Coimbra.

Setembro 2023

Agradecimentos

Gostava de agradecer a todos aqueles que, de forma direta ou indireta, me apoiaram durante as várias fases da elaboração desta tese. Fico grata ao Engenheiro Hugo Amaro pelo o apoio extraordinário que sempre me disponibilizou, por me ter incentivado ao longo deste processo, bem como pelas piadas ocasionais durante a escrita desta tese. Quero também agradecer ao Professor Fernando Penousal Machado, meu orientador, por me incentivar a procurar novos caminhos e explorar mais opções. Aos meus colegas do LIS, por me receberem e disponibilizarem para ajudar, especialmente ao João por me apoiado, literalmente, desde o primeiro dia. Ao André por acreditar em mim e por me abraçar quando o desespero era demais. Aos meus amigos mestres, João, Mariana e Rita, por me incentivarem a continuar e inspirarem a ser melhor. Aos meus amigos de faculdade por compreenderem o que é escrever uma tese, Bia, Diogo, Duarte, Gabriel, Gonçalo, João e Pedro. À maravilhosa Nês por ser paciente, e mesmo longe, se ter preocupado com o meu bem-estar. Ao Abdellahi, António, Bruno, João e Laura, simplesmente por existirem. Aos meus afilhados, Bernardo, Duarte, Marta e Tatiana, por todo o seu carinho e por acreditarem na pessoa que gostava de ser. Ao Carnide, por todo o apoio e motivação que me deu ao longo deste ano. Um obrigado também à Mariana Cunha, por me ter adotado no meu primeiro ano de faculdade. E um agradecimento especial à Harriet pelas dicas de escrita que me ofereceu de bom grado. Finalmente, agradeço à minha família, principalmente àqueles que me ligam constantemente para sondar se já sou mestre. Está quase!

Abstract

The mortality rates associated with the high incidence of cancer in Portugal highlight the urgent need for effective cancer treatments, such as radiotherapy. Intensity modulated radiotherapy (IMRT) is a widely used form of external radiotherapy that allows precise delivery of radiation to a tumour, while minimising the dose delivered to surrounding healthy tissue. One of the crucial aspects of IMRT is the treatment planning process, which involves setting up a set of parameters to determine the alignment, beams and intensity of radiation to be delivered to the patient.

During this process, it is essential to optimize the treatment for the individual patient. This includes finding a configuration that irradiates the target volume as defined in the prescription and minimizes the dose absorbed by healthy organs and tissues. The process of creating treatment plans can be time-consuming, particularly when dealing with the beam angle optimization (BAO) and fluence map optimization (FMO) problems.

Recognizing the complexity of the fluence map optimization problem, it is important to study and combine a multifaceted solution that intertwines the various dimensions of the problem, considering both memory and processing efficiency.

Thus, there is a clear need to reduce the amount of data in the problem, using memory management strategies, such as sparse matrices, in order to save storage space and facilitate complex calculations.

Furthermore, it is relevant to note that fluence map optimization has been carried out using serial algorithms, which can be slow and time-consuming. This leads to a growing interest in using computer graphics and optimization techniques to accelerate the optimization process. In particular, the CUDA programming model emerges as a powerful platform for performing computation on GPUs.

This exploration also extends to the selection of appropriate optimization algorithms to solve this problem, particularly the use of quadratic programming algorithms, together with a knowledge of the intrinsic characteristics of FMO.

This report presents the research and work developed, regarding the use of computer graphics and parallel optimization techniques to accelerate the fluence map optimization problem in IMRT in the context of the ORION project, during the internship in the Master in Computer Engineering at the Faculty of Science and Technology of the University of Coimbra.

Keywords

Radiotherapy, FMO, CPU, GPU, Computer Graphics, CUDA, Sparse Matrix, Quadratic Optimization

Resumo

A elevada incidência do cancro e as taxas de mortalidade que lhe estão associadas em Portugal realçam a necessidade urgente de tratamentos oncológicos eficazes, tais como a radioterapia. A radioterapia modulada por intensidade (IMRT) é uma forma amplamente utilizada de radioterapia externa que permite a entrega precisa de radiação a um tumor, minimizando ao mesmo tempo a dose aplicada ao tecido saudável circundante. Um dos aspetos cruciais da IMRT é o processo de planeamento do tratamento, que envolve a configuração de um conjunto de parâmetros para determinar o alinhamento, os feixes e a intensidade da radiação a ser administrada ao paciente.

Durante este processo é essencial otimizar o tratamento para o paciente individual. Isto inclui encontrar uma configuração que irradie o volume alvo tal como definido na prescrição médica e minimize a dose absorvida por órgãos e tecidos saudáveis. O processo de criação de planos de tratamento pode ser demorado, particularmente quando se trata do problema da otimização dos ângulos dos feixes (BAO) e da otimização dos mapas de fluência (FMO).

Reconhecendo a complexidade do problema de otimização dos mapas de fluência, é importante estudar e conjugar uma solução multifacetada que entrelace as várias dimensões do problema, considerando tanto a eficiência de memória como de processamento.

Desta forma, surge uma necessidade clara de reduzir a dimensão dos dados do problema, utilizando estratégias de gestão de memória, como matrizes esparsas, de forma a economizar o espaço de armazenamento e facilitar cálculos complexos.

Além disto, é relevante notar que a otimização dos mapas de fluência tem sido realizada utilizando algoritmos em série, que podem ser lentos e demorados. O que leva a um interesse crescente na utilização de computação gráfica e técnicas de otimização paralela para acelerar o processo de otimização. Em particular, o modelo de programação CUDA surge como uma plataforma poderosa para executar computação em GPUs.

Esta exploração estende-se também à seleção de algoritmos de otimização apropriados para a resolução deste problema, particularmente o uso de algoritmos de programação quadrática, juntamente com um conhecimento das caraterísticas intrínsecas ao FMO.

Este relatório apresenta a investigação e trabalho desenvolvido, relativamente à utilização de computação paralela e técnicas de otimização para acelerar a otimização de mapas de fluência na IMRT no contexto do projeto ORION, durante o estágio no Mestrado em Engenharia Informática da Faculdade de Ciências e Tecnologia da Universidade de Coimbra.

Palavras-Chave

Radioterapia, FMO, CPU, GPU, Computação Gráfica, CUDA, Matriz Esparsa, Otimização Quadrática

Conteúdo

1 Introdução **1**
- 1.1 Entidade Acolhedora . . . 2
- 1.2 Enquadramento do Projeto . . . 3
- 1.3 Objetivos . . . 4
- 1.4 Estrutura do Documento . . . 5
- 1.5 Planeamento e Requisitos . . . 5

2 Contextualização do Projeto **7**
- 2.1 Processo Radioterapêutico . . . 7
  - 2.1.1 Sequência do Tratamento . . . 8
  - 2.1.2 Módulo de Geração Automática de Planos . . . 11

3 Fundamentos **13**
- 3.1 Problema BAO . . . 13
- 3.2 Problema FMO . . . 14
  - 3.2.1 Modelos Lineares . . . 15
  - 3.2.2 Modelos Lineares Inteiros Mistos . . . 16
  - 3.2.3 Modelos Não Lineares . . . 17
  - 3.2.4 Comparação dos modelos . . . 17
- 3.3 Qualidade do Tratamento . . . 18

4 Estratégias de Implementação **21**
- 4.1 Otimização usando a GPU . . . 21
  - 4.1.1 Contexto histórico . . . 21
  - 4.1.2 CUDA . . . 23
- 4.2 Matriz esparsa . . . 25
- 4.3 Otimização Quadrática . . . 27
  - 4.3.1 Métodos de Otimização Quadrática . . . 27
  - 4.3.2 IPOPT e Método de Pontos Interiores Primal-Dual . . . 28
  - 4.3.3 Fmincon e Método de Região de Confiança Refletivo . . . 29
  - 4.3.4 Resumo IPOPT vs Fmincon . . . 30

5 Experimentação e Resultados **33**
- 5.1 Considerações Inicias . . . 33
  - 5.1.1 Ambiente de Testes . . . 33
  - 5.1.2 Condições de Testagem . . . 35
- 5.2 Matrizes Esparsas . . . 36
- 5.3 Comparação Algoritmos . . . 37
- 5.4 Otimização usando a GPU . . . 40

5.5 Otimização do Acesso da Matriz de Dose 43
5.6 Variação do Ponto Inicial 45

6 Conclusão **55**

Apêndice A Requisitos **65**
A.1 Requisitos Funcionais 65
A.2 Requisitos Não Funcionais 66
A.3 Cumprimento Requisitos 66
A.3.1 Cumprimento Requisitos Funcionais 67
A.3.2 Cumprimento Requisitos Não Funcionais 67

Apêndice B Planeamento **69**

Acrónimos

ALU Arithmetic-Logic Unit.

API Application Programming Interface.

BAO Beam Angle Optimization.

CPU Central Processor Unit.

CTV Clinical Target (or Tumor) Volume.

CUDA Compute Unified Device Architecture.

DICOM Digital Imaging and Communications in Medicine.

DRAM Dynamic Random-Access Memory.

DVH Dose-Volume Histogram.

FMO Fluence Map Optimization.

GPU Graphical Processor Unit.

IMRT Intensity-Modulated Radiation Therapy.

IPN Instituto Pedro Nunes.

IPOPT Interior Point OPTimizer.

linac Linear Accelarator.

LIS Laboratórios de Informática e Sistemas.

LP Linear Programming.

MIP Mixed Integer Programming.

MLC Multileaf Collimators.

NT Normal Tissue.

OAR Organs at Risk.

ORION Optimised Remote radIOtherapy plaNning.

PCIe Peripheral Component Interconnect Express.

PTV Planning Target (or Tumor) Volume.

ROI Region Of Interest.

TAC Tomografia Axial Computorizada.

TPS Treatment Planning System.

UC Universidade de Coimbra.

VOI Volume Of Interest.

Lista de Figuras

2.1 Exemplo TAC da pélvis e conjunto de fatias que a compõem. 9
2.2 Exemplo do delineamento dos volumes de uma TAC da pélvis. . . 9
2.3 Acelerador Linear utilizado na Radioterapia Externa. Retirado de [36] 10
2.4 Esquema da arquitetura do projeto ORION 12

3.1 Histograma Dose-Volume (DVH) de um plano relativo à "cabeça e pescoço". 18

4.1 Comparação arquitetura CPU e GPU. Adaptado de [50]. 22
4.2 Arquitetura CUDA. Adaptado de [7]. 23
4.3 Um exemplo da organização de uma grelha em CUDA. Adaptado de [31]. 24
4.4 Representação visual simplificada da matriz de dose. 25
4.5 Representação formato CSC para matrizes esparsas. Adaptada de [22] 26

5.1 (a) PCIe Riser x1 para x16 real da máquina 1 e (b) PCIe Riser x1 para x16 35
5.2 Comparação DVHs produzidos utilizando IPOPT e fmincon para o ficheiro "Cabeça e Pescoço", onde — indica volumes alvo e - - - indica OARs e NTs. 39
5.3 Comparação da cobertura alvo obtida utilizando IPOPT e fmincon para o ficheiro "Cabeça e Pescoço". Nota: as doses desejadas para CTV63, GTV, PTV63 e PTV70 são 63 Gy, 70Gy, 63 Gy e 70 Gy, respetivamente. 39
5.4 Comparação do tempo de execução na CPU, GPU e GPU Teórico, para os ficheiros "Cabeça e Pescoço" e "Pulmões" usando a máquina 1. 43
5.5 Comparação da dispersão da dose para o tronco cerebral, resultado de variação do vetor de pesos inicial para quatro janelas distintas, para o ficheiro "Cabeça e Pescoço" usando o IPOPT na máquina 1. 48
5.6 Comparação da dispersão da dose para a Laringe, resultado de variação do vetor de pesos inicial para quatro janelas distintas, para o ficheiro "Cabeça e Pescoço" usando o IPOPT na máquina 1. . . . 48
5.7 Comparação da dispersão da dose para o CTV63, resultado de variação do vetor de pesos inicial para quatro janelas distintas, para o ficheiro "Cabeça e Pescoço" usando o IPOPT na máquina 1. . . . 49

5.8 Comparação da dispersão da dose para o PTV70, resultado de variação do vetor de pesos inicial para quatro janelas distintas, para o ficheiro "Cabeça e Pescoço" usando o IPOPT na máquina 1. . . . 49
5.9 Comparação da dispersão da dose para o coração, resultado de variação do vetor de pesos inicial para quatro janelas distintas, para o ficheiro "Pulmões" usando o IPOPT na máquina 1. 50
5.10 Comparação da dispersão da dose para o pulmão esquerdo, resultado de variação do vetor de pesos inicial para quatro janelas distintas, para o ficheiro "Pulmões" usando o IPOPT na máquina 1. 51
5.11 Comparação da dispersão da dose para o ITV, resultado de variação do vetor de pesos inicial para quatro janelas distintas, para o ficheiro "Pulmões" usando o IPOPT na máquina 1. 51
5.12 Comparação da dispersão da dose para o PTV, resultado de variação do vetor de pesos inicial para quatro janelas distintas, para o ficheiro "Pulmões" usando o IPOPT na máquina 1. 52
5.13 Comparação da dispersão da dose para o tronco cerebral, resultado de variação do vetor de pesos inicial para quatro janelas distintas, para o ficheiro "Cabeça e Pescoço" usando o fmincon na máquina 1. 52
5.14 Comparação da dispersão da dose para a Laringe, resultado de variação do vetor de pesos inicial para quatro janelas distintas, para o ficheiro "Cabeça e Pescoço" usando o fmincon na máquina 1. . . 53
5.15 Comparação da dispersão da dose para o CTV63, resultado de variação do vetor de pesos inicial para quatro janelas distintas, para o ficheiro "Cabeça e Pescoço" usando o fmincon na máquina 1. . . 53
5.16 Comparação da dispersão da dose para o PTV70, resultado de variação do vetor de pesos inicial para quatro janelas distintas, para o ficheiro "Cabeça e Pescoço" usando o fmincon na máquina 1. . . 54

B.1 Diagrama de Gantt 1º semestre 69
B.2 Diagrama de Gantt estimado do 2º semestre 70
B.3 Diagrama de Gantt real do 2º semestre 70

Lista de Tabelas

5.1 Especificação das caraterísticas das máquinas de teste 35
5.2 Caraterísticas da matriz de dose para cada caso de teste 36
5.3 Comparação da representação da matriz de dose original e esparsa para cada caso de teste 37
5.4 Comparação do tempo de execução para IPOPT e fmincon, usando a máquina 1 38
5.5 Cobertura alvo (D_{95}) obtida com o IPOPT e *fmincon* para estruturas de volume alvo presentes no ficheiro "Cabeça e Pescoço" 40
5.6 Tempo de execução, em segundos, da versão da multiplicação matriz-vetor na GPU para o ficheiro "Cabeça e Pescoço", usado a máquina 1 (RTX 3080) e a máquina 2 (Quadro P600). 40
5.7 Comparação do tempo de execução entre a versão na CPU e na GPU, para o ficheiro "Cabeça e Pescoço" usado a máquina 1 e a máquina 2. 41
5.8 Tempo de execução, em segundos, da versão da multiplicação matriz-vetor na GPU para os ficheiros "Cabeça e Pescoço" e "Pulmões", usado a máquina 1 (RTX 3080). 42
5.9 Tempo de execução na CPU, GPU e GPU Teórico, para os ficheiros "Cabeça e Pescoço" e "Pulmões" usando a máquina 1. 42
5.10 Tabela gerada pelo *profiler* do MATLAB na análise das linhas que demoraram mais tempo a ser executadas, usando o ficheiro "Cabeça e Pescoço" 44
5.11 Comparação dos tempos de execução da versão do fmincon original e modificada, usando a máquina 1 45
5.12 Comparação dos tempos de execução entre o IPOPT e o fmincon modificada, usando a máquina 1 45
5.13 Tempos de execução, em segundos, para o IPOPT e o fmincon modificado para diferentes tamanhos da janela do peso inicial do subfeixe, usando a máquina 1 46
5.14 Tempos de execução, em segundos, para o ficheiro "Pulmões" para diferentes tamanhos da janela do vetor de pesos inicial, usando o IPOPT na máquina 1 47

A.1 Requisito funcional #1 65
A.2 Requisito funcional #2 66
A.3 Requisito funcional #3 66
A.4 Cumprimento Requisitos Funcionais 67
A.5 Cumprimento Requisitos Não Funcionais 68

Capítulo 1

Introdução

Em Portugal, o cancro continua a representar uma das maiores causas de morte por número total de óbitos, superado apenas por doenças do sistema circulatório, como a insuficiência cardíaca. No ano de 2019, atingiu um valor de 25,5% do valor total de óbitos [42]. Este número reforça a necessidade urgente do investimento em tratamentos que combatam o desenvolvimento de tumores malignos e que sejam capazes de destruir as células cancerígenas, como o caso da radioterapia. Esta ferramenta de combate ao cancro desempenha um papel fundamental na área da oncologia, sendo um tratamento local que utiliza radiação para destruir as células cancerígenas [6]. Efetivamente, é estimado que esta especialidade terapêutica é utilizada em mais de 50% dos doentes oncológicos na Europa [35]. Esta técnica pode ser administrada internamente, designada por braquiterapia, com auxílio de um implante interno que emite radiação, ou externamente através de equipamento externo. Este trabalho vai se debruçar sobre a segunda técnica de radioterapia, particularmente na técnica de Radioterapia Modelada por Intensidade (IMRT).

Neste contexto, um dos processos mais significativos no tratamento por radioterapia é a construção de planos de tratamento, também designados por dosimetrias. Durante este processo é essencial configurar um conjunto de parâmetros que irão estabelecer como será aplicado o tratamento, incluindo o alinhamento do doente perante o equipamento emissor de radiação, o conjunto de feixes e respetiva direção e intensidade, que irão ser emitidos em direção ao volume alvo delineado pelo radioterapeuta. Pode-se resumir os objetivos da radioterapia da seguinte forma: (1) encontrar uma configuração que permita irradiar o volume a tratar de acordo com o definido na prescrição médica e (2) simultaneamente, minimizar a dose absorvida pelos órgãos e tecidos saudáveis [5, 20]. O plano de tratamento é por norma realizado por um dosimetrista, e depois aprovado por um radioterapeuta. Este processo tem vindo a ser automatizado de forma a libertar o dosimetrista de tarefas lentas e fastidiosas.

Atualmente, a construção de planos de tratamento é feita com recurso a *software* especializado (*Treatment Planning System* ou TPS), que permite a obtenção de vários planos consoante a configuração dos parâmetros necessários, como aqueles referidos acima [14]. Contudo, dependendo da sua complexidade, o processo de criação de planos pode ser muito moroso. Em casos mais simples, o período de

construção de planos pode atingir três horas, e em casos mais complexos, dias.

Para além da duração do tratamento, é essencial considerar outros fatores que contribuem para a qualidade do tratamento, podendo variar de paciente para paciente. Entre estes deve-se considerar, a forma como o problema da otimização dos ângulos dos feixes - conhecido como problema BAO - e da otimização dos mapas de fluência - problema FMO - são abordados e resolvidos durante o planeamento [44].

Dada a natureza complexa do problema de otimização dos mapas de fluência, torna-se essencial realizar um estudar profundo, de forma a conjugar uma solução abrangente que entrelace as várias dimensões do problema, levando em consideração tanto a eficiência de memória como de processamento.

Desse modo, surge uma necessidade evidente de reduzir a dimensão do problema utilizando estratégias de gestão de memória. É natural, as matrizes de dose conterem milhões de voxeis (unidades de volume), resultando em matrizes com centenas de GBs. Estratégias como o uso de matrizes esparsas destacam-se de modo a economizar espaço de armazenamento e simplificar cálculos complexos [25].

Por outro lado, é importante notar que a otimização dos mapas de fluência tem sido tradicionalmente realizada por meios que não tiram partido de paralelismo, que, para a quantidade astronómica de valores presentes na obtenção destes mapas, são lentos e podem levar horas ou até dias para serem concluídos. Contudo, nos últimos anos, tem havido um interesse crescente na utilização de computação paralela com recurso a GPUs e técnicas de otimização paralela para acelerar o processo de otimização. Em particular, o modelo de programação CUDA surge como uma plataforma eficaz para realizar computação em GPUs [47].

Esta exploração estende-se também à seleção de algoritmos de otimização apropriados para a resolução deste problema, particularmente o uso de algoritmos de programação quadrática, juntamente com um conhecimento das caraterísticas intrínsecas ao FMO. Destes, destacam-se o Método dos Pontos Interiores, implementado pelo IPOPT, e o algoritmo de Região de Confiança Refletivo, integrante do fmincon do MATLAB.

Nesta tese, foi investigado o uso matrizes esparsas, de computação gráfica e técnicas de transferência de dados, otimização paralela, juntamente com algoritmos de otimização quadráticos, e outras técnicas para acelerar a otimização dos mapas de fluência do módulo de Geração Automática de Planos do projeto ORION.

1.1 Entidade Acolhedora

Este trabalho está a ser desenvolvido sob tutoria e acompanhamento do Instituto Pedro Nunes (IPN) de Coimbra. Este instituto foi criado pela Universidade de Coimbra (UC) em 1991 com o intuito de promover a inovação e transferência de tecnologia, aproximando a comunidade científica e tecnológica do setor produtivo [38].

Atualmente é uma instituição privada sem fins lucrativos e a sua missão abrange três grandes objetivos, sendo eles: (1) a investigação e desenvolvimento tecnológico; (2) a incubação e (3) aceleração de empresas de base tecnológica e a formação especializada.

Nesta vertente de investigação encontra-se o Laboratório de Informática e Sistemas (LIS ou IPNlis) e tem como foco a participação em projetos de Investigação e Desenvolvimento, e a criação de software a pedido para entidades ligadas à área da informática sejam públicas ou privadas. Este atua igualmente sobre as áreas de Engenharia de Software, Inteligência Artificial, Cibersegurança e privacidade, e Infraestruturas de comunicação e serviço.

1.2 Enquadramento do Projeto

O projeto ORION (Optimised Remote radIOtherapy plaNning) integra-se num consórcio entre a Mercurius Health, o IPN e a (UC) e nasceu com o propósito de colmatar vários problemas associados à radioterapia. Entre eles, destacam-se a falta de profissionais especializados em certas zonas geográficas, o elevado custo das licenças dos TPSs e a lentidão da construção de alguns planos de tratamento. A colmatação destas dificuldades tem como meta o desenvolvimento de uma plataforma de gestão e criação de planos e pedidos de tratamento radioterapêutico baseado num modelo de *Gig Economy*, permitindo aos dosimetristas trabalharem por pedidos e consecutivamente trabalhar, ainda que de forma indireta, para qualquer centro de radioterapia.

A Mercurius Health é líder em serviços de consultoria em Física Médica e Proteção Radiológica. Na tentativa de resolver alguns desafios associados à radioterapia, particularmente o seu acesso a países não desenvolvidos, passou a oferecer aos seus clientes serviços de construção de planos de tratamento remotos ou *e-planning*. Esta entidade assoma, portanto, como líder deste projeto.

O IPN surge neste consórcio como parceiro tecnológico capaz de trazer para o projeto o conhecimento científico, a experiência na implementação de projetos de I&D e a aptidão de transferência de conhecimento e tecnologia necessárias à execução do projeto.

A Universidade de Coimbra entra no consórcio como portador de conhecimento científico relacionado com a otimização e automatização do planeamento de tratamentos de radioterapia, bem como uma longa experiência em aplicações de Visão por Computador, formando uma equipa diversificada composta pela Professora Joana Matos Dias, pelo Professor Humberto Rocha e pelo Professor Hélder Araújo.

Este projeto teve um investimento total de 1.2 milhões de euros, sendo financiado pela União Europeia, com uma duração prevista para o mesmo de 33 meses. Devido à sua complexidade, o projeto ORION foi divido em vários módulos, distribuídos por diferentes equipas de desenvolvimento. Englobando portanto a investigação e desenvolvimento do componente de geração automática de pla-

nos de tratamento e sequencialmente o desenvolvimento do *software* ao qual esse componente será alojado com todos os mecanismos de preparação, gestão, criação e pagamento de dosimetrias.

O presente estágio recai sobre o estudo do módulo de Geração Automática de Planos de tratamento, desde a compreensão das suas limitações em termos de performance à otimização do mesmo, via funções de otimização e computação paralela. Assim, este relatório visa ilustrar o percurso e métodos utilizados durante este estágio académico.

1.3 Objetivos

O objetivo principal deste estágio curricular é a otimização da performance do módulo de Geração Automática, que calcula os mapas de fluência ótimos para planos de tratamento. Em particular, ir-se-á focar na otimização da função de cálculo do vetor de fluência ótimo. Este módulo foi desenvolvido em MATLAB e está inserido no projeto ORION.

Com isso em consideração, é possível dividir o problema em objetivos mais pequenos:

- Estudo da criação automática de planos de tratamento:
  - Enquadrar projeto no processo de radioterapia;
  - Estudar problema de otimização dos mapas de fluência;
  - Identificar restrições no processo e possíveis soluções.
- Otimização da função de cálculo de soluções utilizando a GPU:
  - Fazer um levantamento de vantagens e desvantagens para esta abordagem;
  - Implementar função de cálculo do vetor ótimo na GPU;
  - Testar e comparar a performance com diferentes abordagens da gestão de transferências de dados e de computação de soluções;
  - Generalizar algoritmo para GPUs com diferentes características de memória e unidades de computação, e para múltiplas GPUs.
- Refinamento do algoritmo de otimização:
  - Analisar algoritmo quadrático usado para este problema;
  - Selecionar melhor algoritmo, experimentar, testar e validar.

Como será discutido ao longo desta dissertação, os objetivos delineados inicialmente foram alcançados com as adaptações e extensões necessárias. Aqui faremos apenas um resumo das contribuições alcançadas nesta tese. Assim, iniciou-se o estudo do problema associado à criação de mapas de fluência de forma a

identificar as restrições associadas ao módulo de Geração Automática de Planos enquadrado no projeto ORION. Este estudo incluiu a compreensão do fluxo de dados do módulo de geração, bem como as caraterísticas das estruturas de dados geradas. A partir desse ponto, identificaram-se o tamanho da matriz de dose e o número de multiplicações matriz-vetor, como duas das restrições mais marcantes do módulo. Esta descoberta, permitiu eleger como possíveis soluções a utilização de matrizes esparsas e computação gráfica com auxílio de CUDA. Após estas implementações, também se garantiu que o módulo de Geração Automática de Planos era compatível com diferentes máquinas e gráficas. Explorou-se inclusive, as diferenças de dois algoritmos de otimização quadrática, o método de pontos interiores primal-dual implementado pelo IPOPT e o método de região de confiança refletivo implementado pelo fmincon em MATLAB. Abordou-se ainda como a inserção de variabilidade do vetor de pesos inicial afeta o tempo de execução e a qualidade de um plano de tratamento. Finalmente, procedeu-se a uma minuciosa fase de testes e validação das soluções propostas, assegurando assim a eficácia do módulo de geração.

1.4 Estrutura do Documento

O presente capítulo serve como introdução ao projeto, fazendo um enquadramento do mesmo e uma apresentação dos objetivos. No capítulo 2 faz-se uma apresentação do contexto do problema, em concreto uma introdução à radioterapia e ao processo de criação de dosimetrias. No capítulo 3 é feita uma visão geral sobre o problema dos mapas de fluência (FMO) e modelos de programação abordados na literatura. No capítulo 4 são descritas estratégias de implementação para formatos de matrizes esparsas, técnicas de computação paralela e funções de otimização. No capítulo 5 são expostas as experiências e os resultados obtidos com base nas estratégias de implementação apresentadas. Por fim, é feita uma breve conclusão no capítulo 6.

1.5 Planeamento e Requisitos

De forma a cumprir os objetivos apresentados, foi elaborado um planeamento do projeto, onde se definiu as tarefas que se consideraram essenciais para os atingir, juntamente com o tempo estimado necessário. De forma a facilitar a leitura do presente documento, tornando-a mais fluída, o planeamento foi colocado em apêndice, B. Do mesmo modo, foram definidos os requisitos funcionais e não funcionais intrínsecos ao projeto, e colocados no apêndice A.

Capítulo 2

Contextualização do Projeto

Este capítulo serve como contextualização deste projeto, apresentado vários conceitos associados à radioterapia, fases do processo radioterapêutico e identificando onde o módulo de Geração Automática de Planos se insere no projeto ORION.

2.1 Processo Radioterapêutico

A radioterapia é uma técnica terapêutica largamente usada para o combate a doenças oncológicas, caraterizadas por um crescimento anómalo e desgovernado das células, estimando-se que mais de 50% dos doentes oncológicos utilizam esta técnica [35].

A estratégia da radioterapia assenta no facto de as células cancerígenas se focarem numa rápida reprodução e serem incapazes de se reparar quando atingidas por radiação, ao contrário das células saudáveis. Tem como objetivo entregar uma dose de radiação suficiente à região cancerígena de forma a esterilizar o tumor, tentando simultaneamente não comprometer a capacidade de células saudáveis sobreviverem e minimizar os danos aos órgãos e tecidos saudáveis circundantes [44].

De acordo com a forma como a radiação é administrada, a radioterapia divide-se em dois tipos principais, sendo eles, a braquiterapia e a radioterapia externa [5]. A primeira, também conhecida por radioterapia interna, resulta da colocação de fontes de radiação, como implantes, dentro ou o mais próximo possível do tumor [48]. Já na radioterapia externa, aquela que se enquadra no contexto deste projeto, a radiação é administrada por uma fonte externa ao paciente, direcionada para a localização do tumor utilizando máquinas especializadas. Existe um conjunto variado de máquinas que emitem diferentes tipos de radiação, incluindo máquinas Cobalt-60, aceleradores lineares, máquinas de feixes de neutrões, máquinas de feixes de protões, entre outras [20, 48].

A IMRT (Radioterapia de Intensidade Modelada) é um tipo importante de *conformal radiation therapy* em que um feixe de radiação é modelado por um colimador

multifolhas (MLC), podendo ser visto como vários sub-feixes (como *beamlets* ou *bixels*), cada um deles com uma dada fluência (intensidade). É importante compreender que este sub-feixes não existem fisicamente, sendo gerados pelo movimento das folhas do MLC que bloqueiam parte do feixe durante porções do tempo de irradiação.

No fundo, a otimização de um planeamento de tratamento pode ser representada como a seleção ótima de um determinado tratamento entre um conjunto possível de soluções admissíveis [44].

Devido à complexidade do problema é comum que o planeamento do tratamento seja feito através de um procedimento de tentativa e erro, conhecido como *forward planning*. Neste tipo de procedimento define-se uma configuração e as doses que vão ser absorvidas pelo organismo são calculadas. Se estas doses forem aceitáveis, considerando a prescrição médica, então o procedimento termina [44]. Caso contrário, continua, alterando manualmente a planificação do tratamento. Este é um processo demorado e tedioso para o dosimetrista, e não tem garantias de produzir planos de tratamento de alta qualidade [40].

Em contrapartida, o planeamento inverso (*inverse planning*), como é o caso da IMRT, traduz-se no cálculo do tratamento de planeamento óptimo, estabelecendo limites de dose para os órgãos de riscos em função das doses prescritas, de forma a que esta se module ao volume respetivo. Assim, tem a vantagem de permitir a modelação de problemas de planeamento de tratamento complexos, administrando uma maior dose no tumor e diminuindo a dose recebida pelos tecidos adjacentes [5]. Mesmo usando planeamento inverso, a IMRT continua a ser um processo iterativo, podendo haver a necessidade do dosimetrista gerar e avaliar múltiplos planos antes que estes estejam de acordo com o critério de aceitação [14].

2.1.1 Sequência do Tratamento

Para que seja possível compreender onde se enquadra o problema sob o qual se foca este projeto, é imperativo conhecer a sequência do tratamento de radioterapia.

Desta forma, o fluxo do tratamento de radioterapia surge como uma cadeia com os seguintes passos: (1) Imobilização; (2) Imagem; (3) Localização do Tumor; (4) Planeamento; (5) Posicionamento; (6) Tratamento e (7) Garantia de Qualidade [14, 44].

Os três primeiros passos focam-se na identificação e delimitação da forma tridimensional (3D) do tumor e dos órgãos e tecidos circundantes do paciente, através de várias técnicas de imagem avançadas como a tomografia axial computorizada (TAC) ou a ressonância magnética (RM), guardadas em ficheiros DICOM.

Uma imagem 3D obtida por uma TAC pode ser considerada como um conjunto de fatias 2D do corpo. Cada uma dessas fatias representa uma seção transversal do corpo num local específico, mostrando os órgãos internos e os ossos nesse local [14].

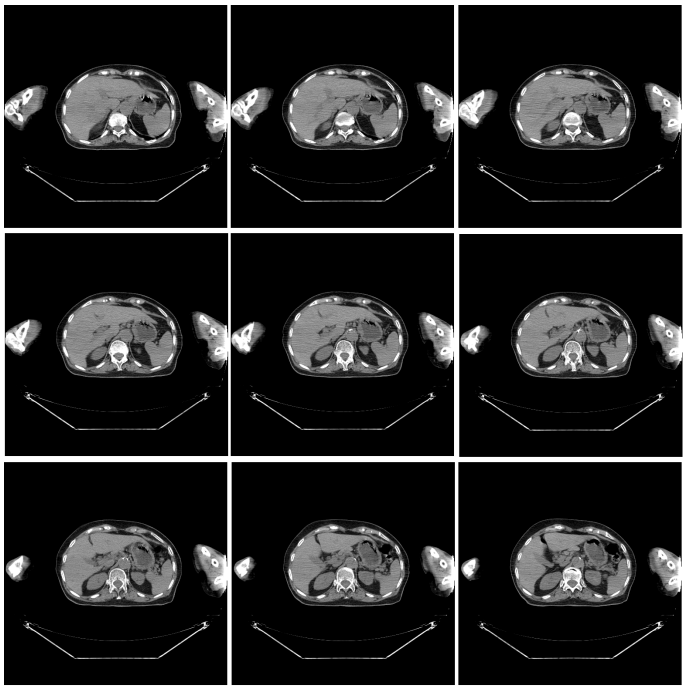


Figura 2.1: Exemplo TAC da pélvis e conjunto de fatias que a compõem.

Com base nestas imagens 3D, o médico radiologista delimita várias estruturas importantes para a zona que irá ser irradiada, entre as quais o CTV, PTV, OAR e NT [41].

O CTV (*Clinical Target Volume*) descreve o volume conhecido do tumor, incluindo a possível propagação microscópica. O PTV (*Planning Target Volume*) representa o volume do tumor conhecido e propagação microscópica (CTV) mais um volume marginal à volta do CTV, relevando-se como a estrutura usualmente utilizada para o desenho de planos de tratamento (consultar figura 2.2). O OAR (*Organs at Risk*) descreve os órgãos que podem ser danificados na vizinhança do tumor. E finalmente, o NT (*Normal Tissue*) identifica os tecidos saudáveis circundantes onde a radiação não deve incidir [14, 41].

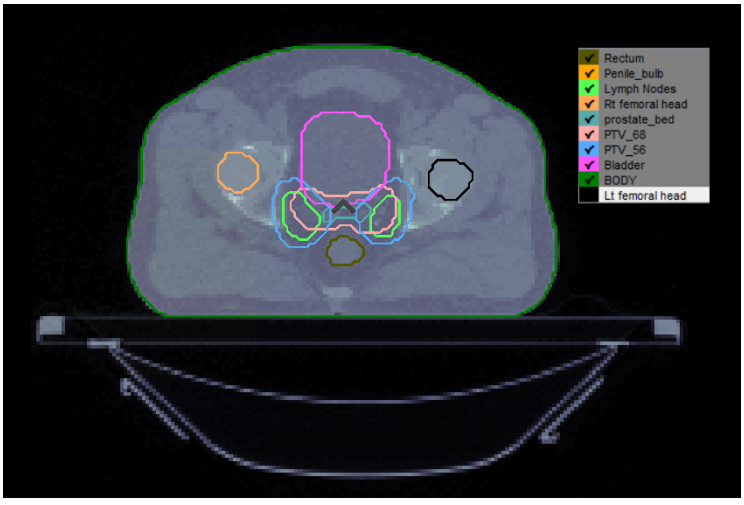


Figura 2.2: Exemplo do delineamento dos volumes de uma TAC da pélvis.

Para efeitos de otimização e de cálculo de depósito de dose de radiação, cada uma destas estruturas de volume é discretizada em voxeis (elementos de volume) [44].

Durante o tratamento radioterapêutico, o paciente encontra-se imobilizado numa maca que pode rodar, recebendo radiação de um acelerador linear (linac) montado num *gantry* que pode igualmente rodar ao longo do eixo central, como se pode observar na figura 2.3 . Esta combinação permite radiação de qualquer ângulo à volta do tumor. Apesar deste facto, na prática apenas se considera ângulos coplanares de forma a simplificar um problema já complexo.

Figura 2.3: Acelerador Linear utilizado na Radioterapia Externa. Retirado de [36]

Colimadores multifolhas encontram-se posicionados em frente dos emissores de partículas e transformam o feixe de radiação numa grelha de *beamlets* (unidades discretizadas de cada um dos feixes) mais pequenos de intensidades independentes.

Para criar o plano de tratamento, é preciso calcular a dose que atinge cada ponto no corpo, ou seja, a porção da dose prescrita que é efetivamente entregue. A energia depositada por unidade de massa de tecido é designada dose absorvida e é expressa em unidades Gray (Gy). A dose total absorvida num ponto específico do corpo pode ser calculada como uma soma ponderada das doses absorvidas devido à intensidade de cada *beamlet*. Assim, o problema de planeamento (quarto passo na cadeia) corresponde à escolha dos ângulos dos feixes e mapas de intensidade, para que a distribuição da dose resultante cumpra com a prescrição da dose pelo médico radiologista e com as restrições para os órgãos e tecidos saudáveis. Este passo está repleto de sub-tarefas iterativas que culminam na aceitação do plano de tratamento, tornando-o assim um dos passos mais longos e complexos da cadeia [14].

Após a produção de um conjunto de mapas de intensidade aceitáveis, é fundamental encontrar uma forma correta de entrega, ou seja, como posicionar e tratar o paciente, correspondentes ao passos cinco e seis da cadeia [44].

Por fim, o último passo da cadeia corresponde à avaliação da qualidade do tratamento, muitas vezes baseada em DVH (histogramas de dose-volume) e comparando-o com um conjunto variado de métricas, que podem mudar consoante o paciente em questão [14].

A qualidade do tratamento depende de vários fatores, entre os quais a forma como o problema da otimização dos ângulos dos feixes - conhecido como BAO - e da otimização dos mapas de fluência - FMO - são abordados e resolvidos durante o planeamento. Sucintamente, o tratamento terá melhor qualidade sempre que deposita a dose prescrita no PTV, minimizando simultaneamente a dose absorvida pelos OARs e tecidos circundantes.

Após termos compreendido todas as etapas do processo de tratamento radiológico, e as dependências entre as mesmas, no contexto deste projeto, vamos nos focar na quarta etapa do processo, o planeamento, particularmente na otimização da solução ao nível do estado da arte para o problema FMO.

2.1.2 Módulo de Geração Automática de Planos

O projeto ORION divide-se internamente em seis módulos, nomeadamente: (1) Importação e Exportação, (2) Plataforma Web, (3) Segmentação Automática, (4) Editor de ROIs, (5) Geração Automática de Planos e (6) Visualizador e Validador.

A primeira etapa corresponde à importação de um ficheiro *dicom* para a plataforma Web através do módulo de Importação e Exportação de ficheiros (1). A partir desse ponto é possível fazer a segmentação automática das estruturas presentes no ficheiro carregado (3). O utilizador poderá igualmente editar a região de interesse (ROI), caso considere que a segmentação necessita de algum ajuste com base no seu caso específico (4). Pode ainda, gerar planos de forma automática através do módulo de Geração Automática de Planos (5). Após este passo, consegue visualizar os resultados do plano gerado (6). E por fim, poderá exportar o plano se assim o desejar (1). A figura 2.4 representa estes módulos e as suas interações.

O módulo de Geração Automática de Planos (5) sobre o qual recai esta dissertação, foi desenvolvido pelos professores Humberto Rocha e Joana Dias da Universidade de Coimbra. Este módulo tem como objetivo automatizar a criação de planos de tratamento radioterapêutico através da resolução do problema FMO na fase de planeamento. Pretende evitar que o dosimetrista invista muito tempo num processo de tentativa-erro e que possa ser assistido por mecanismos autónomos.

Este módulo foi desenvolvido em MATLAB e faz uso da função *ipopt*, usando o algoritmo de pontos interiores. Tem como *input* um conjunto de imagens TAC de um paciente que definem o volume que irá ser irradiado, com os órgãos delineados, e um algoritmo de cálculo de dose que devolve a dose unitária que cada um dos vóxeis do paciente recebe de cada *beamlet*.

A solução presente neste módulo assume que a dose total (D_T) é igual ao produto da matriz da dose unitária (d) pelo vetor das intensidades (w), como enunciado

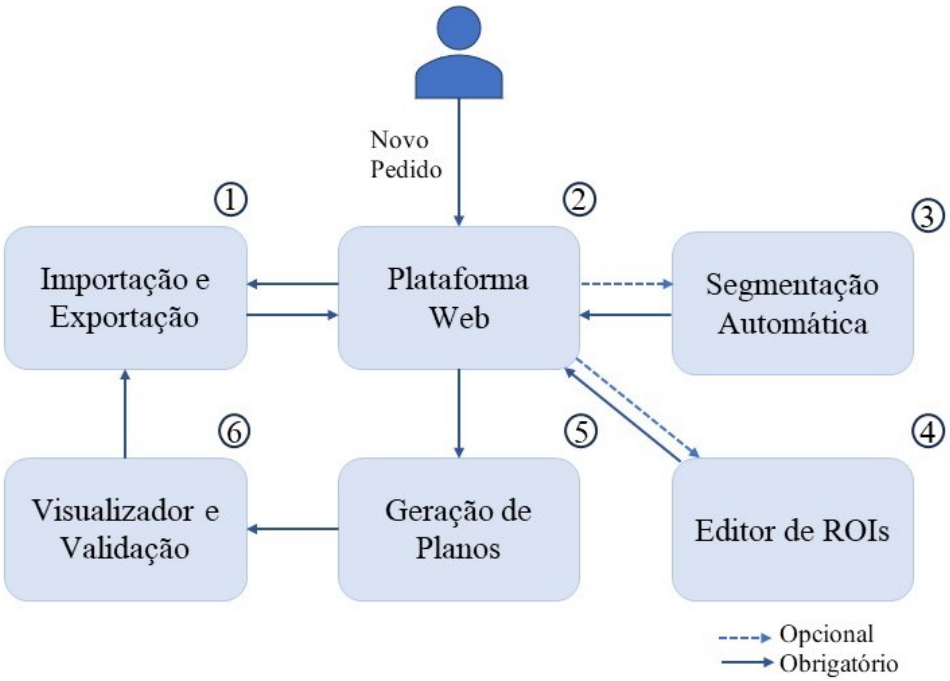


Figura 2.4: Esquema da arquitetura do projeto ORION

em 1 no capítulo 3.2.

Assim, considerando as direções fixas dos feixes (por exemplo, configurações equidistantes), o objetivo é encontrar um vetor w otimizado, isto é, que faça com que a dose total respeite a prescrição médica para cada um dos volumes.

Capítulo 3

Fundamentos

Neste capítulo, será revisto o estado da arte para os problemas de otimização dos ângulos dos feixes (*Beam Angle Optimization* - BAO) e de otimização de mapas de fluência (*Fluence Map Optimization* - FMO) no contexto da IMRT. Particularmente o último, considerando que é o alvo do módulo de Geração Automática de Planos.

Na fase de planeamento do tratamento da IMRT, a escolha de abordar o problema BAO, o problema FMO, ou ambos, depende das características específicas do tratamento e dos objetivos da terapia.

É igualmente possível resolver simultaneamente os problemas de BAO e FMO. Neste caso, seria utilizado um algoritmo de otimização que tivesse em conta tanto os ângulos dos feixes como os mapas de fluência para encontrar a solução ótima que proporcione a distribuição da dose desejada, minimizando ao mesmo tempo a dose para os tecidos circundantes saudáveis. Esta é uma abordagem mais complexa e computacionalmente intensiva.

Assim, serão examinados vários algoritmos propostos para abordar o problema FMO, incluindo métodos lineares e não lineares, e serão discutidos os seus pontos fortes e limitações.

3.1 Problema BAO

O problema da otimização dos ângulos dos feixes (BAO) tem como meta encontrar o número mínimo de feixes e direções correspondentes que satisfazem os objetivos do tratamento, libertando o dosimetrista para que dedique o seu tempo a outros encargos [44].

A resolução deste problema tem impacto tanto na qualidade como na duração do tratamento. Portanto, a escolha de direções adequadas é decisiva para maximizar as doses do tumor e poupar os órgãos em risco [45]. Além disso, alterar as direções dos feixes durante o tratamento consome muito tempo, e com vista o conforto do paciente é aconselhável que o tratamento seja de curta duração. Muitas tentativas para solucionar o problema BAO podem ser encontradas na li-

teratura, como a utilização de algoritmos genéticos ou otimização por enxame de partículas [45].

Tipicamente a resolução do problema BAO requer a resolução do problema FMO, ou seja, para cada conjunto fixo de direções (ou ângulos) dos feixes é necessário saber os mapas de fluência (ou intensidade) ótimos correspondentes. Por este motivo, em muitas abordagens ambos os problemas são resolvidos em conjunto. Contudo, para o contexto do projeto ORION vai se focar apenas na resolução do segundo.

3.2 Problema FMO

Nesta fase, escolhidos os ângulos dos feixes a ser usados, o paciente será tratado com base no plano ótimo obtido ao resolver o problema de otimização dos mapas de fluências (FMO). Este problema visa determinar os pesos de *beamlets* ótimos para ângulos de feixes fixos.

Ao longo dos anos, foram propostos vários algoritmos e modelos matemáticos para resolver este problema, incluindo modelos lineares [46], modelos não-lineares [21] e modelos multiobjetivo [16].

O volume de cada estrutura de volume (PTV, OAR e NT) é discretizado em voxeis, e a dose unitária para cada voxel é calculado considerando a contribuição de cada beamlet através do princípio da superposição [44], enunciado em 1.

Assumindo que há $m \times n$ *beamlets* identificados pelo par (i, j). Cada voxel (x, y, z) em cada estrutura S, sendo $S = PTV \cup OARs \cup NT$, recebe uma dose total identificada pela matriz de dose total, $D_T(x, y, z)$ [44, 49]:

$$D_T(x, y, z) = \sum_{\theta \in \Theta} \sum_{i=1}^{m} \sum_{j=1}^{n} w(\theta, i, j) d_{(\theta,i,j)}(x, y, z) \tag{1}$$

em que o peso (intensidade) do *beamlet* (i, j) entregue sobre o ângulo $\theta \in \Theta$ definido por $w(\theta, i, j)$ e $d_{(\theta,i,j)}(x, y, z)$ corresponde à dose entregue ao voxel (x, y, z) pelo *beamlet* (i, j) do ângulo θ.

O número de linhas da matriz D_T (matriz de Dose Total) é igual ao número de voxeis e o número de colunas é igual ao número de *beamlets* a partir de todos os ângulos considerados. É importante ter em atenção, que muitas vezes o número de voxeis atinge uma ordem de grandeza de milhares, ou até milhões.

Deste modo, compreendemos uma das maiores dificuldades do problema FMO, sendo o tamanho da matriz D_T uma fonte de problemas de escala grande.

3.2.1 Modelos Lineares

A primeira tentativa de resolução deste problema, na literatura, passou pela utilização de modelos lineares (LP) [12]. Um dos fatores para utilizar estes modelos é o facto que a deposição da dose ser linear. A maioria das formulações pertencem à classe de modelos de otimização restritos, de tal forma que uma função objetivo é otimizada ao mesmo tempo que satisfaz os requisitos de dose, através da definição limites inferiores e superiores (*Lower* e *Upper bounds*).

Uma simples formulação para um modelo LP seria uma função objetivo minimizada:

$$\begin{aligned} & min_w f(D_T) \\ & s.q.\, D_T(x,y,z) = \sum_{(\theta,i,j)} w(\theta,i,j) \cdot d_{(\theta,i,j)}(x,y,z), \forall (x,y,z) \in S \\ & \quad LB_{PTV} \leq D_{T_i} w \leq UB_{PTV}, \forall_i \in PTV, \\ & \quad D_{T_i} w \leq UB_{OAR}, \forall_i \in OAR, \\ & \quad D_{T_i} w \leq UB_{NT}, \forall_i \in NT, \\ & \quad 0 \leq w(\theta,i,j) \leq M, \forall \theta \in \Theta, i = 1,...,m, j = 1,...,n. \end{aligned} \tag{2}$$

onde LB_{PTV} e UB_{PTV} correspondem aos limites inferiores e superiores para a dose do PTV, e TG_{PTV} a um dado objetivo alvo para o PTV. O mesmo conceito se aplica aos OARs e NTs. Adicionalmente, w corresponde aos pesos (ou intensidades) de cada *beamlet* (i,j) para o ângulo θ e d à matriz de dose unitária para cada voxel (x,y,z). Esta função objetivo pretende encontrar o mínimo w respeitando as restrições de dose impostas. Deste modo, nas restrições apresentadas, cada linha de D_T é atribuída a um voxel i e multiplicada pelo vetor de pesos w, $(D_{T_i} w)$.

Uma variedade de diferentes modelos podem ser formulados ao combinar diferentes restrições com objetivos. Algumas das funções objetivo mais usadas:

$$\begin{aligned} f(D_T) = \alpha_{ptv} \| D_{T_{PTV}} - TG_{PTV} \|_1 + \alpha_{OAR} \| (D_{T_{OAR}} - UB_{OAR}) \|_1 + \\ \alpha_{NT} \| (D_{T_{NT}} - UB_{NT}) \|_1 \end{aligned} \tag{3}$$

$$\begin{aligned} f(D_T) = \alpha_{ptv} \| D_{T_{PTV}} - TG_{PTV} \|_2^2 + \alpha_{OAR} \| (D_{T_{OAR}} - UB_{OAR}) \|_2^2 + \\ \alpha_{NT} \| (D_{T_{NT}} - UB_{NT}) \|_2^2 \end{aligned} \tag{4}$$

A expressão 3 penaliza o valor absoluto do desvio da dose prescrita em cada voxel, pesando-o de acordo com a região em que cada voxel se encontra. Já expressão 4 penaliza a soma dos quadrados dos desvios, pesando-o pelos fatores definidos. Como visto em 4 e como será discutidos na subsecção 3.2.3, é possível transformar expressões lineares em não lineres. Os fatores de peso, $\alpha_{(.)}$, necessitam de ser definidos pelo dosimetrista. Objetivos diferentes, correspondentes

a diferentes estruturas, podem ser considerados ao usar diferentes normas na mesma função objetivo.

Apesar desta abordagem ser rápida e fácil de formular, a desvantagem em usar modelos lineares assenta na sua falta de flexibilidade [49]. Estes modelos produzem soluções admissíveis que são pontos extremos da região admissível, onde restrições são satisfeitas como equanimidade, pelo que os limites de prescrição são frequentemente atingidos. Ou seja, é frequente que a solução ótima atribua a dose máxima permitida aos OARs e/ou que o PTV receba a dose mínima permitida pela prescrição. Se as restrições forem altas, uma solução viável é difícil de descobrir e a fonte de inviabilidade é desconhecida [44].

3.2.2 Modelos Lineares Inteiros Mistos

Com a introdução das restrições dose-volume, o uso de modelos lineares inteiros mistos (MIP) apareceu naturalmente. Estas restrições são biologicamente significativas e derivaram do facto que por vezes os médicos radiologistas definem os objetivos do tratamento como uma percentagem de cada tipo de tecido que recebe uma dada dose de radiação. Assim, podem sacrificar uma porção de uma região em risco, a fim de aumentar a probabilidade de cura da doença. A título exemplo, é tipicamente aceitável que o PTV receba 95% da dose prescrita [44].

Desta forma, as restrições de dose no modelos LP na subsecção acima (3.2.1) vão ser agora substituídas por restrições dose-volume. Por exemplo, na expressão 2 podemos substituir a restrição do limite superior do PTV da seguinte forma [44]:

$$\begin{aligned} D_i w &\leq (1 + y_i F) UB_{PTV}, \forall_i \in PTV, \\ \sum_{i \in PTV} y_i &\leq P \times card(PTV), \end{aligned} \tag{5}$$

em que y_i são as variáveis binárias associadas a um voxel i, F a sobredosagem máxima permitida e P a percentagem máxima de pontos que podem receber a dose acima da dose máxima do limite superior do PTV. As variáveis binárias y_i associadas a um voxel i para um PTV tem o valor unitário quando a radiação excede a dose do limite superior para o PTV, e nulo em caso contrário.

As restrições dose-volume garantem que não mais de P% do volume do PTV pode ser exceder F% da dose de radiação de UB_{PTV}. Conferindo aos modelos MIP maior flexibilidade evitando problemas de viabilidade.

De facto, a abordagem MIP tem a vantagem de permitir a formulação de modelos mais complexos e flexíveis. Contudo, têm igualmente a desvantagem de ser muito mais difícil de resolver, implicando o uso de algoritmos como *bound-and-branch*, por exemplo [44]. Considerando ainda que o número de inteiros é no mínimo igual ao número de voxeis, encontrar uma solução ótima é muito difícil num tempo clinicamente aceitável. Tem ainda a desvantagem de o processo de otimização poder ficar preso numa solução ótima local, que pode não ser o ótimo global [49].

3.2.3 Modelos Não Lineares

De forma a facilitar a escolha dos pesos e reduzir a dependência do plano em relação a cada estrutura, podemos considerar um desvio da dose média em cada estrutura. Ao considerar-se a norma modificada 4 pode-se originar um modelo quadrático não linear:

$$f(D) = \alpha_{ptv}\frac{||D_{PTV} - TG_{PTV}||_2^2}{card(PTV)} + \alpha_{OAR}\frac{||(D_{OAR} - UB_{OAR})_+||_2^2}{card(OAR)} + \alpha_{NT}\frac{||(D_{NT} - UB_{NT})_+||_2^2}{card(NT)} \tag{6}$$

onde $card(.)$ denota o número total de voxeis para cada estrutura.

A formulação quadrática é a mais prevalente entre modelos não lineares, que garantem a existência de soluções admissíveis. Devido à sua simples formulação matemática, estes modelos têm como vantagem o baixo custo computacional e a convergência rápida para uma solução.

3.2.4 Comparação dos modelos

Modelos lineares (LP), modelos lineares interios mistos (MIP) e modelos quadráticos são técnicas de otimização que podem ser utilizadas para resolver o problema de otimização dos mapas de fluência (FMO) na IMRT.

Os modelos LP são utilizados para otimizar funções objetivo lineares sujeitas a restrições lineares. Estes modelos são relativamente simples e rápidos de resolver, mas só podem ser utilizados para otimizar funções objetivo lineares e restrições lineares. São frequentemente utilizados em situações em que o problema de otimização é relativamente simples, e o número de variáveis e restrições é pequeno [49].

Os modelos MIP, por outro lado, são uma técnica de otimização mais complexa que pode ser utilizada para otimizar funções objetivo lineares e não lineares sujeitas a restrições lineares e não lineares. Estes modelos são capazes de considerar múltiplas restrições e funções objectivo, o que permite um elevado grau de precisão no plano de tratamento final. No entanto, são computacionalmente intensivos e requerem recursos computacionais significativos para serem resolvidos, bem como um elevado nível de conhecimento para serem implementados.

Os modelos quadráticos são utilizados para otimizar uma função objetivo quadrática sujeita a restrições lineares. Utilizam uma formulação matemática simples e direta, o que os torna relativamente fáceis de implementar e compreender. São tipicamente mais rápidos de resolver do que os modelos MIP e requerem menos recursos computacionais. Contudo, podem não ser capazes de fornecer o mesmo nível de precisão que os modelos MIP.

Em conclusão, os modelos LP são simples e rápidos de resolver, mas têm flexi-

bilidade limitada; os modelos MIP são complexos, mas são precisos e flexíveis; e os modelos quadráticos são rápidos, mas menos precisos e flexíveis do que os modelos MIP. A melhor abordagem depende dos requisitos e constrangimentos específicos do problema e dos recursos disponíveis.

3.3 Qualidade do Tratamento

Tal como introduzido na secção 2.1.1, o último passo do processo do tratamento de radioterapia corresponde à avaliação da qualidade do tratamento. Frequentemente, esta avaliação é baseada na observação de DVHs (histogramas de dose-volume), juntamente com outros indicadores de qualidade, e comparando-os com um conjunto variado de métricas, que podem mudar consoante o paciente em questão [14].

Um DVH é uma representação gráfica que fornece informações sobre a distribuição da dose de radiação dentro de uma região de interesse (ROI) no corpo do paciente. DVHs são tipicamente usados no planeamento e avaliação do plano radioterapêutico, de forma a analisar a dose recebida tanto pelo volume alvo quanto pelos órgãos em risco (OARs) que precisam ser poupados da exposição excessiva à radiação. Ao examinar os DVHs, os oncologistas podem avaliar o quão bem o plano de tratamento atinge a distribuição de dose desejada e garantir que a dose prescrita seja administrada com precisão [17, 54].

O DVH é representado através de um gráfico de linhas, onde o eixo x do histograma equivale aos níveis de dose e o eixo y representa o volume ou a percentagem do ROI.

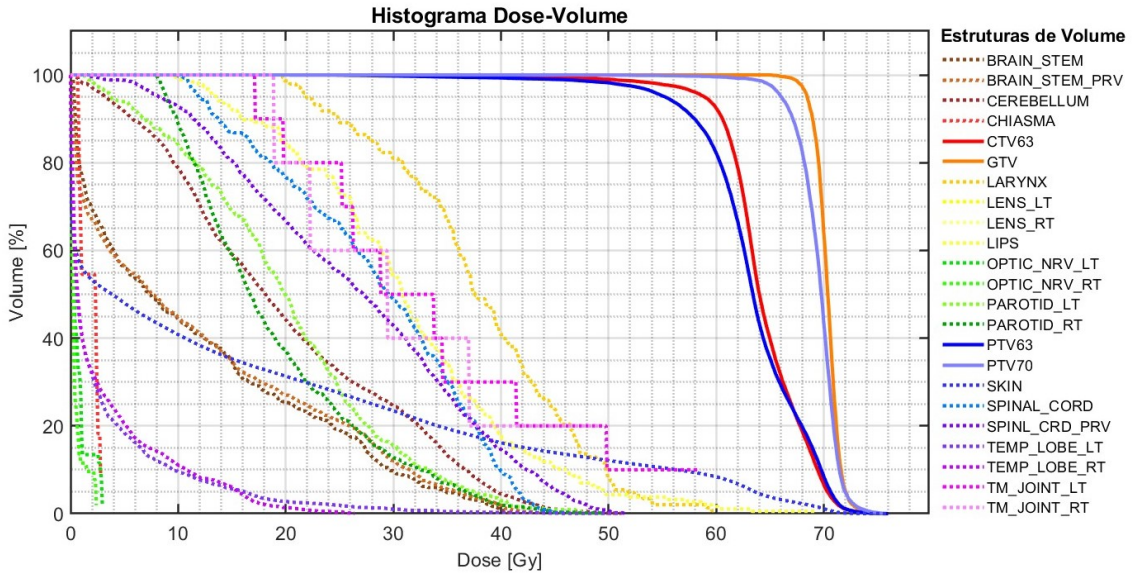


Figura 3.1: Histograma Dose-Volume (DVH) de um plano relativo à "cabeça e pescoço".

Num plano de tratamento, o DVH para o volume alvo fornece informações sobre

a cobertura da dose, ou seja, quanto do volume alvo recebe uma determinada dose. Por exemplo, D_{95} e D_{98} surgem como medidas comummente usadas durante a análise da cobertura alvo e indica qual a percentagem do volume alvo que recebe, respetivamente, 95% e 98% da dose prescrita [44].

Para os estruturas alvo é expetável que uma percentagem significativa do seu volume seja coberta pela a dose prescrita. Se este ponto for respeitado durante a construção do plano, é esperado ver-se no histograma uma linha praticamente horizontal na marca dos 100% do volume até atingir aproximadamente a dose prescrita. Na imagem 3.1 podemos verificar este comportamento quando olhamos, por exemplo, para o CTV63 (linha vermelha), onde a maior parte do seu volume recebe aproximadamente 63 Gy, verificando uma queda a pique quando se aproxima deste objetivo. Para os OARs, o DVH fornece informações sobre a economia de dose, indicando a percentagem do volume do OAR que recebe determinadas doses. Esta informação é essencial para avaliar o potencial risco de efeitos colaterais induzidos por radiação nos tecidos saudáveis.

Capítulo 4

Estratégias de Implementação

Este capítulo pretende elucidar os conceitos de computação em paralelo e funções de otimização abordados neste trabalho. A secção 4.1 irá abordar a utilização da GPU para computação paralela, fazendo uma breve contextualização histórica da sua necessidade e evolução, bem como a introdução de uma plataforma de computação paralela, CUDA. Na secção 4.2 são descritas matrizes esparsas e a sua utilidade para otimização. E finalmente, na secção 4.3 são apresentados alguns métodos de otimização quadrática clássicos, juntamente com alguns solvers que os implementam.

4.1 Otimização usando a GPU

O cálculo dos mapas de fluência na IMRT envolve a utilização de algoritmos de otimização que requerem grandes cálculos matriciais, como multiplicações matriz-vetor (ver 1). Estes cálculos podem ser morosos quando efetuados numa CPU (Central Processing Unit), especialmente para planos de tratamento grandes e complexos.

Assim, a utilização da GPU devido ao seu elevado grau de paralelismo, permite executar muitos cálculos da matriz em paralelo, o que pode resultar em acelerações significativas em comparação com a utilização de uma CPU.

4.1.1 Contexto histórico

Até meados dos anos 2000, apenas se considerava unidades de processamento central (CPU) para tarefas de processamento computacional como cálculos arbitrários. As unidades CPU foram responsáveis pelo aumento do desempenho dos computadores atuais, sendo capazes de executar milhares de milhões de operações por segundo. A sua evolução acompanhou o crescimento exponencial da frequência de relógio, isto é, número de operações executadas por unidade de tempo. Contudo, em 2000 este crescimento experienciou uma quebra abrupta ao atingir a chamada "barreira de potência"(*power wall*) [13].

A par desta estagnação emerge o crescimento da performance das GPUs. Originalmente, as unidades de processamento gráfico foram desenvolvidas com vista a acelerar o processo de renderização de imagens tridimensionais numa matriz bidimensional correspondente aos píxeis num ecrã. Contudo, em meados dos anos 2000, investigadores reconheceram o potencial da GPU, devido às suas capacidade de paralelismo computacional, para aplicações não gráficas, tais como aprendizagem computacional e computação científica [2]. Durante estas investigações perceberam que poderiam iludir a gráfica ao enviar-lhe qualquer tipo de dados, concretamente dados com significado numérico, em vez das *cores* de entrada como esperado, programando os shaders dos píxeis para efetuar cálculos arbitrários sobres estes dados [47]. Assim, com a emergência de incontáveis ferramentas e linguagens de programação, dá-se o florescimento da computação em GPU, tornando esta abordagem mais sofisticada e acessível.

Deste modo, o paradigma relativamente à evolução da performance computacional sofre uma transformação, direcionando o aumento da frequência de relógio para o investimento e investigação do desenvolvimento de instruções paralelas em múltiplos núcleos. Assim, a GPU contém mais cores (e mais simples) que a CPU, de maneira a recolher dados da memória, executar cálculos paralelos e devolvê-los em memória para uso [33].

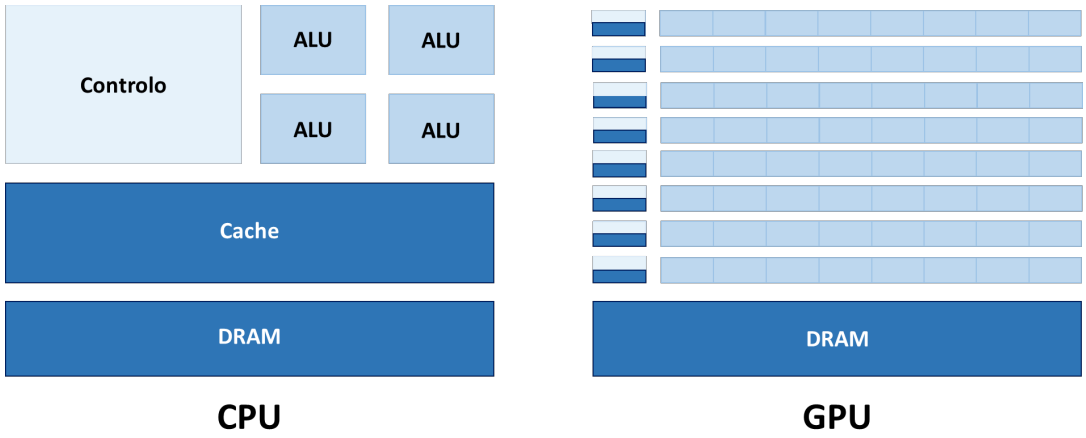


Figura 4.1: Comparação arquitetura CPU e GPU. Adaptado de [50].

Observando para a figura 4.1 pode-se compreender melhor as diferenças entre a arquitetura destas duas unidades de processamento. Por norma, uma CPU contém várias ALUs (Unidades Lógica Aritmética), uma unidade de controlo que por sua vez controla as ALUs, uma memória cache e uma memória de acesso aleatório dinâmica (DRAM). Por seu lado, a GPU contém centenas de ALUs e várias unidades de controlo, bem como várias memórias cache e uma DRAM [50].

Atualmente, a computação gráfica envolve a troca de dados entre a GPU e o host (normalmente a CPU), bem como a sincronização da execução de tarefas entre os dois. Isto pode ser feito utilizando estruturas como CUDA (Compute Unified Device Architecture) [8, 39] e OpenCL (Open Computing Language) [30].

4.1.2 CUDA

Em Novembro de 2006, a NVIDIA apresenta ao mercado a primeira GPU DirectX 10 - a GeForce 8800 GTX - construída com a primeira arquitetura CUDA (*Compute Unified Device Architecture*) [8]. CUDA é uma plataforma de computação paralela e um modelo API (*Application Programming Interface*) [39], que foi desenvolvida especificamente para computação em GPU e tem como objetivo contornar várias limitações dos processadores que limitavam o uso de aplicações não gráficas, permitindo computações em paralelo sem degradar a velocidade de execução [47]. Para além da introdução de uma interface, NVIDIA adicionou unidades de *hardware* especializadas concebidas para efetuar cálculos paralelos rápidos [23].

A arquitetura CUDA consiste nos seguintes componentes [7]:

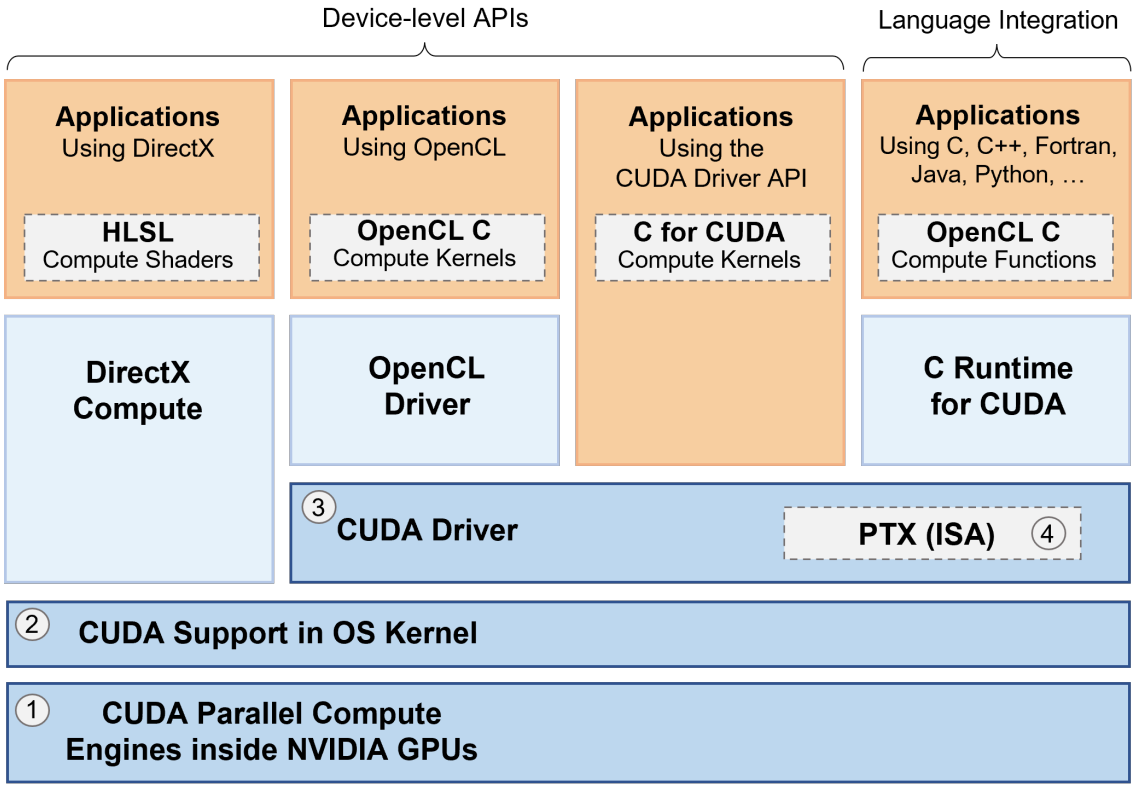


Figura 4.2: Arquitetura CUDA. Adaptado de [7].

1. ***CUDA Parallel Compute Engines*** dentro das GPUs NVIDIA, consistem em grupos de *cores*, de memória partilhada, registos, e unidades de *hardware*, de modo a executar certos tipos de instruções de forma mais eficiente, fornecendo as capacidades de processamento paralelo que permitem à GPU acelerar uma vasta gama de aplicações.

2. ***CUDA support in the OS kernel***, corresponde ao suporte ao nível do *kernel* do sistema operativo para inicialização do *hardware*, configuração, entre outros.

3. ***CUDA Driver***, uma camada de *software* que fornece uma API para interagir com os dispositivos da GPU. Fornece funções para atribuir e desalocar me-

mória na GPU, executar *kernels* da GPU, e sincronizar a execução de tarefas entre o *host* e a GPU.

4. **Arquitetura do conjunto de instruções PTX** (*Parallel Thread Execution ISA*) para *kernels* e funções de computação paralela, permite aos programadores escrever código que pode ser compilado e executado na GPU.

Um programa em CUDA organiza-se num programa *host*, em que a CPU serve como hospedeiro e na qual são executadas uma ou mais *threads*, e num ou mais *kernels* adequados para executar num dispositivo de processamento paralelo como a GPU [24]. O programador organiza um conjunto de *threads* paralelas, executadas por um *kernel*, numa grelha de blocos de *threads* (ver figura 4.3). Estes blocos consistem em grupos independentes de ALUs [9], o que implica que o *kernel* deve executar corretamente qualquer que seja a ordem em que os blocos são executados. Esta restrição das dependências entre os blocos de um *kernel* proporciona escalabilidade. Apesar disso, ao decompor o trabalho paralelo em *kernels* separados é necessário comunicação global e sincronização entre as *threads*.

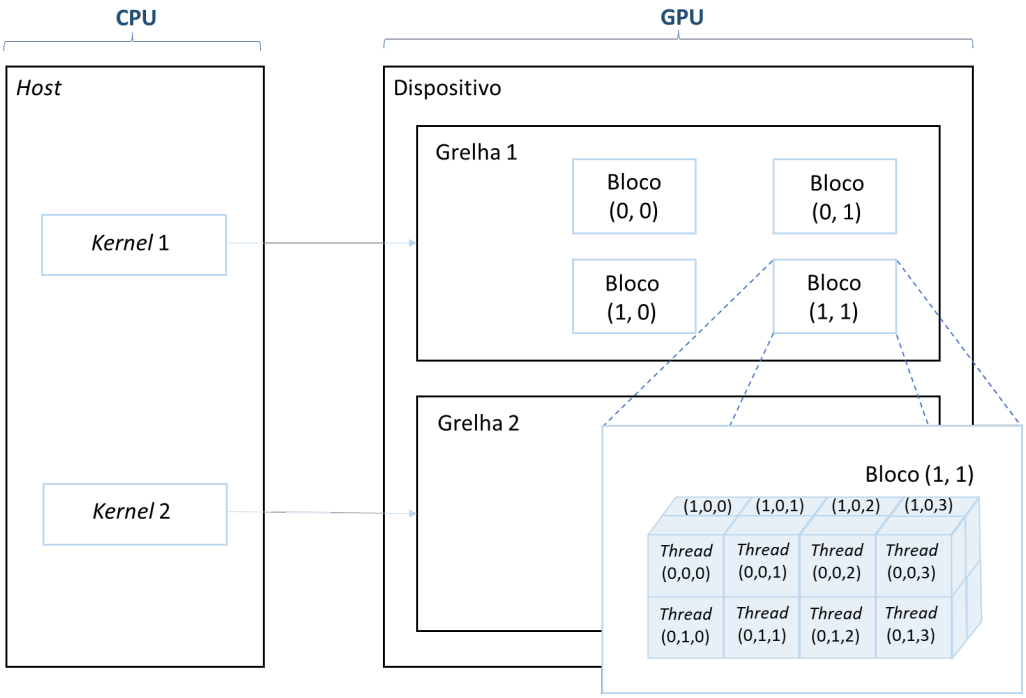


Figura 4.3: Um exemplo da organização de uma grelha em CUDA. Adaptado de [31].

A flexibilidade e programabilidade de CUDA, com mais de 150 bibliotecas baseadas em CUDA, SDKs, e ferramentas de otimização, tornaram-na a plataforma de eleição para a investigação e implementação de novos algoritmos de *deep learning* e de computação paralela [12]. Aplicações escritas em C e C++ podem utilizar diretamente o C Runtime para CUDA. Aplicações escritas em outras linguagens podem aceder ao *runtime* através de ligações de método nativo, permitindo aos programadores usar a arquitetura CUDA para, por exemplo, programas em Fortran (FLAGON Fortran 95), em Java (JaCUDA), em Python (PyCUDA), entre outros [7].

4.2 Matriz esparsa

Matrizes esparsas são frequentemente usadas para representar estruturas de dados em que apenas uma pequena porção de elementos são diferentes de zero. Visam reduzir de forma significativa a quantidade de memória e recursos computacionais necessários para representar e operar a estrutura [25].

A matriz de dose unitária, d, representa a distribuição da dose para todos os voxeis (elementos de volumes) presentes no plano. O seu tamanho corresponde ao número total de voxeis pelo número de sub-feixes (ou *beamlets*). O número de voxeis pode variar dependendo da resolução da imagem 3D, da complexidade dos órgãos presentes, do tamanho do volume alvo, entre outras. Este fator pode levar a matrizes com milhões de voxeis, e consequentemente centenas de GBs. Uma representação simplificada da matriz de dose pode ser observada na figura 4.4.

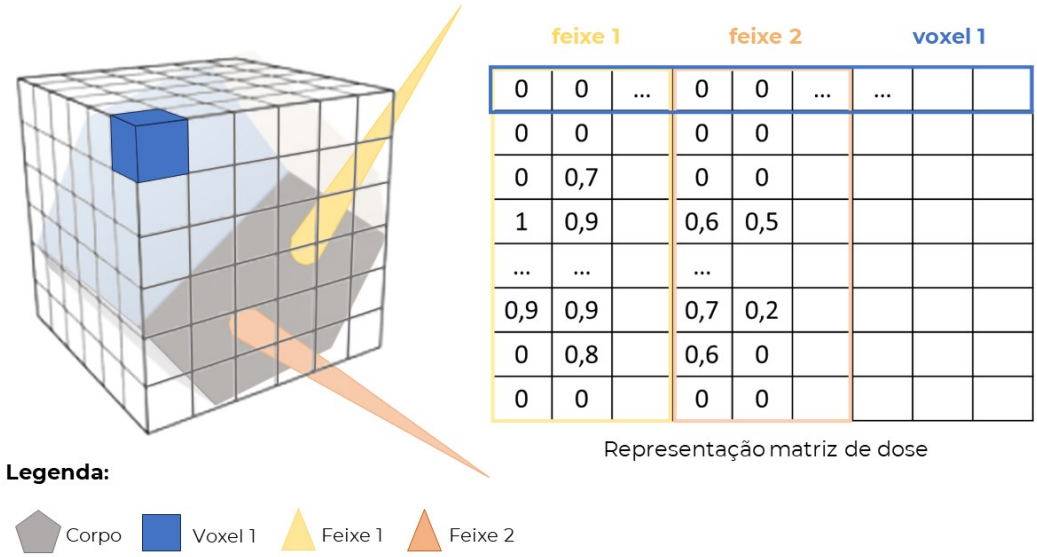


Figura 4.4: Representação visual simplificada da matriz de dose.

Esta matriz apresenta caraterísticas esparsas devido ao facto de os feixes de radiação serem concebidos para fornecer radiação a um volume alvo específico (PTV), minimizando ao mesmo tempo a exposição dos OARs e NT. Portanto, a maioria dos elementos da matriz correspondem a voxeis fora do volume alvo, que recebem muito pouca ou nenhuma radiação, resultando em muitos zeros na matriz. [18].

Existem várias abordagens para representar este tipo de matrizes em memória, das quais se destacam: (1) **Lista de Coordenadas** (COO), (2) **Dicionário de Chaves** (DOK), (3) **Compressed Sparse Row** (CSR) e (4) **Compressed Sparse Column** (CSC) [19, 22].

No formato **COO** (1) a matriz é armazenada como uma lista de tuplos (i, j, valor), onde i e j são os índices de linha e coluna, respetivamente, dos elementos

diferentes de zero, e valor é o valor do elemento. No formato **DOK** (2) a matriz é armazenada como um dicionário onde as chaves são tuplos (i, j) representando os índices dos elementos diferentes de zero, e os valores são os valores dos elementos correspondentes. Já no formato **CSR** (3) a matriz é armazenada como três matrizes unidimensionais: os valores diferentes de zero (na ordem principal da linha), os índices de coluna dos elementos diferentes de zero e o índice inicial de cada linha nos dois primeiros vetores. Finalmente, o formato **CSC** (4) é bastante similar ao CSR, substituindo apenas os índices de linha pelos índices de coluna.

Conhecendo estes quatro formatos, deve-se ter em conta os cenários em que a matriz de esparsa é utilizada durante este projeto, ou seja, é importante compreender que a matriz esparsa está envolvida em operações de acesso a elementos e multiplicações matriz-vetor para o cálculo do vetor de intensidade desejado. Desta forma, ao analisar estes formatos, recolheu-se que:

- O **COO** facilita a construção e manipulação de matrizes esparsas, contudo, considerando que o nosso principal objetivo é executar a multiplicação matriz-vetor e visto que não é uma formato eficiente para memória, poder-se-á excluir [19].
- O **DOK** por sua vez é um formato bom para a construção, mas complexo para o acesso de elementos individuais da matriz uma vez que envolve usar o par (linha, coluna) para concretizar o acesso. Por outro lado, as tabelas hash consomem muito mais memória do que vetores [15, 19].
- Os formatos **CSR** e **CSC** são eficientes para multiplicações matriz-vetor e matriz-matriz, bem como para operações orientadas a linhas/colunas [25]. A compressão das coordenadas das linhas/colunas redundantes aumenta o desempenho de cálculos que normalmente são limitados à largura de banda [da memória], como a multiplicação esparsa de matriz-vetor [32].

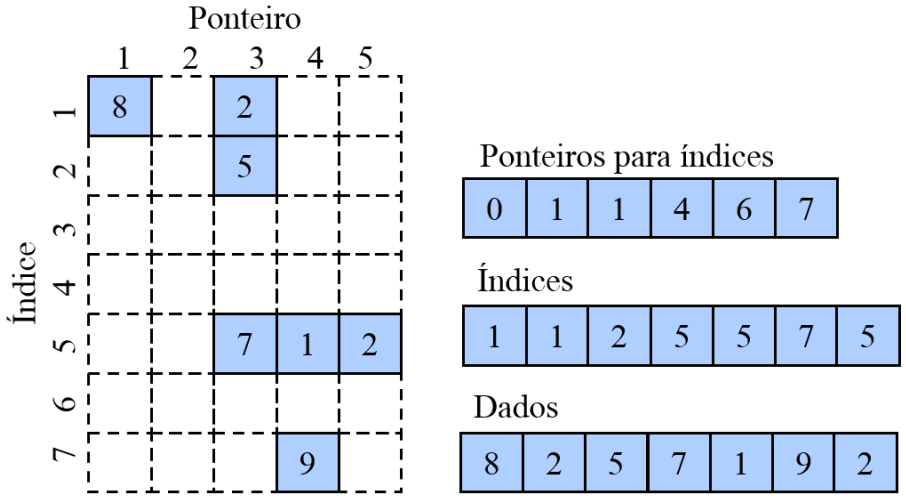


Figura 4.5: Representação formato CSC para matrizes esparsas. Adaptada de [22]

Assim, analisando a natureza do nosso problema e de forma a otimizar as multiplicações matriz-vetor entre a matriz de dose d e o vetor de pesos w, optou-se

por utilizar a representação **CSC** representado na figura 4.5, sendo esta também a representação base do MATLAB para matrizes esparsas.

4.3 Otimização Quadrática

Como analisado anteriormente no capítulo 3, modelos quadráticos podem ser utilizados para otimizar uma função objetivo quadrática sujeita a restrições lineares e destacam-se pela sua rapidez e formulação matemática simples e direta, o que facilita a sua implementação e compreensão.

A otimização quadrática pode ser aplicada na resolução de uma ampla gama de problemas do quotidiano, incluindo engenharia, finanças, aprendizagem computacional e tratamentos médicos. Num contexto de programação, a otimização quadrática refere-se ao processo de otimização de uma função objetivo quadrática sujeita a um conjunto de restrições. Envolve encontrar os valores das variáveis que minimizam ou maximizam a função quadrática enquanto satisfazem as restrições dadas [29, 45]. No contexto do nosso problema, temos como objetivo descobrir o vetor de intensidade, w, ótimo, que faça com que a dose total respeite a prescrição médica para cada um dos volumes, minimizando a dose irradiada aos órgãos em risco e tecidos circundantes do volume-alvo.

4.3.1 Métodos de Otimização Quadrática

Existem vários métodos de otimização quadrática clássicos na literatura, cada um apresenta uma abordagem distinta para resolver problemas de otimização. Entre estes, destacam-se métodos de **Pontos Interiores**; métodos de **Conjunto Ativo**; métodos de **Programação Quadrática Sequencial** e métodos de **Região de Confiança**.

Métodos de **Pontos Interiores** transformam um problema de programação quadrática com restrições de desigualdade num problema de otimização sem restrições, introduzindo uma função de barreira. São capazes de tratar de um grande número de restrições de desigualdade, encontrar ótimos globais eficientemente e convergir para a solução em passos finitos [43].

Métodos de **Conjunto Ativo** por sua vez, abordam uma série de subproblemas onde as restrições de igualdade são satisfeitas. O método visa prever o conjunto ativo, que inclui as restrições satisfeitas exatamente na solução. Funciona em duas fases, onde a primeira garante a viabilidade, enquanto que a segunda otimiza a solução dentro das restrições do conjunto ativo [53].

Métodos de **Programação Quadrática Sequencial (SQP)** apresentam uma técnica de otimização iterativa que usa programação quadrática para encontrar ótimos locais de um problema de otimização não linearmente restrito. Estes métodos surgiram para combater algumas limitações dos métodos de Conjunto Ativo, combinando o mesmo com o método de Newton [27]. Lida com restrições não lineares e funções objetivo não convexas e é capaz de encontrar otimizações locais

eficientemente e melhorar iterativamente a estimativa do ótimo.

Finalmente, métodos de **Região de Confiança** definem uma vizinhança, conhecida como região de confiança, em torno da melhor solução atual. Esta região é representada tipicamente por um modelo quadrático que aproxima a função objetivo. O mecanismo associado a métodos de região de confiança pretende determinar simultaneamente a direção do passo e o tamanho do mesmo para otimização [37, 55]. Para além de serem capazes de resolver problemas lineares e não lineares, estes métodos são adequados para lidar com problemas de grande escala.

Tendo explorado alguns métodos de otimização, cada um com seus pontos fortes e nuances, pode-se aventurar na descoberta de *solvers* criados especificamente para implementar estas metodologias.

Destes, o **IPOPT** [4] e o **fmincon** [3] surgem como exemplos de *solvers* capazes de aplicar estes algoritmos para resolver problemas. O IPOPT (Interior Point OPTimizer), aproveita de forma eficaz os princípios inerentes aos Métodos de Pontos Interiores para navegar em cenários complexos de otimização. Este *solver* surgiu de uma pesquisa rica e extensa ao longo de vários anos, o IPOPT destaca-se como um solver robusto que se diferencia entre outros *solvers open-source*. As suas caraterísticas tornam-no numa escolha adequada para enfrentar problemas de programação quadrática com restrições de desigualdade, facilitando a busca eficiente de ótimos globais.

Por outro lado, o fmincon é um solver pertence ao *toolkit* de otimização do MATLAB e reflete os princípios de vários métodos supracitados, desde os Métodos de Conjuntos Ativos e de Programação Quadrática Sequencial, bem como algoritmos de região de confiança e de pontos interiores. Assim, permite ao utilizador escolher o algoritmo mais adequado para o seu problema específico.

Portanto, a seleção do IPOPT e do fmincon foi motivada por estes se alinharem com os princípios subjacentes às metodologias de otimização elucidadas anteriormente, pela a sua robustez, versatilidade e capacidade de enfrentar com eficiência os complexos desafios de otimização. No contexto de planos de tratamento de IMRT, tornam-se opções formidáveis na busca de soluções, que minimizem a emissão da dose em tecidos e órgãos saudáveis, satisfazendo em simultâneo a dose do volume alvo desejada. Nas próximas secções ir-se-á aprofundar estes *solvers* e os métodos que implementam.

4.3.2 IPOPT e Método de Pontos Interiores Primal-Dual

O **IPOPT** (Interior Point OPTimizer) é um *solver open-source* desenvolvido pela COIN-OR, e foi desenvolvido para resolver problemas de otimização não-lineares de grande escala com variáveis continuas, incluindo problemas de programação quadráticos [4]. Foi criado para ser altamente flexível, permitindo especificar o problema em termos de sua função objetivo, restrições e suposição inicial para a solução. O IPOPT transforma o problema num conjunto de equações não lineares, que posteriormente, são usadas para resolver o método de pontos interiores

primal-dual. É capaz de lidar com restrições não lineares e consegue incorporar tanto restrições de igualdade e desigualdade lineares [51].

O **método de pontos interiores** e o **método de pontos interiores primal-dual** são técnicas de otimização matemática usadas para resolver problemas de programação linear e não linear, sendo o segundo uma extensão do primeiro.

O primeiro método de pontos interiores conhecido é o método de barreira logarítmica de Frisch, que foi desenvolvido para resolver problemas de programação linear em que a complexidade do pior caso era melhor do que o método *simplex* usado na altura [26]. A ideia básica do método é reformular o problema de otimização com restrições num problema sem restrições, introduzindo uma função de barreira na função objetivo. Essa função barreira penaliza soluções que se aproximam do limite da região viável. Com o passar dos anos, este método foi adaptado de forma a ser capaz resolver de também problemas de otimização não linear, sendo atualmente dividido em três categorias: métodos projetivos, métodos de *affine-scaling* e métodos primal-dual.

O método de pontos interiores primal-dual é uma destas variantes e é considerada a classe mais bem-sucedida de métodos de pontos interiores. Em vez de reformular o problema, o método primal-dual incorpora diretamente as restrições no problema de otimização e resolve os problemas primal e dual simultaneamente. A principal diferença deste método centra-se na atualização destas variáveis. As variáveis primárias representam as variáveis no problema de otimização, enquanto as variáveis duais correspondem às restrições. A cada iteração, a atualização é baseada nos valores atuais das variáveis, no gradiente da função objetivo e de restrição. Este método executa um passo de Newton ou uma variação deste consoante a especificação do problema e, em seguida, move-se para o próximo passo, sem ciclos internos e externos separados [51].

4.3.3 Fmincon e Método de Região de Confiança Refletivo

Em contrapartida, o **fmincon** é um *solver* de otimização integrado no MATLAB que foi desenhado para resolver problemas de otimização com variáveis continuas, incluindo problemas de programação quadrática [3].

O fmincon também é capaz de lidar com restrições não lineares e consegue incorporar tanto restrições de igualdade e desigualdade lineares. Permite ainda utilizar vários algoritmos para otimização incluindo `trust-region-reflective`, `interior-point` ou `sqp` [1].

O algoritmo de `interior-point` utilizado no fmincon é baseado no método do pontos interiores. É particularmente eficaz para resolver problemas de otimização condicionada com funções objetivo e restrições suaves. É conhecido pela sua capacidade de lidar com uma vasta gama de restrições, incluindo restrições lineares e não lineares de igualdade e desigualdade. Este algoritmo procura uma solução que satisfaça as restrições, deslocando-se iterativamente para o interior da região viável.

O algoritmo `sqp` - programação quadrática sequencial - aproxima a função ob-

jetivo e as restrições utilizando modelos quadráticos e executa subproblemas de programação quadrática em cada iteração. O algoritmo `sqp` é eficiente para problemas com variáveis e restrições de tamanho moderado.

O algoritmo `trust-region-reflective` é outra alternativa disponível no fmincon. Combina o método da região de confiança com uma estratégia reflexiva para lidar com restrições lineares ou limites. Este algoritmo é frequentemente preferido quando se lida com problemas com a imposição de limites, em que as variáveis estão sujeitas a limites inferiores e superiores.

Existem outros algoritmos disponíveis no fmincon, que não serão aprofundados aqui, e incluem `active-set` e `sqp-legacy`. Cada um destes algoritmos tem diferentes estratégias de otimização e características de desempenho, o que os torna adequados para diferentes tipos de problemas.

Assim, analisando estas caraterísticas percebemos que o fmincon é um *solver* deveras diversificado, que pode ser aplicado a diversos problemas. Não obstante, seguindo as indicações do MATLAB e considerando as especificações do nosso problema, optou-se por utilizar o método `trust-region-reflective`. Este método sobressai-se como o mais adequado tendo em conta a importância da imposição de limites de inferiores e superiores de dose durante o tratamento de radioterapia.

Deste modo, o algoritmo de **região de confiança refletivo** é uma extensão dos métodos de região de confiança abordados na secção 4.3.1.

O algoritmo de região de confiança refletivo é baseado no conceito de regiões de confiança anteriormente discutido. A ideia é aproximar a função objetivo dentro de uma região de confiança, que é uma vizinhança ao redor do ponto atual. O algoritmo então calcula uma etapa de teste dentro da região de confiança e atualiza o ponto atual com base na melhoria da função objetivo. A região de confiança é ajustada em cada iteração com base no progresso feito. Caso contrário, a etapa é rejeitada e a região de confiança é contraída [34, 37].

Complementarmente, o método região de confiança refletivo, implementado pelo fmincon no MATLAB, incorpora várias ideias, incluindo a transformação reflexiva e o uso de um modelo quadrático. Este modelo é usado para aproximar a função objetivo dentro da região de confiança. A transformação reflexiva é a principal diferença desta versão, em que será necessário escolher o melhor passo entre a passo reflexivo, o passo da região de confiança e o passo ao longo do limite da região confiável. Esta transformação é particularmente usada para lidar com situações em que o ponto atual está nos limites do problema [34].

4.3.4 Resumo IPOPT vs Fmincon

Uma das principais diferenças entre o IPOPT e o fmincon enquanto *solvers*,é que o primeiro é um *solver open-source* que pode ser modificado e personalizado pelos utilizadores. Em contraste, o fmincon é um *solver* proprietário, disponível apenas no MATLAB. Isto significa que, tipicamente, os utilizadores têm mais controlo sobre o processo de otimização ao utilizar o IPOPT e que a utilização do fmincon

está limitada ao MATLAB. Contudo, considerando que este projeto foi desenvolvido em MATLAB, este ponto deixa de ser uma limitação para o uso do fmincon.

Outra diferença entre os dois *solvers* é o tipo de algoritmo de otimização que utilizam. O IPOPT utiliza um método de ponto interiores primal-dual, enquanto o fmincon fornece uma maior liberdade relativamente à escolha do seu algoritmo, permitindo o uso de um método de pontos interiores, de região de confiança ou de programação quadrática sequencial, consoante as necessidades do problema em questão [1].

O IPOPT é um *solver* poderoso, particularmente eficiente para problemas de otimização em grande escala [26]. No entanto, requer maior investigação para ser utilizado e personalizado em comparação com o fmincon, uma vez que é um *solver* muito extenso, com muitas classes, onde é difícil de encontrar e selecionar as extensões de código relevante para o nosso problema específico. Adicionalmente, há um suporte formal menor disponível para o IPOPT em comparação com o fmincon, que é um *solver* proprietário suportado pelo MATLAB.

O fmincon, por outro lado, é um *solver* de fácil aplicação que é amplamente utilizado no MATLAB. É adequado para problemas de otimização de pequena e média dimensão e é conhecido por ser robusto e fiável. No entanto, pode não ser tão eficiente como o IPOPT para problemas de grande escala.

Capítulo 5

Experimentação e Resultados

Neste capítulo será discutido o sucesso da implementação das técnicas apresentadas no capítulo anterior (4), bem como o ambiente de execução e os testes realizados para analisar a eficácia de cada implementação. Assim, na secção 5.1, começou-se por fazer algumas considerações iniciais necessárias para compreender o ambiente em que serão realizados os testes. De seguida, são expostos os resultados da implementação das matrizes esparsas na secção 5.2. Avançando para a análise dos resultados de diferentes algoritmos de otimização de programação quadrática, com apoio do IPOPT e *fmincon* na secção 5.3. Consecutivamente, foi avaliado o impacto da utilização da GPU para a realização dos cálculos especificados pela equação (1) na secção 5.4 e de otimizações a nível da eficiência dos acessos à matriz de dose na secção 5.5. Por fim, foi analisado se a variação do ponto inicial, isto é, do vetor de pesos w, influencia a tempo de execução na secção 5.6.

5.1 Considerações Inicias

Antes de serem apresentadas as experiências realizadas, deve-se ter em conta algumas considerações iniciais. Para começar, o projeto desta tese foi desenvolvido e testado em MATLAB, e foram utilizadas várias ferramentas, como *matRad*, *Parallel Computing Toolbox* e *Optimization Toolbox*. De seguida, será explicado o ambiente de trabalho, bem como as condições em que foram executados os testes.

5.1.1 Ambiente de Testes

Tal como descrito anteriormente, este projeto foi desenvolvido e testado em **MATLAB**. Antes de avançar para a realização dos testes, é necessário compreender algumas das ferramentas utilizadas, bem como as estruturas e processos subsequentes, e como estes influenciam a solução.

O **matRad**[1] é um conjunto de ferramentas *open source* de cálculo e otimização de

[1] O repositório pode ser encontrado em `https://github.com/e0404/matRad`

dose na radioterapia. Inteiramente escrito em MATLAB, apresenta uma vasta gama de funcionalidades desde a importação de DICOM e fotões digitalizados, ao cálculo de dose de iões de carbono digitalizado e otimização de dose para visualização do plano [52]. O modelo de Geração Automática de Planos apoia-se neste conjunto de ferramentas para gerar o ficheiro de *steering*, calcular a matriz de dose, calcular o histograma DVH do plano, entre outras funcionalidades.

Analisando o código fornecido pelos professores Humberto Rocha e Joana Dias, foi necessário identificar as estruturas relevantes e perceber onde a informação proveniente de um plano era guardada e mais tarde acedida.

Inicialmente, importa-se um conjunto de ficheiros DICOM, que inclui várias fatias TAC, ficheiros RP (*Radiation Plan*), RD (*Radiation Dose*) e RS (*Radiation Structure*), convertendo-os para um ficheiro do tipo *.mat* com o apoio do MatRad. Desta forma, ao importante o ficheiro *.mat* são criadas duas estruturas (*ct*, *cst*). A primeira variável (*ct*) contém a tomografia axial computorizada (TAC), em que a imagem de dados é armazenada como uma estrutura. É nesta estrutura que é armazenada, por exemplo, a resolução dos voxeis e as fatias da TAC.

Posteriormente, as segmentações correspondentes aos volumes de interesse (VOI) são armazenadas numa matriz em que cada linha é identificada pelo nome da estrutura de volume que lhe corresponde, a qual se designa por *cst*. Para além da lista de índices dos voxeis para cada VOI, a *cst* contém meta dados para a otimização da dose, como a prioridade de sobreposição, restrições e objetivos utilizados para calcular o valor da função objetivo durante o planeamento inverso apresentado na secção 2.

Após a definição do modo de radiação, o tipo de máquina, o número de voxeis, entre outra informação relevante para o plano, pode-se gerar a informação da geometria dos feixes com o auxílio da função `matRad_generateStf` e armazená-la na estrutura *stf*.

Assim que a informação da geometria do feixe estiver disponível, pode-se efetuar o cálculo da dose de fotões ou partículas (`matRad_calcPhotonDose` ou 'matRad_calcParticleDose'). A função de cálculo da dose produz uma estrutura denominada *dij*, que contém a contribuição da dose para cada feixe individual.

A estrutura *dij* conterá informação relativa ao número de *beams* e voxeis, bem como a matriz de dose física que será utilizada para efetuar as operações definidas na equação 1.

Definida a matriz de dose, d, é necessário definir o vetor de pesos iniciais, w, que corresponde a uma razão matemática entre a dose alvo máxima pela média de todas as doses alvo, e preparar os dados de entrada para cada algoritmo de otimização.

Finalmente, a multiplicação matriz-vetor que é discutida ao longo deste projeto, encontra-se definida na função `matRad_backProjection` que será chamada pela função objetivo a cada iteração do algoritmo de otimização. Esta função é implementada pelo matRad e foi modificada pelos professores Humberto Rocha e Joana Dias para as caraterísticas do nosso problema específico.

5.1.2 Condições de Testagem

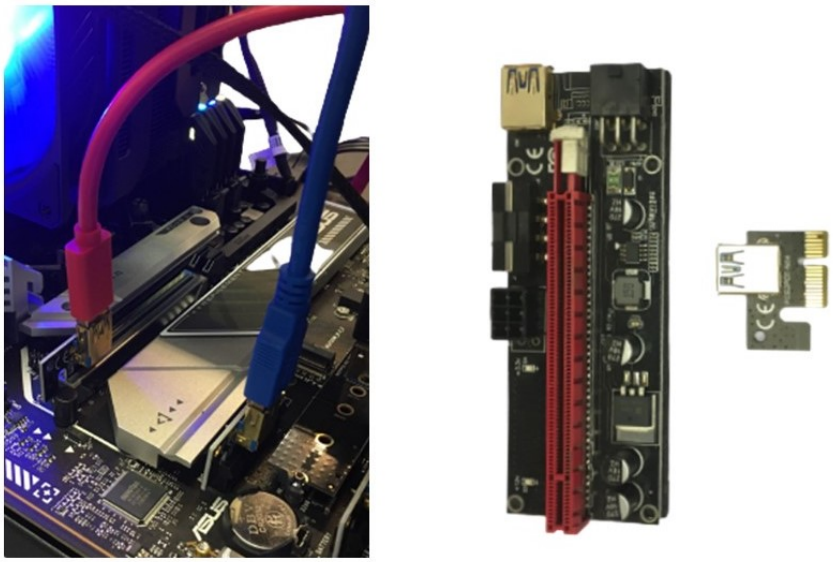
Durante a seguinte experimentação foram executados diversos testes com base em quatro ficheiros DICOM reais disponibilizados pela Mercurius Health. Estes ficheiros DICOM apresentam informações relativas a CTs da região da cabeça e pescoço, pulmões, pélvis e estômago respetivamente.

Os testes foram realizados em duas máquinas diferentes, as quais estarão estipuladas na tabela 5.1:

ID	Processador	Placa Gráfica	VRAM Disponível
Maq-1	AMD 7950x	NVIDIA GeForce RTX 3080	10,50 GB
Maq-2	Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz 2.21 GHz	Quadro P600	4.29 GB

Tabela 5.1: Especificação das caraterísticas das máquinas de teste

É relevante referir que a máquina 1 utiliza um PCIe (Peripheral Component Interconnect Express), que corresponde a uma interface para conectar componentes de alta velocidade, no nosso caso conectar a GPU à *motherboard*. Os PCIes podem apresentar distintas configurações físicas, que podem ser representados pelos fatores x1, x4, x8, x16, x32.



(a) (b)

Figura 5.1: (a) PCIe Riser x1 para x16 real da máquina 1 e (b) PCIe Riser x1 para x16

Como se pode observar pela imagem 5.1(a), no presente cenário temos instalados PCIe Risers x1 para x16, que possui apenas uma pista para transferência de dados, o que implica que pode mover dados a um bit por ciclo [28].

Desta forma, para cada instância diferente do problema, ou seja, cada combinação diferente entre máquina, solução e ficheiro, foram realizadas trinta execuções isoladas e independentes. Portanto, os dados que serão demonstrados ao longo deste capítulo, irão corresponder à média destas trinta execuções. É importante reparar que todas as condições de testagem foram mantidas ao máximo ao longo dos diferentes testes. Contudo, não é possível garantir a isolação a 100%, visto que por vezes as máquinas foram utilizadas pela equipa do LIS para outros projetos.

Conhecidas as especificações do ambiente de testes e as condições de testagem, poderemos passar para a análise das experiências com base nas estratégias de implementação abordadas e discutir os resultados das mesmas.

5.2 Matrizes Esparsas

Tal como discutido na secção 4.2 a utilização de matrizes esparsas pode significar ganhos significativos em termos de armazenamento e performance.

Desta forma, a primeira ronda de testes visou comprovar que de facto, a matriz de dose, d, é uma candidata viável para a aplicação de matrizes esparsas. Como representado na tabela 5.2, começou-se por calcular o número total de elementos da matriz, juntamente com o número de elementos não nulos da mesma. Com estes dados é possível calcular a percentagem de elementos nulos presentes na matriz original.

Ficheiro	Tamanho Original (GB)	Total elementos	Elementos não nulos	Percentagem de elementos nulos (%)
Cabeça e Pescoço	54,1	$7,26E+09$	31338893	99,57
Pulmões	195,5	$2,62E+10$	153495615	99,42
Pélvis	1289,5	$1,73E+11$	921553404	99,47
Estômago	1195,3	$2,62E+12$	676770741	99,58

Tabela 5.2: Caraterísticas da matriz de dose para cada caso de teste

Após a análise dos dados recolhidos, surge um padrão notável: a percentagem de elementos nulos presentes na matriz original varia consistentemente na faixa de 99,4% a 99,6% para todos os ficheiros considerados. Atendendo à regra geral de aplicação de matrizes esparsas que enuncia que matrizes em que o número de elementos nulos corresponde a cerca de dois terços do seu número total de

elementos, são candidatas viáveis para a aplicação deste tipo de representação. Esta regra e a notável uniformidade na dispersão das matrizes originais para os diferentes ficheiros, ressalta a adequação da utilização da matriz esparsa para abordar matrizes de dose.

A percentagem extremamente alta de elementos nulos significa que as matrizes originais são predominantemente compostas de pontos de dados que não recebem radiação. Considerando que na multiplicação matriz-vetor enunciada em 1, estes elementos nulos resultarão em elementos nulos, perdendo-se tempo computacional multiplicando qualquer valor por zero. Assim, a sua representação gera um desperdício de memória e recursos computacionais, que são uma preocupação central no contexto de nosso problema. Assim, a esparsidade identificada valida a eficácia da adoção da representação de matriz esparsa.

A seguinte ronda de testes pretendeu averiguar efetivamente a redução gerada por esta estratégia. Assim, na tabela 5.3 apresenta-se os dados relativos a esta experiência, nomeadamente o tamanho original da matriz de dose (em GB), o tamanho da sua versão esparsa e a percentagem relativa à poupança de armazenamento verificada.

Ficheiro	Representação Original (GB)	Representação Esparsa (GB)	Percentagem de Redução (%)
Cabeça e Pescoço	54,1	0,47	99,13
Pulmões	195,5	2,29	98,83
Pélvis	1289,5	13,73	98,94
Estômago	1195,3	10,08	99,16

Tabela 5.3: Comparação da representação da matriz de dose original e esparsa para cada caso de teste

Ao examinar os resultados da tabela 5.3, torna-se evidente que a integração de matrizes esparsas trouxe benefícios notáveis em termos de eficiência de armazenamento. O tamanho original da matriz de dose, que abrange vários GBs para cada ficheiro, é significativamente reduzido na sua representação esparsa.

A percentagem de redução do armazenamento alcançada varia aproximadamente entre 98,8% e 99,2%, indicando uma redução substancial nos requisitos de armazenamento. A capacidade de transformar uma matriz de dose massiva numa representação altamente compacta sem sacrificar informações críticas ressalta o poder e a eficiência desta abordagem.

5.3 Comparação Algoritmos

Aplicadas as matrizes esparsas, passou-se para uma análise comparativa do desempenho exibido através da utilização dos *solvers* apresentados previamente,

IPOPT e **fmincon**. Relembrando que o IPOPT implementa um método de pontos interiores primal-dual, enquanto que o fmincon implementa um algoritmo de região de confiança refletivo. Os tempos de execução, em segundos, registados para ambos os *solvers* para os quatro ficheiros de teste usando a máquina 1, são apresentados na tabela 5.4.

Ficheiro	IPOPT (s)	Fmincon (s)	Speed-Up (%)
Cabeça e Pescoço	4,68	2244,96	99,79
Pulmões	57,79	9214,83	99,37
Pélvis	102,45	16216,35	99,37
Estômago	305,96	12268,73	97,51

Tabela 5.4: Comparação do tempo de execução para IPOPT e fmincon, usando a máquina 1

Os resultados iniciais destacam inequivocamente que o IPOPT apresenta uma eficiência superior em comparação com o fmincon, como é evidente pelos tempos de execução consideravelmente mais curtos para todos os casos de teste.

No entanto, reconhecendo a complexidade do problema, e considerando que ainda há espaço para mais medidas de otimização, as próximas secções irão apresentar potenciais modificações que poderão impactar o desempenho de ambos os *solvers*. Ao analisar meticulosamente os parâmetros algorítmicos e as caraterísticas específicas do problema, queremos determinar se de facto a superioridade inicial do IPOPT persiste ou se o fmincon exibe um potencial inesperado.

Na imagem 5.2 pode-se observar os DVHs gerados pelos planos obtidos utilizando os dois *solvers* para o ficheiro "Cabeça e Pescoço".

Analisando os DVHs resultantes, percebemos que ambos os planos seguem uma distribuição de dose semelhante, particularmente olhando para a cobertura alvo. A principal diferença entre os mesmos reside na preservação dos órgãos em risco, que advém da forma como ambos os *solvers* lidam com os limites definidos. Nomeadamente, o fmincon usa limites para estes órgãos impostos por mim com base na tabulação de dose definida pelo RTOG (Radiation Therapy Oncology Group). Estes limites posteriormente são usados para calcular o valor da função objetivo e o gradiente. Desta forma, é necessário ter em consideração que estes valores são apenas de referência e devem ser verificados por um oncologista certificado, pelo que não será comparada a preservação dos OARs neste projeto e focar apenas na cobertura alvo.

Observando a figura 5.3 têm-se uma perspetiva mais clara das diferenças obtidas para a cobertura alvo dos planos resultantes usando cada *solver*.

Adicionalmente, a tabela 5.5 apresenta a cobertura alvo para as estruturas de volume alvo presentes na figura 5.5, fazendo uso da métrica D_{95} que indica a qual a dose recebida, em Gy, para 95% do volume da estrutura alvo.

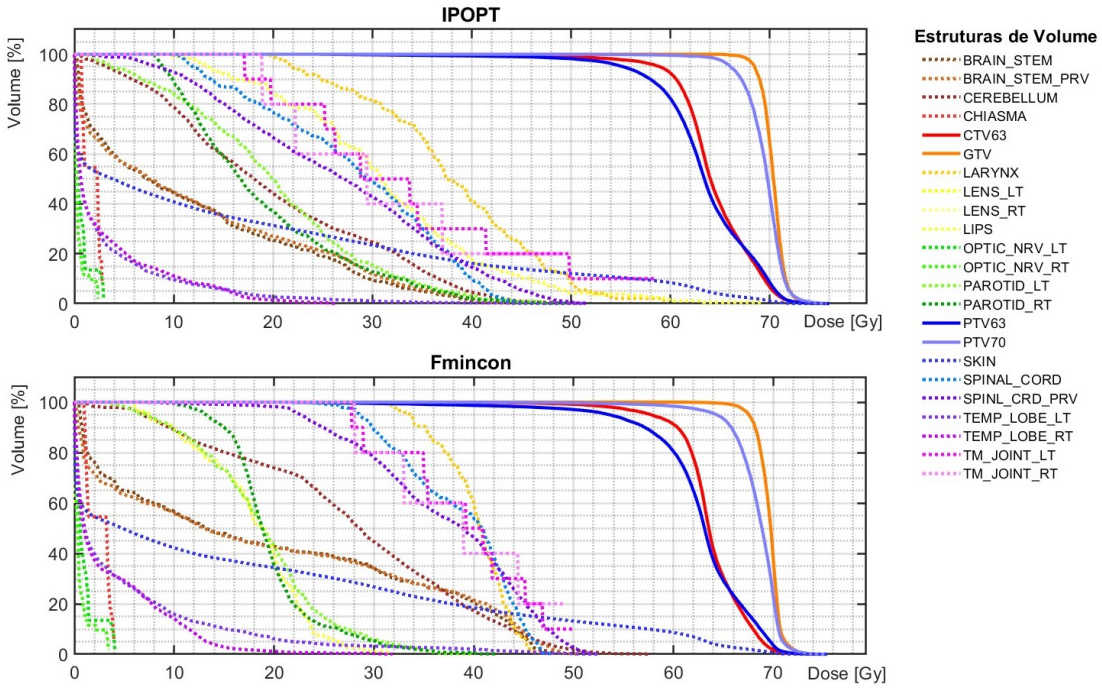


Figura 5.2: Comparação DVHs produzidos utilizando IPOPT e fmincon para o ficheiro "Cabeça e Pescoço", onde — indica volumes alvo e - - - indica OARs e NTs.

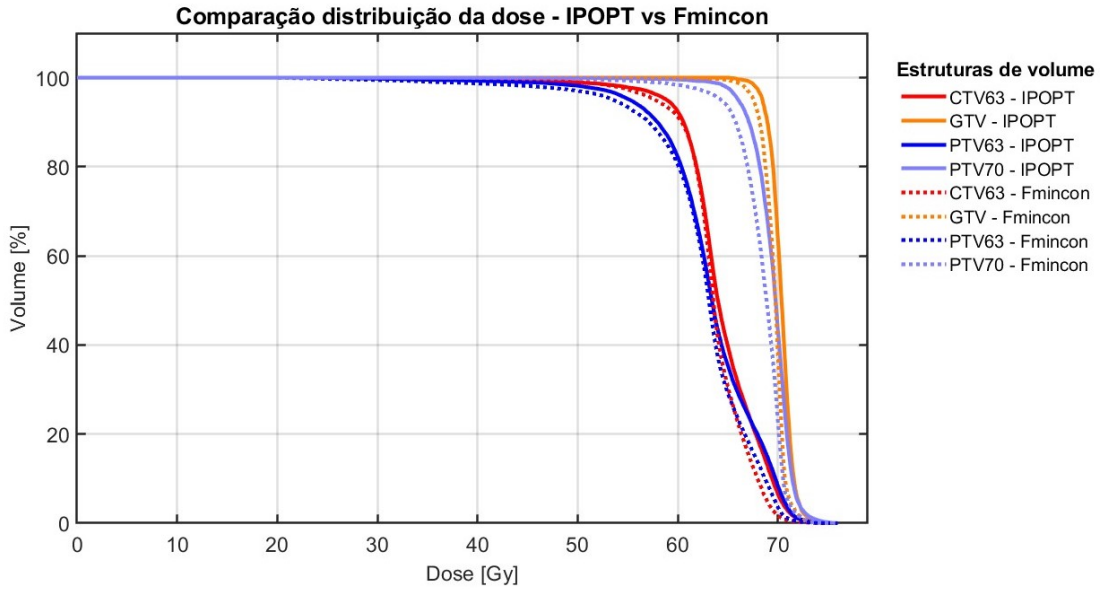


Figura 5.3: Comparação da cobertura alvo obtida utilizando IPOPT e fmincon para o ficheiro "Cabeça e Pescoço". Nota: as doses desejadas para CTV63, GTV, PTV63 e PTV70 são 63 Gy, 70Gy, 63 Gy e 70 Gy, respetivamente.

Cobertura Alvo	IPOPT	Fmincon
D_{95} PTV63	54,91 Gy	54,08 Gy
D_{95} PTV70	66,04 Gy	64,58 Gy
D_{95} CTV63	58,91 Gy	58,19 Gy
D_{95} GTV	68,45 Gy	67,75 Gy

Tabela 5.5: Cobertura alvo (D_{95}) obtida com o IPOPT e *fmincon* para estruturas de volume alvo presentes no ficheiro "Cabeça e Pescoço"

Ambos os *solvers*, IPOPT e Fmincon, resultaram numa cobertura alvo semelhante para todas as estruturas. Essas diferenças estão dentro de uma faixa aceitável, com IPOPT mostrando uma cobertura de dose ligeiramente superior.

5.4 Otimização usando a GPU

Tal como proposto nos objetivos deste projeto, um dos pontos principais do mesmo, é utilizar a **GPU** de forma a otimizar os cálculos necessários para resolver o problema FMO. O cálculo dos mapas de fluência na IMRT envolve a utilização de algoritmos de otimização que requerem grandes cálculos matriciais, nomeadamente multiplicações matriz-vetor. Estes cálculos podem ser morosos quando efetuados numa CPU, especialmente para planos de tratamento grandes e complexos. Desta forma, foi carregada a matriz de dose d e o vetor de pesos w para a GPU utilizando CUDA. De seguida, foi executado o cálculo especificado na equação 1. Após, a computação da solução na GPU é feito o *download* da mesma novamente para a CPU.

A tabela 5.6 apresenta o tempo de execução para cada uma das etapas descrevidas, em segundos, relativamente ao ficheiro "Cabeça e Pescoço" usado a máquina 1 e a máquina 2.

Etapa	RTX 3080 (s)	Quadro P600 (s)
Upload Matriz	0,912	0,582
Upload Vetor	0,0063	0,0086
Multiplicação	0,041	0,0071
Downnload	1,946	1,256
Total	2,905	1,854

Tabela 5.6: Tempo de execução, em segundos, da versão da multiplicação matriz-vetor na GPU para o ficheiro "Cabeça e Pescoço", usado a máquina 1 (RTX 3080) e a máquina 2 (Quadro P600).

É importante notar que a matriz de dose, d, foi carregada apenas uma vez e mantida na memória da gráfica durante toda a execução do algoritmo, enquanto que o vetor de intensidade atualizado foi carregado e descarregado por cada iteração do algoritmo.

A tabela 5.7 apresenta o tempo de execução, em segundos, comparando a performance da multiplicação matriz-vetor nas diferentes unidades de processamento para o ficheiro "Cabeça e Pescoço" usado a máquina 1 e a máquina 2.

Unidade de Processamento	Máquina 1 (s)	Máquina 2 (s)
CPU	1,355	3,145
GPU	2,905	1,854

Tabela 5.7: Comparação do tempo de execução entre a versão na CPU e na GPU, para o ficheiro "Cabeça e Pescoço" usado a máquina 1 e a máquina 2.

Observando as tabelas 5.6 e 5.7 pode-se retirar variadas conclusões sobre os dados recolhidos. Em primeiro, é possível verificar uma discrepância nos resultados entre as máquinas utilizadas. Para a máquina 2 (Quadro P600) o tempo de processamento da GPU (1,85 segundos), incluindo o *upload* e *download* de dados, é inferior ao tempo de processamento da CPU (3,15 segundos), indicando que o processamento da GPU é mais eficiente. Portanto, para a máquina 2 enviar os dados para a GPU e realizar o cálculo na mesma resulta num speed-up de 41%. Em contrapartida, para a máquina 1 (RTX 3080) o tempo de processamento total da GPU (2,905 segundos) é superior ao tempo de processamento da CPU (1,36 segundos), sugerindo que a vantagem da GPU vista com a Quadro P600 não se pronuncia neste cenário.

Além disso, reparou-se que para qualquer cenário, a eficácia da multiplicação matriz-vetor na GPU é claramente superior face à CPU. Isto é evidenciado pelos tempos de execução consideravelmente mais baixos (0,041 e 0,0071 segundos) quando comparados com a CPU. Para quantificar com precisão esta melhoria, calculou-se o fator de aceleração comparando os tempos de execução de ambas as unidades, o que resulta numa aceleração de aproximadamente 33 vezes para a máquina 1 (RTX 3080) e 443 vezes para a máquina 2 (Quadro P600).

No entanto, verifica-se que a rapidez da multiplicação matriz-vetor na GPU é contraposta face à sobrecarga introduzida pelos *uploads* e *downloads* de dados entre a CPU e a GPU. Particularmente, o tempo despendido no *download* do resultado da multiplicação da GPU para a CPU a cada iteração do algoritmo. Uma possível solução para aliviar esta sobrecarga, particularmente na máquina 1, seria uma intervenção de *Hardware*.

Tal como descrito na subsecção 5.1.2, a configuração da máquina 1 apresenta uma interface PCIe x16 com velocidades PCIex1, devido ao uso de PCIe Risers x1 para x16. Esta interface consiste em uma única via de transmissão de dados, fornecendo uma largura de banda bastante reduzida comparada com configurações

superiores. Assim, aproveitado todo o potencial da interface PCIe x16 teríamos dezasseis pistas dedicadas à transmissão de dados. Este aumento acentuado no número de faixas possibilita paralelismo e traduz-se numa melhoria substancial na capacidade de transferência de dados, possibilitando diminuir o tempo de transferência em 16 vezes.

A tabela 5.8 apresenta o tempo de execução, em segundos, comparando agora os resultados para dois casos de testes, "Cabeça e Pescoço" e "Pulmões", usado a máquina 1. Para além de se definir o tempo de execução real, apresenta-se o tempo de execução teórico que poderíamos obter atualizando o *hardware*, de forma a usarmos as capacidades completas da PCIe x16 da máquina 1.

Etapa	Cabeça e Pescoço (s)	Cabeça e Pescoço - Teórico (s)	Pulmões (s)	Pulmões - Teórico (s)
Upload Matriz	0,91	0,057	3,53	0,2204
Upload Vetor	0,006	0,0004	0,019	0,0012
Multiplicação	0,041	0,041	0,138	0,138
Download	1,946	0,122	130,06	8,129
Total	2,905	0,220	133,747	8,489

Tabela 5.8: Tempo de execução, em segundos, da versão da multiplicação matriz-vetor na GPU para os ficheiros "Cabeça e Pescoço" e "Pulmões", usado a máquina 1 (RTX 3080).

Esta alteração de *hardware* representa um *speed-up* em relação à solução real na GPU, de aproximadamente 92-94%.

A tabela 5.9 apresenta o tempo de execução, em segundos, para a versão original (CPU), a versão modificada (GPU) e a versão modificada teórica (GPU teórico), para os ficheiros "Cabeça e Pescoço" e "Pulmões" usando a máquina 1.

Ficheiro	CPU (s)	GPU (s)	GPU Teórico (s)	Speed-Up Teórico (%)
Cabeça e Pescoço	1,355	2,905	0,22	83,76
Pulmões	55,01	133,75	8,49	84,57

Tabela 5.9: Tempo de execução na CPU, GPU e GPU Teórico, para os ficheiros "Cabeça e Pescoço" e "Pulmões" usando a máquina 1.

Analisando os resultados da tabela 5.9 verifica-se que a combinação da integração da GPU com as devidas atualizações, pode gerar um *speed-up* em relação ao código original na CPU de aproximadamente 83 a 85%. A figura 5.4 apresenta uma

representação gráfica destes resultados para trinta execuções usando a máquina 1.

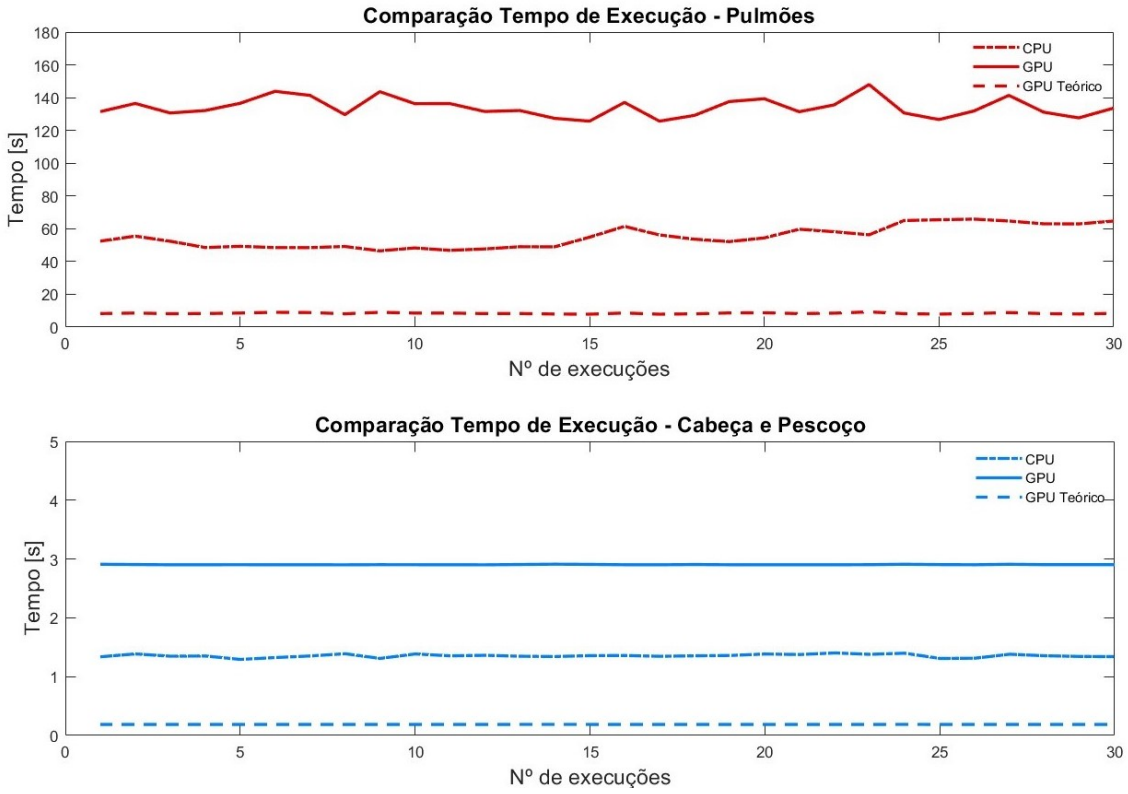


Figura 5.4: Comparação do tempo de execução na CPU, GPU e GPU Teórico, para os ficheiros "Cabeça e Pescoço" e "Pulmões" usando a máquina 1.

Resumindo, a análise dos resultados destaca que a multiplicação matriz-vetor é significativamente acelerada quando realizada na GPU em comparação com a CPU. Embora a transferência de dados de e para a GPU introduza sobrecarga no tempo de execução, a otimização das configurações de *hardware* pode atenuar esse atraso. Assim, a nossa abordagem na GPU apresenta vantagens potenciais, desde que exista um equilíbrio adequado do *hardware* de forma a agilizar a transferência de dados e aproveitar todos os benefícios da aceleração da GPU.

5.5 Otimização do Acesso da Matriz de Dose

Durante a realização dos testes aplicando o *fmincon*, deparou-se que o algoritmo apresentava um tempo de execução muito superior ao esperado. De forma a entender a origem desta surpresa, iniciou-se uma investigação completa com o apoio do *profiler* do MATLAB. Esta ferramenta visa ajudar o utilizador a compreender as características de desempenho do seu código, fornecendo informações detalhadas sobre quanto tempo é despendido em cada parte do programa, quais funções são chamadas e quantas vezes são chamadas.

Na tabela 5.10 encontra-se um exemplo de uma análise executada pelo *profiler*. É importante perceber que os valores temporais fornecidos pelo *profiler* não são

valores reais, apenas aproximações dos mesmos.

Linha	Código	Chamadas	Tempo Total (s)	Tempo (%)
64	$inflTmp = d(cst\{structIndex(i), 4\}\{1\}, :);$	1632	1818,99	62,6
40	$d = matRad\_backProjection (w, dij, options);$	204	869,89	29,9
72	$grad(:, 1) = ... * Ftemp/numVoxels;$	1632	176,89	6,1
88	$grad(:, 1) = ... * Ftemp/numVoxels;$	408	35,09	1.2
92	$H = hessian;$	204	2,71	0,1
Restantes	-	-	-	0

Tabela 5.10: Tabela gerada pelo *profiler* do MATLAB na análise das linhas que demoraram mais tempo a ser executadas, usando o ficheiro "Cabeça e Pescoço"

Através desta análise, descobriu-se que o método de **acesso aos dados** da matriz de dose (primeira linha da tabela 5.10) para o cálculo do gradiente era notavelmente **ineficiente**, levando a um desempenho lento. Em vez de se aceder diretamente à matriz de dose, *d*, o acesso era encadeado, sendo necessário aceder primeiro ao índice da estrutura de volume desejada, depois recolher os índices correspondentes a essa estrutura guardados na *cst* e por fim, utilizar esse conjunto de índices para aceder aos valores de dose registados na matriz de dose.

Dado que esta operação é executada várias centenas de vezes, acaba por afetar negativamente o desempenho geral do algoritmo, resultando em aproximadamente 62% do tempo despendido pela função objetivo. Reconhecendo esta restrição, alterou-se este acesso de forma a tornar-lho mais direto. Assim, antes de executar o *fmincon*, separou-se a matriz num vetor de matrizes, em que cada célula corresponde a todos os pontos de dose de uma estrutura de volume específica (PTV, coração, pulmão, por exemplo). Logo, para cada chamada da função objetivo durante a execução o algoritmo, basta selecionar a célula correspondente à estrutura de volume desejada: *dPerStruct(i)*.

Esta intervenção resultou num processo significativamente simplificado e otimizado, permitindo uma maior eficiência computacional, que pode ser observada com base nos resultados reais das duas versões utilizando o *fmincon*, presentes na tabela 5.11.

Ficheiro	Tempo Original (s)	Tempo Modificado (s)	Speed Up (%)
Cabeça e Pescoço	2244,963	97,859	95,64
Pulmões	9214,834	462,758	94,97
Estômago	12268,73	2361,391	80,75
Pélvis	16216,349	3783,76	76,67

Tabela 5.11: Comparação dos tempos de execução da versão do fmincon original e modificada, usando a máquina 1

Assim, analisando a tabela 5.11, pode-se testemunhar que a alteração do acesso à matriz de dose resultou num tempo de execução significativamente inferior, convertendo-se num *speed-up* de aproximadamente 77% a 96%. Sendo que quanto menor é o tamanho da matriz de dose, como a matriz do ficheiro "Cabeça e Pescoço", maior é o ganho obtido. Para este ficheiro, verifica-se ainda que com esta modificação é possível gerar vinte e dois planos de tratamento no tempo em que se geraria apenas um plano com a versão original do fmincon.

Após estas alterações comparou-se os tempos de execução do IPOPT e fmincon, estes dados encontram-se tabelados em 5.12.

Ficheiro	IPOPT (s)	Fmincon Modificado (s)
Cabeça e Pescoço	4,68	97,859
Pulmões	57,79	462,758
Estômago	102,45	2361,391
Pélvis	305,96	3783,76

Tabela 5.12: Comparação dos tempos de execução entre o IPOPT e o fmincon modificada, usando a máquina 1

Em última análise, apesar de se ter reduzido significativamente o tempo de execução do fmincon com estas modificações, o IPOPT continua a ser o algoritmo mais eficiente em termos temporais.

5.6 Variação do Ponto Inicial

Após estas experiências e a aquisição de um maior conhecimento do problema e do código de cada algoritmo, foram ponderadas outras formas de exploração do

comportamento dos mesmos. Assim, tanto para o fmincon como para o IPOPT foi analisada como a **variação do vetor de pesos inicial**, w_i, afeta o tempo e resultado do problema. Os dois objetivos principais desta estratégia são perceber se é possível gerar mais do que um plano admissível e se existe alguma janela de pontos que convirja mais rapidamente para a solução desejada. Além disso, deseja-se igualmente observar como a variação deste vetor inicial afeta a qualidade da solução.

Assim, para o nosso problema original, cada célula do vetor de pesos inicial corresponde à razão matemática entre a dose alvo máxima do plano de tratamento e a média de todas as doses alvo presentes no mesmo. O resultado desta razão corresponde ao peso de cada sub-feixe. Para introduzir variabilidade na solução, criou-se uma janela de valores aleatórios em torno desta razão, e foi-se alargando o tamanho da mesma durante as experiências. Desta forma, injetou-se no algoritmo, sub-feixes com diferentes intensidades. Para estas experiências usaram-se os ficheiros "Cabeça e Pescoço" e "Pulmões", em que os pesos do sub-feixe iniciais são 21,64 Gy e 10,87 Gy, respetivamente. Portanto, foram gerados trinta planos para cada ficheiro e cada tamanho da janela de valores, usando o IPOPT e o fmincon.

Para clarificação, quando se refere que o tamanho de janela é igual a um, por exemplo, significa que o novo intervalo para o vetor de pesos inicial, seguirá a forma: $v_1 = [r \pm 1]$, onde r é a razão calculada.

Numa primeira análise, pode-se observar pela tabela 5.13 que por norma, ampliar o tamanho da janela de valores iniciais, aumenta o tempo de execução médio.

		Cabeça e Pescoço		Pulmões	
Solver	Janela	Média (s)	Std (s)	Média (s)	Std (s)
IPOPT	0	4,677	0,094	57,791	1,826
	1	4,972	0,364	68,282	11,249
	2	5,026	0,433	67,492	13,039
	4	5,104	0,364	73,859	13,872
fmincon	0	96,507	3,91	455,678	28,688
	1	99,367	5,391	440,183	14,073
	2	102,796	4,21	441,545	10,738
	4	105,679	13,719	436,44	12,561

Tabela 5.13: Tempos de execução, em segundos, para o IPOPT e o fmincon modificado para diferentes tamanhos da janela do peso inicial do sub-feixe, usando a máquina 1

Apesar disto, reparou-se que o desvio padrão do tempo de execução também aumenta. O que isto nos sugere que ao variar o vetor de pontos iniciais, o conjunto

de tempo de execução é maior e mais variado. Desta forma, com diferente variações do vetor de pesos iniciais, podemos ter planos mais rápidos ou mais mais lentos.

Na tabela 5.14 pode-se ver com mais detalhe os tempos de execução mínimos e máximos obtidos para o ficheiro "Pulmões" usando o IPOPT. Assim, verifica-se que o tempo médio de execução para o vetor de pesos estáticos é 57,8 segundos. Porém, variando a janela de valores do vetor de pesos, notou-se que o plano mais rápido a ser gerado demorou 43,5 segundos, enquanto que o plano mais lento demorou 98,3 segundos.

Pulmões					
Solver	Janela	Média (s)	Std (s)	Min (s)	Max (s)
IPOPT	0	57,791	1,826	54,582	61,388
	1	68,282	11,249	43,537	87,650
	2	67,492	13,039	44,833	87,529
	4	73,859	13,872	50,726	98,263

Tabela 5.14: Tempos de execução, em segundos, para o ficheiro "Pulmões" para diferentes tamanhos da janela do vetor de pesos inicial, usando o IPOPT na máquina 1

Numa segunda análise, verificou-se como é que a variação do vetor de pesos inicial afeta a distribuição das doses para as diferentes estruturas de volume.

Assim, os gráficos das figuras 5.5, 5.6, 5.7 e 5.8 ilustram a dispersão da dose para quatro estruturas de volume (dois OARs - tronco cerebral e laringe - e dois volumes alvo - CTV63 e PTV70) relativos ao ficheiro "Cabeça e Pescoço", variando o tamanho da janela, usando o IPOPT.

Para ambos os órgãos em risco representados em 5.5 e 5.6 pode-se ver que, usando o IPOPT, a dispersão da dose aumenta consoante o tamanho da janela de valores escolhida. Por outro lado, reparou-se também que a média da dispersão da dose, mantém o mesmo modelo, mesmo variando o tamanho da janela. O mesmo comportamento é verificado nos restantes OARs e NTs.

Em contrapartida, para os volumes alvo representados em 5.7 e 5.8 pode-se observar que aumentar o tamanho da janela não afeta de forma expressiva a distribuição da dose alvo.

Estes dados permitem-nos supor que usando o IPOPT e variando os valores de pesos iniciais dos sub-feixes, é possível obter planos de tratamento diferentes sem comprometer a cobertura alvo. Esta alteração permite portanto selecionar o plano que melhor minimiza a dose absorvida pelos órgãos em risco e tecidos normais.

Analisando agora os dados para o ficheiro "Pulmões" usando o IPOPT, queremos perceber se de facto esta tendência se mantém. Deste modo, as figuras 5.9, 5.10,

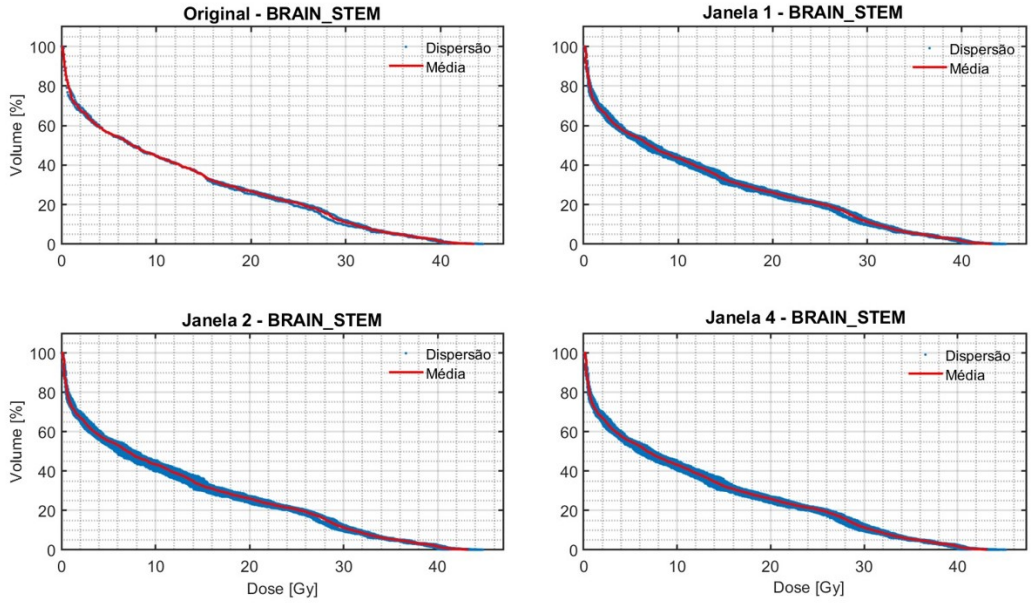


Figura 5.5: Comparação da dispersão da dose para o tronco cerebral, resultado de variação do vetor de pesos inicial para quatro janelas distintas, para o ficheiro "Cabeça e Pescoço" usando o IPOPT na máquina 1.

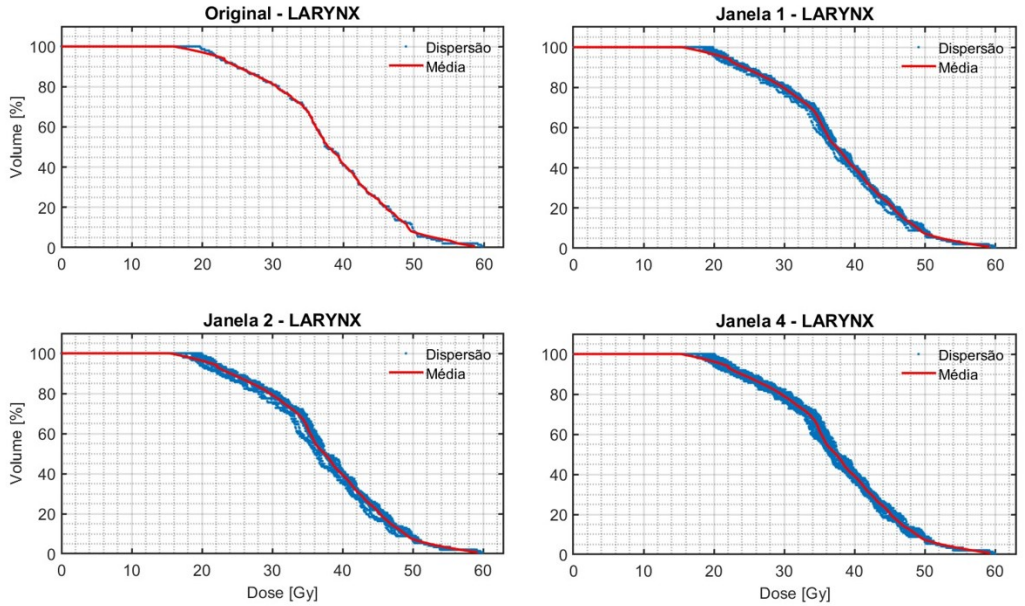


Figura 5.6: Comparação da dispersão da dose para a Laringe, resultado de variação do vetor de pesos inicial para quatro janelas distintas, para o ficheiro "Cabeça e Pescoço" usando o IPOPT na máquina 1.

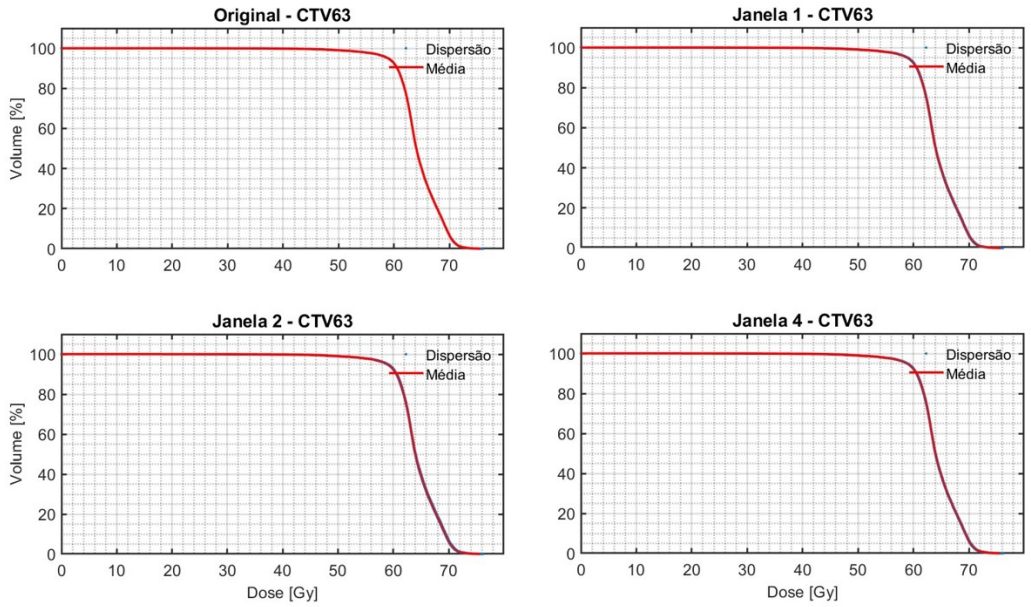


Figura 5.7: Comparação da dispersão da dose para o CTV63, resultado de variação do vetor de pesos inicial para quatro janelas distintas, para o ficheiro "Cabeça e Pescoço" usando o IPOPT na máquina 1.

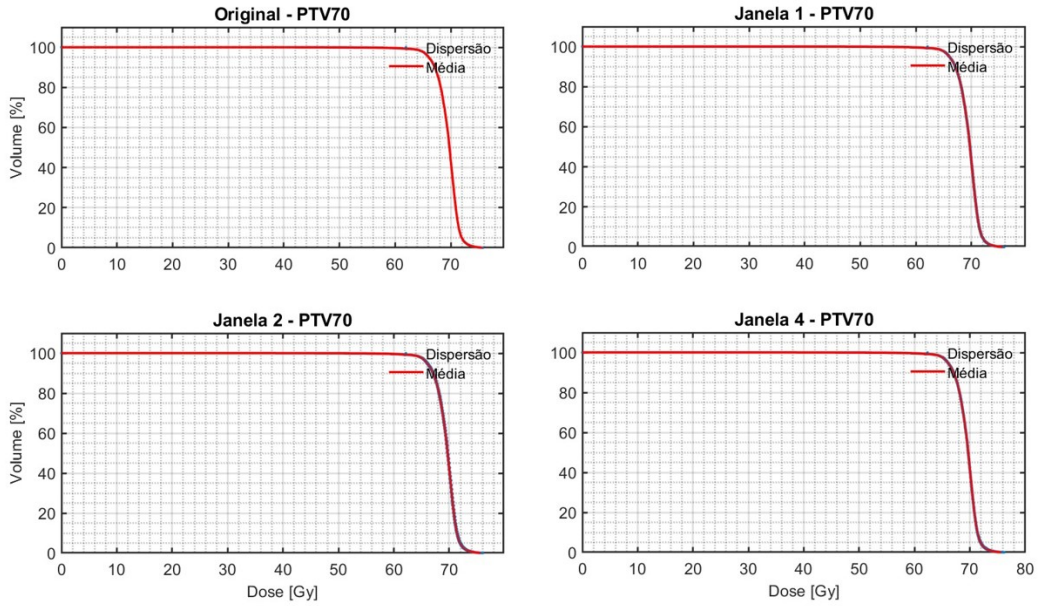


Figura 5.8: Comparação da dispersão da dose para o PTV70, resultado de variação do vetor de pesos inicial para quatro janelas distintas, para o ficheiro "Cabeça e Pescoço" usando o IPOPT na máquina 1.

5.11 e 5.12, expõem a dispersão da dose para quatro estruturas de volume (dois OARs - coração e pulmão esquerdo - e dois volumes alvo - PTV e ITV) variando o tamanho da janela.

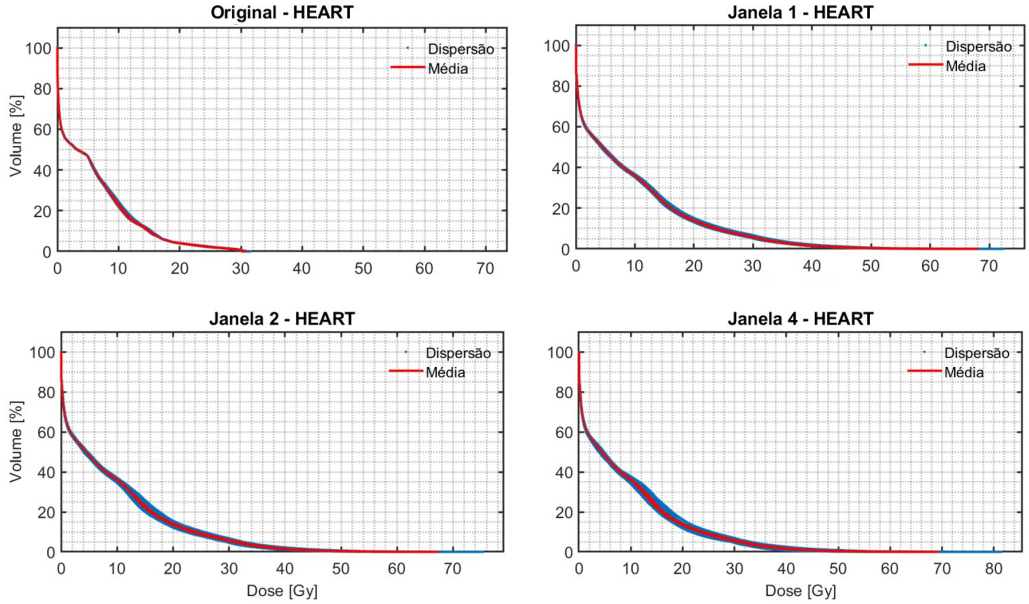


Figura 5.9: Comparação da dispersão da dose para o coração, resultado de variação do vetor de pesos inicial para quatro janelas distintas, para o ficheiro "Pulmões" usando o IPOPT na máquina 1.

Para este caso de teste, constatou-se que a distribuição de dose dos OARs e NTs continua a variar com a variação do tamanho da janela. Este comportamento poder ser observado nos gráficos das figuras 5.9 e 5.10.

Já para os volumes alvo representados em 5.11 e 5.12 pode-se ver que o comportamento registado anteriormente, também se mantém, pelo que aumentar o tamanho da janela altera a distribuição da dose alvo de forma expressiva.

O padrão destes resultados apoia a hipótese que é possível, usando o IPOPT e variando os valores de pesos iniciais dos sub-feixes, gerar planos de tratamento diferentes, que não prejudicam a cobertura alvo, e poder selecionar aqueles que melhor minimizam a dose absorvida pelos órgãos em risco e tecidos normais.

A mesma análise foi estendida aos resultados obtidos pelo fmincon. Assim para o ficheiro "Cabeça e Pescoço" usando o fmincon, selecionou-se quatro estruturas de volume (dois OARs - tronco cerebral e laringe - e dois volumes alvo - CTV63 e PTV70) como anteriormente, representadas nas figuras 5.13, 5.14, 5.15 e 5.16, de forma a verificar se neste caso, a dispersão da dose também varia com o tamanho da janela.

Outra vez, vê-se que para ambos os órgãos em risco (5.13 e 5.14) usando o fmincon, a dispersão da dose aumenta com o aumento do tamanho da janela de valores escolhida e que a tendência da dispersão da dose mantém o mesmo modelo ao longos dos vários tamanhos da janela. O mesmo comportamento é verificado nos restantes OARs e NTs do plano de tratamento.

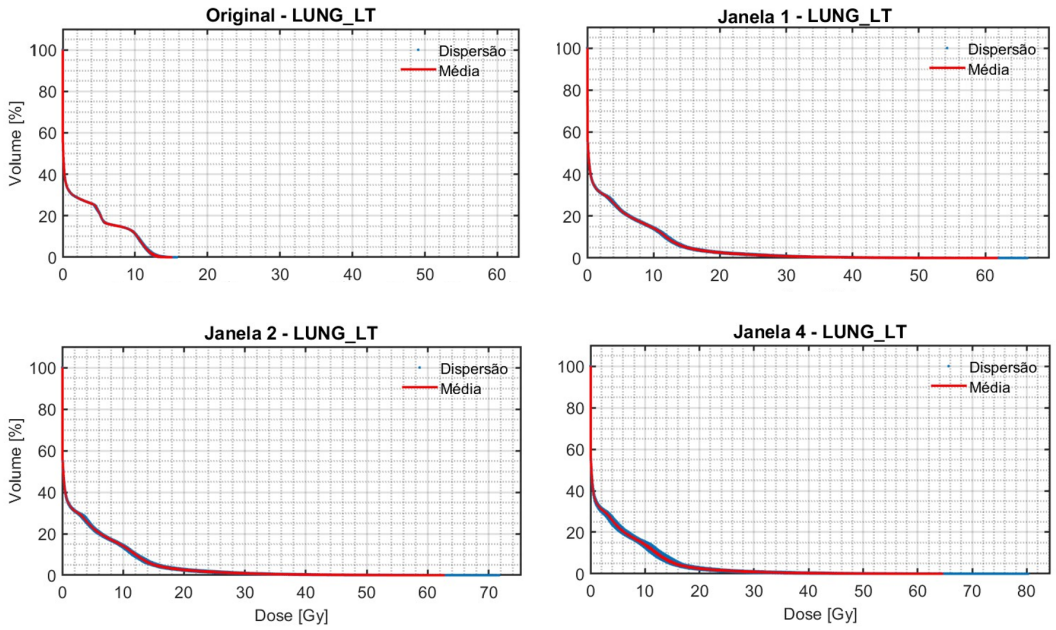


Figura 5.10: Comparação da dispersão da dose para o pulmão esquerdo, resultado de variação do vetor de pesos inicial para quatro janelas distintas, para o ficheiro "Pulmões" usando o IPOPT na máquina 1.

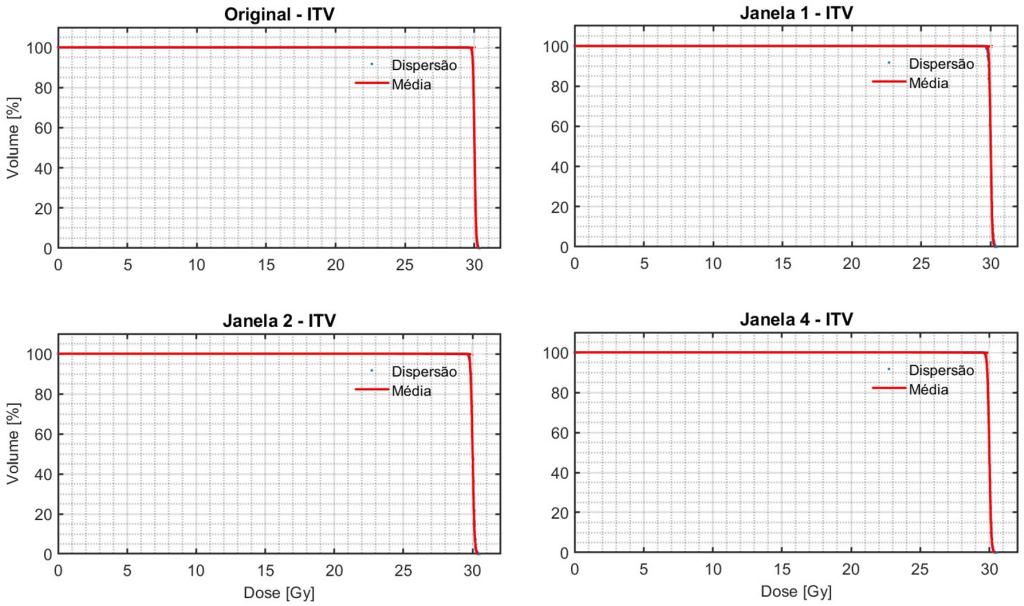


Figura 5.11: Comparação da dispersão da dose para o ITV, resultado de variação do vetor de pesos inicial para quatro janelas distintas, para o ficheiro "Pulmões" usando o IPOPT na máquina 1.

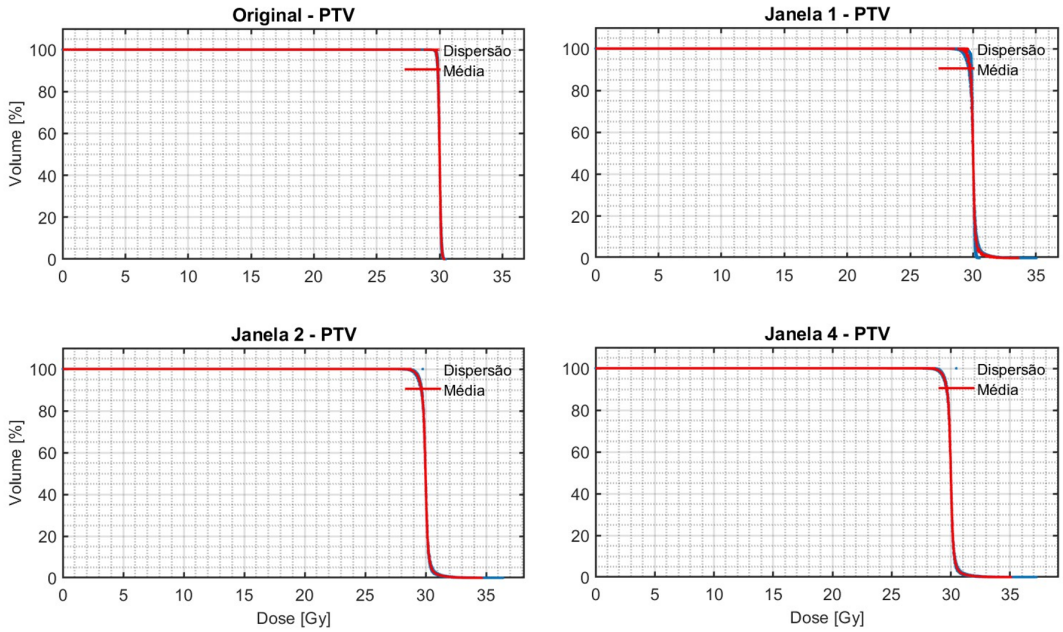


Figura 5.12: Comparação da dispersão da dose para o PTV, resultado de variação do vetor de pesos inicial para quatro janelas distintas, para o ficheiro "Pulmões" usando o IPOPT na máquina 1.

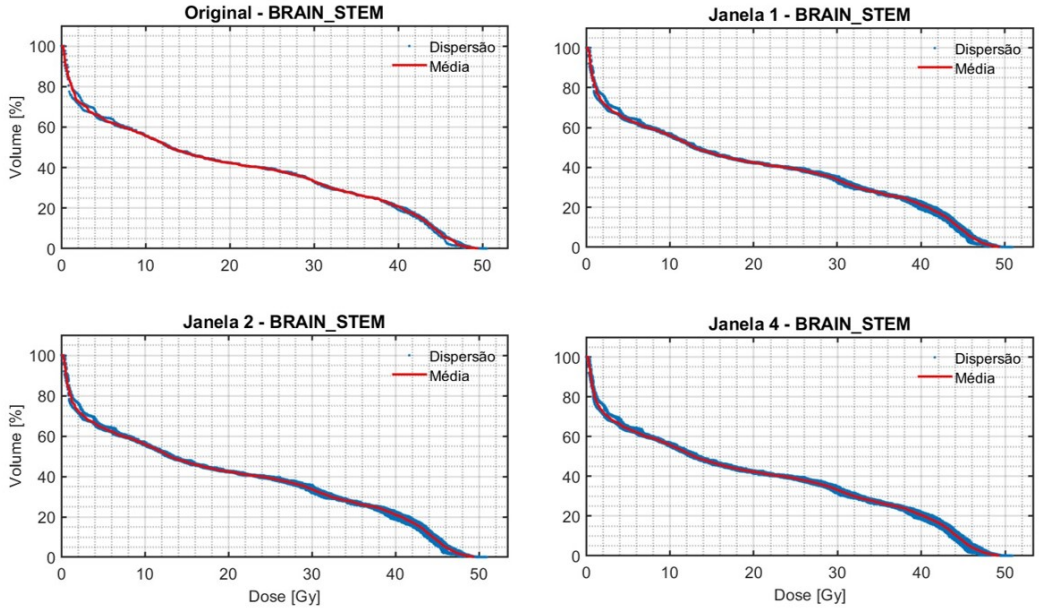


Figura 5.13: Comparação da dispersão da dose para o tronco cerebral, resultado de variação do vetor de pesos inicial para quatro janelas distintas, para o ficheiro "Cabeça e Pescoço" usando o fmincon na máquina 1.

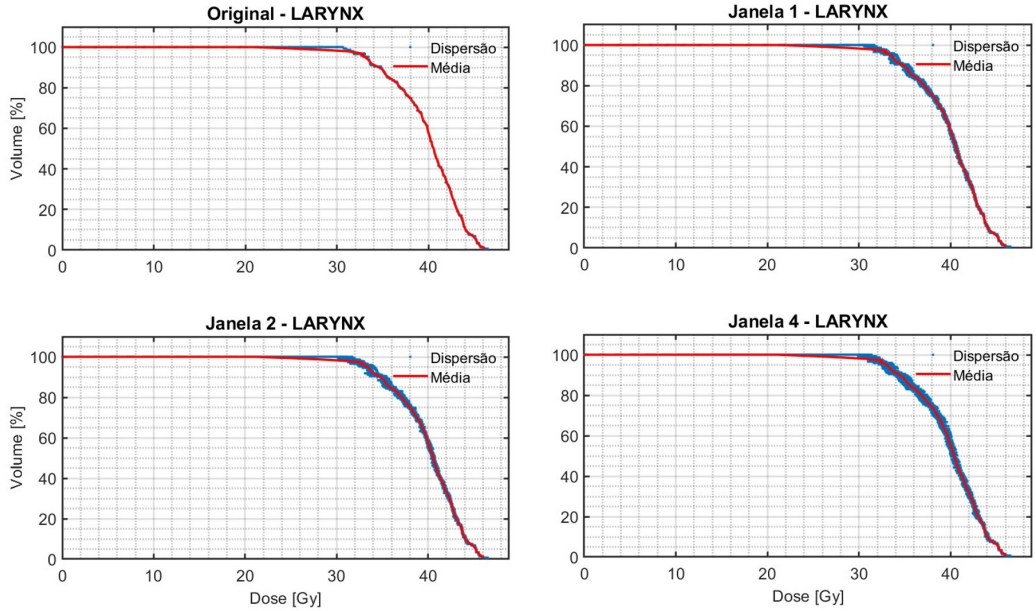


Figura 5.14: Comparação da dispersão da dose para a Laringe, resultado de variação do vetor de pesos inicial para quatro janelas distintas, para o ficheiro "Cabeça e Pescoço" usando o fmincon na máquina 1.

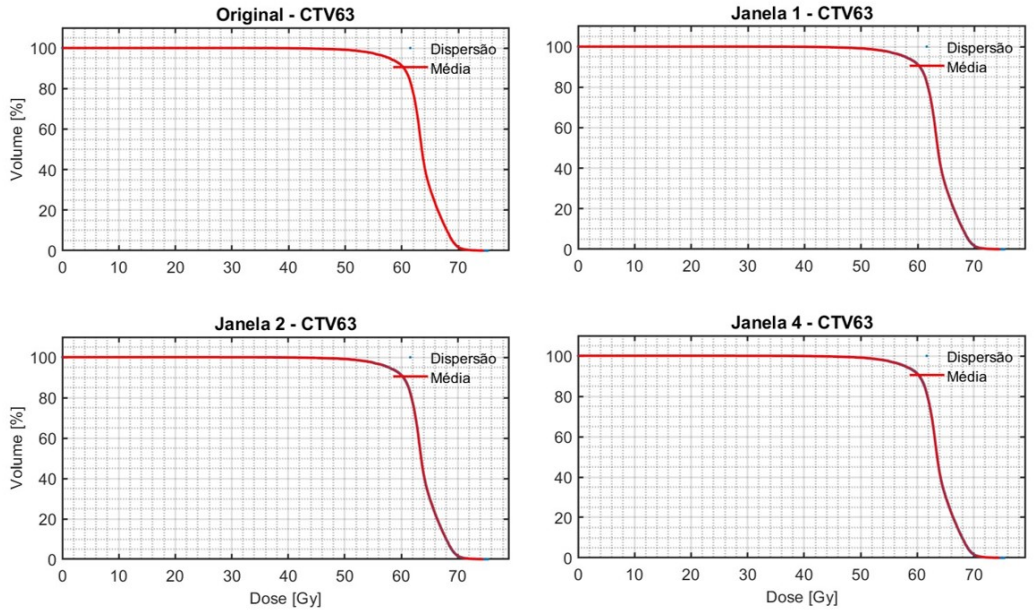


Figura 5.15: Comparação da dispersão da dose para o CTV63, resultado de variação do vetor de pesos inicial para quatro janelas distintas, para o ficheiro "Cabeça e Pescoço" usando o fmincon na máquina 1.

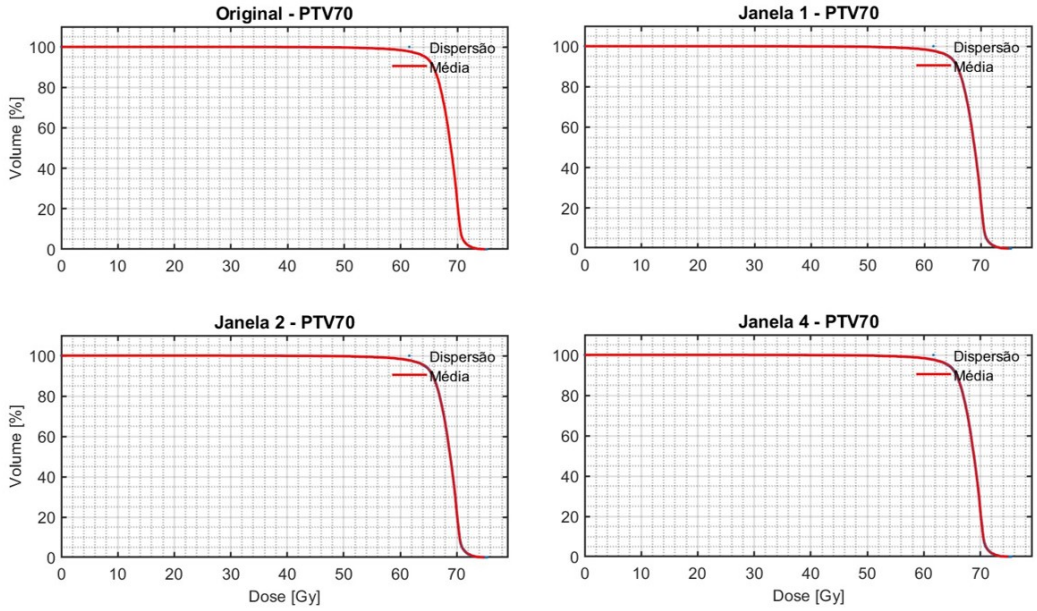


Figura 5.16: Comparação da dispersão da dose para o PTV70, resultado de variação do vetor de pesos inicial para quatro janelas distintas, para o ficheiro "Cabeça e Pescoço" usando o fmincon na máquina 1.

De forma similar, usando o fmincon, constatou-se que para os volumes alvo (5.15 e 5.16), aumentar o tamanho da janela não afeta a tendência da distribuição da dose alvo.

Estes dados salientam que **independentemente do algoritmo** de otimização usado (IPOPT ou fmincon), variar os valores de pesos iniciais dos sub-feixes permitem-nos obter **planos de tratamento distintos** que não comprometem a cobertura alvo. Isto permite ao dosimetrista ter alternativas que pode analisar e comparar para escolher o plano que melhor se adequa ao paciente em questão, valorizando o seu histórico pessoal e a sua individualidade, podendo inclusivamente focar-se na preservação de órgãos em risco e tecidos normais específicos.

Capítulo 6

Conclusão

Resumindo, a radioterapia modelada por intensidade (IMRT) é uma técnica de radioterapia que usa o planeamento inverso e permite a entrega de uma dose de radiação suficiente à região cancerígena de forma a esterilizar o tumor, minimizando ao mesmo tempo a dose recebida pelos tecidos saudáveis circundantes. Esta técnica levanta vários desafios associados à criação de planos de tratamento, nos quais se destaca o problema de otimização de mapas de fluências (FMO). Esta dissertação, enquadrada dentro do projeto ORION, intentou facilitar a geração automática de planos de tratamento ao resolver este problema de forma mais eficiência.

Uma das etapas do problema FMO envolve o cálculo da dose absorvida em cada ponto do corpo do paciente, devido ao efeito de cada feixe por cada ângulo de incidência. Isto resulta numa matriz de dose, d, que contém os valores de dose unitários para cada *beamlet* em cada ponto do corpo do paciente. O tamanho desta matriz varia muito em função do plano de tratamento específico, da anatomia do paciente ou da resolução da imagem 3D (TAC). Portanto, o seu tamanho pode atingir a ordem de várias centenas de *megabytes* ou mesmo de alguns *gigabytes*.

Para minimizar o impacto deste problema comprometemos-nos a usar estruturas de dados e algoritmos eficientes para representar e manipular a matriz de dose. Assim, para tirar partido da quantidade de zeros presentes nesta matriz, que resultam dos pontos do corpo que não irão receber radiação, utilizou-se uma matriz esparsa de forma a economizar espaço de armazenamento, bem como tempo de *upload* e *download* da matriz e cálculos matriciais. Desta forma, verificou-se que para os quatro casos de teste estudados, é possível reduzir o tamanho da matriz de dose em 98,8% a 99,2%.

Outra abordagem que se propôs examinar, foi a utilização da GPU para acelerar o cálculo do vetor de pesos otimizado. O cálculo dos mapas de fluência na IMRT envolve a utilização de algoritmos de otimização que requerem grandes cálculos matriciais, nomeadamente multiplicações matriz-vetor, entre a matriz de dose, d, e o vetor de pesos, w. Assim, com o auxílio de CUDA, uma plataforma de computação paralela e modelo de programação desenvolvido pela NVIDIA para computação de uso geral em GPUs, enviou-se a matriz de dose e o vetor de pesos para a GPU de forma a otimizar a multiplicação entre estes elementos. A utiliza-

ção da GPU para multiplicação matriz-vetor apresentou um aumento notável de desempenho, com um tempo de execução 33 a 443 vezes inferior àquele registada na CPU, para as máquina 1 e 2, respetivamente. Embora as operações de transferência de dados, *upload* e *download*, entre a CPU e GPU tenham contrabalançado o sucesso dos cálculos na GPU, principalmente na máquina 1, ilustrou-se que, com as configurações de *hardware* corretas, esses problemas podem ser atenuados.

Além disto, estudou-se vários algoritmos de otimização quadráticos, focando-nos no método de pontos interiores primal-dual implementado pelo IPOPT e no método de região de confiança refletivo implementado pelo fmincon em MATLAB. Numa primeira análise, o *solver* IPOPT mostrou resultados deveras interessantes, superando o fmincon para qualquer caso de teste. Mesmo com as modificações ao acesso de dados no fmincon, que demonstraram melhorias notáveis no tempo de execução, obtendo um *speed-up* de 77% a 96%, o IPOPT continua a superar o fmincon por uma margem substancial. Assim, comparando o IPOPT com a versão modificada do fmincon, o primeiro prova ser 8 a 23 vezes mais rápido para os quatro casos de teste.

Finalmente, explorou-se o impacto da introdução de variabilidade no vetor de pesos inicial usado pelo algoritmo de otimização. Os resultados obtidos revelaram que esta variação tendeu a prolongar o tempo médio de geração de um plano. No entanto, também se verificou que a dispersão dos tempos de execução aumenta e que é possível obter planos mais rápidos do que quando se usou um vetor inicial estático. Portanto, embora esta variabilidade possa tornar a geração um pouco mais lenta, em média, também abre a porta para a realização ocasional de planos de tratamento num período de tempo mais rápido.

Também se reparou, que esta variabilidade não comprometeu o alcance da cobertura alvo nos planos de tratamento. Por outro lado, notou-se a variação na distribuição da dose absorvida pelos órgãos em risco e tecidos normais. Esta observação sugere que, ao abraçar a variabilidade do vetor de pesos inicial, pode-se gerar um conjunto de planos de tratamento diferentes que atendem aos critérios de dose-alvo. Esta flexibilidade abre possibilidades para a seleção de planos que não aderem apenas a esses critérios, mas também exibem preservação de OARs e NTs superior.

Concluindo, começou-se por estudar o problema associado à criação de mapas de fluência de forma a identificar as restrições associadas ao módulo de Geração Automática de Planos enquadrado no projeto ORION. Este estudo incluiu a compreensão do fluxo de dados do módulo de geração, bem como as caraterísticas das estruturas de dados geradas. A partir desse ponto, identificaram-se o tamanho da matriz de dose, bem como o elevado número de multiplicações matriz-vetor, como duas das restrições mais marcantes. O que permitiu eleger como possíveis soluções, a utilização de matrizes esparsas e computação gráfica com auxílio de CUDA. Também se garantiu que o módulo de Geração Automática de Planos era compatível com diferentes máquinas e gráficas. Ainda foram exploradas as diferenças de dois algoritmos de otimização quadrática, o método de pontos interiores primal-dual implementado pelo IPOPT e o método de região de confiança refletivo implementado pelo fmincon em MATLAB. Por fim, foi abordada como a inserção de variabilidade do vetor de pesos inicial afeta o

tempo de execução e a qualidade de um plano de tratamento.

Desta forma, ao refletir sobre o culminar deste projeto, concluímos que se atingiu os nossos objetivos através de uma pesquisa marcada por uma exploração minuciosa do problema, juntamente com a proposta de várias de estratégias para lidar com as restrições identificadas.

Referências

[1] Choosing the algorithm. "https://uk.mathworks.com/help/optim/ug/choosing-the-algorithm.html". (Acedido em April 2023).

[2] Cpu vs gpu. "https://www.heavy.ai/technical-glossary/cpu-vs-gpu". (Acedido em Outubro 2022).

[3] fmincon. "https://www.mathworks.com/help/optim/ug/fmincon.html". (Acedido em Janeiro 2023).

[4] Ipopt documentation. "https://coin-or.github.io/Ipopt/". (Acedido em Março 2023).

[5] Radioterapia. "https://www.ipolisboa.min-saude.pt/sobre-o-cancro/tratamento/radioterapia/". (Acedido em Setembro 2022).

[6] Tipos de tratamento do cancro: Radioterapia. "https://www.sns24.gov.pt/tema/doencas-oncologicas/tipos-de-tratamento-do-cancro/radioterapia/". (Acedido em Setembro 2022).

[7] Nvidia cuda architecture - introduction & overview. "https://developer.download.nvidia.com/compute/cuda/docs/CUDA_Architecture_Overview.pdf", apr 2009. (Acedido em Outubro 2022).

[8] It is time to finally retire your faithful nvidia geforce 8800 gtx. "https://www.cclonline.com/article/281/Blog/Graphics-Cards/It-is-time-to-retire-your-faithful-Nvidia-GeForce-8800-GTX/", apr 2012. (Acedido em Outubro 2022).

[9] Introduction to cuda programming. "https://www.geeksforgeeks.org/introduction-to-cuda-programming/", 2021. (Acedido em Novembro 2022).

[10] Data integrity: Types, threats, and countermeasures. "https://www.altexsoft.com/blog/data-integrity/", apr 2022. (Acedido em Janeiro 2023).

[11] Non-functional requirements: Examples, types, how to approach. "https://www.altexsoft.com/blog/non-functional-requirements/", jun 2022. (Acedido em Janeiro 2023).

[12] G. K. Bahr, J. G. Kereiakes, H. Horwitz, R. Finney, J. Galvin, and K. Goode. The method of linear programming applied to radiation treatment planning. *Radiology*, 91(4):686–693, 1968. PMID: 5677503.

[13] P. Bose. *Power Wall*, pages 1593–1608. Springer US, Boston, MA, 2011.

[14] L. W. Brady and T. E. Yaeger. Encyclopedia of radiation oncology. 2013.

[15] W. Brehm. Memory efficient hash tables and pseudorandom ordering. "https://1ykos.github.io/patchmap/". (Acedido em Março 2023).

[16] R. Cao, L. Si, X. Li, Y. Guang, C. Wang, Y. Tian, X. Pei, and X. Zhang. A conjugate gradient-assisted multi-objective evolutionary algorithm for fluence map optimization in radiotherapy treatment. *Complex & Intelligent Systems*, pages 1–27, 2022.

[17] C.-W. Cheng and I. J. Das. Treatment plan evaluation using dose–volume histogram (dvh) and spatial dose–volume histogram (zdvh). *International Journal of Radiation Oncology*Biology*Physics*, 43(5):1143–1150, 1999.

[18] P. S. Cho and M. H. Phillips. Reduction of computational dimensionality in inverse radiotherapy planning using sparse matrix operations. *Physics in Medicine & Biology*, 46(5):N117, 2001.

[19] S. Chou, F. Kjolstad, and S. Amarasinghe. Format abstraction for sparse tensor algebra compilers. *Proceedings of the ACM on Programming Languages*, 2(OOPSLA):1–30, 2018.

[20] D. C. W. M. D. Radiation therapy for treatment of cancer. "https://news.cancerconnect.com/treatment-care/radiation-therapy-for-treatment-of-cancer", 2021. (Acedido em Outubro 2022).

[21] J. Dias, H. Rocha, T. Ventura, B. Ferreira, and M. d. C. Lopes. Automated fluence map optimization based on fuzzy inference systems. *Medical physics*, 43(3):1083–1095, 2016.

[22] M. Eding. Data structures - sparse matrices. "https://matteding.github.io/2019/04/25/sparse-matrices/", 2019. (Acedido em Março 2023).

[23] M. Garland. *NVIDIA GPU*, pages 1339–1345. Springer US, Boston, MA, 2011.

[24] M. Garland, S. Le Grand, J. Nickolls, J. Anderson, J. Hardwick, S. Morton, E. Phillips, Y. Zhang, and V. Volkov. Parallel computing experiences with cuda. *IEEE Micro*, 28(4):13–27, 2008.

[25] J. R. Gilbert, C. Moler, and R. Schreiber. Sparse matrices in matlab: Design and implementation. *SIAM journal on matrix analysis and applications*, 13(1):333–356, 1992.

[26] M. Glavic. Interior point methods: A survey, short survey of applications to power systems, and research opportunities. 2004.

[27] B. Goodman. Sequential quadratic programming. "https://optimization.cbe.cornell.edu/index.php?title=Sequential_quadratic_programming#The_Active_Set_Method_and_its_Limitations", 2016. (Acedido em Abril 2023).

[28] S. Harding. What is pcie? a basic definition. "https://www.tomshardware.com/reviews/pcie-definition,5754.html", 2022. (Acedido em Junho 2023).

[29] E. L. Keller. The general quadratic optimization problem. *Mathematical Programming*, 5:311–337, 1973.

[30] J. Kessenich, D. Baldwin, and R. Rost. The opengl shading language, 2004. (Acedido em Novembro 2022).

[31] D. B. Kirk and W. H. Wen-Mei. *Programming massively parallel processors: a hands-on approach*. Morgan kaufmann, 2016.

[32] D. Langr and P. Tvrdik. Evaluation criteria for sparse matrix storage formats. *IEEE Transactions on parallel and distributed systems*, 27(2):428–440, 2015.

[33] M. Levinas. Gpu vs cpu: What are the key differences? "https://www.cherryservers.com/blog/gpu-vs-cpu-what-are-the-key-differences", nov 2020. (Acedido em Outubro 2022).

[34] Y. Li. Centering, trust region, reflective techniques for nonlinear minimization subject to bounds. Technical report, Cornell University, 1993.

[35] Y. Lievens, U. Ricardi, P. Poortmans, D. Verellen, C. Gasparotto, C. Verfaillie, and A. J. Cortese. Radiation oncology. optimal health for all, together. estro vision, 2030. *Radiotherapy and Oncology*, 136:86–97, 2019.

[36] Narenfox. Linear accelerator. "https://commons.wikimedia.org/wiki/File:Linear_Accelerator_-_Elekta_Compact_model.jpg". (Acedido em Janeiro 2023).

[37] J. Nocedal and S. J. Wright. Trust-region methods. *Numerical Optimization*, pages 66–100, 2006.

[38] I. P. Nunes. Instituto pedro nunes. "https://www.ipn.pt/ipn". (Acedido em Setembro 2022).

[39] F. Oh. What is cuda. "https://blogs.nvidia.com/blog/2012/09/10/what-is-cuda-2/", 2012. (Acedido em Novembro 2022).

[40] M. Oldham, A. Neal, and S. Webb. A comparison of conventional 'forward planning'with inverse planning for 3d conformal radiotherapy of the prostate. *Radiotherapy and Oncology*, 35(3):248–262, 1995.

[41] W. Parker and H. Patrocinio. Clinical treatment planning in external photon beam radiotherapy. *Radiation oncology physics: A handbook for teachers and students. Vienna: IAEA*, 219, 2005.

[42] Pordata. Óbitos por algumas causas de morte em % do total de óbitos. "https://www.pordata.pt/europa/obitos+por+algumas+causas+de+morte+em+percentagem+do+total+de+obitos-2484". (Acedido em Setembro 2022).

[43] F. A. Potra and S. J. Wright. Interior-point methods. *Journal of computational and applied mathematics*, 124(1-2):281–302, 2000.

[44] H. Rocha and J. M. Dias. On the optimization of radiation therapy planning. set 2019.

[45] H. Rocha, J. M. Dias, B. C. Ferreira, and M. Lopes. Selection of intensity modulated radiation therapy treatment beam directions using radial basis functions within a pattern search methods framework. *Journal of Global Optimization*, 57(4):1065–1089, 2013.

[46] H. E. Romeijn, R. K. Ahuja, J. F. Dempsey, A. Kumar, and J. G. Li. A novel linear programming approach to fluence map optimization for intensity modulated radiation therapy treatment planning. *Physics in Medicine & Biology*, 48(21):3521, 2003.

[47] J. Sanders and E. Kandrot. *CUDA by example: an introduction to general-purpose GPU programming*. Addison-Wesley Professional, 2010.

[48] SEER Training Modules. Radiation. `https://training.seer.cancer.gov/`. (Acedido em Setembro 2022).

[49] D. M. Shepard, M. C. Ferris, G. H. Olivera, and T. R. Mackie. Optimizing the delivery of radiation therapy to cancer patients. *Siam Review*, 41(4):721–744, 1999.

[50] D. Thambawita, R. Ragel, and D. Elkaduwe. To use or not to use: Graphics processing units (gpus) for pattern matching algorithms. In *7th International Conference on Information and Automation for Sustainability*, pages 1–4. IEEE, 2014.

[51] A. Wächter and L. T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical programming*, 106:25–57, 2006.

[52] H.-P. Wieser, E. Cisternas, N. Wahl, S. Ulrich, A. Stadler, H. Mescher, L.-R. Müller, T. Klinge, H. Gabrys, L. Burigo, et al. Development of the open-source dose calculation and optimization toolkit matrad. *Medical physics*, 44(6):2556–2568, 2017.

[53] E. Wong. *Active-set methods for quadratic programming*. University of California, San Diego, 2011.

[54] X. Yi, W.-l. Lu, J. Dang, W. Huang, H.-x. Cui, W.-c. Wu, Y. Li, and Q.-f. Jiang. A comprehensive and clinical-oriented evaluation criteria based on dvh information and gamma passing rates analysis for imrt plan 3d verification. *Journal of applied clinical medical physics*, 21(8):47–55, 2020.

[55] Y.-x. Yuan. Recent advances in trust region algorithms. *Mathematical Programming*, 151:249–281, 2015.

Appendices

Apêndice A

Requisitos

Neste apêndice serão apresentados os requisitos deste projeto. Começaremos por explorar os requisitos funcionais que descrevem as principais funcionalidades e tarefas que se espera que nossa solução realize, na secção A.1. De seguida, na secção A.2 fazemos a transição para os requisitos não funcionais, esclarecendo alguns atributos de qualidade que a implementação do projeto deve seguir. Finalmente, faremos uma análise do cumprimento destes requisitos na secção A.3.

A.1 Requisitos Funcionais

Esta secção enumera os requisitos funcionais definidos para a otimização do algoritmo de cálculo inserido no módulo de Geração Automática de Planos. Relembrando que o algoritmo de cálculo supracitado, consiste no cálculo e seleção do vetor de intensidades ótimo para o plano radioterapêutico desejado.

RF#1	Fazer *upload* de dados
Prioridade	Must Have
Descrição	O módulo de geração automática de planos deve ser capaz de gerir *uploads* da matriz de dose, d, e vetores de intensidades, w, da RAM da CPU para a VRAM da GPU.

Tabela A.1: Requisito funcional #1

RF#2	Fazer *download* de dados
Prioridade	Must Have
Descrição	O módulo de geração automática de planos deve ser capaz de gerir *downloads* do vetor de pesos otimizado, w_{opt}, da VRAM da GPU para a RAM da CPU.

Tabela A.2: Requisito funcional #2

RF#3	Gerir agendamento de cálculos
Prioridade	Should Have
Descrição	O módulo de geração automática de planos deve ser capaz de gerir o agendamento de cálculos através do uso de *threads* na GPU de forma a otimizar o cálculo entre a matriz de dose unitária e os vetores de intensidade.

Tabela A.3: Requisito funcional #3

A.2 Requisitos Não Funcionais

Nesta secção serão abordados os requisitos não funcionais identificados para o *scope* desta tese. Estes requisitos descrevem as caraterísticas que devem ser atingidas durante o processo de otimização:

1. **Portabilidade**: implica que um *hardware* ou *software* seja capaz de ser executado em diferentes ambientes [11].

   Cenário: O módulo de geração automática de planos deve conseguir se adaptar a diferentes capacidade de memória e diferentes GPUs.

2. **Integridade dos dados**: garante que ao longo do ciclo de vida dos dados, estes se encontram completos, consistentes, precisos e atualizados [10].

   Cenário: Os cálculos efetuados na GPU devem resultar em soluções 100% idênticas aos efetuados pela CPU, salvo erros de precisão em tipos de dados de vírgula flutuante de 64 bits.

A.3 Cumprimento Requisitos

Esta secção faz uma reflexão sobre o cumprimento dos requisitos funcionais e não funcionais supracitados.

A.3.1 Cumprimento Requisitos Funcionais

Na seguinte tabela resumem-se os requisitos funcionais e discute-se o seu cumprimento.

Requisito	Cumprido	Descrição
RF#1	Sim	Através do auxílio da ferramenta CUDA, o módulo de Geração Automática de Planos é capaz de carregar a matriz de dose, d, e os vetores de intensidades, w, da RAM da CPU para a VRAM da GPU.
RF#2	Sim	Através do auxílio da ferramenta CUDA, o módulo de Geração Automática de Planos é capaz de descarregar o vetor de intensidade, w_{opt}, da VRAM da GPU para a RAM da CPU.
RF#3	Não	No contexto desta dissertação não se recorreu ao uso implícito de *threads* para o agendamento de cálculos. Este requisto foi apoderado pela equipa do ORION através de processos em *python*.

Tabela A.4: Cumprimento Requisitos Funcionais

A.3.2 Cumprimento Requisitos Não Funcionais

Tal como visto anteriormente definiu-se **portabilidade** e **integridade dos dados** como dois objetivos que se deveriam alcançar neste projeto. Na tabela A.3.2 pode-se ver que ambos os requisitos não funcionais foram cumpridos e testados de acordo com a especificação desejada.

Requisito	Cumprido	Descrição

Portabilidade	Sim	Uma vez que o módulo de Geração Automática de Planos foi compilado usando MATLAB, este poderá ser executado em qualquer cenário que inclua um compilador de MATLAB. Além disso, o módulo de Geração Automática de Planos foi executado em duas máquinas com diferentes GPUS (Quadro P600 e RTX 3080) e diferentes capacidades de memória VRAM (aproximadamente 4GB e 10GB).
Integridade dos dados	Sim	Foi calculado o RMSE (Root-mean-square deviation) entre a multiplicação na CPU e GPU, obtendo-se um resultado de $5,12e{-}15$ para a máquina 1 e $5,51e{-}15$ para a máquina 2. Uma vez que estamos a trabalhar com valores reais, este valor de RMSE tão pequeno comprova que os cálculos efetuados na GPU resultaram em soluções 100% idênticas aos efetuados pela CPU, salvo erros de precisão resultante dos tipos de dados de vírgula flutuante.

Tabela A.5: Cumprimento Requisitos Não Funcionais

Apêndice B

Planeamento

Este apêndice serve para apresentar o trabalho desenvolvido durante a totalidade deste estágio curricular.

O primeiro semestre serviu de preparação para o trabalho que seria desenvolvido no futuro. A primeira etapa, foi a familiarização com o problema, que consistiu na apresentação do problema e estudo no mesmo através da documentação apropriada, bem como a definição de objetivos. Nesta fase de familiarização estiveram incluídas reuniões com os professores Humberto Rocha e Joana Dias, autores do "Módulo de Geração Automática de Planos". A segunda tarefa passou por perceber o contexto do problema e porquê que precisava de ser resolvido. De seguida, definiu-se quais as estratégias de implementação que seriam aplicadas no segundo semestre, e quais os requisitos do projeto.

A par destas tarefas esteve presente a redação do relatório intermédio de estágio. A distribuição temporal das tarefas pode ser vista no diagrama de Gantt da figura B.1.

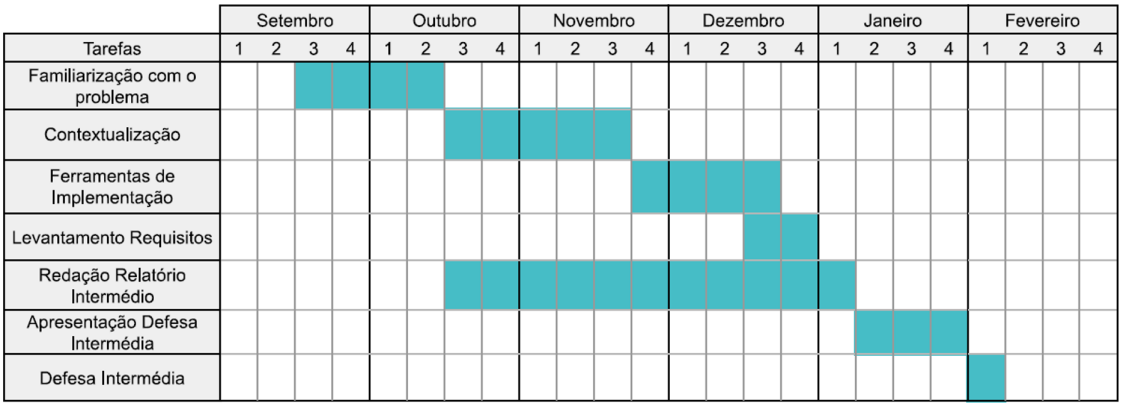


Figura B.1: Diagrama de Gantt 1º semestre

O trabalho do segundo semestre focou-se em otimizar a solução proposta pelos professores Humberto Rocha e Joana Dias, para o problema FMO no "Módulo de Geração de Planos de Tratamento".

No planeamento inicial foram definidas seis tarefas principais: (i) identificação

de constrangimentos associados, (ii) implementação da matriz esparsa, (iii) implementação na GPU, (iv) análise de soluções para restrições identificadas, (v) refinamento do algoritmo de otimização e (vi) redação do relatório de estágio final. A distribuição temporal das tarefas pode ser vista no diagrama de Gantt da figura B.2.

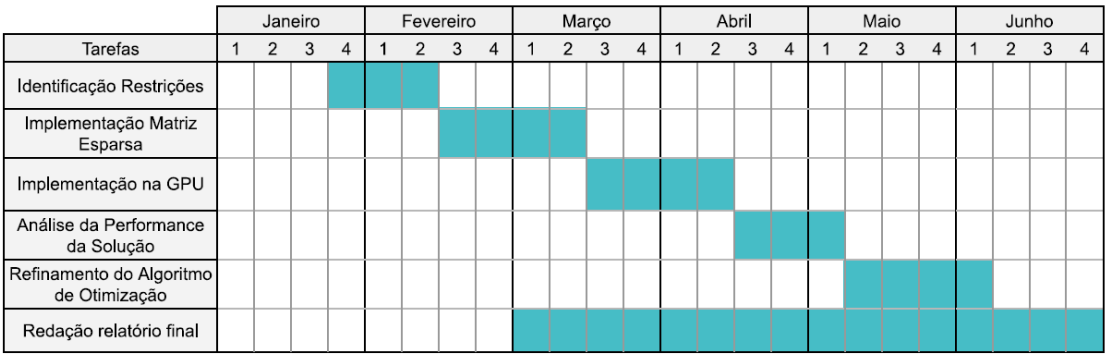


Figura B.2: Diagrama de Gantt estimado do 2º semestre

Estas seis tarefas foram cumpridas, mas foi necessário alterar o tempo anteriormente estimado e modificar e incluir novas tarefas que surgiram como essenciais.

Desta forma, introduziram-se as seguintes tarefas: (I) instalação do módulo de geração automática de planos, (II) análise do código do módulo de geração automática de planos, (III) análise das estruturas de dados utilizadas, (IV) implementação e refinação do Fmincon, (V) comparação da performance do IPOPT e Fmincon e (VI) variação do vetor de pesos inicial.

A distribuição temporal real pode ser consultada no diagrama de Gantt da figura B.3.

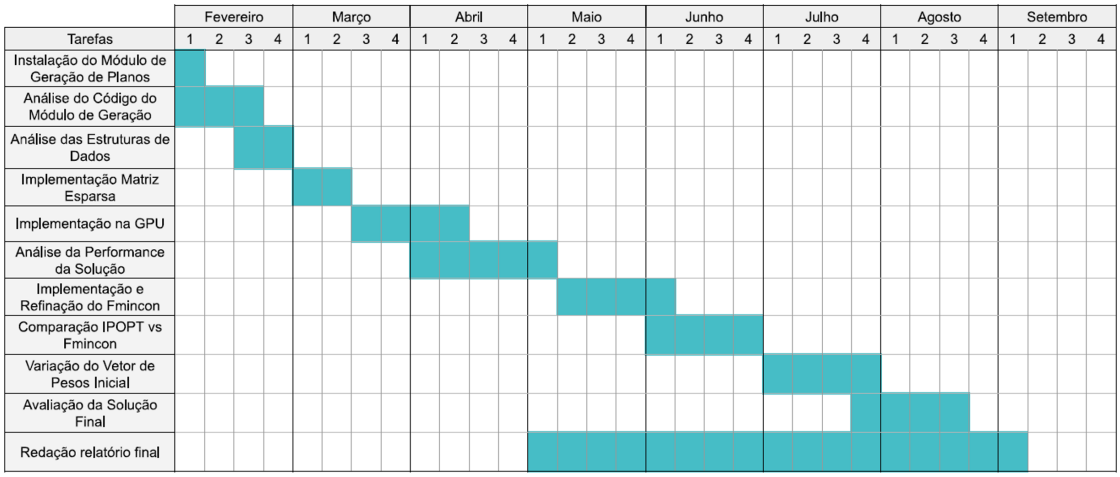


Figura B.3: Diagrama de Gantt real do 2º semestre

Para clarificação, é importante indicar que, a tarefa (i) definida no plano inicial foi integrada na tarefa (II) do plano real. Durante a análise do código foram procuradas mais restrições e possíveis soluções. Já a tarefa (v) foi convertida na tarefa (IV), onde foi implementado o Fmincon com possível substituto do IPOPT.

As novas tarefas foram definidas com a supervisão dos meus orientadores.