Nuno Gonçalo Barreto de Sá Miranda Fernandes

Licenciatura em Engenharia Biomédica

Optimização de uma ferramenta de segmentação de tecidos em imagens de ressonância magnética

Dissertação para obtenção do Grau de Mestre em
Engenharia Biomédica

Orientador: Doutor Ricardo Nuno Pereira Verga e Afonso Vigário, Professor Auxiliar Convidado, Faculdade de Ciências e Tecnologias da Universidade Nova de Lisboa.

Constituição do Júri:

Presidente:

- Doutora Célia Maria Reis Henriques, Professora Auxiliar da Universidade Nova de Lisboa – Faculdade de Ciências e Tecnologia.

Vogais:

- Doutor Ricardo Nuno Pereira Verga e Afonso Vigário, Professor Auxiliar Convidado da Universidade Nova de Lisboa – Faculdade de Ciências e Tecnologia, orientador.

- Doutora Rita Homem de Gouveia Constanzo Nunes, Professora Auxiliar do Instituto Superior Técnico da Universidade de Lisboa, arguente.

FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

Setembro, 2017

Optimização de uma ferramenta de segmentação de tecidos em imagens de ressonância magnética

Dedico esta dissertação a todos aqueles que me acompanharam ao longo dos anos, em especial aos meus dois avôs, que não puderam estar cá para me ver alcançar este objectivo…

Agradecimentos

Agradeço, em primeiro lugar, ao Professor Doutor Ricardo Vigário, pelo apoio que deu e paciência que teve ao longo de todos estes meses de trabalho nesta dissertação e ainda pela forma como deu as cadeiras de Métodos de Imagem Médica e Imagiologia, o que me cativou e levou a optar pela imagem médica como tema de tese e como a minha prioridade no que toca a Engenharia Biomédica.

Em segundo lugar, agradeço à Faculdade de Ciências e Tecnologias da Universidade Nova de Lisboa, por me acolher ao longo dos meus cinco anos de curso e ser a casa fora de casa.

À minha família, em especial aos meus pais, que por muito que, por vezes me custasse, sempre me forçaram a estudar e a trabalhar, quando me mostrava mais preguiçoso, e me apoiaram e não deixaram que me faltasse nada, em todos os meus anos de escola e faculdade.

Aos meus amigos, família, da faculdade e fora dela, ao Pedro Santos, Francisco Santos, Margarida Dias, David Melo, Paulo Teixeira, Miguel Baptista, Gonçalo Costa, Gonçalo Luís, Pedro Costa e Afonso Moreira, por serem os melhores amigos, verdadeiros irmãos, que alguém pode ter, pelo apoio em todos estes anos.

Um especial obrigado ao Afonso Moreira que, para além de seres um irmão para mim, me ajudaste quando o trabalho da tese esteve mais complicado, sempre disponível.

Um grande obrigado a todos os que fizeram parte desta jornada.

Resumo

A segmentação de tecidos cerebrais tem sido uma área de estudo importante no que toca a Imagens por Ressonância Magnética. Um diagnóstico de qualidade pode ser potenciado pela correcta identificação de lesões nos tecidos, não só em termos de saber dizer que é um tumor ou hemorragia, ou ainda um caso de degeneração da matéria branca, tecidos afectados em esclerose múltipla, mas também pela correcta avaliação volumétrica das zonas de interesse. Esta última é de particular interesse quando se pretende aferir da região foco para aquisição de imagem, aquando de um tratamento oncológico.

O método proposto nesta dissertação revela ser interessante nesta abordagem ao problema por, não só identificar as zonas lesadas dos tecidos, mas também por prever algum tipo de desenvolvimento de lesão em zonas aparentemente saudáveis, com o cálculo de probabilidades de pertença de determinados pixéis/vóxeis a várias classes de tecidos. Apresenta, ainda, outra vertente: a aplicação em imagens multi-espectrais, ou seja, obtidas com um conjunto de diversas sequências de aquisição de imagem, o que permite uma melhor validação e observação dos resultados. De facto, o algoritmo usado nesta dissertação, tanto *Self Organizing Maps* como o *Discriminative Clustering* só funciona se as imagens forem multi-espectrais. A optimização deste algoritmo mostrou resultados coerentes com os propostos na primeira implementação, em *Linux*, e foi criada uma interface de fácil utilização, bem como a adaptação do algoritmo para *Windows*, dada a anterior implementação em *Linux*.

Este método poderá, no futuro, ser aplicado na procura de melhorar a qualidade de imagens hospitalares, com optimizações a nível da implementação na pós-aquisição de imagem e avaliação imediata das estructuras presentes nas imagens adquiridas nos exames.

Palavras-chave: MRI, Segmentação, Processamento, Classificação, SOM, DC

Abstract

Brain tissue Segmentation has been an important front on which concerns Magnetic Resonance Imaging. A good diagnosis can be powered by the correct identification of tissue lesions, not only in a way of knowing if it is a tumor or a hemorrhage or even a case of white matter degeneration, but also by the correct volumetric evaluation of the region of interest. The last is of particular interest when we pretend to study the region of interest in case of an oncologic treatment.

The proposed method in this dissertation is interesting not only by identifying regions of tissues that contain lesions, but also by predicting some kind of development of the lesion in question in areas that apparently are healthy, with the acquisition of belonging rates of pixels/voxels to the tissues. Yet it reveals another front: the application to multi-spectral images, images obtained with a set of different sequences, which allows for a better validation and observation of the results. In fact, the algorithm used in this dissertation only functions if the images used are multi-spectral. The algorithm's optimization has showed consisted results when compared to the previous implementation on *Linux*. It was also created an interface allowing an easy manipulation of the algorithm by the user and the adaptation to *Windows*, because of the prevoious *Linux* implementation.

This method, in the future, can be applied to the of incriesed quality of hospital images, with optimization on the field of implementation in post-acquisition of the image and immediate evaluation of the structures present in the acquired images during the examination.

Keywords: MRI, Segmentation, Processing, Classification, SOM, DC

Conteúdo

Índice de Figuras

Índice de Equações

Lista de Abreviaturas

CSF	Cerebral Spinal Fluid (Líquido Céfalo-Raquidiano)
FCM	Fuzzy C-means
FLAIR	Fluid Attenuation Inverse Recovery
FN	False Negative (Falsos Negativos)
FP	False Positive (Falsos Positivos)
GM	Grey Matter (Matéria Cinzenta)
MS	Multiple Sclerosis (Esclerose Múltipla)
MTI	Magnetic Transfer Imaging
TN	True Negative (Verdadeiros Negativos)
TP	True Positive (Verdadeiros Positivos)
WM	White Matter (Matéria Branca)

1 Introdução

A Imagem por Ressonância Magnética (MRI) é um método cada vez mais usado para observação estructural e funcional de lesões cerebrais como tumores, esclerose múltipla ou mesmo hemorragias traumáticas, causadas por acidentes [1][2][3][4].

A MRI tem vantagem sobre outros métodos de imagem médica como a CT (*Computed Tomography*), por não recorrer a radiação ionizante para obtenção da imagem, mas sim a campos magnéticos estáticos e pulsados. Aliado a esta vertente, o facto de a resolução espacial ser bastante boa, leva a que a sua utilização para estudos estructurais do cérebro e observação de lesões seja da maior utilidade.

Actualmente, uma precisa identificação de áreas lesadas é da maior importância para rápida acção por parte do médico, e para redução da probabilidade de erros aquando do tratamento/intervenção cirúrgica. Quanto mais precisa e correcta a identificação das lesões, menor a probabilidade de afectar os tecidos saudáveis que possam estar na vizinhança das mesmas, atacando apenas a lesão em si.

É para melhorar estes diagnósticos que, no campo da MRI, existem em cada vez maior número de estudos e aplicações de métodos que visam melhorar a segmentação de tecidos e, em particular, a correcta distinção entre lesões e tecido saudável.

Com este objectivo têm sido desenvolvidos e apresentados diversos métodos de segmentação, desde os estritamente manuais aos completamente automáticos. Actualmente, visa-se encontrar processos que sejam bastante automáticos e não requeiram uma grande quantidade de tempo para efectuar a segmentação dos tecidos e que, ao mesmo tempo, apresentem a menor margem de erro possível, para obtenção de uma melhor identificação dos vários tecidos presentes na imagem.

O desenvolvimento de técnicas capazes de uma boa segmentação de tecidos em MRI tem sido orientado, não só para identificação de tecidos, mas também para a previsão de lesões e progressão de doenças degenerativas [5]. Esta evolução representa uma mais valia no desenvolvimento de novas tecnologias de análise.

A importância do presente trabalho, prende-se, portanto, com a optimização de um destes métodos, previamente desenvolvido em *Linux*, e pela uniformização de programas

usados para a automação de segmentação de tecidos em ressonância magnética bem como a sua automatização.

O resultado deste trabalho é um conjunto uniformizado de programas capazes de produzir uma classificação mais detalhada do que o "*hard-clustering*", normalmente utilizado na segmentação de tecidos. Atribui probabilidades de pertença aos diferentes tipos de tecido, o que é particularmente importante quando a distinção entre tecidos adjacentes não pode ser feita por simples binarização. De notar que estes casos são exactamente os que estão presentes em vóxeis associados a tecidos afectados de forma degenerativa.

Realça-se ainda a importância do método proposto no que toca à utilização de várias imagens, obtidas por manipulação dos parâmetros de MRI, ou seja, imagens multi-espectrais, obtidas através de um conjunto alargado de sequências de imagem. É ainda de salientar a importância do conceito de Volume Parcial. Cada pixel/vóxel pode incluir, ou não, mais do que um tipo de tecido ou classe. Um exemplo deste caso, pode ser um vóxel apresentar 50% de matéria branca e 50% de matéria cinzenta. Assim, este algoritmo terá em conta as contribuições dos vários tipos de tecido, presentes em cada pixel/vóxel.

Neste trabalho existe um foco na classificação dos pixéis/vóxeis de uma imagem por SOM, para obtenção de *labels* dos pixéis, e DC para classificação. A aquisição de imagem e o seu pré-processamento não são do âmbito desta dissertação, partindo-se do princípio que as imagens usadas se encontram nas condições certas para classificação, sendo que o pré-processamento, que inclui alinhamentos da imagem, correção de inhomogeneidades do campo magnético e identificação de regiões de interesse, estando ligado à classificação, faz parte do trabalho para a dissertação do colega Afonso Moreira [29].

2 Conceitos Fundamentais

2.1 O cérebro

O cérebro pode ser considerado como o centro de comandos do corpo. Composto por diversos lobos, é responsável pelo processamento da maior parte da informação que recolhemos através dos sentidos de que dispomos, visão, audição, paladar, tacto, olfacto.

Biologicamente, a informação é transmitida de neurónio para neurónio, até chegar aos locais alvo do corpo que receberam certo estímulo e darão resposta a esse estímulo.

Podemos encontrar dois tipos de neurónios diferentes no que toca à existência, ou não, de revestimento no axónio, responsável pela passagem do impulso nervoso do corpo celular do neurónio para o terminal das dendrites que, por sua, vez efectuam a passagem do sinal para o próximo neurónio, processo identificado com Sinapse. Este revestimento é visível na **Figura 2.1.**, que representa um neurónio. A mielina confere ao axónio uma cor branca e permite que o impulso se propague, de forma bastante eficiente, ao longo de um axónio "saltando" entre os intervalos na mielina, chamados nódulos de Ranvier.

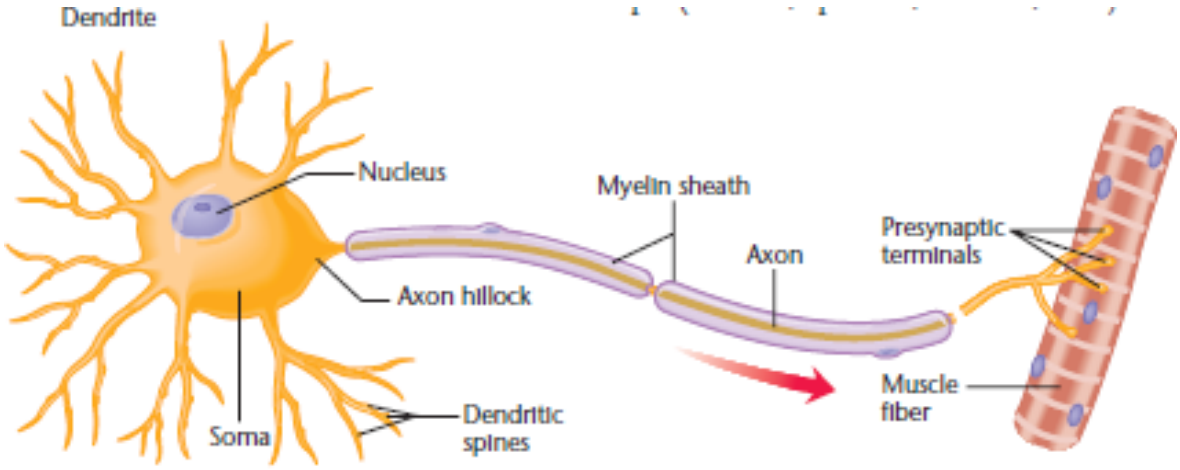


Figura 2.1 - Representação da estrutura de um neurónio. É possível observar o revestimento de mielina em torno do axónio, bem como os Nódulos de Ranvier que permitem acelerar a passagem do impulso nervos pelos axónios [31].

A cor branca da mielina e a sua presença à volta dos axónios dos neurónios dá o nome de Matéria Branca (White Matter, WM) à área do cérebro composta maioritariamente por axónios revestidos por mielina. Por outro lado, à área do cérebro

composta por neurónios cujos axónios não se encontram revestidos de mielina, chama-se Matéria Cinzenta (Grey Matter, GM), onde também se encontra a maior parte dos corpos celulares dos neurónios. Na **Figura 2.2.** é possível observar um corte, ao longo do plano coronal, de um cérebro, revelando as áreas correspondentes à WM e aquelas que são constituídas por GM.

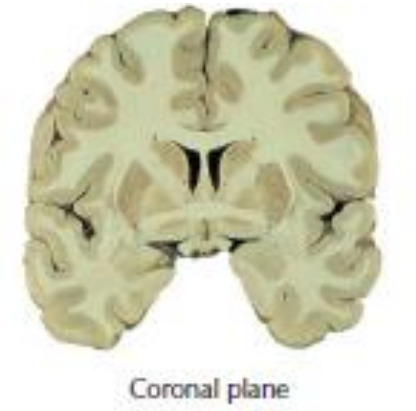


Figura 2.2 - Corte coronal do cérebro. É de notar que a localização da WM remete para zonas mais interiores do cérebro, enquanto que a GM se encontra maioritariamente na periferia do cérebro ao longo das convoluções do córtex [31].

No âmbito desta dissertação, é também importante referir o Líquido Céfalo-Raquidiano (mencionado ao longo do texto como CSF), maioritariamente localizado nos ventrículos do cérebro no centro deste, com as funções de nutrição dos tecidos cerebrais, suporte do cérebro e acondicionamento do mesmo. Na **Figura 2.3.** encontra-se representado um cérebro, com especial relevância nos ventrículos laterais e central, ilustrados a azul [31].

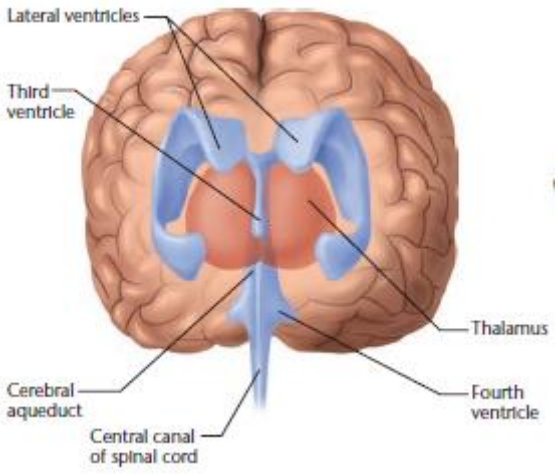


Figura 2.3 – Representação do cérebro com, especial relevância para os ventrículos laterais e central, ilustrados a azul, em torno do centro [31].

Estes três constituintes do cérebro, WM, GM e CSF farão parte dos tecidos identificados no processo de classificação no qual este trabalho se concentrará. A adicionar a estes tipos de tecidos encontram-se as lesões que serão o principal foco desta dissertação.

No entanto, antes de se proceder à contextualização de lesões usualmente identificadas em Imagem por Ressonância Magnética (MRI), torna-se importante referir os princípios básicos da MRI bem como o que está por detrás das imagens multi-espectrais usadas para estudo.

2.2 Imagem por Ressonância Magnética (MRI)

A técnica de MRI baseia-se na manipulação de campos magnéticos, aplicados ao objecto de estudo, de forma a obter uma imagem. Uma vez que, cada tecido no corpo é composto por elementos com variado número de protões (este número é relevante para ponderações em Densidade Protónica, explicada mais abaixo), a aplicação de um campo magnético, $B_{0,}$ leva a que os *spins*, momento angular intrínseco a partículas subatómicas (neste caso), precessem, no seu estado normal, em torno de um eixo zz, num referencial a três dimensões. A aplicação de um impulso de rádio frequência, no plano ortogonal a B_0, origina a alteração das direções de spins dos protões presentes nos núcleos dos átomos que constituem cada tecido, o que leva a que cada um dos tecidos presentes nas áreas estudadas, dependendo dos seus tempo de relaxação T1 e T2, apareça nas MRI com intensidades de graus de cinzento diferentes. As alterações das direções de spins podem ser obtidas de maneiras variadas por aplicação de impulsos de 90º ou 180º com ordens de aplicação diferentes ou por aplicação sucessiva de vários impulsos. O grau de variação destas direções pode ser diferente, o que leva a que seja dito que as imagens, obtidas com a aplicação de diferentes sequências, sejam ponderadas em diferentes valores de parâmetros, os quais são manipulados consoante o que se pretende observar. Estes parâmetros são **Tempo de Repetição (TR)** e **Tempo de Eco (TE)**.

- **Tempo de Repetição**: caracteriza-se pelo tempo entre a repetição de dois pulsos de rádio frequências. Num caso de *spin-echo*, em que, sucessivamente, se aplica um pulso de 90º, seguido por um de 180º, para a obtenção de um eco, o tempo de repetição irá corresponder, ao tempo decorrido entre a aplicação do primeiro pulso de 90º e a aplicação do segundo pulso de 90º, para criação e um novo eco. De uma forma simplificada, o tempo de repetição, TE, corresponde ao tempo entre dois processos de imagem.

- **Tempo de Eco**: o tempo de eco baseia-se no intervalo de tempo decorrido entre a aplicação de um pulso de rádio frequências de 90º e uma amostra do sinal de ressonância magnética, ao qual corresponde o máximo do eco. Recorrendo ao mesmo exemplo dado para o TE, o caso do *spin-echo*, o uso de um pulso de 180º dá-se a meio do TE, ou seja $\frac{TE}{2}$.

Na **Figura 2.4.** encontram-se ilustrados os tempos acima explicados, para um caso de *spin-echo*, onde se pode ver o TR como o intervalo entre a aplicação de dois pulso de rádio frequências distintos e o TE como o intervalo de tempo entre o pulso aplicado de 90º e o pico do eco medido.

Figura 2.4 – Representação do TR e TE para o caso de aplicação de um spin-echo. De notar que o TR é medido entre dois pulsos de 90º e o TE entre um pulso de 90º e o máximo de eco [32].

A manipulação destes tempos leva a imagens ponderadas, por aplicação de diferentes sequências, de forma diferente, consoante o que se pretende estudar e observar com maior clareza. Por exemplo, no caso das ponderações T1, T2 e DP (Densidade protónica), obtemos imagens ponderadas em T1 (tempo de relaxação longitudinal) utilizando-se TR e TE curtos; para imagens ponderadas em T2 (tempo de relaxação transversal), faz-se uso de TR e TE longos, enquanto imagens ponderadas em PD (do inglês *Proton Density*) aplicam-se TR longo, que leva a que a diferença nos tecidos dependa só do número de protões dos átomos, e TE curto, que leva a que não haja muito tempo entre aplicação de ecos, ou seja, sejam aplicados vários impulsos seguido a um ritmo relativamente rápido.

Existem casos em que, com o objectivo de enfatizar, por exemplo, matéria branca ou cinzenta e, de certa forma, suprimir um pouco o CSF, se recorre à recuperação inversa. Esta, ao invés do *spin-echo*, começa pela aplicação de um pulso de 180º, seguido de um

de 90º e de outro de 180º. Este acrescento de um pulso de 180º na aplicação dos pulsos leva a que o vector de **Magnetizaçã**o seja movido para o eixo negativo dos zz. Consequentemente, o tempo de recuperação da magnetização para a sua posição inicial e estável, precessão dos *spins* em torno do eixo zz positivo, aumenta, ocorrendo também aumento do espaçamento entre os máximos registados para cada tecido. Desta forma, recorrendo à recuperação inversa, consegue-se obter uma maior diferenciação entre tecidos e supressão selectiva de outros [32].

Considera-se assim um novo tempo, o **Tempo de Inversão (TI)**:

- **Tempo de Inversão**: caracteriza-se pelo intervalo de tempo entre a aplicação do primeiro pulso de 180º e a aplicação do pulso de 90º para obtenção de magnetização transversal. Dependendo da manipulação do TI, a sua alteração significará a supressão de tecidos diferentes.

A **Figura 2.5.** a obtenção do sinal por recuperação inversa. Pode observar-se o pulso inicial de 180º antes da aplicação do de 90º, com indicação do TI.

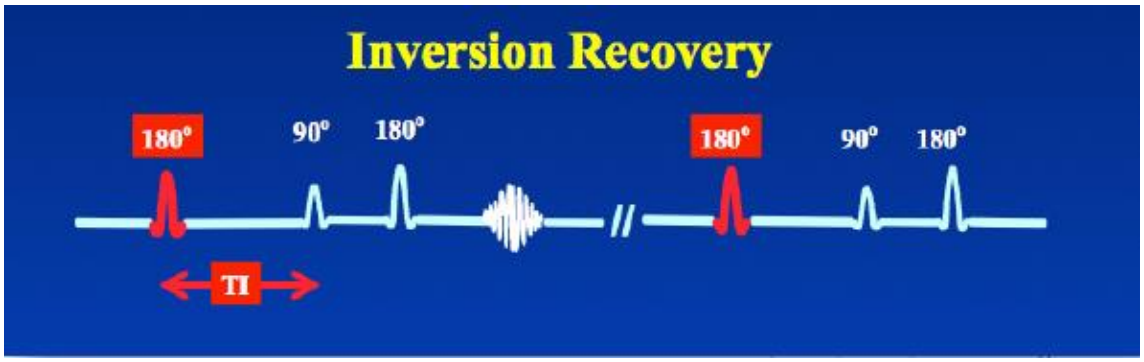


Figura 2.5 – Representação do TI para o caso de aplicação de uma recuperação inversa. Note-se que o TI caracteriza o intervalo de tempo entre o primeiro pulso de 180º e o de 90º [32].

A **Figura 2.6.** representa um corte transversal do cérebro, obtido por MRI com indicação de WM, GM, CSF e crânio.

Em contraste, a **Figura 2.7.** representa um cérebro que aparenta, claramente, apresentar degeneração da matéria branca no hemisfério direito [30], manifestada na alteração dos valores de cinzento na região anterior do hemisfério esquerdo.

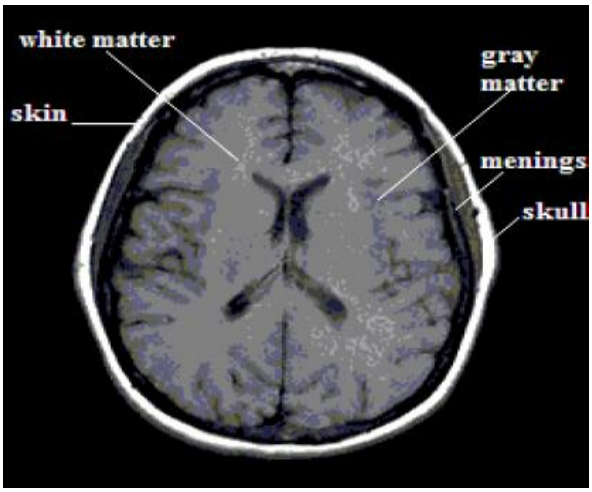


Figura 2.6 - Imagem de ressonância magnética de um cérebro normal [2]. Note-se a GM na zona exterior do cérebro, a WM mais no interior, a rodear os ventrículos, com CSF.

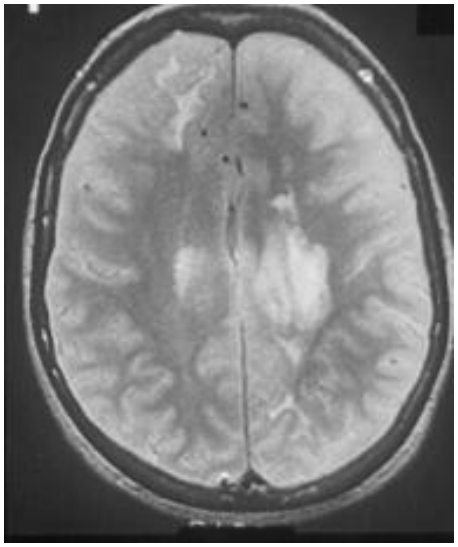


Figura 2.7 - - Representação de um cérebro patológico, apresentando desmielinização, ou seja, degeneração da matéria branca, mais notada no hemisfério esquerdo (mancha mais intensa quase ao centro) [30].

2.3 Lesões cerebrais

As lesões cerebrais requerem particular cuidado no seu tratamento, quer sejam tumores, hemorragias ou resultantes de esclerose múltipla. São, normalmente, distinguíveis, em MRI, dos restantes tecidos, em particular quando são escolhidas determinadas sequências de imagem. No entanto, e como se pode observar nas **Figuras 2.6-8**, raramente a mesma sequência que enfatiza numa dada lesão cerebral, também apresenta uma discriminação anatómica adequada para os outros tecidos cerebrais.

Assim, é normal, num estudo imagiológico, recolher várias imagens complementares, baseadas num conjunto distinto de sequências. Ao conjunto de imagens assim produzido chama-se multi-espectral. Veremos, agora, alguns exemplos de lesão, ao nível do cérebro, ilustradas com imagens obtidas com sequências diferentes, consoante a lesão apresentada.

2.3.1 Tumores

Os tumores, como apresentado na **Figura 2.8.**, são caracterizados por um crescimento descontrolado e invasivo, sendo que podem ser benignos ou malignos e que, numa imagem de ressonância magnética, aparecem como massas distintas do resto dos tecidos [4].

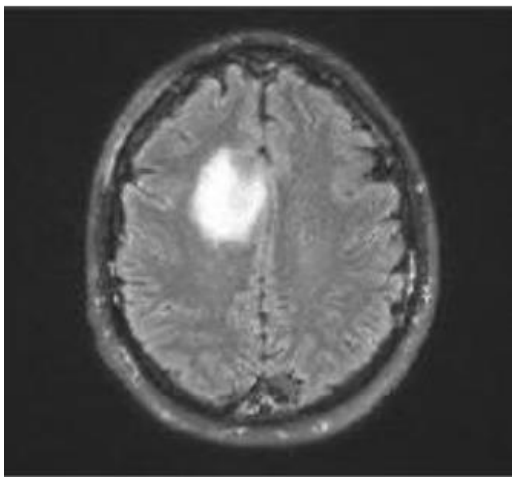


Figura 2.8 - Imagem de ressonância magnética de um tumor no cérebro [4] obtida por FLAIR, ou seja, por recuperação inversa.

2.3.2 Hemorragias

As hemorragias, com a representada na **Figura 2.9.** são, na maioria das vezes, causadas por acontecimentos traumáticos, como impactos fortes ao nível da cabeça, mas pode também ser causada por tromboses podendo afectar grandes partes do cérebro. Ao contrário dos tumores, muitas vezes há sobreposição do tecido hemorrágico com o tecido normal, levando a que o mesmo vóxel tenha contribuições de vários tipos de tecidos. Este fenónmeno torna a identificação da zona hemorrágica bastante difícil [1].

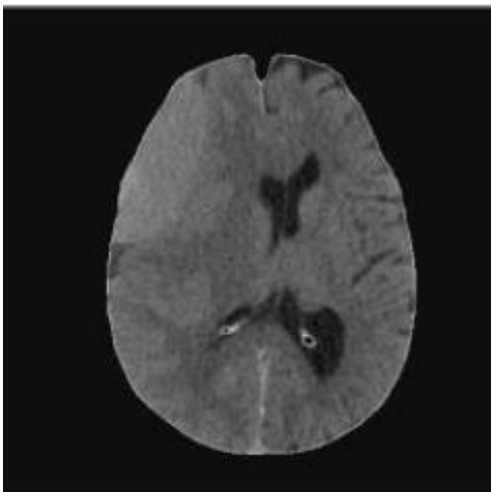


Figura 2.9 - Imagem de Ressonância magnética de um cérebro com hemorragia [1] obtida por recuperação inversa..

2.3.3 Esclerose Múltipla (MS)

MS é uma doença progressiva do cérebro que leva a um declínio cognitivo e a uma deficiência física. Caracteriza-se por uma acção inflamatória que danifica as bainhas de mielina, e manifestada por zonas brancas em imagens ponderadas em T2, como a representada na **Figura 2.10.** Esta desmielinização afecta a comunicação dos impulsos nervosos o que afecta o paciente tanto na sua capacidade motora, como também na cognitiva [6].

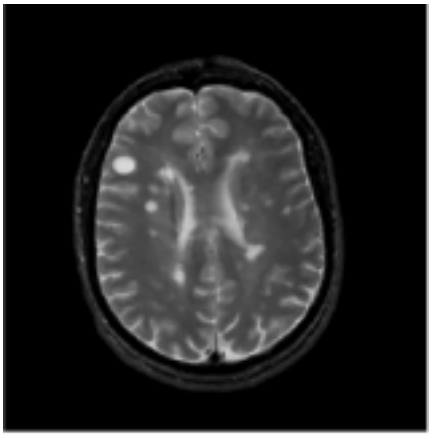


Figura 2.10 - Imagem de ressonância magnética, ponderada em T2 ,de um cérebro afectado por esclerose múltipla [6].

2.4 Análise das Imagens

2.4.1 Pré-processamento e considerações prévias

Tal como mencionado anteriormente, MRI é um método de imagem médica que permite estudos estructurais e funcionais de um paciente, através de um complexo jogo de interação entre campos magnéticos aplicados e os spins de átomos de hidrogéneo presentes no corpo.

Neste trabalho a avaliação de lesões cerebrais será feita pelo estudo de diversas imagens obtidas em MRI, e ponderadas em *T1*, *T2*, *DP* (Densidade Protónica), *Difusão*, *FLAIR* (*Fluid Attenuation Inverse Recovery*), ou seja, recolhas multi-espectrais, obtidas com várias sequências de imagem diferentes. Estas imagens irão passar por um pré-processamento e uma subsequente análise.

O pré-processamento, que não será a incidência principal desta dissertação, passa pela preparação das imagens para melhor análise. Numa tese adicional [29], apresentada ao mesmo tempo que o presente trabalho, estes assuntos serão tratados em mais detalhe. São exemplos de pré-processamento, o alinhamento de imagens, a remoção de artefactos, a remoção de ruído ou a remoção de tecidos não cerebrais, como é o caso do crânio. Também uma análise dos histogramas das imagens pode ser feita, para possível equalização das mesmas, por forma a conseguir um maior equilíbrio dos graus de cinzento nas imagens.

O processamento das imagens, sobre o qual esta tese incide, consiste na segmentação de tecidos nas imagens do cérebro recolhidas, optimizando-se um método de segmentação que separa os vários tipos de tecidos e atribuindo probabilidades de pertença de certos pixéis/vóxeis a determinado tipo de tecido.

2.4.2 Segmentação

O conceito de base por trás da segmentação é a determinação de características comuns aos vóxeis pertencentes a cada tecido. Das várias formas de o fazer, podemos distinguir os métodos de classificação, em que é necessário ter informação sobre as classes a que pertencem os vários vóxeis, e o *clustering*, onde se agrupa os dados com base em características comuns aos vários elementos do grupo.

Clustering, traduz-se na agregação de pixéis ou vóxeis e pela atribuição de protótipos associados a cada grupo (centroídes), representando características comuns aos vários elementos do grupo. Esta agregação pode ser obtida por diversos métodos, com base em diferentes assunções algorítmicas. *Clustering* pode ser visto como o primeiro passo no processo de segmentação, para a correcta identificação de regiões lesadas e não lesadas, uma vez que não utiliza explicitamente nenhuma informação de classe. Existem diversos métodos que aplicam esta noção:

- **K-means**: baseia-se na criação de *clusters* (agrupamentos) pela atribuição de pixéis/vóxeis a determinados grupos baseado em cálculos de distâncias euclidianas entre pixéis com características semelhantes, tais como a intensidade dos graus de cinzento [7][8]. Pode, então, definir-se um centroíde para representar cada agrupamento. A atribuição dos pixéis/vóxeis de um determinado cluster é feita comparando as distâncias dos mesmos aos centroídes propostos, num espaço tridimensional (no caso dos dados usados neste trabalho, em que foram usadas imagens multi-espectrais com três sequências distintas). O pixel/vóxel é atribuído ao cluster cujo centroíde esteja à menor distância calculada. Este método é apelidado de "*hard clustering*"[5], por determinar que um pixel pertence inteiramente a um tipo de cluster ou tecido. É um algoritmo que apresenta a vantagem de ser de computação rápida, mas que tem as desvantagens de, primeiramente, depender da distribuição inicial dos centros dos clusters e de, em segundo lugar, não permitir o discernimento de tecidos entre os quais o contraste não seja muito acentuado, ou seja, dois tecidos cujas intensidades dos respectivos pixéis sejam próximas irão ser considerados pelo algoritmo como pertencendo ao mesmo tecido. A **Equação 2.1.** representa a função objectivo (função que deve ser optimizada para se obter o objectivo desejado, muiyo utilizada em problemas de optimização em Programação Linear) que infere acerca da pertença de um pixel a um cluster, calculando a mínima variância desse pixel-n (x_n) ao centroíde (U_j) do cluster. Visa separar N pixéis em k clusters (valor que deve ser já conhecido), sendo j a ordem do centroíde, $1<j<k$. O pixel irá pertencer ao cluster cujo centroíde corresponda à menor variância. C_j, existente na inicialização do segundo somatório, ao qual devem pertencer os pixéis, representa a classe desse pixel.

$$J = \sum_{j=1}^{k} \sum_{n \in C_j} \left(x_n - U_j \right)^2$$

Equação 2.1 – Função objectivo para minimização da variância entre pixeis e centroídes dos clusters por K–means [12].

- **Fuzzy C-means (FCM)**: este algoritmo é semelhante ao K-means. A sua implementação também se baseia no cálculo de distâncias dos pixéis aos centros dos clusters. A diferença entre este algoritmo e o anteriormente apresentado está no tipo de clustering. Ao contrário do algoritmo K-means, caracteriza-se por ser de "*soft clustering*" [12][16][19]. Esta classificação deve-se ao facto de os pixéis poderem pertencer a mais do que um cluster, com uma determinada probabilidade de pertença [5][9]. Este método tem a desvantagem de cada pixel/vóxel não ser influenciado pelos vizinhos. O facto de este método não cobrir o envolvimento com os pixéis vizinhos, não os ter em conta, pode levar ao aparecimento de artefactos na imagem, por má identificação da contribuição do tecido na região tida em conta, o que, posteriormente, afecta a segmentação por atribuição errada de pixéis a determinadas classes. A **Equação 2.2.** é muito semelhante à **Equação 2.1.** tendo, no entanto, em conta a contribuição do mesmo pixel em mais do um tecido, u_i, ou seja, o coeficiente de pertença. X_k representa os pixéis/vóxeis em questão e u_i, os centros dos clusters considerados para cálculo da distância entre si e os pixéis/vóxeis.

$$J = \sum_{i=1}^{c} \sum_{k=1}^{N} u_{ik}^{P} \|x_k - v_i\|^2$$

Equação 2.2 - Função a optimizar pelo método FCM. Representando a distância entre os píxeis e os centros dos clusters, multiplicada por uma variável representativa do peso da pertença do pixel a um tipo de tecido [27].

- **Método de Otsu**: este processo, pertencente à classe de análise discriminante, minimizando a variância entre as características de pixéis constituintes de um dado agrupamento, enquanto maximiza a variância entre agrupamentos. Assim, o Método de Otsu encontra o limiar que melhor discrimina entre as várias classes de pixéis/vóxeis [12]. Baseia-se no cálculo das probabilidades de encontrar cada um dos clusters, tendo em conta os pixéis que os formam, dentro do total de pixéis da imagem em que Pi é a probabilidade de encontrar pixéis em cada grau de cinzento abaixo, ou acima, do threshold, **Equação 2.3.** (em que P_i é a probabilidade de encontrar cada conjunto de pixéis com certo valor de grau de cinzento, $P_r=w_2$, que é a probabilidade de encontrar pieis com certos graus de cinzento dentro da classe C_2), no cálculo das médias de graus de cinzento de cada grupo, **Equação 2.4.** (em que L é o total de graus de cinzento disponíveis, i são os índices desses graus de cinzentos e t o número de graus de cinzentos dentro de cada classe), média total das intensidades inter-cluster dos pixéis, **Equação 2.5.**, e no consequente cálculo das variâncias intra-cluster e da variância total, **Equação 2.6.**, sendo

que a variância intra-cluster terá de ser o mais baixa possível, de forma a traduzir a uniformidade de características dos pixéis que constituem cada um dos clusters. No entanto, como dito, a variância entre os diferentes clusters terá de ser bastante maior, ou mesmo o mais alta possível, de forma a garantir uma separação clara entre as características de cada cluster [13].

$$w_2 = P_r(C_2) = \sum_{i=t+1}^{L-1} P_i$$

Equação 2.3 - Probabilidades de cada cluster, com Pi a ser a probabilidade de pertença de cada grau de cinzento dentro do cluster [12].

$$u_1 = \sum_{i=0}^{t} iP_i / w_1$$

$$u_2 = \sum_{i=t+1}^{L-1} iP_i / w_2$$

Equação 2.4 - Cálculo das médias de cada uma das classes ou clusters [12].

$$u_T = w_1 u_1 + w_2 u_2$$

Equação 2.5 - Cálculo da média total [12].

$$\sigma_1^2 = \sum_{i=0}^{t} (i - u_1)^2 P_i / w_1$$

$$\sigma_2^2 = \sum_{i=t+1}^{L-1} (i - u_2)^2 P_i / w_2$$

Equação 2.6 - Cálculo das variâncias intra-cluster [12].

- **Thresholding**: é um método de binarização (0 e 1) que se baseia na escolha de um limite (*Threshold*) acima do qual um pixel/vóxel toma o valor 1, enquanto pixéis/vóxeis com valores abaixo, tomam o valor 0, [10][11][5]. Assim, pixéis abaixo do threshold são agrupados num cluster e os restantes, acima do threshold, são agrupados noutro [2], tal como mostra a **Equação 2.7.** em que ρ é o valor do threshold. O método tem a

desvantagem de poder ignorar as áreas da lesão, se o limite escolhido não for o mais apropriado. A escolha destes valores é, desta forma, bastante importante. Assim, a segmentação resultante poderá ser pouco selectiva. No caso de ser impossível definir um único threshold para toda a imagem, por exemplo, baseado na avaliação do histograma da imagem, existe o chamado "*local thresholding*" que se caracteriza pela definição de um limite numa determinada região, em vez de o fazer na imagem total "*global thresholding*" [11]. No entanto o aumento do número de limiares a usar acarreta um aumento concomitante da subjectividade do método.

$$g(x,y) = \begin{cases} 1 & \text{if } f(x,y) \geq \rho \\ 0 & \text{otherwise} \end{cases}$$

Equação 2.7 - Resultado da binarização, baseada num limite escolhido [11].

- **Watershed:** É uma técnica de gradiente, segue uma determinada direcção e sentido (o sentido do gradiente de intensidades), baseada no percurso da água ao seguir "caminhos" nas rochas, ou seja, ao deslizar por efeito da gravidade por entre as rochas e as suas fissuras. Parte do princípio que, durante o seu percurso, os contornos na imagem vão ser criados quando duas, ou mais, massas de água se juntam. A imagem passa, então, a ser segmentada por esses contornos [16] [21]. Este algoritmo consegue delimitar várias regiões em simultâneo, e fazer sobressair contornos, não os desfocando [11]. Costuma ser usado sequencialmente com K-means, de forma a contribuir com a informação espacial que não é, normalmente, tida em conta por aquele método de *clustering* [7] [8]. Uma aplicação deste algoritmo é na passagem de RGB (Red-Green-Blue, codificação típica para as cores de uma imagem) para uma escala de cinzentos. Neste ponto, cada grau de cinzento será visto como uma altura, uma distância vertical, onde começa a segmentação, a delimitação de contornos, até atingir um mínimo local. Pense-se nesta delimitação como a água a seguir determinado percurso ao escorregar por uma montanha.

A desvantagem desta técnica é o facto de ser muito sensível à quantidade de ruído da imagem,ou seja, o SNR (Signal Noise Ratio) tem de ser o maior possível, [4]. Para além disso, o Efeito de Volume Parcial (presença de mais do que um tipo de tecido no num vóxel) não pode ser muito importante, ou seja, a contribuição de vários tecidos dentro de um mesmo pixel/vóxel não deve levar a grandes variações de intensidade, que

podem levar a buracos (zonas a preto) ou a sobre-segmentação [11] [22], o que levaria um dado tecido a ser considerado como dois ou mais.

- **Region Growing:** ao contrário das técnicas anteriores, excluindo Watershed, que não têm em conta onde se encontram os pixéis/vóxeis de cada cluster, esta técnica isola regiões contínuas de pixéis/vóxeis que apresentam características semelhantes. Normalmente é escolhido um pixel na região de interesse, por exemplo, um tumor. De seguida os pixéis na vizinhança directa do pixel escolhido são testados, no que toca à sua semelhança com o primeiro pixel em termos de, por exemplo, a sua intensidade, medida em graus de cinzento. No caso das ditas semelhanças se confirmarem então é concluído que os pixéis vizinhos pertencem, de facto, à mesma região do pixel inicial, caso contrário define-se que o pixel testado se encontra na fronteira da região de interesse. Um dos parâmetros estudados para encontrar determinada região pode ser um intervalo específico de intensidades de graus de cinzento. Quando duas regiões são vizinhas, e respeitam a condição de semelhança, definida, por exemplo, através de um limite de intensidade, elas juntam-se numa só. No entanto, em concordância com outros métodos baseados em regiões, como Watershed, apresenta a desvantagem de não conseguir contornar o efeito de volume parcial, o que pode afectar a intensidade dos pixéis estudados e, por sua vez, a atribuição dos mesmos às regiões correctas [11].

-**Componentes ligados:** este método baseia-se na delimitação do perímetro de uma determinada região, com o objectivo de fazer sobressair os seus contornos. Dependendo do tipo de forma da região, é definido um tipo de conectividade, **Figura 2.11**, sendo que, por exemplo, para regiões bastantes simples, como regiões que possuam arestas e cantos, usa-se conectividades simples com 4 ou 8 pontos. A delimitação do perímetro traduz-se na delimitação dos contornos dos clusters. Pode ser usado para tentar corrigir o *blur* de contornos mal diferenciados, recorrendo às adjacências para seguir uma direção de ligação dos pixéis e, consequentemente, dar origem a uma delimitação de contornos [33].

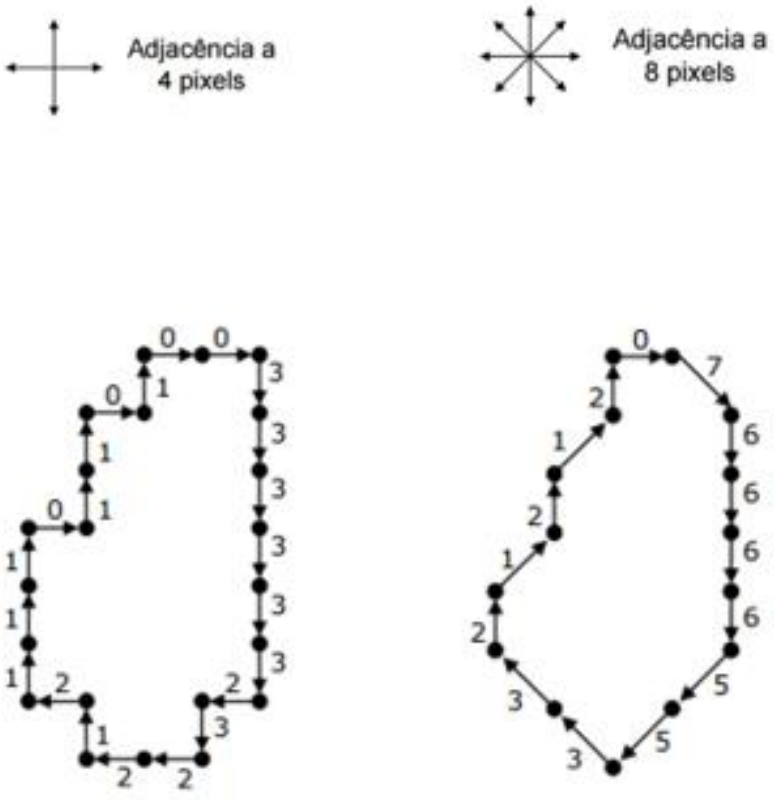


Figura 2.11 - Adjacências a 4 e a 8, na técnica de componentes ligados [33].

- **Particle Swarm Optimisation (PSO)**: neste método, obtido através da observação de pássaros e peixes em movimentos bidimensionais [14] [15], aplicado a características dos pixéis testa, a adequação do pixel/vóxel e dos seus vizinhos aos agrupamentos de forma a atribuir novas posições. Como nos métodos anteriores o objectivo é agrupar pixéis com o mesmo "comportamento", ou seja, que apresentem semelhanças entre características como a intensidade dos graus de cinzento [7]. Tem a vantagem de poder ser utilizado em espaços de grande dimensão ou, no caso de imagens de ressonância magnética, em pixéis de maiores dimensões.

- **Markov Random Fields (MRF)**: integra a informação espacial na criação de clusters, o que não acontece na maioria dos métodos apresentados, permitindo preservar a informação local [16]. A possibilidade da utilização desta informação espacial permite evitar a sobreposição de regiões bem como minimizar o efeito do ruído, dado que , sendo um método estatístico, descarta valores que sejam muito à parte daquilo que é, de facto, a imagem. Segundo Gordillo N, et al, State of the art survey on MRI brain tumor

segmentation, Magn Reson Imaging (2013), "In the particular case of brain tumor segmentation, if a pixel is strongly labeled as tumor (non-tumor), it suggests that its neighbors will have a tumor (non-tumor) label.", ou seja, o facto de, normalmente, não se recorrer a informação espacial, leva a uma possivelmente má classificação de pixéis vizinhos, por possivelmente sobre-estimar a extensão do que se procura e se acreditar que o está à volta é do mesmo tipo. No entanto, este processo apresenta a desvantagem de ser computacionalmente pesado. Apesar desta desvantagem, o MRF tem sido usado para modelação de texturas e para ultrapassar inhomogeneidades de intensidade de campo magnético nas imagens, por a sobre-estimção do que está à volta de um pixel acabar por funcionar como um filtro passa-baixo [11].

- **Self Organizing Maps (SOM)**: com este método, um aglomerador apresenta uma série de opções de características, que passam os valores de input (valores dos pixéis), à entrada da rede neural, e matematicamente, é dada uma classificação nos nós finais [11] [17]. SOM é um exemplo de Redes Neuronais Artificiais (Artificial Neural Networks, ANN) [11] [5]. SOM caracteriza-se por ser uma técnica que possibilita o mapeamento de inputs pertencentes a espaços de alta dimensão em elementos de um espaço de menor dimensão, conseguindo manter a informação de vizinhança em relação ao input inicial. Garante-se que unidades vizinhas no mapa correspondem a pixéis/vóxeis com comportamento espectral semelhante no conjunto de imagens iniciais. As dimensões e forma do mapa têm de ser escolhidas logo no início do processo. Durante o processo de amostragem de um dado de entrada, a unidade activa dentro do mapa, o "*winning neuron"* é aquele que apresenta características mais aproximadas do valor do input, usando o mínimo da distância euclidiana. Este processo encontra-se sumarizado na **Equação 2.8**, onde *c* é o resultado obtido para a distância, *x(n)* representa os inputs e m_k representa o neurónio específico para calculo. O objectivo é descobrir a *"best matching unit"*.

$$c = c(\boldsymbol{x}(n), t) = \arg\min_k \|\boldsymbol{x}(n) - \boldsymbol{m}_k(t)\| \ ,$$

Equação 2.8 - Mínimo da distância euclidiana (nó mais próximo) entre o vector de input e o neurónio k , de forma a encontrar o "*winning neuron*" [5].

Este método é diferente da maioria dos métodos ANN, por se basear numa aprendizagem competitiva, ao invés da habitual aprendizagem por erro. A aprendizagem

competitiva é um tipo de aprendizagem não-supervisionada, no qual os nós "competem" entre eles com o objectivo final de se especializarem numa determinada característica, o que se revela útil na atribuição de pixéis a clusters. Depois de aprender o melhor mapa, é possível adjudicar uma classe a cada entrada, observando a sua projeção no mapa resultante.

Para o uso de SOM, neste trabalho recorrer-se-á a várias corridas do algoritmo, o que aumenta o número de propostas e partição dos pixéis em variados clusters. Podemos ainda aumentar a confiança na criação de agrupamentos consistentes, dando lugar ao aparecimento de um conjunto de *labels* atribuídas pelo SOM. Cada mapa é inicializado aleatoriamente e os pixéis/vóxeis usados para super-vóxeis, i.e. vóxeis com *label*, são aqueles que, em múltiplas corridas, aparecem maioritariamente agrupados [5]. A maior parte das soluções diferem de corrida para corrida. No entanto, há alguns vóxeis que, consistentemente, se agrupam independentemente da inicialização.

A distribuição espacial dos mapas finais é, em parte, função da inicialização dos mesmos. Assim, se se correr várias vezes o mesmo algoritmo, utilizando os mesmos dados, para inicializações aleatórias, obtemos resultados ligeiramente diferentes. Podemos explorar esta diferença, efectuando múltiplas corridas, o que leva mais tarde, à seleção de super-vóxeis.

- **Técnicas de Projeção**: Dada a existência de várias imagens por sujeito, por existirem várias sequências de imagem, o que quer dizer que se considera um espaço de altas dimensões. Em cada imagem derivará, então, de um sinal que será a soma pesada de vários tecidos, com pesos diferentes para cada imagem. ICA e PCA separam os tecidos/formam projeções que maximizam independência não-gaussiana e descorrelação com máxima variância, respectivamente.

i) **Análise de Componentes Independentes (ICA):** Este método permite a separação do sinal em várias componentes estatisticamente independentes para uma melhor interpretação dos resultados. Procura projeções não-gaussianas que são aproximadamente independentes umas das outras. O sinal, neste caso de MRI, é a soma pesada de cada tecido.

É um método bastante útil quando se pretende isolar várias fontes, que contribuem para um sinal obtido, ou, por exemplo, na fMRI pode recorrer-se ao método ICA para

isolar áreas activas do cérebro durante determinadas tarefas ou aplicação de estímulos [23] [24].

ii) **Análise de Componentes Principais (PCA):** recorre a um critério de máxima variância sendo uma técnica muito comum. Utiliza transformações ortogonais, de forma a projectar os dados obtidos num conjunto de dados descorrelacionados. O número de componentes obtidas terá de ser menor ou igual ao número de variáveis. Produz projeções ortogonais de um espaço de alta dimensão, em que cada projeção maximiza a variância dos dados projectados. Dentro do tema da segmentação de tecidos, o PCA, pode ser usado como "substituto" dos métodos de clustering usais, como k-means ou FCM, quando os dados usados são multi-espectrais, como aqueles utilizados neste trabalho. Apresenta sub-métodos que têm por base análises probabilísticas, que permitem inferir sobre a pertença de um pixel a determinado tecido, o que vai contra o normal *hard clustering*, e mais na direcção ao *soft clustering* [25] [26].

- **Métodos baseados em Densidades de Probabilidades**: nestes processos os clusters são gerados por uma mistura de densidades de probabilidade Gaussianas das intensidades dos pixéis do conjunto. Este método é baseado no algoritmo Expectation Maximization (EM), que, primeiramente, cria uma função representativa da semelhança entre pixéis e, em segundo, procura maximizar essa função, procurando os pixéis que apresentam maior semelhança. Desta forma, é possível estimar os clusters que serão utilizados para posterior segmentação [5].

O algoritmo EM tem por base três passos.

Em primeiro lugar, calcula-se a média e a covariância das intensidades dos pixéis, recorrendo ao método K-means, e a probabilidade inicial de determinado grau de cinzento no interior do cluster (o que acaba por funcionar como uma inicialização). De seguida, tem-se o E-step (Expectation), que estima a probabilidade de pertença a um cluster, para cada pixel/vóxel, por inferência sobre a semelhança entre as características dos vários pixéis/vóxeis. Por último, tem-se o M-step (Maximization), o qual utiliza o modelo encontrado no E-step, fazendo um *fit* do modelo, ou seja, um ajuste dos parâmetros do modelo, sendo um método interactivo. Estes três passos são repetidos até se verificar convergência, obtendo-se um vector de Gaussianas para cada classe/cluster [16] [18].

Um exemplo de métodos baseados em densidade de probabilidade é a técnica de "*Descriminative Clustering*", um método semi-supervisionado, onde o treino é feito recorrendo a alguns dados com supervisão, *labels*. No caso deste estudo, esses são fornecidos pelo SOM. É um algoritmo que procura maximizar as margens, locais onde a variação dos valores dos graus de cinzentos é maior, por haver separação de tecidos diferentes, para, posteriormente, se efectuar uma melhor segmentação, dado que, quanto mais diferenciadas estiverem as fronteiras entre tecidos, mais fácil será a separação de cada tipo de tecido, ou seja, a segmentação. O método funciona melhor quando existe igual número de casos por tecido ou classe. No entanto, esse é raramento o caso em MRI. Assim, o DC poderá atribuir pixéis/vóxeis a classes de forma errónea, com base em informação incompleta ou número reduzido de pixéis para estudo de uma dada classe. Uma forma de contornar esta sobrelotação é a utilização do mesmo número de pontos por cluster [5] [19] [20].

Sendo o DC o método maioritariamente estudado, em parceria com o SOM, verifica-se que a sua utilização não depende de informação espacial como alguns dos métodos anteriormente apresentados. Apresenta, no entanto, falhas a nível de imagens que apresentem ruído, efeitos de inhomogeneidade do campo magnético ou desalinhamentos nas imagens. Por esta razão, neste estudo, utilizam-se imagens devidamente pré-processadas, isto é, sem ruído ou inhomogeneidades de campo e nas quais um pixel/vóxel se encontra na mesma posição em todas as imagens. Este pré-processamento, que não é aqui abordado, fará parte, como dito anteriormente da dissertação do colega Afonso Moreira [29].

3 Estado da Arte

Tendo em conta os métodos apresentados, muitos foram os estudos ao longo dos anos que propuseram a segmentação de tecidos em imagens de ressonância magnética.

Soltaninejad *et al* [1] propuseram um método de segmentação de hemorragias cerebrais baseado nas técnicas de *thresholding* e usando distância euclidiana. Na **Figura 3.1.** está representado o processo de detecção de uma hemorragia. Em primeiro lugar, recorrendo à definição de um limiar de intensidade, consegue-se separar tecido mole do crânio, não interessante para o estudo da hemorragia. De seguida, fez-se a divisão da imagem em múltiplos clusters, de forma quadrada, a que chamaram de super-pixéis, **Figura 3.1. (b)**. Estes primeiros agrupamentos serviram para atribuir, a cada pixel, um cluster, usando a distância entre os mesmos e os centros dos super-pixéis. Os pixéis foram atribuídos aos clusters cujos centros estivessem à menor distância do pixel escolhido. O subsequente agrupamento dos clusters/super-pixéis fez-se por análise das intensidades dos vizinhos. Os agrupamentos de super-pixéis com intensidades semelhantes foi o primeiro passo para delimitação dos contornos, **Figura 3.1. (c)**. Por último a imagem binarizada, a partir de um *threshold* determinado, revela a zona da hemorragia, **Figura 3.1. (d)**.

De forma a testar a qualidade do resultado proposto pelo método calcularam a sobreposição entre a segmentação obtida e o conhecimento da região real (*ground-truth*). Verificaram que, para hemorragias de pequenas dimensões, o método só seria viável se fosse feita uma redução das dimensões dos super-pixéis no início da segmentação [1].

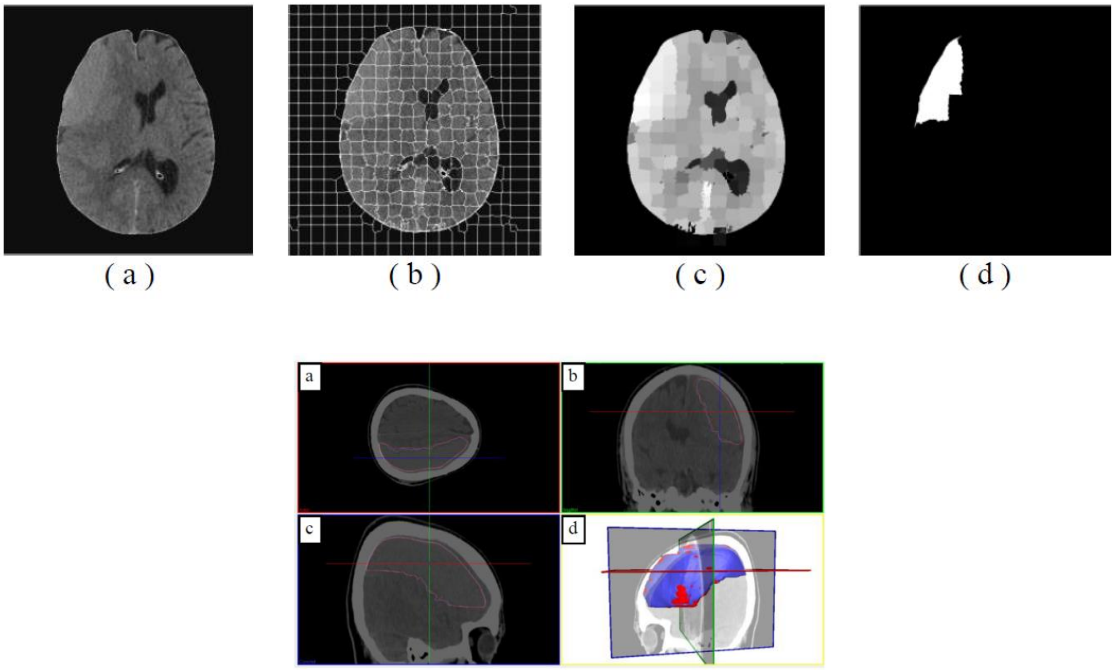


Figura 3.1 - Representação do processo. **Em cima: a)** Imagem recolhida; **b)** agrupamentos iniciais, em que a variação das dimensões irá alterar bastante o resultado final; **c)** agrupamento dos superpixeis; **d)** segmentação da zona hemorrágica. **Em baixo:** ilustração da hemorragia em 3D [1].

Ahmed *et al* [27], recorreu ao método Fuzzy C-means, optimizando-o de forma a conseguir ultrapassar os efeitos das inhomogeneidades do campo magnético, sumarizado na **Equação 3.1.** (sendo x_k o valor real da intensidade do pixel e β_k a contribuição da inhomogeneidade do campo no valor observado do pixel). A utilização do método mantém o uso da distância euclidiana e da atribuição das probabilidades de pertença de tecidos a diferentes clusters. A optimização vem com o acrescento de um termo que permite que a classificação de um pixel/vóxel fosse influenciada pelos seus vizinhos, por o método ser aplicado localmente, como ilustrado na **Equação 3.2.** (utilização do algoritmo FCM com o termo que traduz a contribuição dos pixéis vizinhos). Esta influência vem depender de um parâmetro α, coeficiente que considera a contribuição da vizinhança do pixel em questão, e do SNR. Quanto maior o SNR, menor deverá ser o parâmetro α.

O algoritmo parte também do princípio que o valor do pixel k depende da sua intensidade observada, e que esta inclui os efeitos da inhomogeneidade do campo. N_R traduz o número de vizinhos considerados. Os parâmetros usados nas distâncias euclidianas são os mesmos usados no método Fuzzy C-means e explicados no **Capítulo 2**.

$$y_k = x_k + \beta_k \qquad \forall\, k \in \{1, 2, \ldots, N\}$$

Equação 3.1 - Valor real da intensidade do pixel k, dado pela soma da intensidade observada com a contribuição da inhomogeneidade do campo magnético [27].

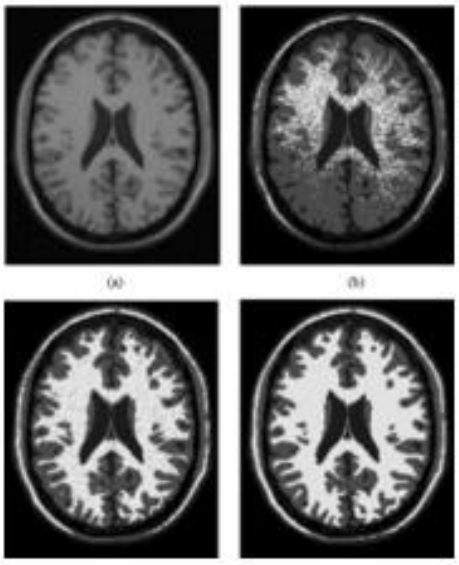
$$J_m = \sum_{i=1}^{c}\sum_{k=1}^{N} u_{ik}^{p}\|x_k - v_i\|^2 + \frac{\alpha}{N_R}\sum_{i=1}^{c}\sum_{k=1}^{N} u_{ik}^{p}\left(\sum_{x_r \in \mathcal{N}_k}\|x_r - v_i\|^2\right)$$

Equação 3.2 - Método FCM, melhorado pelo termo de classificação dos vizinhos, sendo Nr o número desses vizinhos [27].

O algoritmo pôde ser descrito como a optimização dos clusters normalmente criados pelo algoritmo FCM, estimando a contribuição da inhomogeneidade do campo magnético para a segmentação dos tecidos. Desta forma, é possível remover essa contribuição, e obter assim uma melhor classificação dos pixéis/vóxeis.

Os resultados obtidos foram comparados com o método FCM tradicional, com o método EM. A imagem utilizada continha 5% de ruído gaussiano e 20% de intensidade da inhomogeneidade.

Os resultados para diferentes níveis de SNR encontram-se na tabela da **Figura 3.2.** juntamente com as imagens de IRM obtidas com este método.



Segmentation Method	SNR		
	13db	10 db	8db
FCM	98.92	86.24	78.9
EM	99.12	93.53	85.11
BCFCM	99.25	97.3	93.7

Figura 3.2 - À esquerda: a)Imagem original pesada em T1; b) Segmentação usando FCM; c) Segmentação usando EM; d) segmentação usando FCM optimizado. **À direita:** Tabela com SNR para os três métodos comparados, verificando-se melhor qualidade de segmentação, dada pelos coeficientes, no método proposto por Ahmed *et al* [27].

Assim, conseguiram optimizar um método que, não só foi capaz de segmentar os tecidos da imagem, mas também conseguiu filtrar a imagem de maneira a retirar o máximo de ruído possível, daí o SNR do método proposto ser razoavelmente superior que os métodos com o qual foi comparado [27].

Considerando, agora, métodos de *thresholding*, apresentam-se estudos nos quais investigadores procederam à segmentação de imagens por definição de limiares de intensidades a partir de diferentes métodos.

Roy *et al* [6] propuseram um método para prever e segmentar lesões cerebrais, causadas por esclerose múltipla. A chave deste método foi recorrer a um método de Global Thresholding para remoção do crânio da imagem de interesse. O uso desse limiar permite que a binarização da imagem faça sobressair as zonas mais intensas, como as lesões causadas pela esclerose múltipla e o corpo caloso.

Após o processo de binarização da imagem, o fundo desta foi subtraído de forma a restarem apenas as lesões e o corpo caloso, **Figura 3.3. (b)**.

Sendo que a simples binarização da imagem não chega, por si só, para a correcta identificação das lesões, Roy et al, recorreram a três processos de *level set* (método que também é usado para correções de inhomogeneidades [28] e que obtem contronos por interseção de um plano com a superfície me causa) antes da aplicação do threshold. Estes

três passos evitam sobre-segmentação e também sub-segmentação, "arrumando" os pixéis, ou seja, os autores atribuíram delimitações que depois, consoante o peso dos pixéis na região, ou vizinhos, são movidas de acordo com o peso dos pixéis vizinhos para as fronteiras correctas dos objectos [6], o que se traduz num refinamento dos resultados depois de uma primeira estimativa. Para uniformização do processo de segmentação a "arrumação" dos pixéis levou à melhor representação dos contornos, **Figura 3.3. (c)** e **Figura 3.3. (d)**. De seguida foi adicionado o fundo para preparar a imagem para o processo de binarização, **Figura 3.3. (e)**. Com o processo de binarização, **Figura 3.3. (f)**, foi possível fazer sobressair as lesões de esclerose múltipla do resto dos tecidos do cérebro (fundo). A remoção dos tecidos saudáveis possibilitou a obtenção das áreas lesadas, **Figura 3.3. (g)**. Por último, **Figura 3.3. (h)**, as lesões foram sobrepostas à imagem inicial, para obter a sua localização espacial no cérebro.

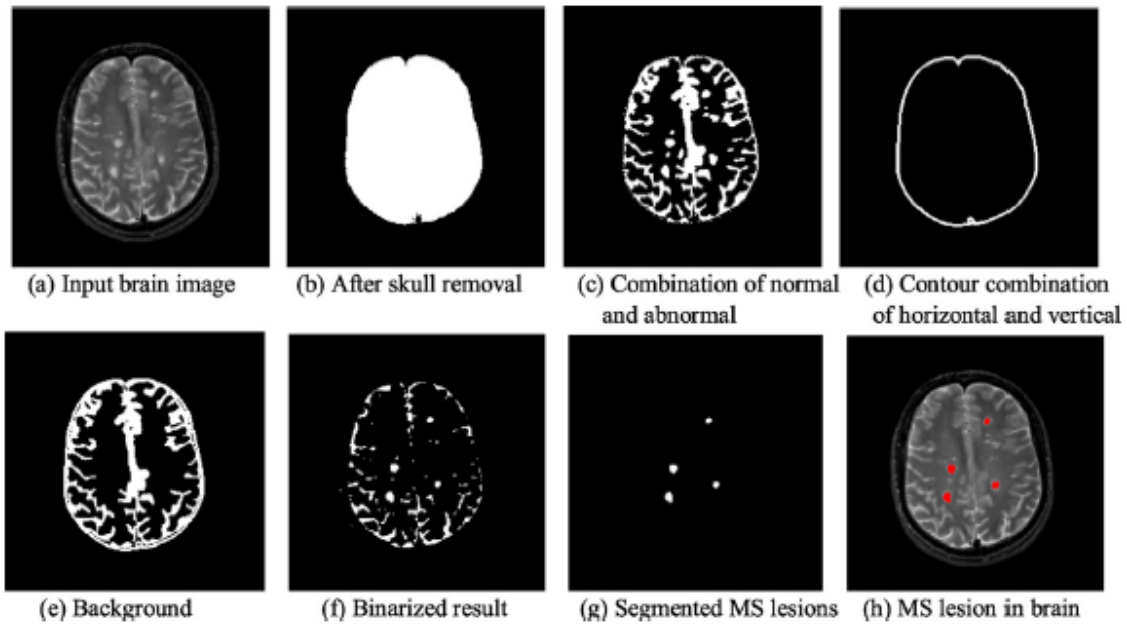


Figura 3.3 - Detecção das lesões de Esclerose Múltipla, partindo da remoção do crânio, level set, binarização e segmentação das lesões. Na imagem final, (h), observa-se as lesões identificadas sobrepostas à imagem original [6].

Joseph *et al* [8] propuseram um método de segmentação de imagens de MRI baseado em K-means, para imagens de cérebros com tumor, com o objectivo de isolar e identificar a área de lesão.

Como explicado anteriormente, K-means é um método de "*hard clustering*", ou seja, um pixel pertence a um e só um cluster. Também foi notado atrás que tanto a técnica de K-means, como a de FCM (Fuzzy C-means), não têm em conta a informação espacial

ou morfológica da imagem. Para isto, Joseph *et al* recorreram ao uso de filtros morfológicos para uma melhor detecção da região do tumor.

Os filtros referidos incluem técnicas de erosão e dilatação nas quais, aquando da sobreposição do *kernel* (filtro, na forma de matriz, aplicado à matriz da imagem) nos pixéis da imagem. O pixel-alvo, que fica no centro do *kernel*, toma o valor mais baixo ou mais alto, respectivamente, da área abrangida pelo *kernel*.

Com este método, sumarizado no diagrama apresentado na **Figura 3.4.**, conseguiram segmentar diversos tumores de imagens de MRI retiradas de uma base de dados [8]. Como mostrado na **Figura 3.5.**, as imagens da esquerda correspondem à identificação de clusters. A delimitação dos contornos das imagens de cima corresponde ao uso de operações morfológicas que facilitam a identificação da zona lesada pelo tumor. Este processo irá funcionar como auxiliar do método de K-means, melhorando a segmentação, como se comprova na imagem da direita.

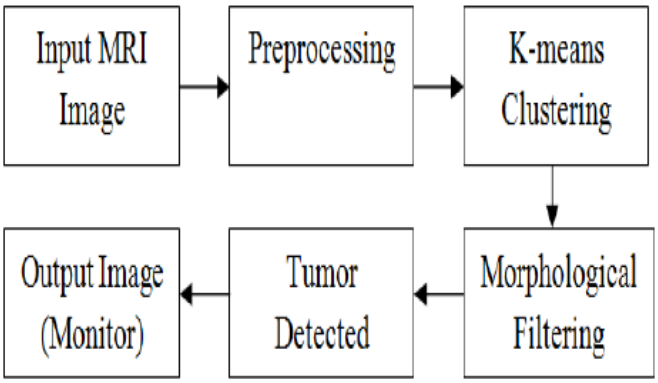


Figura 3.4 - Diagrama de blocos proposto [8].

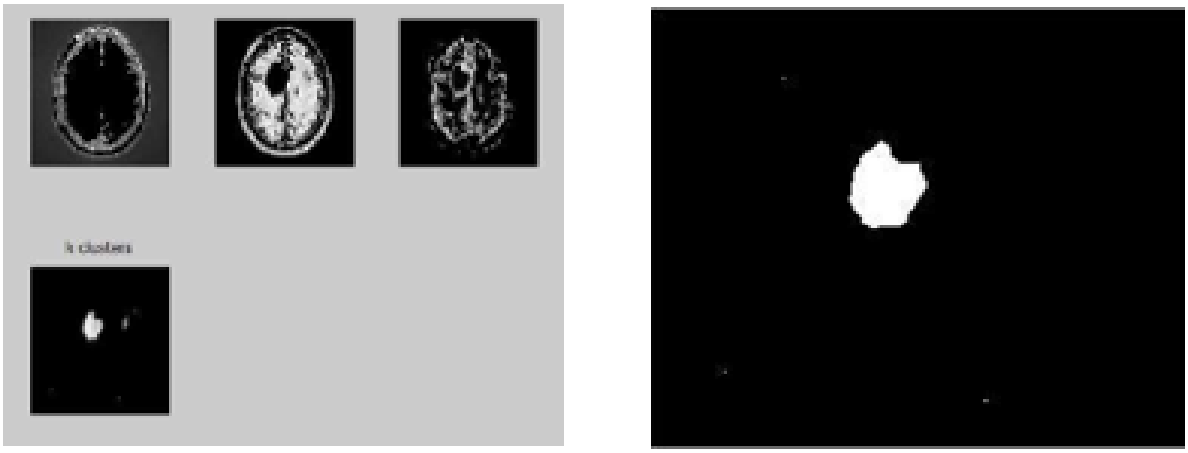


Figura 3.5 - **À esquerda**: Detecção de clusters através do método K-means seguido de erosão/dilatação. **À direita**: Detecção do tumor [8].

Patil *et al* [4] recorreram também ao método k-means para agrupamento de pixéis em clusters, na detecção de tumores cerebrais, usando imagens de pacientes em tempo real. No entanto, em vez do uso apenas de operações morfológicas para a segmentação dos tecidos, recorreram ao método de threshold como pré-processamento seguido da técnica de watershed.

A aplicação destas técnicas às imagens tumorais foi antecedida de um pré-processamento que passou pela conversação de RGB para uma escala de cinzentos, a passagem por um filtro passa-alto, e um filtro de mediana para melhorar a qualidade da imagem.

Depois deste pré-processamento aplicaram a técnica de K-means para encontrar um threshold fiável para a binarização da imagem, **Figura 3.6**. Em sequência, recorreram ao método de watershed, cuja aplicação facilitou a distinção das delimitações dos segmentos criados pela binarização. Por último, o uso de operações morfológicas, serviu para alisamento das fronteiras, com a finalidade de melhor a qualidade da detecção [4]. O desenvolvimento deste processo encontra-se ilustrado na **Figura 3.7**.

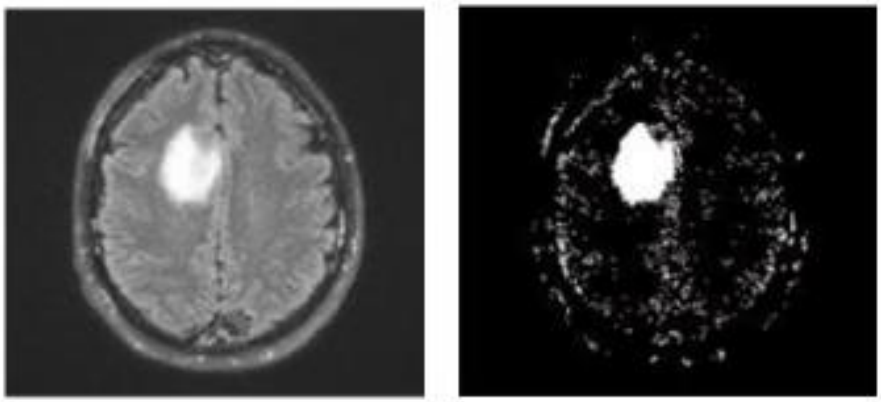


Figura 3.6 - À esquerda: imagem obtida por pré-processamento; à direita: imagem binarizada após aplicação de k-means [4].

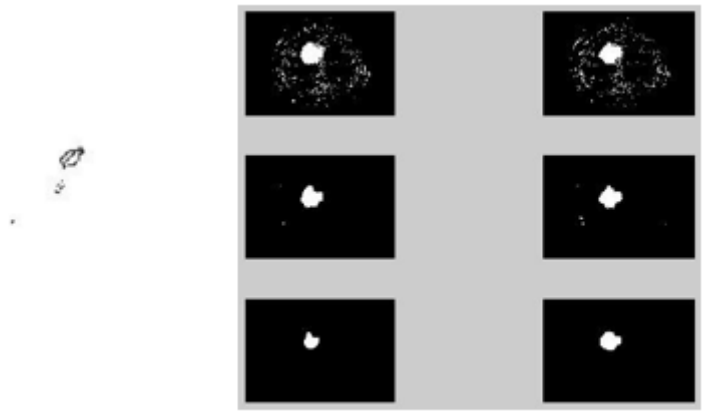


Figura 3.7 - À esquerda: aplicação da técnica wathershed para detecção do tumor; à direita: aplicação de operações morfológicas para eliminar componentes restantes da imagem que não façam parte da área do tumor [4].

Atkins *et al* [10] propuseram um método para segmentação do cérebro recorrendo à criação de uma máscara, aplicada a imagens de MRI obtidas com diferentes resoluções e através de diferentes sequências de eco.

O método de segmentação proposto foi dividido em três passos:

- **Cabeça:** O primeiro passo baseou-se na obtenção do histograma com as intensidades dos pixéis da imagem para remoção de ruído de fundo e criação de uma máscara inicial opara identificação da cabeça.

- **Correcção de inhomogeneidades**: O segundo passo consistiu na produção de uma máscara que identificasse correctamente as margens do cérebro. Usando um filtro de difusão anisotrópico não-linear (os valores dos graus de cinzentos dos pixéis são combinados com uma função não-linear para detecção de, por exemplo, cantos), foi

possível contrariar as inhomogeneidades do campo e reduzir as intensidades de zonas não pertencentes ao cérebro, como o escalpe.

- **Segmentação**: No terceiro e último passo geram a máscara final recorrendo a um algoritmo de detecção das margens do cérebro "(…)even in the presence of parcial volume effects." [10].

A máscara obtida é binarizada por thresholding das projecções verticais e horizontais da imagem, ou seja, o número de pixéis contados em x e em y.

O valor de threshold é escolhido pela sobreposição da curva gaussiana que melhor se ajusta ao histograma da imagem ponderada em T2 [10] (**Figura 3.8.**).

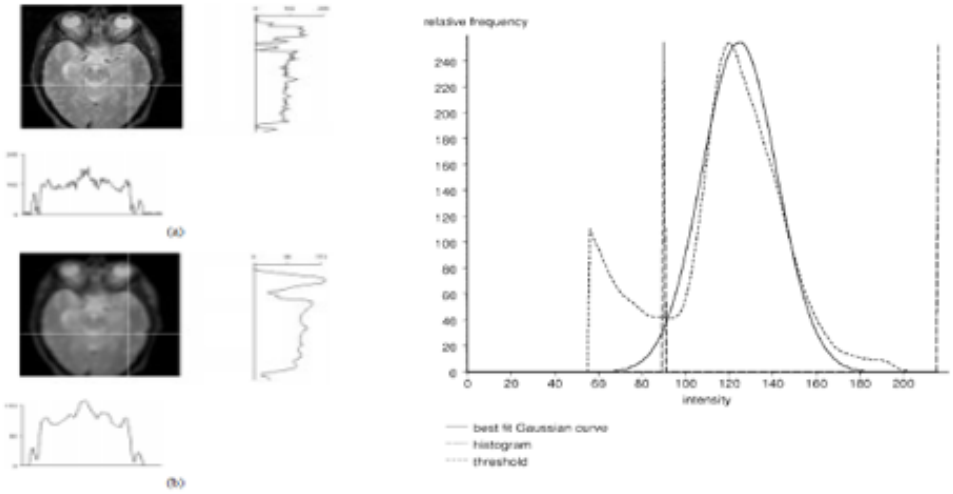


Figura 3.8 - **À esquerda**: projeções das (a) imagem ponderada em T2 e (b) imagem obtida por difusão; **À direita**: cálculo do *threshold*, por sobreposição da curva gaussiana ao histograma da imagem [10].

A aplicação da máscara às imagens obtidas discriminou tecidos que não fazem parte do cérebro, como os olhos, e, como dito, suavizou a intensidade do crânio, por isso a sua utilização para segmentação do cérebro em diferentes imagens revelou ser fiável, por reduzir as hipóteses de existirem pixéis com intensidades irreais que levam a uma má classificação do pixel em questão, existindo uma boa detecção de margens/fronteiras do cérebro como se verifica pela **Figura 3.9.**.

Temos, ainda, uma primeira utilização de IRM multi-espectral e baseada na recolha de várias imagens com sequências diferentes.

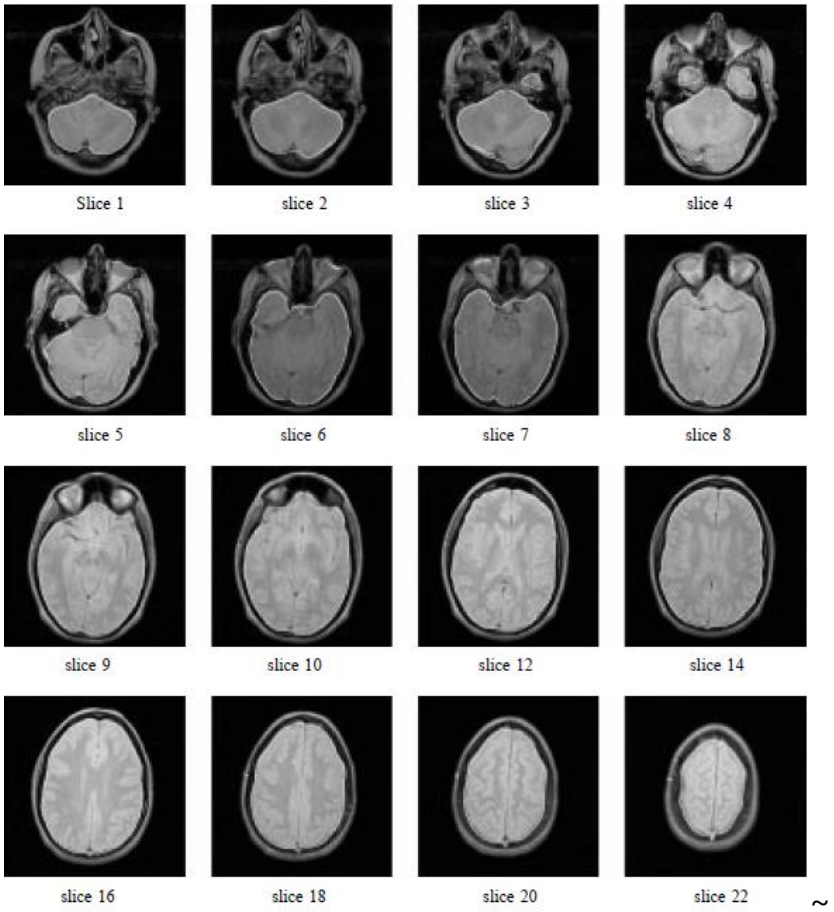


Figura 3.9 - Resultado da aplicação da máscara às diferentes imagens usadas [10].

Nicolau Gonçalves [5] recorreu a um método auto-supervisionado (a aprendizagem não requere a inserção de informação de classe pelo utilizador, sendo a atribuição de *labels*, i.e. atribuição de pixéis/vóxeis a determinadas classes, autónoma) para segmentação de tecidos, que em termos de qualidade de segmentação e previsão de evolução de lesões degenerativas revela ser dos métodos mais adequados, quando comparado com os estudos anteriormente descritos. Será, também este, o método no qual se irá focar esta dissertação.

Dado que a maioria dos métodos até agora propostos, aposta em técnicas de "*Hard Clustering"*, cada pixel só pode ser atribuído a um tipo de tecido e, por isso, a um cluster. Naquele trabalho foi apresentado um método de "*Soft Clustering"* que reconhece que cada pixel/vóxel representa entre um a vários tipos de tecidos, o que implica que existe uma probabilidade de pertença que tem de ser atribuída.

O método proposto começa pelo *labelling* de alguns pixéis, num processo consistência de *clustering*, usando SOM (*Self Organizing Maps*).

De seguida, por *Discriminative Clustering* (DC), os pixéis/vóxeis são distribuídos por clusters, com uma determinada probabilidade associada. Esta atribuição de probabilidades possibilita a previsão de futuras lesões e o acompanhamento de doenças degenerativas, uma vez que permite ter acesso a informação ainda não completamente declarada correspondente à lesão.

As doenças degenerativas podem, por vezes, causar imprecisões na identificação de tecidos, por não existir uma "fronteira" definida entre os segmentos ou os clusters que se pretende diferenciar [5]. Na **Figura 3.10.** é possível observar o resultado da segmentação pelo método apresentado.

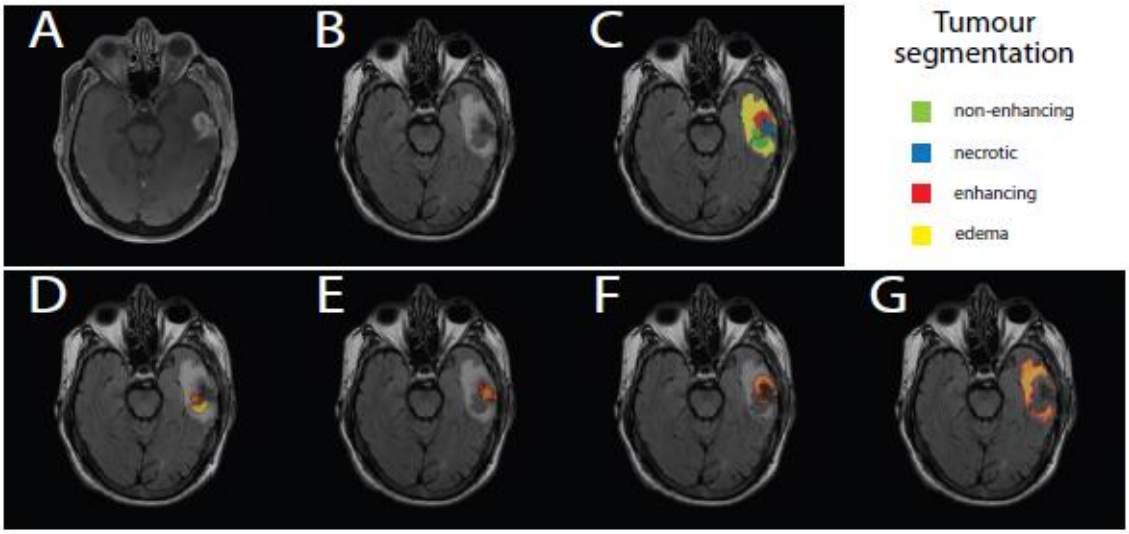


Figura 3.10 - Segmentação de tumor usando o método descrito. **A**: ponderação em T1; **B**: Imagem com ponderação FLAIR; **C**: Segmentação por *Hard Clustering*, recorrendo ao métod K-means; **D a G**: segmentação de várias estructuras que compõem o tumor baseado em *Soft Clustering* para diferentes áreas do tumor [5].

4 Objectivos específicos

O objectivo deste trabalho é optimizar um conjunto de ferramentas que permitam segmentar tecidos em imagem de ressonância magnética.

Porque grande parte dos estudos feitos até agora, como aqueles apresentados no capítulo anterior, utilizam métodos de binarização, que atribuem apenas um tipo de tecido a cada pixel/vóxel, existe a necessidade de desenvolver um método que tenha em conta a presença de vários tipos de tecidos em cada pixel/vóxel manifestado através de volumes parciais. Pretendemos, assim, atribuir probabilidades ou graus de pertença de cada pixel/vóxel a diferentes tipos de tecidos.

Através de imagens de ressonância magnética multi-espectrais, ou seja, obtidas com mais do que um tipo de sequência de imagem, pretende-se optimizar as ferramentas para segmentação de tecidos desenvolvidas até agora.

De forma mais específica, o trabalho desenvolvido nesta dissertação irá procurar melhorar as técnicas de segmentação para aumento da qualidade de diagnóstico de doenças/lesões cerebrais como, por exemplo, tumores, hemorragias, Esclerose Múltipla, doença de Alzheimer, entre outras. Para atingir o objectivo deste trabalho, utilizar-se-á imagens de ressonância multi-espectrais, isto é, obtidas com várias sequências. O método a optimizar fará uso dessa informação acrescida na identificação da probabilidade acima.

5 Materiais e Métodos

Os resultados finais desta dissertação serão suportados por dois passos gerais para identificação de lesões cerebrais e estudo de MS, em imagens adquiridas em Helsinquia, Finlândia, referentes a um paciente com Esclerose Múltipla, com um intervalo de três anos entre as primeiras aquisições e as segundas, de forma a avaliar o desenvolvimento das lesões ou o aparecimento de novas.

O primeiro passo será um pré-processamento das imagens recolhidas, que irá corresponder a outra dissertação, no âmbito de uma tese de mestrado em Engenharia Biomédica, mas que complementa o trabalho desenvolvido na presente tese. Este pré-processamento terá como objectivo o alinhamento de imagens de forma a garantir que, em imagens obtidas com sequências diferentes, os pixéis/vóxeis correspondam ao mesmo local. Irá também remover ruido, melhorando o SNR, equalizar os histogramas da imagem para equilíbrio de graus de cinzento e, se possível, remover partes da imagem que não farão parte do cérebro como, por exemplo, o crânio.

Referente à segmentação em si, o procedimento experimental irá organizar-se da seguinte maneira:

- *Labelling* dos pixéis/vóxeis usando o método "*Self Organizing Maps"*. A aplicação múltipla de SOM e subsequente análise de consistência, irá levar à identificação dos *labels* desejados.

- Uso da técnica "*Discriminative Clustering"*, para agrupar os pixéis/vóxeis, e para a obtenção das probabilidades de pertença de cada pixel/vóxel a um tipo de tecido.

A aplicação destas técnicas terá como base o uso de *Matlab* e de *Python* para escrita do código, em todos os passos anteriormente identificados.

No seguinte diagrama, **Figura 5.1.**, estão representados os passos a tomar para a concretização do objectivo. A presente dissertação, irá partir de imagens multi-espectrais de MRI, já sujeitas ao pré-processamento identificado em cima. De seguida irá passar por propostas de agrupamento de pixéis/vóxeis com base na utilização de SOM. Por último, através de DC obter-se-ão os clusters juntamente com as probabilidades de pertença associadas aos pixéis. Desta forma, conseguir-se-á proceder à segmentação dos vários tecidos presentes nas imagens de MRI, como ilustrado no diagrama da **Figura 5.1**.

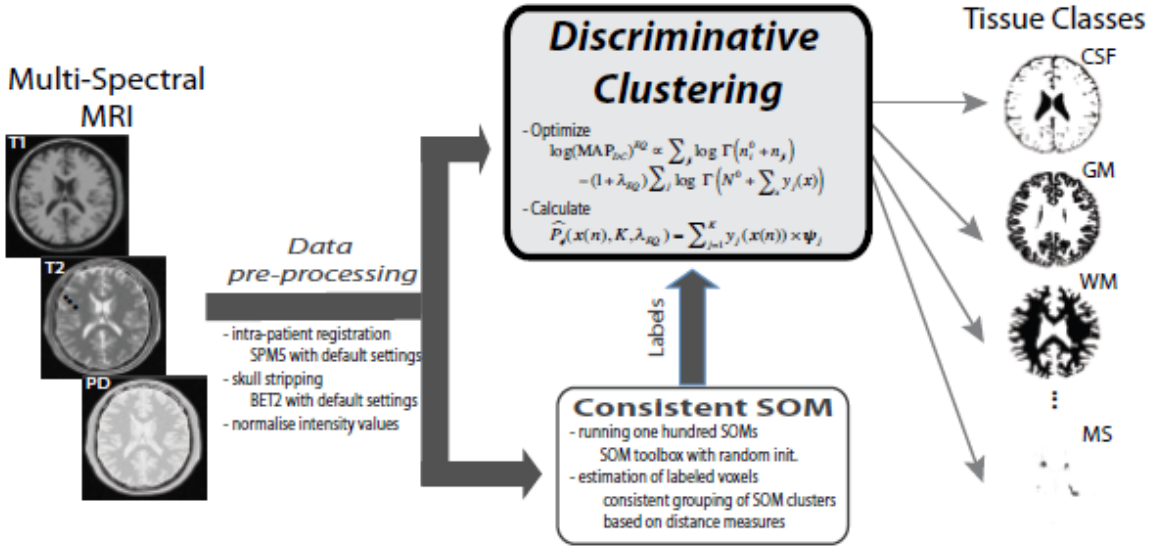


Figura 5.1 - Diagrama representativo do procedimento experimental, para a realização dos objectivos desta dissertação [5].

5.1 Estructuras

A aplicação do algoritmo proposto, como dito acima, é feita em imagens de ressonância magnética multi-espectrais, apresentando-se, neste caso, sob a forma de três sequências diferentes: FLAIR, T2, MTI (Transferência Magnética por acoplamento de *spins* não nulos).

Dado que o que é utilizado no decorrer do algoritmo não são, apenas, as imagens, foi necessário guardar a informação em estructuras, necessárias ao longo dos vários processos algorítmicos. Informação esta que se baseou em segmentações feitas por software específico para o efeito, bem como para a existência de *ground-truth*, vectores contendo as alturas com informação mais úteis ou o número de imagens (sequências) usadas, ou ainda os parâmetros específicos a cada algoritmo.

Foram usadas cinco dados diferentes para teste do algoritmo, de nome: DATA_HE42, contendo os dados originais, DATA_HE42_followup, contendo os dados de evolução das lesões do paciente, DATA_HE42_followup_masked, contendo os dados de evolução mascarados, DATA_HE42_masked, com os dados originais mascarados e DATA_HE42_merge, contendo uma sobreposição dos dados originais e evolutivos. No entanto, os dados realmente usados são baseados em estructuras, ficheiros *.mat*, que albergam vários tipos de informação:

- **Info**: nome do ficheiro que pode variar consoante as cinco estruturas identificadas anteriormente, identificando-se como HE42, HE42_followup, HE42_followup_masked, HE42_masked e HE42_merge. Esta informação é frequentemente útil para a criação de novas estruturas contendo os resultados obtidos e definição de directórios para a gravação desses resultados.

-**Dataname**: este campo identifica as diferentes sequências que se encontram dentro da estrutura. No caso de estudo, FLAIR, T2 e MTI.

-**Height**: representa uma das dimensões das imagens, a Altura, e toma o valor de 256.

-**Width**: Caracteriza a outra dimensão das imagens, a Largura, e, à semelhança da altura, também toma o valor de 256.

-**ORIG**: este campo é, em si, uma estructura, representando o Ground Truth para comparação dos resultados obtidos. Dentro deste campo existem três variáveis, Volume, CRISP_GROUND_TRUTH e FUZZY_GROUND_TRUTH (este último, vazio). Volume representa as dimensões das imagens com todas as fatias consideradas, num total de 19 fatias.

-**Crisp_truth**: Obtido com ferramentas de segmentação como o SPM, representa o *ground-truth*, que se baseia na segmentação das imagens. Esta é produzida/estimada com base em software que apresenta modelos específicos para o fazer.

-**Classes_truth**: representa as classes idealmente presentes nas imagens estudadas e representativas dos tecidos encontrados. Este campo é um vector de dimensão cinco, cada uma respectiva a um tecido, WM (White Matter), GM (Grey Matter), CSF (Cerebral Spinal Fluid), MS (Multiple Sclerosys) e Background, correspondente ao fundo das imagens.

-**Use_mask**: neste campo, existe apensa a indicação de se foi utilizada máscara previamente para remover pixéis/vóxeis que não pertencessem ao cérebro como, por exemplo, o crânio.

-**Mask**: representa a máscara usada em cada uma das fatias, para retirar elementos que não pertencem ao cérebro, daí ter dimensões de 256 por 256 por 19.

-**Z**: representa as fatias onde, efectivamente se encontra cérebro. Apesar de existirem 19 fatias, grande parte delas não apanha o suficiente do cérebro ou não o apanha

de todo, o que leva à seleção de fatias em que o corte revele toda a informação que é útil para o processo.

-**n_images**: o número de imagens diz respeito às imagens obtidas de cada sequência. No caso em estudo, uma imagem por sequência.

-**Volume**: representa o volume das imagens tendo em conta a altura, largura e número de fatias.

-**I**: baseado em z, é utilizado para o *plot* das imagens, no fim de cada passo.

-**DataMasked**: apresenta o resultado da aplicação da máscara nas imagens das várias sequências. Como exemplo, a remoção do crânio e dos olhos das imagens iniciais. A codificação é sob a forma de três vectores normalizados apenas com os valores de pixéis não nulos.

-**Algorithm**: identifica o algoritmo usado para a formação de clusters originais, no caso desta dissertação, SOM.

Durante as corridas do algoritmo vão sendo formadas mais estructuras, cada uma delas representativa das corridas do SOM; dos clusters formados; dos super-vóxeis criados; dos resultados do DC, contendo informação sobre, também, as suas corridas. São, assim, uma forma simples e organizada de guardar a constante criação de bastante informação, como matrizes de grandes dimensões, e que permite fácil acesso a essa mesma informação, como se pode verificar na **Figura 5.2.**

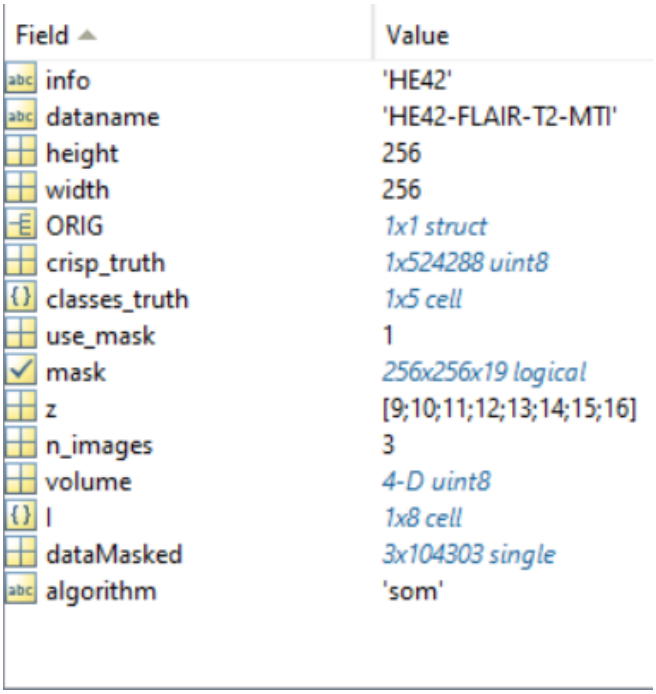


Figura 5.2 - Imagem representativas das estructuras usadas, neste estudo, DATA_HE42.

Com a utilização das estructuras, procedeu-se à optimização, propriamente dito, do algoritmo.

5.2 Optimização do algoritmo

O desenvolvimento do algoritmo (*flow* no **Anexo 9.1.**) assenta em várias funções escritas em Matlab (esqueleto do algoritmo e SOM) e em Python (desenvolvimento do DC). No entanto o script onde se encontram esta funções, encadeadas, tem o nome de *script_segmentation*. Este script representa o guião de todo o processo e de onde é possível recolher os resultados dos SOM ou do DC, bem como as imagens importantes como, por exemplo, as resultantes de pequenas filtragens, dos agrupamentos de pixéis ou da averiguação de erros e eficácia do algoritmo final.

Em primeiro lugar, encontra-se o carregamento dos dados, para que possam ser utilizados.

5.2.1 Loading Data

O primeiro passo para a optimização do algoritmo passa pelo carregamento dos dados a trabalhar, guardando-os numa variável. Para isto existe a função *load_data* que apresenta o output DATA que permite a posterior utilização da estrutura com as imagens de ressonância magnética. As entradas desta função são *data_type*, que é o nome da estrutura original, e *only_data_vector*, que averigua da existência de conteúdo na estrutur,a e que é inicialmente colocado a *false*.

Esta função encontra a directoria a utilizar, onde estão guardadas as estruturas de dados, directório esse que é definido à vontade do utilizador.

De seguida verifica a existência de um *path* que leve ao ficheiro em causa para ser carregado, analisando tanto o nome do ficheiro como a sua extensão.

Nesta função pode observar-se a ocorrência de diversas condições de *if*. Isto prende-se com o facto de o *load_data* ter sido desenvolvido não só para verificar a existência da estructura no directório escolhido, mas também para analisar o conteúdo da estructura, ou seja, os seus diversos campos, e carregá-los individualmente.

Após o carregamento dos dados, foi incluído, apenas para acompanhamento do conteúdo dos campos, um ciclo *for* com o objectivo de observar as imagens no campo da máscara, do *ground_truth* e da imagem com a máscara aplicada o que fornece uma ideia do foco principal do algoritmo e da fasquia a tentar atingir em termos de segmentação e, também das imagens as utilizar.

Apesar de as imagens usadas não precisarem de passar por um pré-processamento, por já estarem alinhadas e sem inhomogeneidades de campo magnético, procedeu-se a uma pequena suavização das mesmas, de forma a reduzir um possível ruído que pudesse existir.

5.2.2 Data Preprocessing

Esta componente do algoritmo, traduzida na função *preprocess*, permite fazer um *smooth* às imagens, de forma a reduzir pequenos ruídos ou a melhorar um pouco a qualidade da imagem após o pré-processamento e antes das corridas do SOM. Passa, assim, por dois processos:

i) Preprocess Filtering

Neste caso, recorre-se à filtragem das imagens (função *preprocess_filtering*), que implica a utilização da máscara e o Z (fatias da imagem que, efectivamente, continham imagem) e aplicar uma normalização da matriz da imagem. Esta filtragem foi, assim, inserida no campo *dataMasked* da estructura DATA.

O objectivo principal deste passo, é realçar um pouco as fronteiras entre tecidos, e facilitar o *clustering*.

ii) Preprocess ICA

Como dito anteriormente, a técnica de ICA baseia-se na separação do sinal em componentes independentes. Recorrer a esta técnica, no caso de filtragem, significou usar as componentes independentes para identificar possíveis ruídos, ou frequências fora do comum, de forma a eliminar esses intervenientes.

Para aumentar a robustez dos resultados encontrados por ICA, este algoritmo é corrido 100 vezes, e os seus resultados, agrupados por semelhança.

Acontecendo estes dois processos de filtragem, o ficheiro *preprocess.mat* converge as duas técnicas de forma a existir uma melhor filtragem das imagens, alterando o campo *dataMasked* da estructura DATA através da normalização de matrizes e da aplicação dos filtros.

Com a garantia de que qualquer ruído existente ou, pelo menos a grande maioria, foi removida da imagem, procedeu-se à realização das múltiplas corridas do SOM para obtenção de *labels* para alguns pixéis/vóxeis.

5.2.3 Run Clustering Method (SOM) and Clusters Creation

O SOM, tal como dito anteriormente, é um método de clustering que varia com a inicialização que lhe é feita. Assim, é para aumentar o poder de separação, bem como a confiança na mesma, que se treinam vários SOM, em inicializações aleatórias e forma-se o conjunto de agrupamento mais robusto. Representado na função *do_clustering*, o SOM é aplicado com 100 corridas. Foi necessário modificar as dimensões dos mapas do SOM para obter uma discriminação mais acentuada dos protótipos propostos pelo algoritmo.

Aplicar 100 corridas, apesar de computacionalmente mais pesado que, por exemplo, uma ou duas corridas, leva a uma maior especificidade. Se nas 100 corridas, 90 mapas revelam que, recorrentemente, o mesmo conjunto de pixéis se encontra agrupado, então conseguimos dizer que aqueles pixéis, provavelmente, pertencem a uma mesma classe (ou mesmo tipo de tecido). Se existirem pixéis erráticos que não revelam consistência, corrida após corrida, ou seja, que numa estão agrupados com certos pixéis, mas nas seguintes já estão noutro agrupamento, não podemos dizer que aquele pixel pertence a qualquer tecido especifico (será sobre esses pixéis que, mais tarde, será aplicado o DC).

Também o aumento das dimensões dos mapas, leva a que o tempo de computação seja mais elevado, devido à dispersão dos pixéis por uma área maior. Esta dispersão leva a que o SOM extraia dos mapas mais agrupamentos, que poderão, ou não, ter um número mais reduzido de pixéis, o que aumenta a sua especificidade. O uso de dimensões muito elevadas para os mapas poderá levar a uma sobre-segmentação e à separação de pixéis em diferentes agrupamentos que, teoricamente, deveriam estar presentes no mesmo agrupamento ou super-vóxel (agrupamento de vários vóxeis com características semelhantes, para atribuição a uma classe).

Para a criação dos clusters, no que toca ao agrupamento entre várias corridas, definiu-se um *threshold* de **0,865** a partir da normalização, ou seja, os valores do *threshold* deveriam estar entre 0 e 1.

A escolha deste valor teve de ser cuidadosa, porque um valor muito baixo não permitiria a formação de vários agrupamentos, nem a diferenciação dos diferentes *clusters*. Por outro lado, um valor muito elevado levaria a uma sobre segmentação de agrupamentos, constituídos por um número baixo de pixéis o que, mais uma vez, levaria a uma discriminação menos boa dos tecidos.

A criação dos *clusters* traduz-se na função *clustering_create_clusters*. O critério para o agrupamento baseia-se numa correlação de 0,9 (90%) entre candidatos a um agrupamento. Isto quer dizer que, para que dois pixéis pertençam ao mesmo agrupamento, será necessário que o seu conteúdo, ou que o seu valor, apresente uma semelhança de 90%. Este critério, definido por K-means, revela ser útil, na medida em que permite um aumento de especificidade e rigor na formação de *clusters*, e impede que pixéis com características diferentes sejam agrupados sob a mesma classe.

Como dito anteriormente, ocorre formação de estructuras no decorrer do processo de segmentação. No caso do SOM, forma-se uma estructura de nome CLUSTERS_SOM_MASK_HE42 (no caso original). O nome que aparece no fim da designação desta estructura vem da variável *data_info*, que guarda o nome da estructura em estudo. Nesta estructura, são gravadas três variáveis: *clusters*, *clusters_1* e *vars_saved*. As duas primeiras são, também elas, estructuras cuja diferença de valores está na diferença dos valores das correlações (0.9 ou *threshold*). A última variável indica os nomes das variáveis que estão a ser guardadas (as três acima).

Esta função de criação dos *clusters* chama um script de Matlab que, por si mesmo, tem a definição do algoritmo do SOM. O que se requer que o utilizador faça é, como dito acima, que designe um número específico de corridas a serem feitas.

No caso de já existirem *clusters* formados, existe a função *load_clusters* que faz o carregamento das mesmas.

5.2.4 Super-Voxels Selection

É importante distinguir a criação de *clusters* da criação de super-vóxeis.

Clusters traduz-se no agrupamento de pixéis/vóxeis que apresentam características semelhantes, podendo representar a mesma classe, ou tecido. Como dito anteriormente, podemos falar de intensidades de tons de cinzento, ao falarmos destas semelhanças.

Por outro lado, super-vóxeis são os agrupamentos de vóxeis que, efectivamente, se encontram sempre juntos em todas as corridas de SOM. Cada mapa proposto pelo o SOM, inicializado aleatoriamente, propõe agrupamentos de pixéis/vóxeis que podem variar nas localizações propostas pelo SOM. Os super-vóxeis correspondem aos pixéis/vóxeis que aparecem, em grande parte das corridas, sempre juntos.

Na verdade, os super-vóxeis são os pixéis-vóxeis que apresentam características mais semelhantes entre eles.

Esta selecção é feita com uma interface (*super_selection_gui*) cujos inputs são o *data_info* (nome da imagem) e o *data_alg* (o algoritmo usado, neste caso o SOM). Esta interface mostra os agrupamentos propostos pelas 100 corridas de SOM com base no *threshold* anteriormente definido para combinação de *clusters*. Com um *threshold* baixo não existe realmente uma separação clara que indicasse diferença de tecidos. Ao invés,

com um *threshold* alto, existe uma separação mais específica e rigorosa dos agrupamentos, o que pode levar a uma possível sobre-segmentação, mas também a um critério de seleção melhor, no caso de o utilizador ser mais conservador na sua escolha. A **Figura 5.3.** mostra um exemplo do conteúdo da interface. É possível observar que além das imagens com as propostas de agrupamentos existem também gráficos de densidade de probabilidade que funcionam como auxiliadores para diferenciação de tecidos, mostrando o peso dos pixéis atribuídos a cada tecido, dentro de toda a imagem.

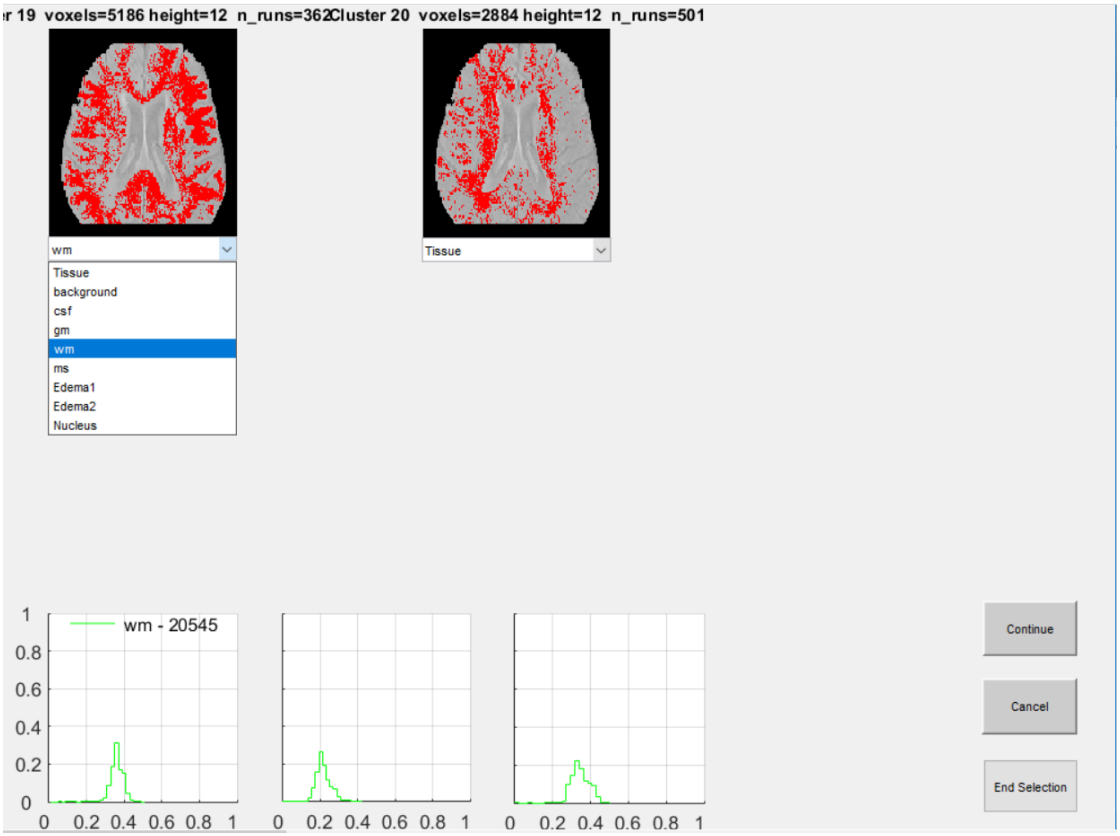


Figura 5.3 - Representação da interface de seleção dos super-vóxeis, demonstrando as propostas de agrupamentos do SOM, os possíveis tecidos e as distribuições de probabilidades para cada tecido.

Após a seleção os super-vóxeis são carregados através da função *load_super*, que procura a existência de super-vóxeis gravados na pasta escolhida para armazenamento, e os carrega. Utiliza-se um *try/catch* para, no caso em que, devido a um erro de compilação, os super-vóxeis não tenham sido guardados, a consola corra um *print* que indique que o ficheiro que os continha não foi encontrado. A utilização do *try/catch* possibilita a

continuação do programa, permitindo o *override* do erro sem paragens. O resultado poderá não ser o mesmo, mas será possível correr o algoritmo até ao fim.

Com o intuito de observar os resultados das corridas do SOM, recorreu-se à função *show_images*. Esta função utiliza a altura dada como argumento, assim como a estructura que guarda os super-vóxeis, para apresentar o resultado do algoritmo SOM. Na única intervenção humana, o utilizador, visualizando as propostas de super-vóxeis pode escolher de entre as classes de tecidos possíveis para relacionar os super-vóxeis a *labels*.

A altura escolhida como argumento da função *show_images* tem em conta que, dentro dessa função, existe uma outra que converte a altura para estar de acordo com um índice máximo de 32, no caso da estructura inicial, ou seja, neste caso 4 imagens para 8 alturas em que o corte transversal revela uma boa observação de todos os tecidos. Neste caso da estructura inicial, utilizou-se o valor 4 para que a conversão originasse a altura 12, coincidente com aquela que melhor discrimina os tecidos presentes e com o campo *DATA.z*.

Posto isto, o facto de já existirem pixéis/vóxeis com *labels* atribuídas, possibilitou a realização do DC. O DC necessita de algumas *labels*, de forma a inferir sobre a atribuição dos restantes pixéis às diferentes classes. A realização do SOM, e seleção de super-vóxeis deu, assim, as bases para a classificação feita pelo DC, a seguir explicada.

5.2.5 Classification

A classificação passou por atribuir graus de pertença aos vóxeis por agrupar, ou seja, que não tinham sido considerados como super-vóxeis durante as múltiplas corridas do SOM. Isto foi feito recorrendo ao método de *Discriminative Clustering* (DC).

Foi necessário envolver certos parâmetros na criação das corridas de DC, pela implementação da função *DC_create_runs*. Estas foram definidas para serem 144, uma vez que, tal como no SOM, quanto mais corridas, mais especificidade se obtém. Tal como o Som, também o DC beneficia de uma análise de robustez, traduzida com múltiplas corridas do algoritmo.

Criar a estrutura que traduz as corridas de DC implica designar todos os parâmetros em cada corrida. Foi usado um ciclo *for* para que, a cada corrida, fossem adicionados os parâmetros pela ordem: *n_clusters*, *entropy*, *use_equal*, *sigma*, *voxel_stats*, *lambda* e *use*.

Estes parâmetros apresentam-se essenciais às corridas do DC e os seus valores foram selecionados por experimentação, estando presentes na versão original da função *DC_create_runs*.

A **Figura 5.4.** representa a estrutura criada pela função *DC_create_runs*, com os seus parâmetros discriminados.

Fields	n_clusters	entropy	use_equal	sigma	lambda	use
1	7	0.2000	0	0.1500	0	0
2	7	0.4000	0	0.1500	0	0
3	7	0.6000	0	0.1500	0	0
4	7	0.8000	0	0.1500	0	0
5	7	1	0	0.1500	0	1
6	7	0.2000	0	0.1500	0.3000	0
7	7	0.4000	0	0.1500	0.3000	0
8	7	0.6000	0	0.1500	0.3000	0
9	7	0.8000	0	0.1500	0.3000	0
10	7	1	0	0.1500	0.3000	0
11	7	0.2000	0	0.1500	0.6000	0
12	7	0.4000	0	0.1500	0.6000	0
13	7	0.6000	0	0.1500	0.6000	0
14	7	0.8000	0	0.1500	0.6000	0

Figura 5.4 - Conteúdo da estructura de dc_runs com os parâmetros representados.

Criando as corridas foi possível correr o DC propriamente dito, através da função *do_DC* que, por sua vez, chama duas funções distintas: *DC_classify* e *DC_clean_voxstats*.

i) *DC_classify*: sendo o núcleo do DC, este *script* alberga a função de escolha de directoria (*DC_get_directory*), para guardar a informação resultante do DC; toda a transformação dos ficheiros de input para a extensão *.h5* (*DC_create_hdf5_files*) para serem usados na classificação do DC; bem como a chamada dos *scripts* que, efectivamente, correm o DC (*DC_run_scripts*). Posteriormente efectua a leitura dos resultados, transformando-os de novo no formato original (*DC_read_hdf5_files*).

- *DC_get_directory* (selecionar a pasta onde trabalhar): a seleção da directoria onde gravar os resultados do DC está sujeita ao tipo de instalação em que está a correr o algoritmo. As opções foram dadas por um *switch*, demonstrando os casos de *"laptop"* e

"local". Neste caso específico o caso foi *"local"*. Foram criadas pastas para guardar a DATA em *data.h5*, as corridas do DC em *data_run.h5* e os resultados de cada corrida do DC em *results.#.h5*. Foram também criadas pastas para ficheiros *.txt*, que guardam os parâmetros do algoritmo, os centróides, representando cada agrupamento, e o grau de pertença de cada vóxel aos vários tecidos.

-*DC_create_hdf5_files* (criação dos ficheiros de informação): a necessidade de compressão e uso rápido dos dados levou à criação desta função. Esta usa o *data_info*, *dc_runs* e *directory_to_use*, para guardar os dados do DC num ficheiro *hdf5*. Este formato permite guardar quantidades complexas de informação, independentemente do seu tamanho, sendo de rápida leitura. O uso do *data_info* teve como objectivo associar os resultados à estructura em uso. Ao guardar as corridas do DC neste formato foi possível ter toda a informação, que estas corridas contêm, discriminada, desde parâmetros de treino até aos centróides inicialmente considerados, bem como ao grau de pertença. Visto que estes ficheiros foram posteriormente usados como input dos *scripts* de DC, é necessário que façam uso da estructura inicial DATA, bem como dos resultados obtidos previamente pelo SOM, i.e. dos super-vóxeis. Para isto acontecer, foi necessário voltar a chamar as funções *load_data* e *load_super*.

- *run_DC_scripts* (correr o DC): os *scripts* que, de facto, correm o algoritmo DC, e que se encontram escritos em Python. Como o esqueleto deste trabalho foi desenvolvido em Matlab, tornou-se necessário construir uma função que fizesse a chamada dos *scripts* de Python dentro do ambiente Matlab.

Por outro lado, dado que a implementação inicial deste algoritmo para segmentação foi pensada para Linux, certos comandos usados apresentavam uma construção diferente daquela que foi necessária para tornar este algoritmo utilizável em Windows. Assim, foi necessário alterar o comando de chamada dos ficheiros de Python para o Matlab, consoante o directório em que tinha sido instalado o executável do Python 2.7.

Deparou-se com alguns problemas de instalação, **Anexo 9.2**, tendo sido necessário alteração da codificação de caracteres no Python, **Figura 5.5.**

```
155 ]
156
157 class PathNotFoundException(Exception): pass
158
159 def get_path(folderid, user_handle=UserHandle.common):
160     fid = GUID(folderid)
161     pPath = ctypes.c_wchar_p()
162     S_OK = 0
163     if _SHGetKnownFolderPath(ctypes.byref(fid), 0, user_handle, ctypes.byref(pPath)) != S_OK:
164         raise PathNotFoundException()
165     path = pPath.value
166     _CoTaskMemFree(pPath)
167     # convert to unicode
168     reload(sys)
169     sys.setdefaultencoding('utf-8')
170     #if not codeo:
171         # codeo="utf-8"
172     #if hasattr(path, "decode"):
173         # path = path.decode(codeo)
174     return path
175
176
```

Figura 5.5 - Correção da codificação de base; implementação da codificação **uint8** para chamada sem problemas dos *scripts* de DC.

Existem no total quatro *scripts* de Python necessários para que o algoritmo DC possa ser executado: *DC_script*, *rdc2*, *nonlinear_optimization*, *Simplestat*.

1. *DC_script*: é o *script* principal. Importa o algoritmo propriamente dito, do *rdc2*, com as respectivas optimizações. Cria as pastas de treino, de centros iniciais, parâmetros, e pertença necessário para caracterizar cada corrida de DC. Este ficheiro usa como *input* os ficheiros *hfd5*, criados com a função *DC_create_hdf5_files* no Matlab,e a gravação destas pastas é feita dentro de uma estructura de 1 por 144 (por serem 144 corridas de DC), chamada *DC_data*.

2. *rdc2*: é o *script* com o DC desenvolvido. Tem em conta os parâmetros das corridas de DC e o número de centros que existem e, no caso de não encontrar nenhum centro, procede à escolha de novos centros. Foi neste ficheiro que se deu o problema referido anteriormente, um erro de compatibilidade entre código feito para Linux e código feito para Windows.

Com a utilização dos *breakpoints* foi possível encontrar o local exacto do erro. Este deveu-se ao facto de a definição de alguns parâmetros ter sido inserida numa condição *if/else*. Ao ter sido feito esta definição sujeita a uma condição, várias linhas abaixo, quando um dos parâmetros, *self.dim*, era chamado, a janela de comandos emitia um erro de compilação o que levou a que certos parâmetros tivessem de ser alterados, mantendo-se apenas a escolha dos novos centros, referidos acima, no *else* da preposição condicional.

3. *nonlinear_optimization*: faz parte do algoritmo DC, e é usado para a optimização dos resultados a nível de diferenciação de classes e fronteiras, ou seja, zonas de transição entre diferentes classes.

4. *Simplestat*: o objectivo desta função é o cálculo de algumas estatísticas, em relação ao DC, que posteriormente se fizeram. Estas estatísticas incluem com Médias, Variâncias, Covariâncias, Correlações, Normalizações e Distâncias Euclidianas, usadas para auxiliar o DC com a atribuição dos pixéis/vóxeis não classificados pelo SOM e que vão entrar nos clusters antes formados.

- *DC_read_hfd5_files*: Esta função foi desenvolvida para ler os ficheiro hfd5. Á semelhança da função *DC_create_hfd5_files*, os inputs voltam a ser *data_info, dc_runs* e *directory_to_use*. Mais uma vez foi feito o carregamento dos dados da estructura e dos super-vóxeis. Dado que os ficheiros *results.#.h5* provêm das corridas do DC, existe, o mesmo número de ficheiro deste tipo que corridas, 144 portanto.

De seguida, são analisados os resultados de todas as corridas de forma a verificar se não existe alguma que apresente valores discrepantes do resto das corridas.

Assim, é feita a leitura dos resultados e a sua correção, no caso da existência de valores discrepantes nos resultados do DC, o que possibilita a passagem ao passo seguinte do algoritmo sem que exista algum valor fora do comum em relação aos resultados das restantes corridas. Como visto na **Figura 5.6.** tanto os ficheiros de input do DC como os resultados das corridas feitas ficam armazenados no directório escolhido para serem lidos.

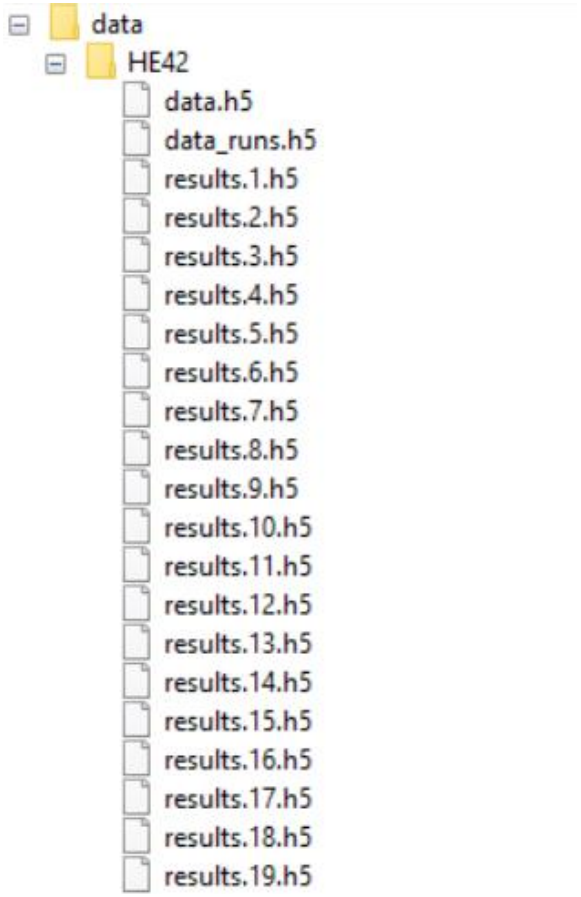


Figura 5.6 - Exemplo da gravação de DATA, corridas e resultados de DC no formato *hfd5*. Neste caso usando a estructura *DATA_HE42.mat*.

ii) *DC_clean_voxstats*: tem como entrada *dc_runs* e *DC_data*, de forma a analisar quais as corridas que devem ser rejeitadas. Recorre ao parâmetro *voxel_stats*, não usando aqueles em que os tecidos não incluem super-vóxeis, e removendo corridas em que haja mistura de super-vóxeis no mesmo *cluster*. Procura a existência, em primeiro lugar, de um campo, nas corridas de DC, que seja *voxel_stats* e as corridas que apresentem, nesse campo, *"common_with_correct"*. Este campo não será preenchido se não existir *ground-truth*, ou se este não for concordante com a imagem em estudo. No caso de não existir nada, de acordo com esta condição, a corrida é marcada como retirada.

A avaliação das corridas quanto à sua possível utilização, para a obtenção de resultados do DC, teve também em consideração o grau de pertença. Foi então definido que se, em corridas consecutivas, o grau de pertença da segunda fosse superior a 50% do grau de pertença da primeira, a corrida seria rejeitada, uma vez que é demonstrada a fraca consistência das atribuições feitas pelo DC. Por último a *DC_data* seria rejeitada se não fossem encontradas todas as classes que deveriam existir em determinada imagem.

Após a execução de todas as funções apresentadas, a função *do_DC* grava numa estructura chamada, por exemplo, *DC_HE42* (no geral, DC_+*data_info*), no caso da estructura original; as corridas de DC (*dc_runs*); e a estructura com toda a informação resultante do DC (*DC_data*).

Este último processo marca o fim do algoritmo DC, indicando com um *print* o seu término.

A função *do_DC*, quando chamada no esqueleto do algoritmo, *script_segmentation*, foi colocada dentro de um *tic/toc*. A sua utilização permitiu medir o tempo de corrida do algoritmo DC, que foi de cerca de **2 horas** pelas 144 corridas executadas.

Para a obtenção de resultados, foi definida a função *load_data_DC*, para carregamento das estructuras representativas dos resultados das corridas de DC. Esta função foi inserida noutra, *DC_get_results*, com o propósito de, não só fornecer os resultados**,** mas também de encontrar a melhor corrida das 144 obtidas. O que procura são campos com a informação e com as classes presentes nela, verificando o erro em cada corrida. Este processo procura a corrida que melhor represente a atribuição de pixéis/vóxeis às classes atribuídas (*DC_find_best_run*).

Obtidos os resultados do DC, o passo seguinte foi proceder à obtenção das imagens representativas da segmentação resultante do DC, usando as funções para o efeito.

5.2.6 Real Data

Este passo do algoritmo de segmentação baseou-se na criação das imagens que representassem o resultado da segmentação feita pelo DC, e que pudessem ser usadas para avaliar a qualidades dos resultados. Para isso, recorreu-se à função *DC_create_images*, que incorporou a função *DC_prediction*, importante para a previsão da atribuição de pixéis aos tecidos em causa.

DC_prediction tem como objectivo criar as previsões de tecidos, a contagem do número de tecidos e a variância entre os pixéis/vóxeis que os formam. Utiliza um *threshold* para ajudar na discriminação dos vários tecidos, e torna a previsão mais rigorosa. É tida também em conta a corrida em questão do DC, dado que, no decorrer do processo de segmentação, teve de ser implementada uma segunda corrida, sobre as imagens originais sobrepostas com uma máscara, para melhorar os resultados obtidos, ao

suprimir a maioria dos tecidos que não são MS para que o DC possa ser mais selectivo sobre os dados que usa, ao ter maioritariamente MS.

A previsão de tecidos foi feita em três níveis. Comparando o grau de pertença de um certo pixel/vóxel com a metade do grau de pertença verificado nos pixéis de cada classe, dado que já é um valor suficientemente alto para considerar a existência de um determinado tipo de tecido dentro de um pixel/vóxel, foi possível prever se de facto o pixel pertencia a determinada classe. O facto de este processo ter sido realizado três vezes, em que a cada vez o estudo da pertença é mais rigoroso, o que leva a que a cada volta, o foco seja em pixéis/vóxeis cada vez com maior especificidade no que toca ao número de classes presentes, eleva o grau de especificidade na atribuição de pixéis a classes, e diminui a probabilidade de estes sofrerem erros de atribuição por, na realidade, poderem pertencer mais a um tipo de tecido mais do que àquele com que aparentam ter mais semelhança.

Feito isto, fez-se o chamamento da função *DC_prediction* dentro da função *DC_create_images* para que as imagens criadas e representativas da segmentação feita tivessem em conta as previsões das pertenças dos pixéis/vóxeis às classes propostas.

De forma a manter o rigor na criação das imagens, recorreu-se a um coeficiente de confiança para, mais uma vez, garantir um certo nível de rigor na formação das imagens. Este coeficiente foi definido como 0.8 (80%) e serviu como medida de consistência entre os resultados obtidos para as várias corridas. Quando mais alto o coeficiente, mais estrito é o critério de validação das soluções propostas. Ao invés, quanto menor o valor, mais permissivo é o critério, agrupando mais facilmente solução que não são exactamente as mesmas, o que se traduz numa perda de especificidade.

O último passo passou pela normalização das imagens, para que cada tecido fosse claramente representado como um todo, com a mesma intensidade, e pela definição da representação em quatro colunas e oito linhas (número de linhas representativas das alturas definidas em *DATA.z*). Cada coluna representou, respectivamente, CSF, GM, WM e MS.

Como, após a obtenção de resultados, foi necessário avaliar a sua qualidade, estatisticamente, e a possível evolução das zonas lesadas. Para isto, procedeu-se ao cálculo das estatísticas relativas aos resultados obtidos da classificação do DC.

5.2.7 Simulated Data

O principal objectivo deste passo foi verificar a eficácia do algoritmo, por comparação entre os resultados obtidos e o *ground truth* fornecido na estructura inicial.

Baseia-se em duas componentes: a primeira, a função *DC_analyse_voxstats* e a segunda, *get_error_results*.

- *DC_analyse_voxstats*: o conteúdo desta função é usado como um passo para a comparação dos resultados obtidos do algoritmo de segmentação. Permite a avaliação das estatísticas dos pixéis/vóxeis em relação a **Verdadeiros Positivos** (**TP**, pixéis/vóxeis correctamente incluídos nas classes), **Falsos Positivos** (**FP**, vóxeis incorrectamente incluídos em classes), **Verdadeiros Negativos** (**TN**, pixéis/vóxeis correctamente excluídos das classes) e **Falsos Negativos** (**FN**, pixéis/vóxeis incorrectamente excluídos de classes).

- *get_error_results*: esta função foi planeada para devolver as estatísticas referentes ao processo de segmentação. Houve que ter em conta que durante este processo, existiram pixéis/vóxeis que puderam não ser correctamente atribuídos, ou que tenham sido excluídos da região de interesse. No entanto o cálculo destas estatísticas só foi possível com a existência de um *ground truth*. Este, permite comparar a classificação feita pelo algoritmo usado com os resultados obtidos por classificadores especialistas, que estabelecem a fasquia.

Foram usados vários métodos para avaliar a qualidade dos resultados da utilização do algoritmo com base em **TP**, **TN**, **FP** e **FN**:

i) Dice Coefficient e *Jaccard Coefficient*: quando se pôde recorrer a *ground truth*, fez-se uso destes dois coeficientes que usam distâncias em segmentação de tecidos para inquirir sobre a percentagem de semelhança entre a segmentação proposta e o *ground truth*. As **Equação 5.1.** e **Equação 5.2.** mostram a forma de calcular o coeficiente de *Dice* e o coeficiente de *Jaccard*, recorrendo aos valores obtidos para os pixéis/vóxeis classificados como TP, TN, FP ou FN. Estas medidas avaliam a *perfomance* do algoritmo entre 0 e 1, sendo 0 uma classificação mal efectuada e 1, uma classificação perfeita.

Apesar de muito semelhantes, a diferença entre os dois coeficientes prende-se com o facto de o coeficiente de *Dice* não ter em consideração a desigualdade de triângulos, consequência do Teorema de Pitágoras, ao contrário do coeficiente de *Jaccard*. No

entanto, para o estudo em causa, o cálculo de ambos os coeficientes prende-se com uma melhor confirmação estatística dos resultados.

$$\kappa_d = \frac{2TP}{2TP + FP + FN}$$

Equação 5.1 - *Dice Coefficient.*

$$\kappa_j = \frac{TP}{TP + FP + FN}$$

Equação 5.2 - Jaccard Coefficient.

ii) *Sensitivity*: caracteriza o número de pixéis/vóxeis que se encontram correctamente segmentados. Encontra-se representada na **Equação 5.3.**, onde é calculada a proporção, em relação ao total de TP e FN, de pixéis/vóxeis que pertencem realmente à região de interesse ou à classe em questão.

$$\eta_{svt} = \frac{TP}{TP + FN}$$

Equação 5.3 - Fórmula da *Sensitivity*.

iii) *Specificity*: acaba por ser o contrário da anterior, contendo a proporção, no total de pixéis/vóxeis, que são correctamente excluídos (total de TN e FP, representado na **Equação 5.4.**).

$$\eta_{spf} = \frac{TN}{TN + FP}$$

Equação 5.4 - Fórmula da *Specificity*.

iv) *Sensibility*: Difere da *Specificity*, na medida em que procura não incluir os TN. A inclusão dos TN na anterior levanta problemas de dimensões, ou seja, os pixéis/vóxeis que são excluídos dependem muito da distância dos mesmos às regiões de interesse. Quanto maior for a dimensão da imagem, mior será o número de TN. Assim, *Sensibility*

é uma boa alternativa. A **Equação 5.5.** demonstra o cálculo feito em alternativa à *Specificity*.

$$\eta_{sbl} = 1 - \frac{FP}{TP + FN}$$

Equação 5.5 - Fórmula da *Sensibility*.

v) *Conformity*: é uma medida mais rigorosa e sensível que os coeficientes de Dice e Jaccard, permitindo uma maior discriminação de pequenas variações em imagens segmentadas. A **Equação 5.6.** representa a formulação desta medida. O uso da *Conformity* é restringido à existência de pixéis/vóxeis correctamente classificados (*TP>0*).

$$\kappa_c = 1 - \frac{FP + FN}{TP}, \text{ if } TP > 0$$

Equação 5.6 - Fórmula da Conformity.

Tanto a *Conformity* como a *Sensibility* podem tomar valores negativos, principalmente quando a segmentação e o *ground truth* não apresentam qualquer tipo de sobreposição. Neste caso, a *Conformity* toma o valor de menos infinito. A *Sensibility* apresenta-se negativa para regiões de interesse superiores às usadas como referência.

Em circunstâncias, como a desta dissertação, em que a classificação não é do tipo *hard classification* (sendo *soft classification*) recorreu-se também a erros *rms* para *hard classification* e *soft classification*, que tiveram em consideração a probabilidade estimada, $\hat{P}_c$, e verdadeira do pixel/vóxel pertencer a determinada classe, P_c, **Equação 5.7.**

$$rms = \sqrt{\frac{1}{N} \sum_{\boldsymbol{c}} (\hat{P}_{\boldsymbol{c}}\boldsymbol{x}(n) - P_{\boldsymbol{c}}\boldsymbol{x}(n))^2}$$

Equação 5.7 - Fórmula do *rms*, com as respectivas componentes de *Probabilidade estimada*, $\hat{P}_c$, e *Probabilidade Verdadeira*, P_c.

É de notar que o cálculo dos coeficientes anteriores esteve dependente da existência de *ground-truth*. No caso de não existir ou de não ser da melhor qualidade, os valores obtidos nestes coeficientes tornam-se desenquadrados com o estudo efectuado.

Numa tentativa de melhorar os resultados e, depois de obter uma primeira avaliação do algoritmo, procedeu-se a uma segunda corrida de DC sendo que, para isso, foi necessário mascarar os dados obtidos na primeira corrida.

5.3 Segunda Corrida do DC

Com o objectivo de melhorar a qualidade da segmentação obtida na primeira corrida do algoritmo, recorreu-se a uma segunda corrida. No entanto, esta não pôde ser feita sobre as imagens originais, mas sim sobre o resultado da segmentação prévia. Para isso foi necessário criar uma máscara com o intuito de sobrepor às imagens originais.

Ao aplicar a máscara, a extensão dos tecidos do cérebro é diferente da inicial e mais pequena, sendo MS predominante.

5.3.1 Preparação da Segunda Corrida

5.3.1.1 Crop image matrix to obtain lesion column only

Dado que o principal objectivo desta dissertação foi a identificação e previsão de lesões, na matriz da imagem recolhida, foi retirada a coluna correspondente apenas à lesão, e criou-se uma máscara da imagem original, contendo apenas os pixéis/vóxeis correspondentes às propostas de lesão feitas pelo DC. O corte foi feito por uma estimativa da largura de cada coluna da imagem apresentada após o DC, correspondente a cada tecido, tendo em conta uma imagem de **2048 por 1024**. A divisão da largura da matriz *img_DC* (representativa das imagens segmentadas vindas do DC) foi feita tendo em conta quatro classes (WM, GM, CSF e MS ou lesão). Assim, a largura da coluna de MS foi considerada como um quarto da largura total da matriz obtida de *img_DC*, sendo desta forma, **256**.

5.3.1.2 Crop lesion column to obtain lesion images for each sequence and mask dilation

Visando uma maior facilidade na aplicação da máscara, produzida por binarização, às imagens originais das três sequências, foram criadas três máscaras idênticas, cada uma visando ser integrada em cada sequência. Apesar de, aparentemente redundante, a criação de três máscaras deveu-se a uma conformidade de dimensões das matrizes, como mostra a **Figura 5.7.** onde se pode observar três colunas, cada uma com a mesma máscara, dilatada a partir de uma **estructura elíptica com raio de um pixel**, que visualmente, se enquadrou melhor com as formas das lesões, com raio de um pixel. A diferença entre este passo e o anterior, prende-se com, não só a dilatação da máscara, mas também com a obtenção de dimensões que fossem coincidentes com o corte do volume original. Assim, e visando uma concordância de dimensões das matrizes a serem multiplicadas, garantiu-se que não existiriam problemas derivados de diferenças de dimensões.

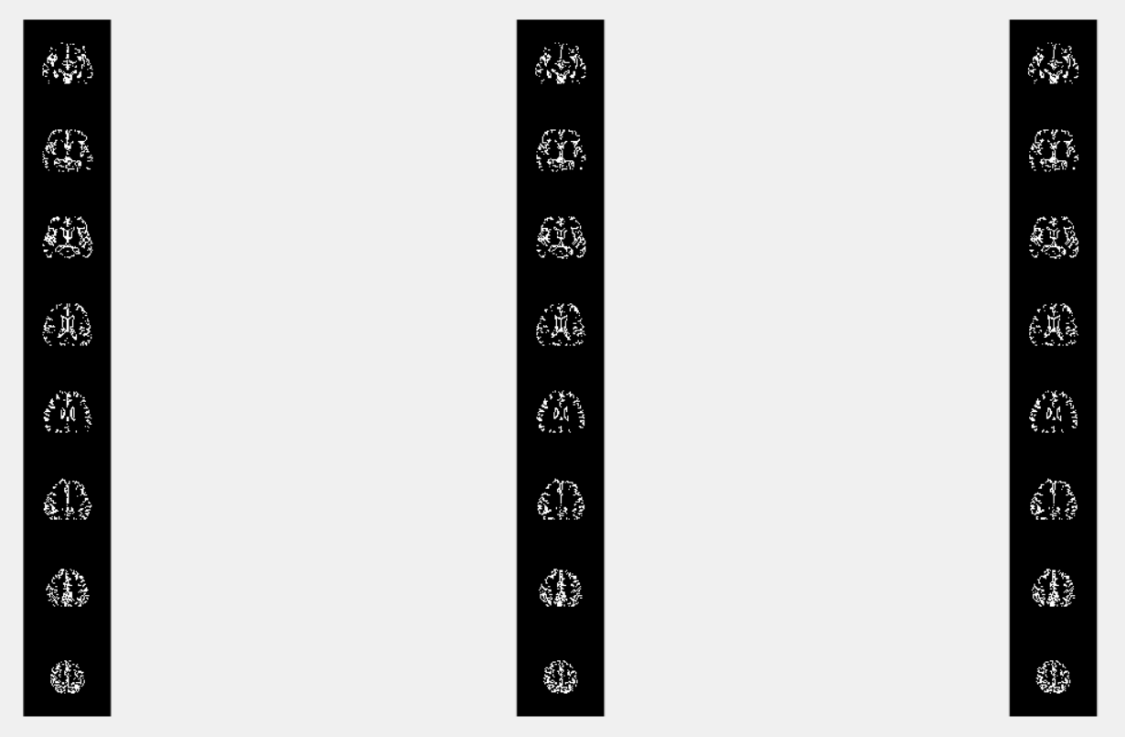


Figura 5.7 - Máscara feita para o número de sequências usado. As máscaras foram iguais para que as dimensões da matriz coincidissem com as dimensões da matriz da imagem.

Uma vez que apenas algumas alturas de imagem contêm informação relevante à caracterização da lesão, presente no estudo, somente essas serão consideradas para processamento. Essa informação encontra-se no campo ***z*** da estrutura *DATA*. Na **Figura**

5.8. estão representadas, para cada altura de *DATA.z*, as fatias representativas de cada sequência utilizada.

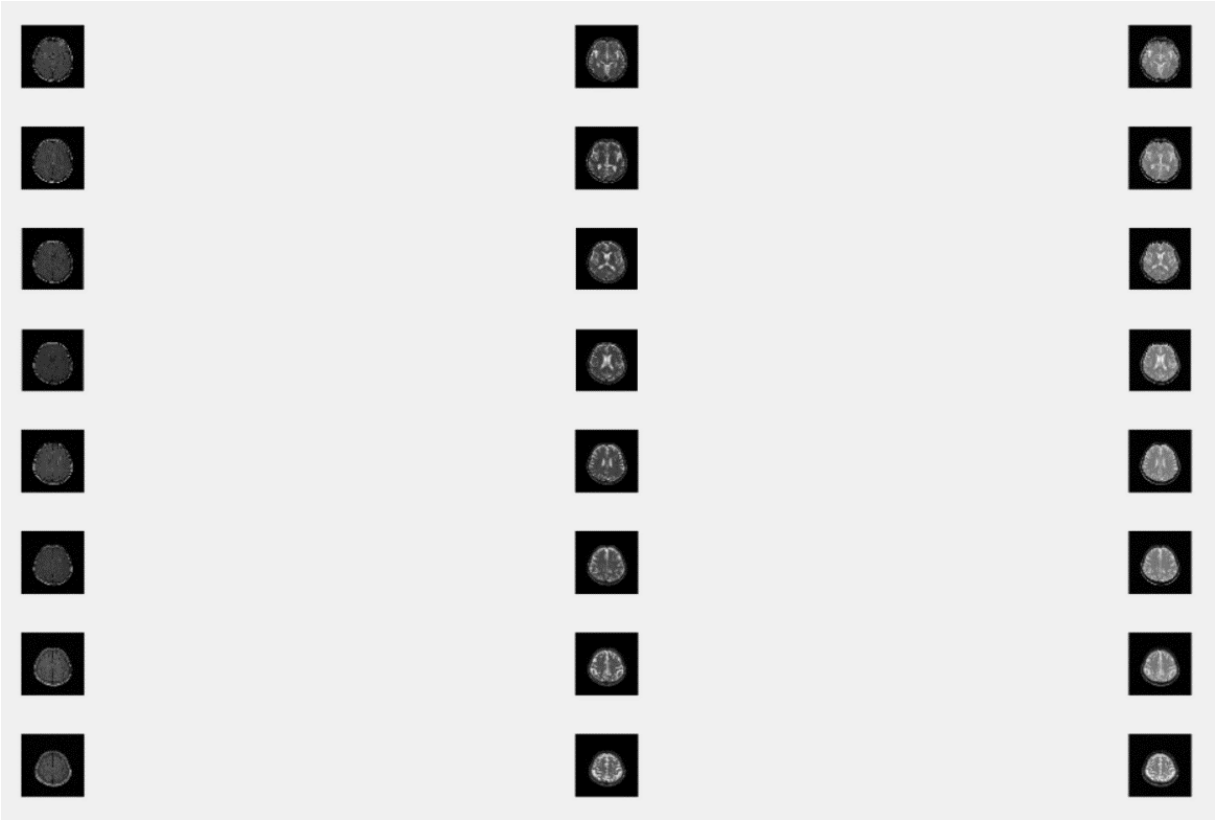


Figura 5.8 - Alturas em *DATA.z*, para cada sequência usada, correspondentes ao volume original e nas quais será sobreposta a máscara criada.

O uso das alturas de *DATA.z* deu origem a uma matriz de dimensões iguais à da matriz da máscara, o que de um ponto de vista de sobreposição das duas imagens, revela ser mais simples, pois, assim, foi possível a multiplicação ponto a ponto das matrizes, mantendo as dimensões, tal como representa a **Figura 5.9.**

Figura 5.9 - Aplicação da máscara às imagens com alturas de DATA.z, para cada sequência.

Como último passo para a criação das novas imagens, a serem usadas na segunda corrida do algoritmo, o novo volume foi gravado onde antes se verificavam as imagens originais depois de criada uma cópia da estructura inicial. De seguida, houve necessidade de guardar o volume em vectores, de forma a existir conformidade com o formato lido pelo DC. Estes vectores foram guardados em *DATA.dataMasked*, campo da estructura *DATA* que é carregado no início do DC.

Todas as estructuras criadas durante a corrida do algoritmo foram duplicadas e alteradas, para que não existisse *overwrite* das estructuras existentes, acrescentando o prefixo *"_2nd"* nas designações das próprias estructuras.

A **Figura 5.10.** ilustra as alterações feitas na estructura *DATA* no que toca ao volume e ao campo *dataMasked*.

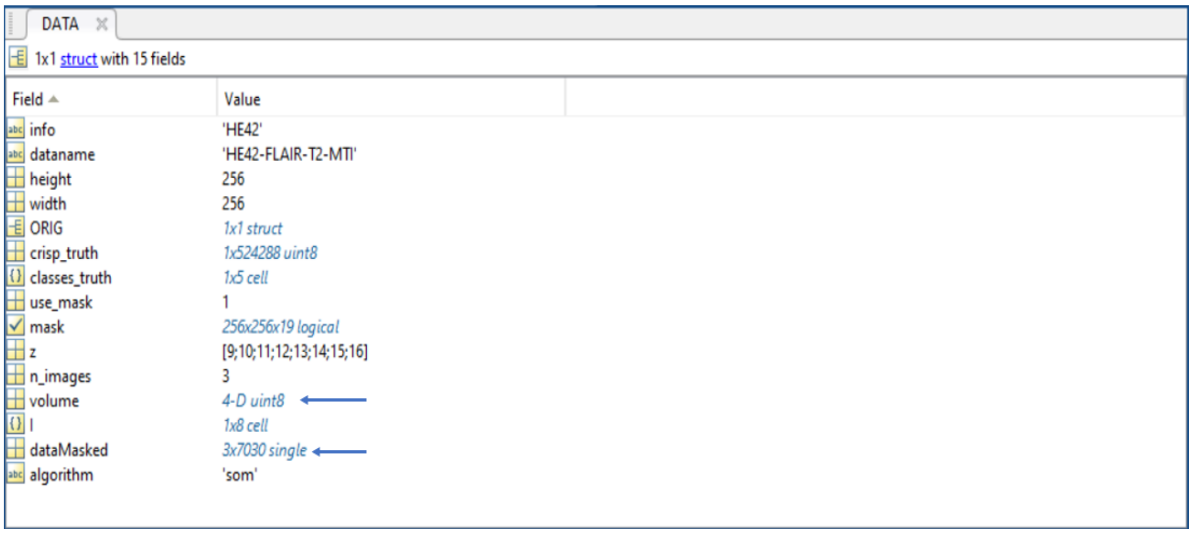


Figura 5.10 - Conteúdo da nova estructura *DATA*. As alterações indicadas pelas setas representam a codificação feita ao volume, para coincidir com a do volume inicial. O novo *dataMasked*, serve de entrada para o DC e representa as novas imagens mascaradas em três vectores de intensidades normalizadas.

5.3.2 Second DC run

Existindo a opção de, não só, correr o DC pela segunda vez, mas também o SOM, optou-se por não efectuar o SOM, por não existir uma mudança significativa dos *clusters* apresentados em relação à primeira corrida o que não resultaria numa melhoria significativa da segmentação feita pela segunda corrida do DC. Isto deveu-se ao facto de tal não trazer benefícios relevantes ao resultado final, por o agrupamento de pixéis/vóxeis em super-vóxeis já ter sido proposto, na altura da segunda corrida, na primeira corrida.

5.4 Interface

Com o objectivo de o algoritmo ser fácil de usar e de que a necessidade de recorrer ao código fosse o mais pequena possível, criou-se uma interface de fácil acesso e relativamente simples, desenhada para executar todos os comandos do algoritmo, passo a passo, e que permitiu ao utilizador guardar, em cada momento, as imagens resultantes dos comandos, bem como as variáveis criadas e usadas no decorrer do processo. Desta forma, visou-se o acompanhamento progressivo de um utilizador que, de certa forma, não estivesse a par do funcionamento do algoritmo em si, mas que pudesse utilizá-lo com o mínimo de limitações possível.

5.4.1 Interface Conjunta

Dada a complementaridade entre o trabalho desta dissertação e o trabalho desenvolvido pelo meu colega Afonso Moreira, na componente de pré-processamento anteriormente referida, foi construída, por acordo de ambos, uma interface intermediária, com o objectivo de permitir ao utilizador a escolha entre Pré-Processamento e Classificação, conforme a qualidade e tratamento das imagens que pretenda testar. A **Figura 5.11.** ilustra a interface construída. Apresenta uma mensagem de explicação da janela em questão, **Figura 5.11.a)**, com um botão de "Preprocess" para a aplicação dos algoritmos desenvolvidos pelo Afonso Moreira, **Figura 5.11.b)** e um botão *"Classification"*, **Figura 5.11.c)**, para aplicação dos processos de segmentação, estudados nesta dissertação. Por último, incluiu-se um botão *"Exit"* para possibilitar ao utilizador fechar o programa, **Figura 5.11.d)**.

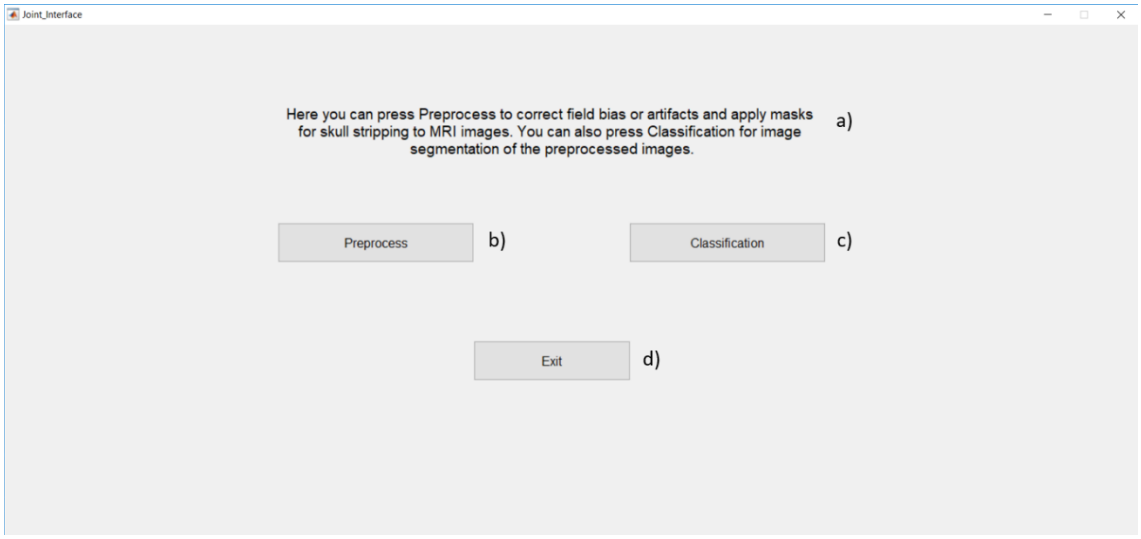


Figura 5.11 - Representação da interface que possibilita a ponte entre o trabalho desenvolvido nesta dissertação e o trabalho desenvolvido na dissertação do Afonso Moreira. **a)** Mensagem informativa do propósito da janela em questão. **b)** Botão "Preprocess" que permite a execução do pré-processamento das imagens de MRI. **c)** Botão "Classification" que permite proceder à segmentação dos tecidos nas imagens pré-processadas. **d)** Botão "Exit" que permite ao utilizador fechar a aplicação.

5.4.2 Mensagem Inicial

Esta primeira interface, *Inicial_Message*, funcionou como uma pequena introdução ao algoritmo e, de certa forma, como uma mensagem de boas vindas ao utilizador. Foi projectada para apresentar apenas uma pequena descrição do objectivo do algoritmo de segmentação e informar o utilizador da possibilidade de estudar cada passo feito

individualmente. Foi também embutido um botão de avanço (*"Continue"*) que permitiu a passagem para a janela principal, *Segmentation_Interface*, onde se implementou a maior parte do algoritmo. A **Figura 5.12.** mostra a apresentação desta primeira janela de apresentação.

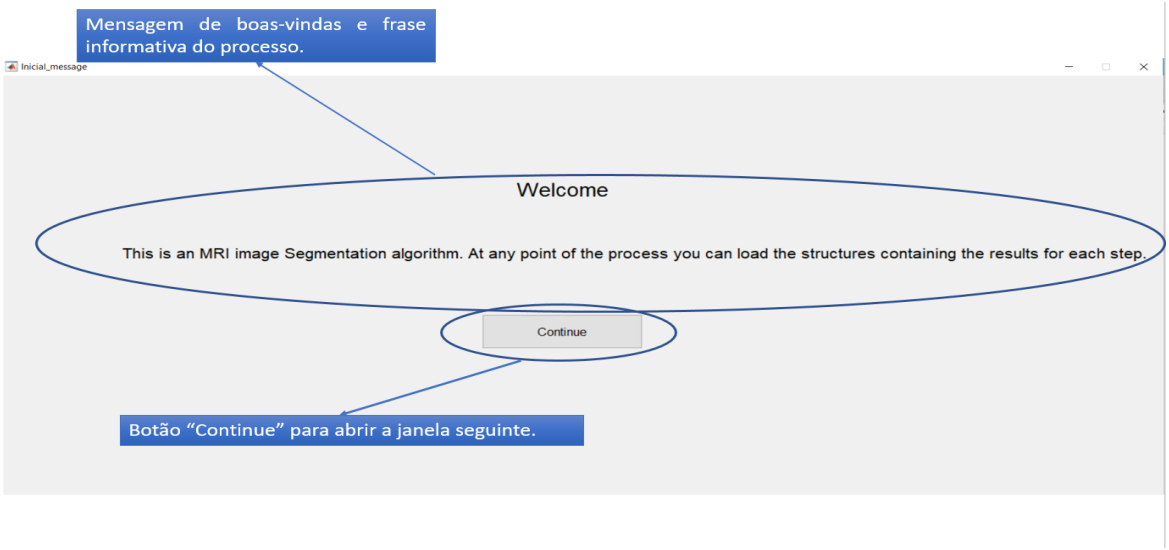


Figura 5.12 - Janela inicial, com mensagem de boas vindas e um botão que permite avançar para a janela seguinte.

5.4.3 Interface de Segmentação

Segmentation_interface foi a janela onde se colocaram todos os passos do algoritmo. Foi onde se apresentaram todas as imagens resultantes dos processos aplicados. Em primeiro lugar apresenta um *"Dropdown Menu"*, com o objectivo de permitir ao utilizador selecionar, através do *data_type*, a estrutura a usar, como mostra a **Figura 5.13.a)**. De seguida, criou-se um botão de *"Load Previous"*, **Figura 5.13.b)** com o objectivo de permitir ao utilizador carregar todas as estructuras necessárias para realizar cada passo do processamento, e não seja necessário correr todo o algoritmo, que computacionalmente leva algum tempo a correr, para obter informação sobre, por exemplo, as imagens resultantes do DC. Foi criada outra janela para carregamento das estructuras de cada passo, explicada mais à frente. Para carregar a estructura inicial e visualizar as imagens originais, programou-se um botão *"Load Data"*, **Figura 5.13.c)**. Pressionando este botão, dá-se inicio à função *load_data*, que guarda a estrutura toda numa variável. De seguida incluiu-se um botão *"Data Preprocess"*, ilustrado na **Figura 5.13.d)**, que executa o *smoothing* das imagens originais, chamando a função *preprocess*.

A utilização deste botão leva ao aparecimento das imagens filtradas. O botão seguinte, *"SOM"* foi criado para correr o algoritmo SOM, executando as corridas deste algoritmo, e levando à criação dos clusters consistentes, **Figura 5.13.e)**. O uso deste botão permite o uso de outro botão, *"Super Voxel Selection"*, que chama a interface de seleção dos super-vóxeis, explicada no capítulo *"5.2.4. Super Voxel Selection"*, ver **Figura 5.13.f)**. Ao serem selecionados os super-vóxeis é possível proceder à corrida do DC, utilizando para tal, o botão com o mesmo nome. Este chama todas as funções que dizem respeito à chamada do DC, ou seja, efectua a conversão para ficheiros *hfd5* e corre os *scripts* de python que criam a *DC_data*. Este botão encontra-se representado na **Figura 5.13.g)**.

Num óptica mais direccionada para a obtenção de resultados de segmentação, foram propostos os botões *"Get Segmentation Images"* e *"Get Statistics"*, **Figura 5.13.h)** e **Figura 5.13.i).** Correspondem à chamada dos resultados do DC, às suas conversões em imagens observáveis, e ao cálculo das estatísticas (no caso de existir *ground-truth*). Produz ainda a imagem com os pixéis/vóxeis atribuídos a cada classe, bem como aqueles em que existe sobreposição de tecidos.

Mais ligados à preparação das imagens para a segunda corrida do algoritmo estão os botões *"Binarized Lesion Column"*, *"Dilated Mask"* e *"Apply Mask"*.

O primeiro, *"Binarized Lesion Column"*, visível na **Figura 5.13.j)**, foi criado com o intuito de isolar a coluna da lesão, e efectuar o primeiro passo de criação da máscara em torno da lesão.

O segundo, *"Dilated Mask"* ilustrado na **Figura 5.13.k)**, foi projectado para proceder à dilatação da máscara obtida, no passo anterior, por binarização.

Por último, *"Apply Mask"* representado na **Figura 5.13.l)**, foi implementado para aplicar a máscara obtida nas imagens originais, levando à criação de novas imagens, para a segunda corrida do algoritmo.

De forma a permitir ao utilizador prosseguir para a segunda corrida de DC, e respectiva obtenção de imagens e estatísticas, procedeu-se à criação de mais três botões com funções semelhantes aos da **g)**, **h)** e **i)**. Assim, temos os botões *"2nd DC"*, **Figura 5.13.m)**, *"Get Segmentation Images for 2nd"*, **Figura 5.13.n)** e *"Get Statistics for 2nd"*, **Figura 5.13.o)**.

Esta interface foi também incorporada com um botão *"Reset all"*, **Figura 5.13.p)**, que permite limpar todas as variáveis do *Workspace*, no caso de ter existido algum erro

de compilação, e um botão *"Clear all Figures"*, **Figura 5.13.q)** utilizado para limpar as duas zonas de imagem, **Figura 5.13.r)**. Foram apenas colocadas duas uma vez que não se pretende sobrelotar a interface e evitar a representação de muitas imagens criadas em simultâneo. Desta forma cada imagem nova, resultante de uma parte nova do processo, substitui a anterior. Para que não se perca nenhuma imagem com a actualização das voisualizações, foram criados dois botões para guardar cada imagem, **Figura 5.13.s)**, *"Save Figure 1"* e *"Save Figure 2"*, correspondendo à imagem da esquerda e da direita, respectivamente.

Além do botão *"Clear all Figures"* que limpa as duas imagens em simultâneo, cada imagem teve associado um botão específico *"Clear Figure"*, **Figura 5.13.t)**.

No caso de, o utilizador, pretender voltar atrás para repetir o algoritmo de Pré-Processamento, foi adicionado um botão *"Back"*, **Figura 5.13.u)** que, quando premido, levasse o utilizador de volta à interface *"Joint_Interface"*, que faz a ponte entre este trabalho e aquele desenvolvido pelo Afonso Moreira.

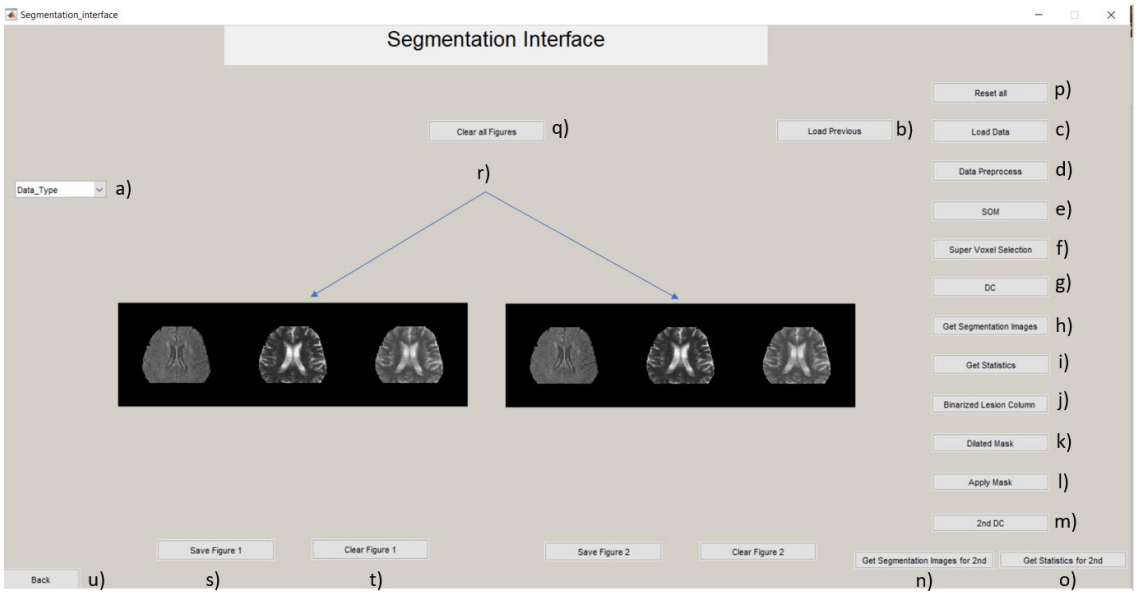


Figura 5.13 - Representação da interface de segmentação. **a)** escolha do *data_type* para seleção de estructura a usar. **b)** carregar estructuras necessárias a cada passo para evitar o tempo de corrida do algoritmo. **c)** carregar a estructura numa variável e apresentar imagens originais. **d)** efectuar filtragem das imagens e apresentar imagens resultantes. **e)** executar SOM. **f)** selecionar super-vóxeis. **g)** executar DC. **h)** obter resultados do DC e apresentar imagens. **i)** calcular estatísticas e apresentar imagem com *overlaps*. **j)** binarizar a coluna da lesão e apresentar resultado. **k)** dilatar a máscara binarizadaa e apresentar resultado. **l)** aplicar a máscara binarizada e dilatada às imagens iniciais das alturas de *DATA.z*. **m)** efectuar segunda corrida do DC. **n)** obter as imagens para a segunda corrida do DC. **o)** obter as estatísticas resultantes da segunda corrida **p)** botão *"Reset all"* para apagar todas variáveis e recomeçar. **q)** botão *"Clear all figures"* para limpara ambos os eixos em simultâneo. **r)** os dois *axes* usados. **s)** botões *"Save Figure"* para guardar as imagens em separado. **t)** botões *"Clear Figure"* referentes a cada eixo para apagar o conteúdo do respectivo eixo. **u)** botão *"Back"* que remete o utilizador para a interface *"Joint_Interface"*.

5.4.4 Carregamento dos dados

"Load_Previous" representa a interface que foi criada para possibilitar ao utilizador carregar todas as estructuras necessárias a cada passo que pretender executar, **Figura 5.14.**

Apresenta uma mensagem informativa do objectivo da janela em questão, aberta com a utilização do botão *"Load Previous"*, a partir da interface *"Segmentation Interface"*.

Foi construída com quatro botões, cada um possibilitando o carregamento das estructuras existentes:

- *Load Clusters Information*, **Figura 5.14.a)**: possibilita o carregamento da estrutura que contém os clusters obtidos por SOM;

- *Load SOM Information*, **Figura 5.14.b)**: carrega toda a informação relativa ao número de corridas de SOM;

- *Load Super Voxel Information*, **Figura 5.14.c)**: carrega a informação referente ao numero de super-vóxeis que pertencem a cada tecido;

- *Load DC Information*, **Figura 5.14.d)**: carrega informação referente às corridas de DC, bem como as saídas do DC.

- *Load 2nd DC Information*, **Figura 5.14.e)**: carrega a informação referente à corrida de DC, aplicada às imagens mascaradas.

Foi incluído também, nesta interface, um botão *"Back"*, **Figura 5.14.f)**, que permite ao utilizador voltar à janela de *"Segmentation_interface"* e, ao mesmo tempo, fechar a janela de *"Load_Previous"*.

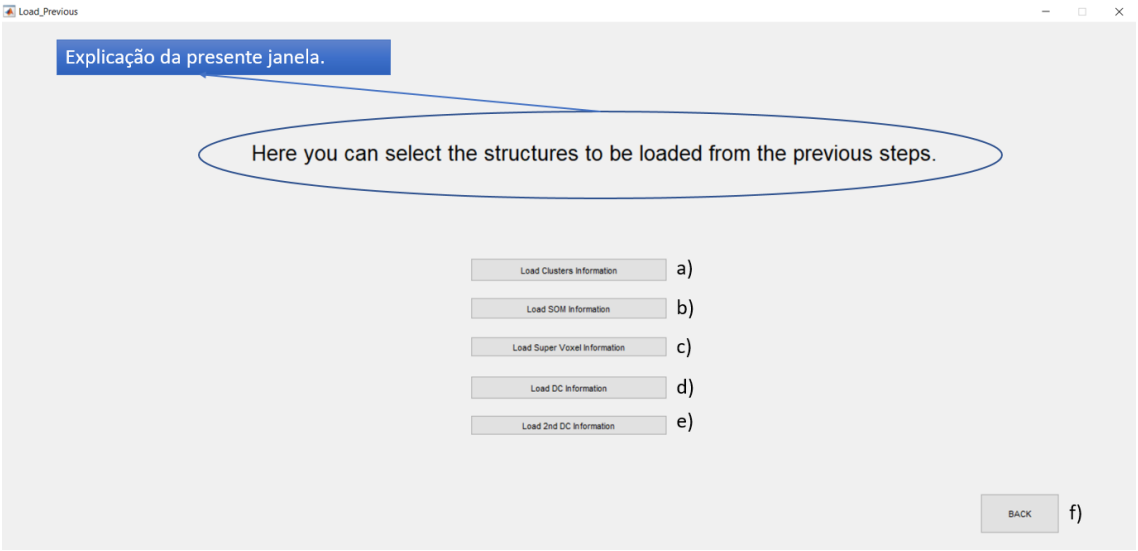


Figura 5.14 - Representação da janela *Load_Previous* que possibilita o carregamento de dados prévios necessários a cada passo. **a)** Carregar informação de clusters. **b)** Carregar informação sobre as corridas do SOM. **c)** Carregar os super-vóxeis formados. **d)** Carregar a informação sobre as corridas do DC. **e)** Carregar a informação referente à segunda corrida do DC sobre as imagens mascaradas. **f)** botão de saída da interface que remeta para a *Segmentation_Interface*.

O carregamento bem sucedido destas estructuras foi projectado para levar a uma nova janela, *"Load_Successful"*, apresentada na **Figura 5.15.** contendo uma mensagem a indicar esse sucesso e um botão de "OK" que remete de novo para a janela *"Load Previous"*, **Figura 5.15.a)**. Foi criada outra janela com o comportamento oposto, *"Load_Not_Successful"*, **Figura 5.16.**, que emite uma mensagem de carregamento mal sucedido e, mais uma vez um botão, **Figura5.16.a)**, que remete para a interface anterior.

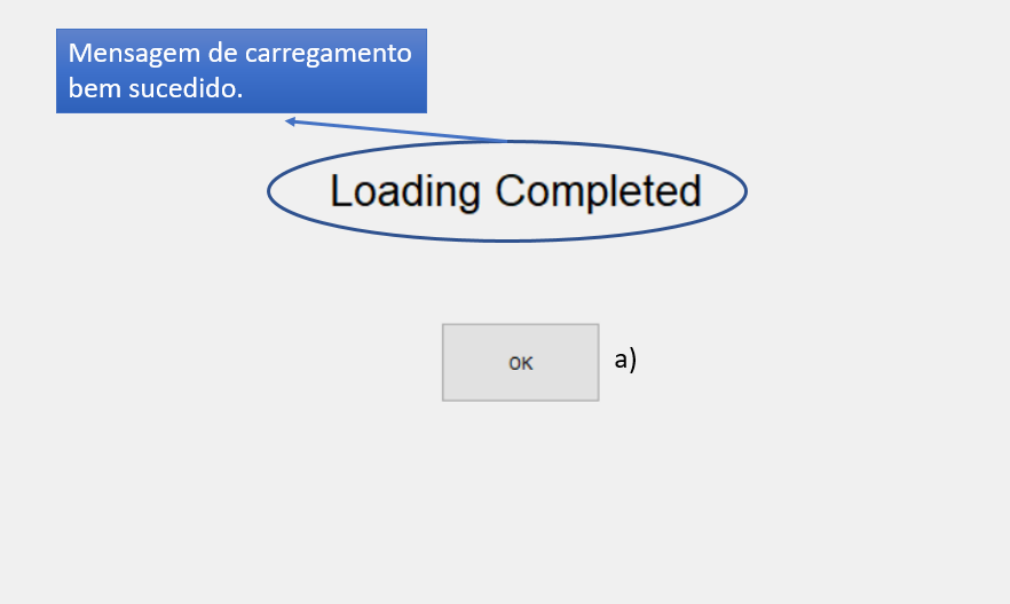


Figura 5.15 - Mensagem de carregamento bem sucedido. **a)** botão "OK" permitindo regressar à janela anterior.

Figura 5.16 - Mensagem de carregamento não efectuado. **a)** Botão que permite regressar à janela anterior.

5.4.5 Save Image Directory

Interface criada para aparecer após a utilização dos botões de "guardar imagem" na janela *"Segmentation_interface"*. Esta nova janela, apresentada na **Figura 5.17.** permite ao utilizador inserir a pasta onde quer que as imagens, de cada imagem apresentada nas caixas de diálogo anteriores sejam guardadas, **Figura 5.17.a)**, possuindo também um botão, **Figura 5.17.b)**, que devolve o utilizador à janela principal de segmentação de imagens. Foi desenhada uma para cada botão de "guardar imagem" referente à imagem da esquerda e da direita, *"Save_Image_Directory"* e *"Save_Image_Directory_2"*, respectivamente.

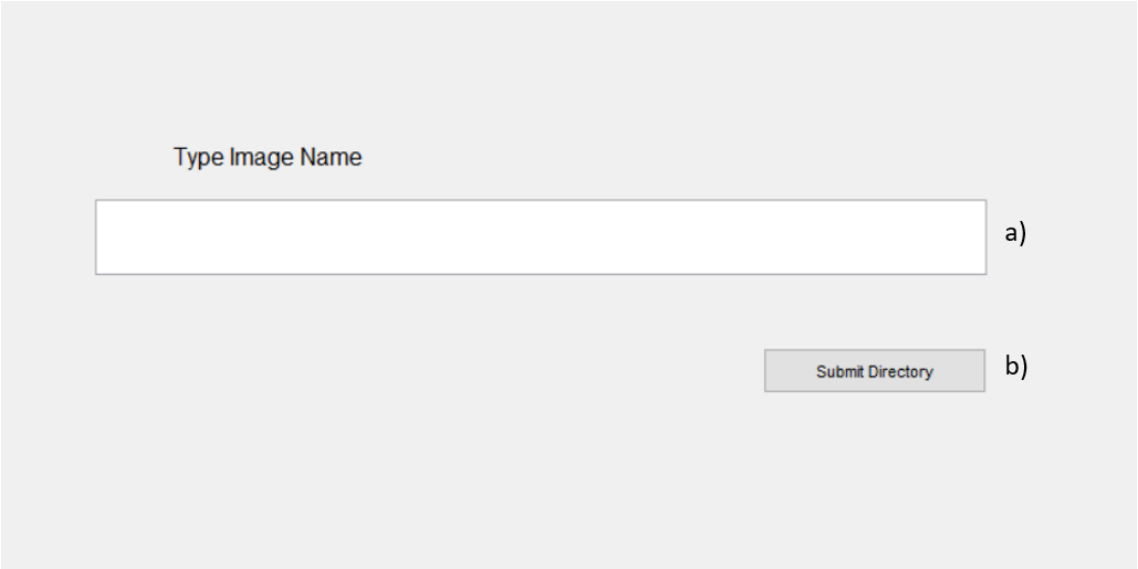


Figura 5.17 - Janela de escolha da pasta onde guardar a imagem. **a)** *edit box* para preenchimento da pasta de escolha pelo utilizador. **b)** Botão de submissão da pasta para gravar a imagem.

Pretendeu-se com este conjunto de interfaces, facilitar o uso a utilizador que não fossem familiarizados com o processo, tendo assim uma interface fácil de utilizar e que diminui bastante a necessidade de consultar o código.

A seguir, apresentam-se os resultados obtidos no processo de segmentação.

6 Resultados

A apresentação dos resultados obtidos será feita cobrindo os processos usados para obtenção dos resultados. Dado que o *output* de cada passo do algoritmo é uma imagem com base em *clusters*, ou segmentação ou obtenção e aplicação de máscaras binarizadas, apresentar-se-á, em sequência os resultados de cada estágio do processo.

6.1 DATA_HE42

6.1.1 Filtragem

Num primeiro passo, *Data Preprocess*, procedeu-se à aplicação de uma suavização da imagem do volume original. Na **Figura 6.1.** estão representadas as imagens correspondentes ao volume inicial, nas três sequências usadas, **Figura 6.1.a)**, e ao volume suavizado, **Figura 6.1.b)**. É de notar a diferença em zonas onde a variação de intensidades dos pixéis/vóxeis é maior, e nas zonas exteriores das imagens. Nessas regiões houve uma suavização dos bordos das imagens, e um equilíbrio maior nas intensidades. Dado que objectivo era apenas melhorar um pouco a imagem, e retirar algum ruído residual que ainda existisse, as alterações verificadas não foram muito significativas, uma vez que o volume inicial já apresentava uma boa qualidade em termos de percentagem de ruído e uniformidades dos bordos das imagens.

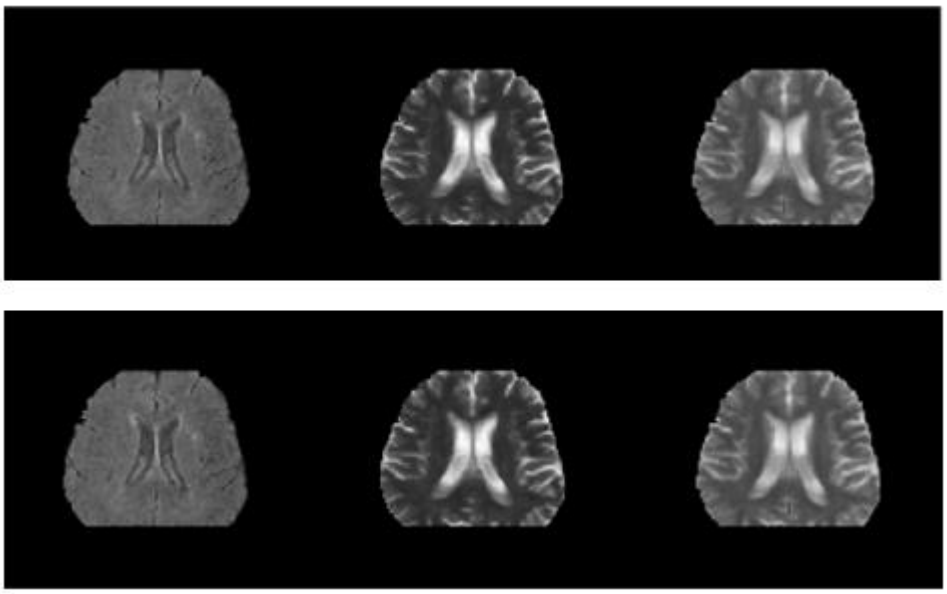


Figura 6.1 – **Em cima:** Representação do volume inicial nas três sequências. **Em baixo:** Representação das imagens filtradas. Note-se uma diferença na suavidade dos bordos e em zonas de maior variação de intensidades dos pixéis/vóxeis.

6.1.2 *Clustering* e SOM

A aplicação do SOM, para criação dos *clusters*, levou à escolha de um conjunto de super-vóxeis, baseada nas corridas feitas pelo SOM, e nos agrupamentos propostos. Na **Figura 6.2.** está representada a divisão dos pixéis pelos vários tecidos considerados, WM, GM, CSF e MS. Algo que foi transversal a todas as estructuras testadas, foi a fraca existência de protótipos para GM. Esta fraca identificação de protótipos levou a problemas na segmentação feita pelo DC que, mais à frente serão abordados e identificados. Na **Figura 6.2.a)** podem observar-se os pixéis/vóxeis identificados como CSF. Na **Figura 6.2.b)**, estão identificados os classificados como GM, onde se pode ver a fraca consistência dos agrupamentos. Na **Figura 6.2.c)**, mostra-se a WM, que apresenta boa consistência, e na **Figura 6.2.d)**, apresentam-se os resultados do SOM para MS. Note-se que há, neste últimos, vóxeis de GM mal itequetados.

Figura 6.2 - Apresentação dos resultados da escolha dos super-vóxeis por processamento das corridas de SOM. **a)** Pixéis caracterizados como CSF. **b)** Pixéis caracterizados como GM (fraca consistência). **c)** Pixéis caracterizados como WM. **d)** Pixéis caracterizados como MS.

6.1.3 DC: Primeira corrida

Foi utilizando estes pixéis agrupados pelo SOM que se procedeu à primeira corrida do DC, para obtenção das imagens finais de segmentação. A **Figura 6.3.** representa a segmentação obtida pela aplicação do DC. A **Figura 6.3.a)** apresenta o CSF obtido. A **Figura 6.3.b)** mostra a GM obtida pelo DC, por utilização dos resultados do SOM, daí ser a coluna que pior resultado apresenta, graças aos fracos protótipos do SOM para GM. A **Figura 6.3.c)** apresenta a obtenção da WM e, por último, a **Figura 6.3.d)** mostra a obtenção de MS. É de notar, na **Figura 6.3.d)**, perto dos ventrículos, mais nas pontas, claramente a existência de lesão correspondente à degeneração da matéria branca.

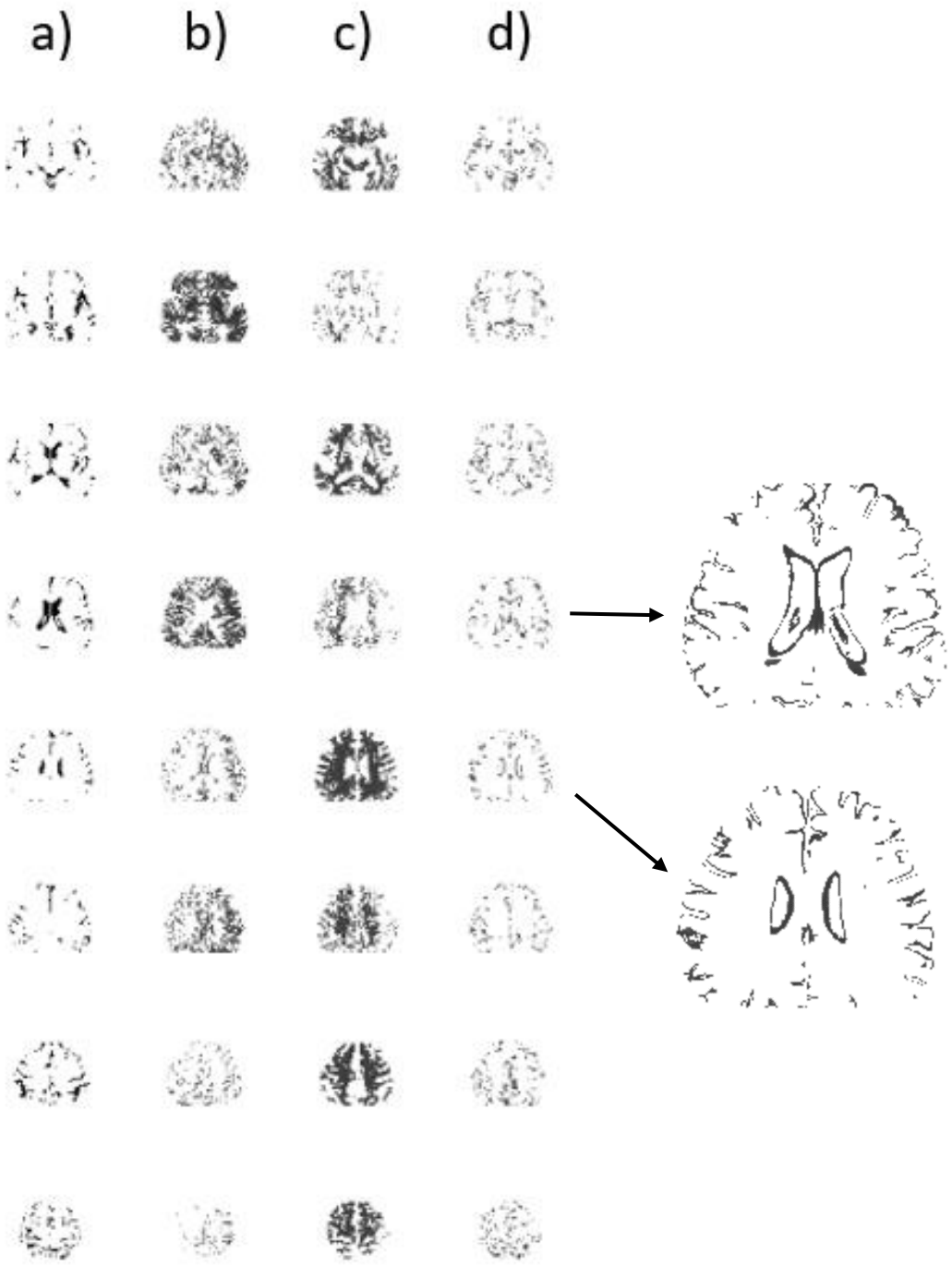


Figura 6.3 - Resultado da segmentação pela primeira corrida do DC. **a)** Resultado para CSF. **b)** Resultado para GM. **c)** Resultado para WM. **d)** Resultado para MS. De notar as claras lesões, por degeneração da matéria branca, nas extremidades dos ventrículos.

6.1.4 Estatísticas

O cálculo das estatísticas tornou-se inapropriado por, ou não existir *ground-truth* ou por este não estar bem feito. No entanto conseguiu-se obter uma imagem sumarizante dos resultados da classificação do pixéis com MS, onde se pode observar a **vermelho**, pixéis que são completamente MS, com probabilidade alta de 66% a 100%, a **azul** aqueles que estão em transição com uma probabilidade entre 33% e 66% e a **verde** pixéis/vóxeis com baixa probabilidade de conterem MS, entre 0% e 33%. A **Figura 6.4.** mostra a imagem representativa da avaliação de pixéis classificados como MS.

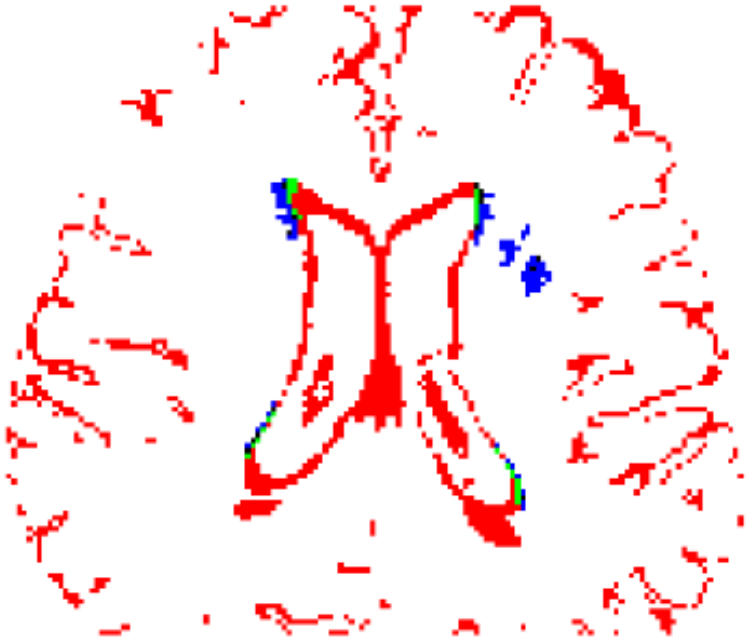


Figura 6.4 - Representação dos pixéis/vóxeis classificados como MS. O esquema de cores remete para a percentagem de tecido de MS presente no pixel. A vermelho, os pixéis/vóxeis com probabilidade de conterem **MS entre 66% e 100%**. A azul aqueles com probabilidades **entre os 33% e os 66%**. Por último, a verde, os pixéis/vóxeis com probabilidade de conter MS **entre os 0% e os 33%**.

6.1.5 Aplicação da máscara de lesão

Para a segunda corrida de DC foi usada uma máscara focando nas lesões encontradas na primeira corrida, e aplicada aos dados originais. O processo encontra-se explicado no **Capítulo 5.3.1.** ***"Preparação da Segunda Corrida"***. Na **Figura 6.5.** está

representado o resultado da aplicação da máscara para cada imagem das aulturas de *DATA.z* para cada sequência, dado anteriormente como exemplo, no capítulo ***"Materiais e Métodos"***, sendo que a **Figura 6.5.a)** corresponde à imagem ponderada em FLAIR, a **Figura 6.5.b)** à ponderada em T2 e a **Figura 6.5.c)** à obtida pela sequência MTI.

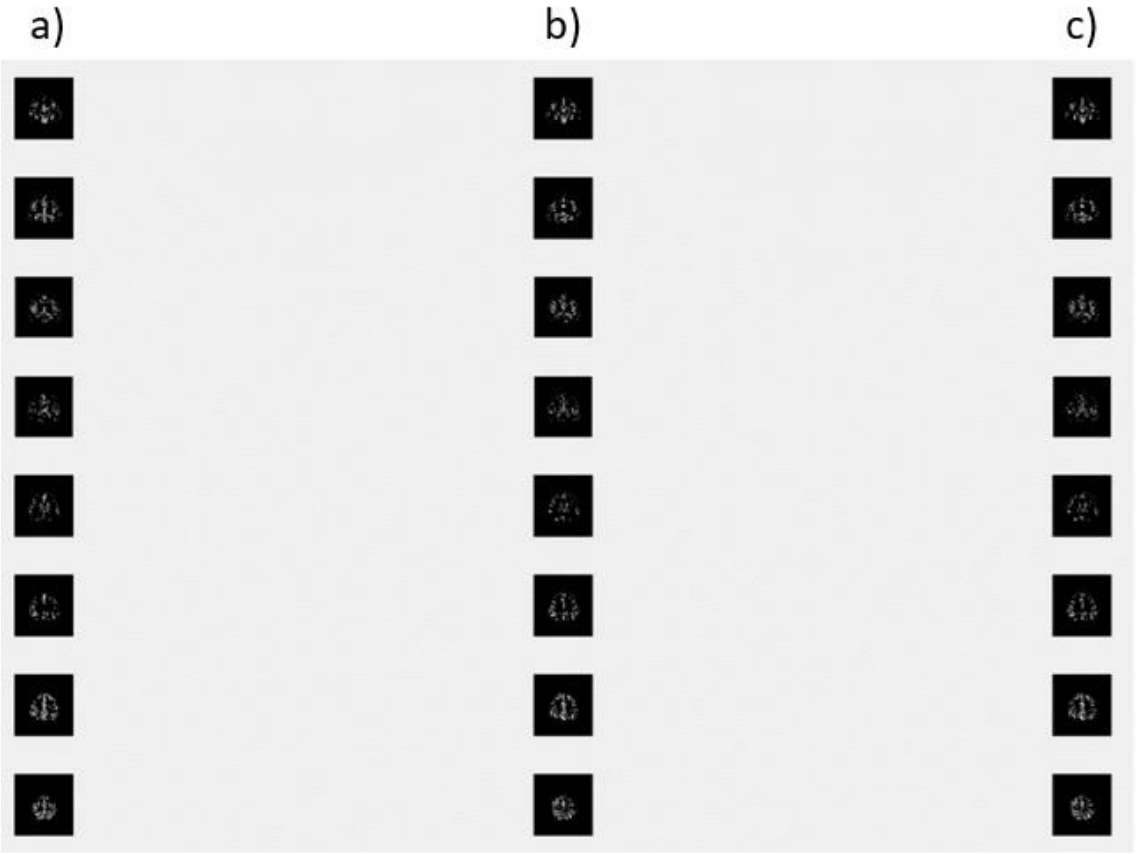


Figura 6.5 - Resultado da aplicação da máscara aplicada a cada altura, sobre as imagens originais, correspondentes às alturas de *DATA.z*, para cada sequência. **a)** FLAIR. **b)** T2. **c)** MTI.

6.1.6 DC: Segunda corrida

O uso da máscara permitiu efectuar uma segunda corrida, na esperança de melhorar os resultados da segmentação, obtidos com a primeira corrida do DC. No entanto, os resultados obtidos na segunda corrida do algoritmo não apresentaram melhorias significativas. Na **Figura 6.6.** podem ser observados os resultados da segunda corrida do DC, onde se nota uma pequena melhoria no que toca à estimativa de GM e MS, tendo existido o aparecimento de mais alguns pixéis correspondentes a cada destes dois tecidos. À semelhança da **Figura 6.3., a)**, **b)**, **c)** e **d)** correspondem aos tecidos CSF, GM, WM e MS, respectivamente.

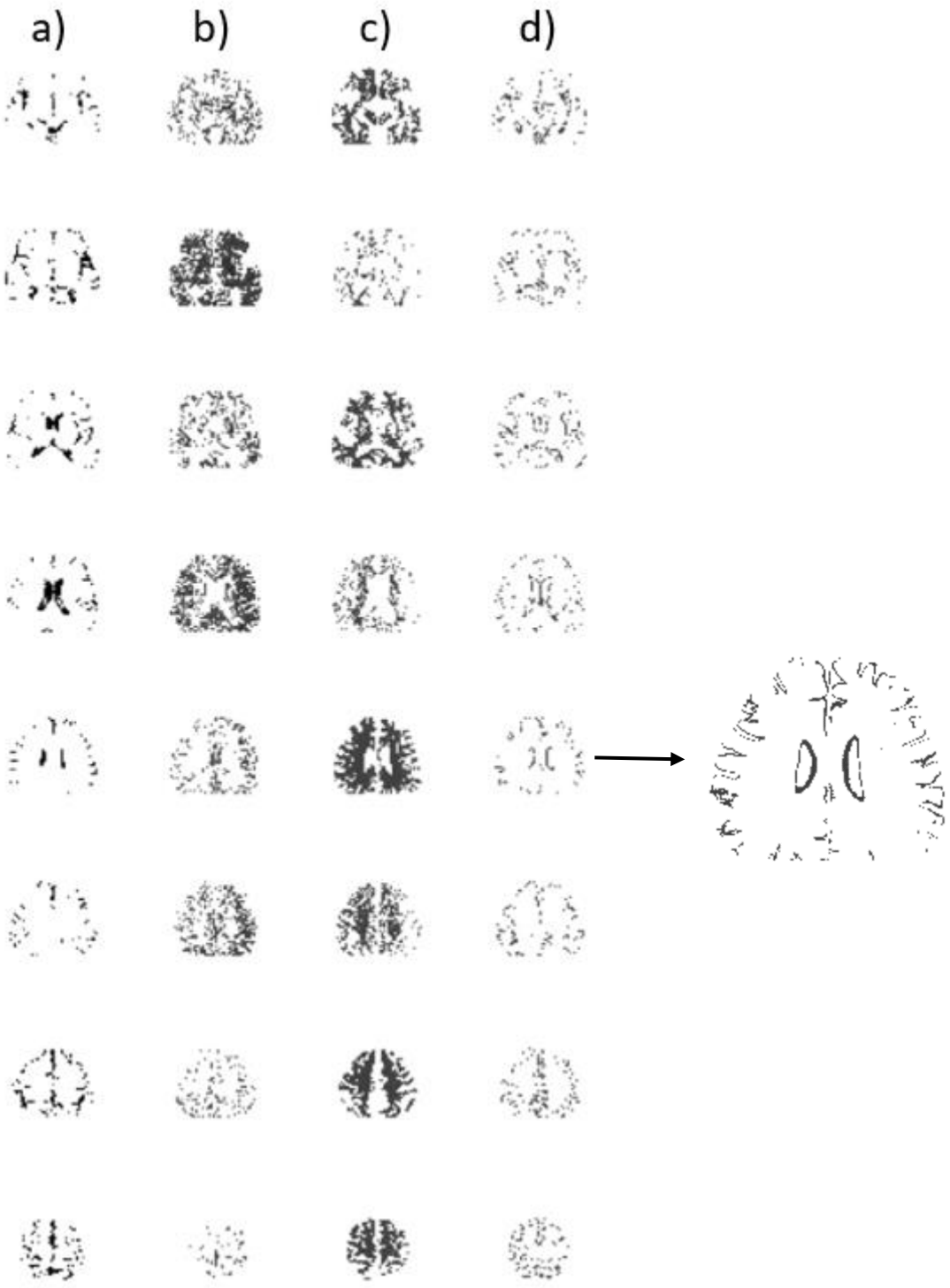


Figura 6.6 - Resultado da segmentação pela segunda corrida do DC. **a)** Resultado para CSF. **b)** Resultado para GM. **c)** Resultado para WM. **d)** Resultado para MS. De notar as claras lesões, por degeneração da matéria branca, nas extremidades dos ventrículos. Note-se também uma ligeira melhoria no número de pixéis mal atribuídos como MS nas margens da imagem, em relação à primeira corrida.

6.2 DATA_HE42_followup

6.2.1 Filtragem

O uso desta estructura deveu-se ao facto de se querer testar imagens do mesmo paciente, mas com o objectivo de estudar o desenvolvimento das lesões detectadas em *DATA_HE42*, agora em imagens recolhidas três anos após as primeiras. Na **Figura 6.7.** tem-se, mais uma vez, a comparação das imagens representativas do volume inicial, com as obtidas após suavização. À semelhança da **Figura 6.1.**, a **Figura 6.6.a)** e a **Figura 6.6.b)**, representam, respectivamente, o volume inicial e o volume filtrado, havendo especial destaque para os contornos da imagem da **Figura 6.7.b)**, mais suavizados que na imagem da **Figura 6.7.a)**.

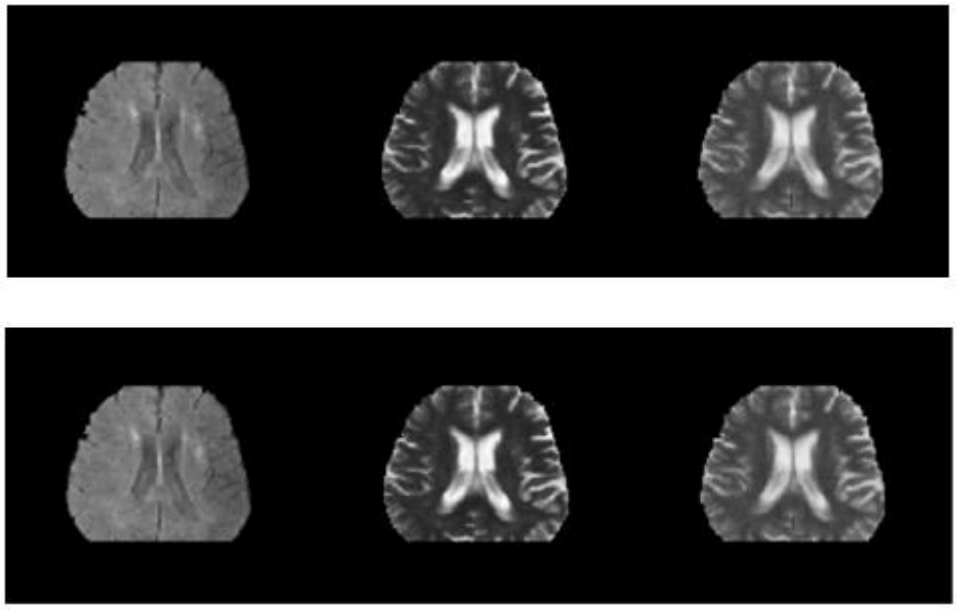


Figura 6.7 - Representação da diferença entre as imagens antese e depois da suavização dos bordos e das diferenças de intensidade dos pixéis/vóxeis. **a)** Imagem representativa do volume inicial, para as três sequências usadas. **b)** Imagem representativa do volume após suavização, para as três sequências. Note-se, mais uma vez, a uniformização dos contornos das imagens.

6.2.2 *Clustering* e SOM

Também para a estructura em causa, contendo as imagens de desenvolvimento das lesões, por degeneração de matéria branca, foram obtidos os protótipos de super-vóxeis resultantes das corridas do algoritmo SOM. Na **Figura 6.8.** estão representados os pixéis/vóxeis atribuídos a cada classe. Como anteriormente, a **Figura 6.8.a)** corresponde

ao CSF, a **Figura 6.8.b)**, a GM, a **Figura 6.8.c)**, a WM e, por último, a **Figura 6.8.d)** corresponde a MS. Mais, uma vez, verificou-se a fraca existência de protótipos para GM.

Figura 6.8 - Representação dos agrupamentos resultantes da atribuição de pixéis/vóxeis às diferentes classes. **a)** Pixéis caracterizados como CSF. **b)** Pixéis caracterizados como GM. **c)** Pixéis caracterizados como WM. **d)** Pixéis caracterizados como MS.

6.2.3 DC: Primeira corrida

Mais uma vez, com as *labels* atribuídas pelo SOM, procedeu-se à primeira corrida de DC. Na **Figura 6.9.** está representada a segmentação, ou classificação proposta como resultado do DC. Como no caso da **Figura 6.3.**, as colunas **a)**, **b)**, **c)** e **d)**, correspondem, respectivamente às classes CSF, GM, WM e MS. Verificou-se, mais uma vez, a influência do pequeno número de protótipos de GM no resultado do DC, que resultou num não reconhecimento de GM, nas imagens da estructura DATA_HE42_followup.

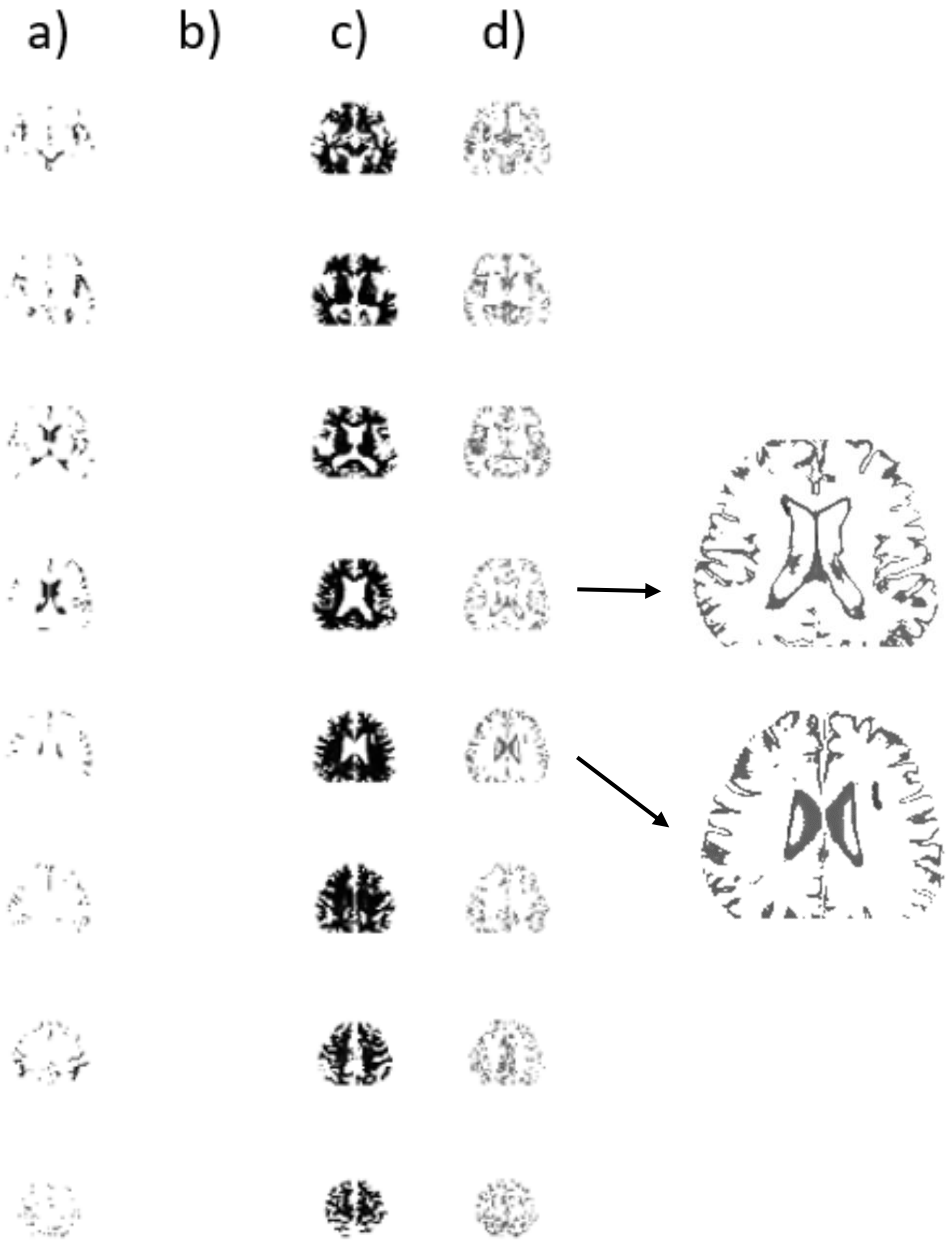


Figura 6.9 - Resultado da segmentação pela primeira corrida do DC. **a)** Resultado para CSF. **b)** Resultado para GM. **c)** Resultado para WM. **d)** Resultado para MS. De notar as claras lesões, por degeneração da matéria branca, nas extremidades dos ventrículos e o desenvolvimento de outra ao lado do ventrículo direito.

6.2.4 Estatísticas

No caso desta estructura, *DATA_HE42_followup*, não foi implementado um campo de *ground_truth*, pelo que, não existindo termo de comparação para os resultados da segmentação, obtidos pelo DC, não houve base para calcular os coeficientes que traduzissem a viabilidade do método.

6.2.5 Aplicação da máscara de lesão

Tal como foi feito para a estructura anterior, foi criada uma máscara baseada nos resultados da primeira corrida do DC. Esta máscara foi aplicada nas imagens correspondentes às alturas contidas em *DATA.z*. A **Figura 6.10.** representa o resultado da aplicação da máscara ao volume em causa, para cada uma das sequências usadas. A **Figura 6.10.a)** corresponde à imagem ponderada em FLAIR, a **Figura 6.10.b)** à ponderada em T2 e a **Figura 6.10.c)** à obtida pela sequência MTI.

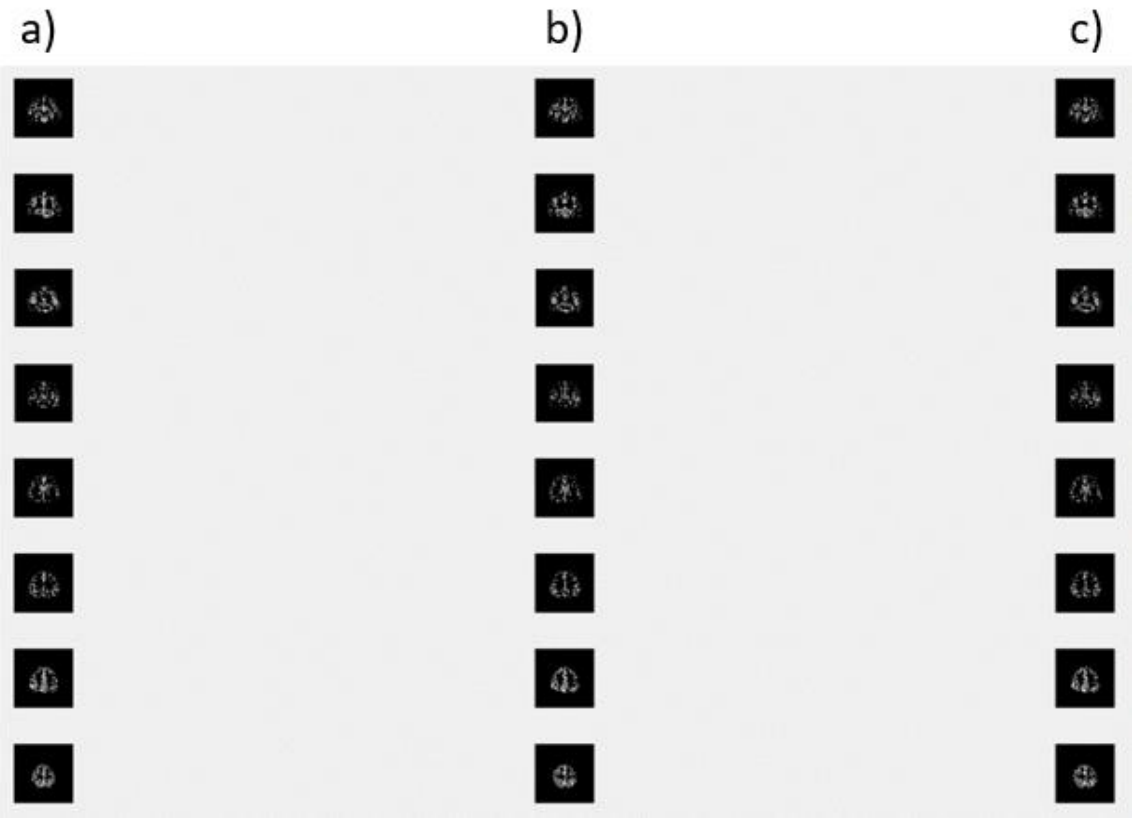


Figura 6.10 - Resultado da máscara aplicada ao volume, em preparação para a segunda corrida de DC, e às três sequências usadas. **a)** FLAIR. **b)** T2. **c)** MTI.

6.2.6 DC: Segunda corrida

Assim, com o novo volume mascarado, efectuou-se uma segunda corrida de DC. As melhorias observadas não foram muito significativas, tendo-se verificado um ligeiro aumento de especificidade na deteção de lesão. Na **Figura 6.11**. pode observar-se os resultados da segmentação, obtidos pelo DC, aquando da segunda corrida. A **Figura 6.11.a)** mostra as várias alturas respectivas ao CSF. A **Figura 6.11.b)** refere-se à GM, revelando, mais uma vez, um mau resultado no que toca à GM. As **Figuras 6.11.c)** e **6.11.d)** remetem para WM e MS, respectivamente, notando-se, no caso de MS, uma ligeira melhoria na deteção da lesão, ao lado do ventrículo direito, mais visível na quinta fatia da coluna de MS a contar de cima.

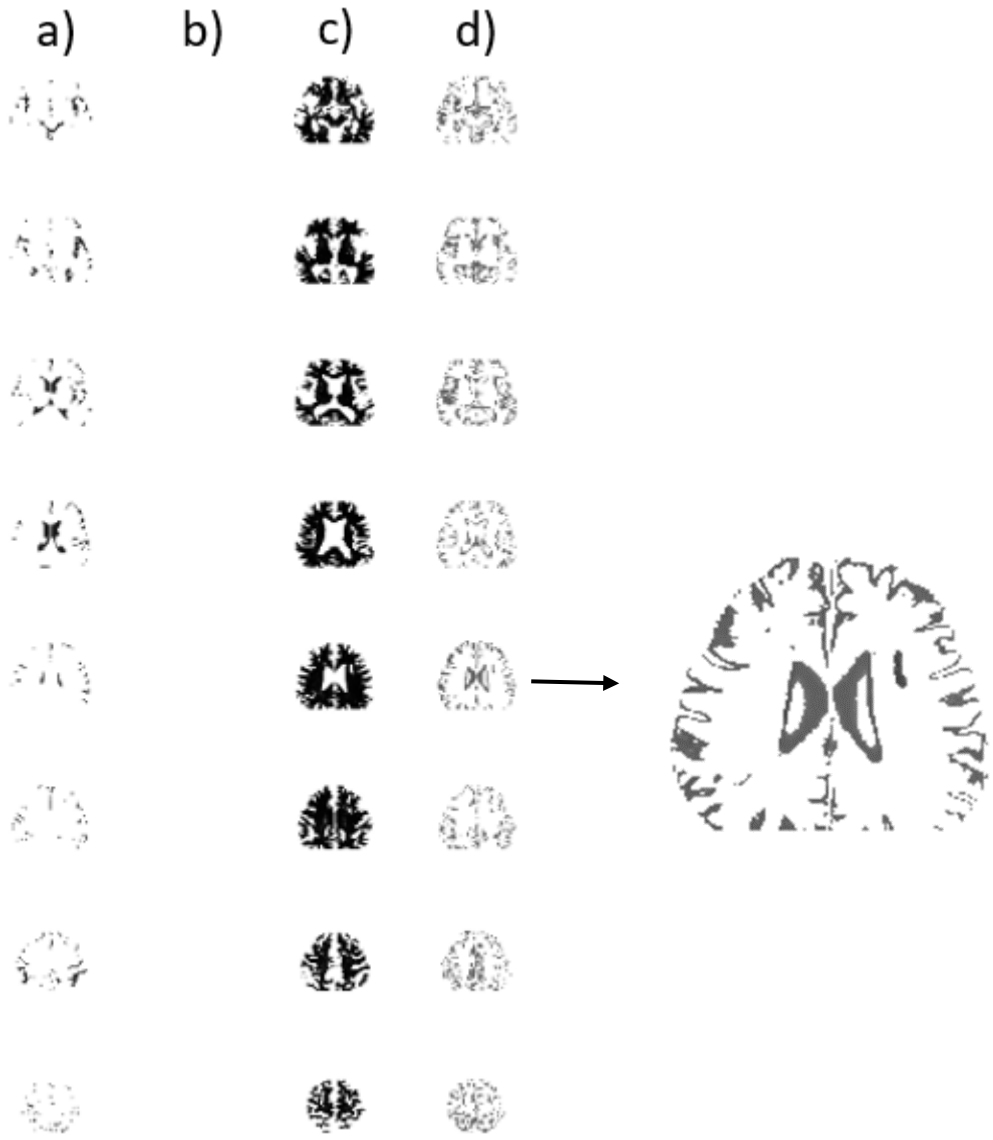


Figura 6.11 - Resultado da segmentação pela segunda corrida do DC. **a)** Resultado para CSF. **b)** Resultado para GM. **c)** Resultado para WM. **d)** Resultado para MS. De notar as claras lesões desenvolvidas, por degeneração da matéria branca, nas extremidades dos ventrículos e o melhoramento da deteção da lesão ao lado do ventrículo direito, mais visível na quinta fatia.

7 Discussão

Com esta dissertação, propôs-se a optimização de uma ferramenta de segmentação de tecidos em Imagens multi-espectrais de Ressonância Magnética. Esta optimização passou por orientar a ferramenta para que se tornasse fácil de utilizar por um utilizador com pouca formação técnica, i.e., *"user friendly"*, possibilitando a este recorrer à ferramenta sem ter de consultar o código. Pretendeu-se ainda que fosse passível de ser utilizado em sistemas operativos mais comuns, como é o caso do *Windows*, que é utilizado pela maioria das pessoas. O algoritmo foi originalmente desenvolvido em ambiente *Linux*.

A realização desta dissertação passa, assim, pelos pontos: segmentação orientada para imagens multi-espectrais, optimização da ferramenta, com particular ênfase na facilidade na facilidade de utilização.

Nos subcapítulos que se seguem, expõe-se as metas cumpridas e as que não foram atingidas na totalidade, justificando o que poderá ter contribuído para certos problemas e incapacidades verificadas.

7.1 Segmentação de Imagens Multi-Espectrais

Sendo o principal foco da dissertação, a segmentação de imagens multi-espectrais, foi o maior motivo de preocupação durante a realização do trabalho. Os resultados apresentados, apesar de, efectivamente, demonstrarem segmentação dos tecidos presentes nas imagens testadas, revelam certas falhas a nível da classificação de alguns tipos de pixéis/vóxeis.

Como apontado no **Capítulo 6**, não foi possível produzir uma correcta classificação da matéria cinzenta, que, por sua vez, levou à classificação em excesso de pixéis como pertencendo à classe de MS. De um ponto de vista teórico, este tipo de resultados deveu-se à fraca criação de protótipos de GM pelo SOM, o que, no momento de atribuir os super-vóxeis às diferentes classes, levou à propagação desse erro aos demais pixéis/vóxeis de MS. Com efeito, o DC acabou por considerar que a matéria cinzenta se encontrava na mesma classe de MS, levando a um agrupamento dessas estruturas na classificação final.

A tentativa de corrigir este problema foi modificar as dimensões dos mapas do SOM. Sendo que as dimensões *default* dos mapas são 12 por 12, foi necessário expandi-los. Os valores usados para os resultados finais para obtenção de protótipo de SOM foram 21 por 21. A razão para estas dimensões foi a de que era um aumento suficientemente grande para que houvesse maior discriminação entre pixéis/vóxeis de diferentes tecidos, mas não eram tão grandes para que existisse sobre-segmentação das imagens, que foi o que se verificou para dimensões 22 por 22 dos mapas. No entanto, este aumento das dimensões dos mapas criou mais protótipos de GM, mas não foi suficiente para melhorar significativamente a classificação feita pelo DC.

No caso da estructura *DATA_HE42_followup*, verificou-se que os fracos protótipos de GM levaram a que o DC não classificasse, de todo, pixéis com GM. Como se verificou nos resultados obtidos para MS, continuou a existir classificação errónea de pixéis que pertenciam a GM.

Outra possível razão para a não total segmentação dos tecidos, pode prender-se com o facto de WM e CSF aparecerem em bastante maior quantidade que GM e MS.

Também a atribuição de super-vóxeis, pelo utilizador, às classes pode ter influenciado os poucos protótipos de GM, apesar de tão pouca quantidade de protótipos deixar algumas dúvidas se, de facto, a subjectividade na seleção de super-vóxeis terá tido influência nas imagens observáveis de GM.

No entanto, é de verificar também que, apesar dos problemas de classificação de GM e à sua influência na classificação de pixéis como lesão, obteve-se correcta classificação das lesões propriamente ditas. Tanto para a estructura *DATA_HE42* como para a estructura *DATA_HE42_followup*.

As lesões presentes nos cantos dos ventrículos, no caso de *DATA_HE42*, que a olho eram facilmente identificadas, foram correctamente identificadas como lesão pelo processo de segmentação e no caso de *DATA_HE42_followup*, além de novamente identificadas essas lesões, foi detectado o desenvolvimento de uma lesão à direita do ventrículo direito.

Estes resultados levam a considerar que a segmentação obtida foi relativamente bem sucedida, considerando que em termos de CSF, o resultado revela ser excelente, o mesmo acontecendo com WM. Apenas no que toca a GM, os resultados foram maus, por falta de protótipos relevantes para se poder afirmar que certo pixel seria, de facto, GM.

Por último, o mais importante, que foi a deteção de lesões por degeneração de matéria branca, pode dizer-se que o resultado foi razoável, no sentido de deteção de lesão, mas deixou a desejar no que toca à especificidade da classificação, por considerar pixéis não contendo lesão, como lesão. Assim, embora com um conjunto de Falsos Positivos, os resultados de segmentação não aparentam ter tido um elevado número de Falsos Negativos na identificação de MS. Inclusivé, comparando com a segmentação obtida em *Linux*, é possível dizer que o conjunto de falsos positivos, pode depender das imagens utilizadas e, consequentemente da anatomia do paciente em causa, afectando o resultado do processo de segmentação, como se pode verificar na Figura 7.1., onde a qualidade dos protótipos de GM das imagens usadas são ligeiramente melhores que os das imagens utilizadas neste trabalho o que reduz o número de FP na classificação de MS. No entanto esse número, apesar de menor, não é nulo, pelo que na imagem se verifca a presença de pixéis que deveriam ser classificados com GM mas que foram classificados como MS.

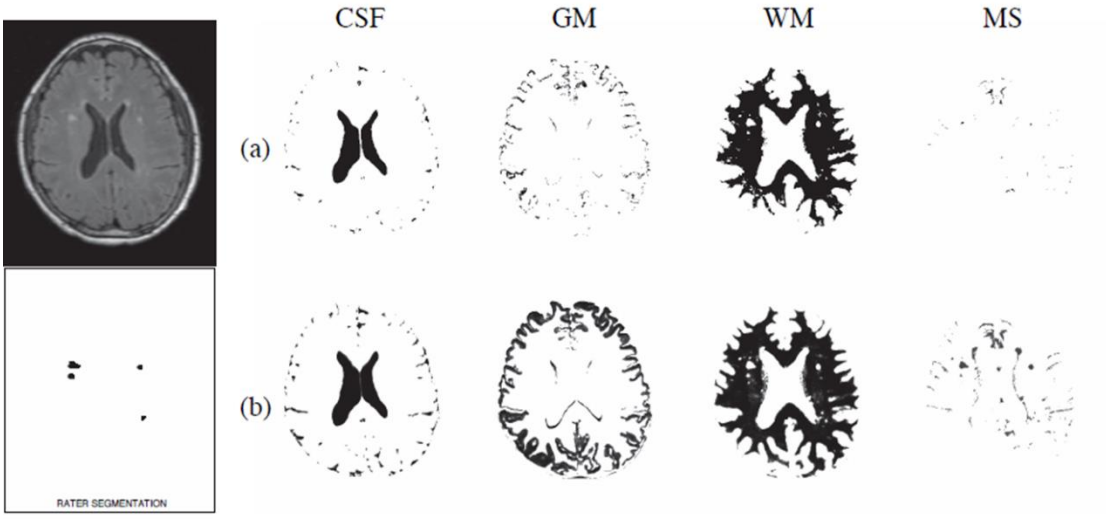


Figura 7.1 – Representação do resultado da classificação em Linux. Verifica-se a presença de pixéis em MS que deveriam pertencer a GM, o que suporta a ideia de que a qualidade dos protótipos obtidos depende das imagens usadas para segmentação [5].

Assim, em termos de avaliação e segmentação de imagens multi-espectrais, poder-se-á dizer que se mostrou algo promissora.

7.2 Foco no utilizador

No que toca à evolução da ferramenta, para se tornar de mais fácil utilização para qualquer utilizador, pode concluir-se que este objectivo foi cumprido. A criação de uma

interface de fácil execução para qualquer utilizador, foi um passo importante na abordagem a este ponto.

Com efeito, aquilo que é pedido do utilizador é que altere o nome das estructuras a usar, dado que os nomes usados foram aqueles que correspondiam aos dados usados nesta dissertação. A interface é simples, desde a observação das imagens passo a passo, até à obtenção de um resultado final, passando pelo possível carregamento de estructuras já criadas, correspondentes ao SOM ou ao DC, para que, dado que são passos de alguma duração em termos de compilação, o utilizador possa obter resultados de forma rápida, no caso de uma primeira passagem pela interface. A possibilidade de o utilizador poder gravar cada imagem individualmente, com a designação que for de melhor compreensão para si mesmo, contorna a necessidade de compreensão de designações feitas por outrem.

7.3 Compatibilidade do algoritmo com Windows

Inicialmente, o algoritmo foi desenvolvido, para ser aplicado em *Linux*. No caso desta dissertação, foi necessário fazer a optimização para *Windows*. A existência de alguns comandos escritos de maneira diferente tanto para um, como para outro sistema operativo, levou à alteração de algum código, como já foi mencionado no **Capítulo 5**.

No entanto, é possível afirmar que, de facto, o algoritmo foi optimizado para *Windows* e que, sendo o sistema operativo mais utilizado no dia à dia pela maioria dos utilizadores de computadores, será uma mais valia introduzida por este trabalho.

Aquilo que é um ponto a favor da optimização desta ferramenta, levanta também uma questão importante, intimamente ligada com o resultado da segmentação em si. Esta questão prende-se com a possibilidade de ainda existirem certas incompatibilidades resultantes da programação em *Linux* e que não se tenham manifestado como erros, mas que tenham, de algum modo, levado à fraca aquisição de protótipos de GM e à consequente má segmentação de GM pelo DC.

7.4 Considerações Final

Olhando para o que foi feito, e os objectivos atingidos com este trabalho, conclui-se que ainda existiria algum trabalho a desenvolver, de maneira a garantir a robustez do

método, como retirar a componente subjectiva da atribuição de super-vóxeis a classes, automatizando também esse passo. No entanto, seria algo que demoraria algum tempo a estudar e a desenvolver pelo que, dado o tempo disponível não foi possível executar.

O estudo da falha de compatibilidade em algum comando ao longo do algoritmo seria outro ponto a abordar com mais tempo, dado que, não sabendo, ao certo, onde se encontra a incompatibilidade, ter-se-ia de testar linha a linha o código, processo que, certamente, demoraria muito.

No geral, com algumas nuances, o algoritmo faz o que se propõe a fazer, ou seja, a segmentação de tecidos em imagens multi-espectrais, obtidas por MRI e, acompanhado de uma interface intuitiva, torna-se agora, de mais fácil utilização, diminuindo bastante a necessidade de quem o utilizar, ter de recorrer a uma leitura cuidada do código.

Em conclusão, ainda existem muitos e possíveis avanços a fazer neste processo, que certamente só viriam a beneficiar o processamento de imagem em MRI. No entanto, não deixa de ser um processo bastante avançado, quando comparado com outras formas e algoritmos de segmentação utilizados, como foi descrito no **Capítulo 2**.

8 Referências

[1] **M. Soltaninejad, T. Lambrou, A. Qureshi, N. Allinson, and X. Ye, "A hybrid method for haemorrhage segmentation in trauma brain CT" , *Conference Paper*, pp. 1–6, 2014.**

[2] **J. Patel and K. Doshi, "Study of Segmentation Methods for Detection of Tumor in Brain MRI," *Adv. Electron. Electr. Eng.*, vol. 4, no. 3, pp. 279–284, 2014.**

[3] **A. V Prabu, A. Bharti, N. Guru, and S. Tripathy, "Brain Tumour Detection in MRI Images Using Matlab" , *International Journal of scientific Research in Science, Engineering and Technology*, vol. 2, no. 2, pp. 1230–1233, 2016.**

[4] **R. C. Patil and a S. Bhalchandra, "Brain Tumour Extraction from MRI Images Using MATLAB," *Int. J. Electron. Commun. Soft Comput. Sci. Eng.*, vol. 2, no. 1, pp. 2–5, 2013.**

[5] **N. Gonçalves, *Advances in Analysis and Exploration in Medical Imaging*. PhD Thesis, Aalto University. 2014.**

[6] **S. Roy, D. Bhattacharyya, S. K. Bandyopadhyay, and T.-H. Kim, "An Effective Method for Computerized Prediction and Segmentation of Multiple Sclerosis Lesions in Brain MRI," *Comput. Methods Programs Biomed.*, vol. 140, pp. 307–320, 2017.**

[7] **E. Küçükkülahli, "Brain MRI Segmentation based on Different Clustering Algorithms", *International Journal of Computer Applications*, vol. 155, no. 3, pp. 37–40, 2016.**

[8] **R. P. Joseph and C. S. Singh, "Brain Tumor Mri Image Segmentation and Detection in Image Processing," *Int. J. Res. Eng. Technol.*, vol. 3, no. 1, pp. 1–5, 2014.**

[9] **K.-S. Chuang, H.-L. Tzeng, S. Chen, J. Wu, and T.-J. Chen, "Fuzzy c-means clustering with spatial information for image segmentation," *Comput. Med. Imaging Graph.*, vol. 30, no. 1, pp. 9–15, 2006.**

[10] **S. Atkins and B. Mackiewich, "Fully automatic segmentation of the brain in MRI," *IEEE Trans. Med. Imaging*, vol. 17, no. 1, pp. 98–107, 1998.**

[11] **N. Gordillo, E. Montseny, and P. Sobrevilla, "State of the art survey on MRI brain tumor segmentation," *Magn. Reson. Imaging*, vol. 31, no. 8, pp. 1426–1438, 2013.**

[12] **I. S. Isa, S. N. Sulaiman, and M. Mustapha, "The automated segmentation techniques of T2-weighted MRI images using K-means clustering and Otsu-based thresholding method," *J. Teknol.*, vol. 78, no. 6–4, pp. 41–48, 2016.**

[13] **N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Trans. Syst. Man. Cybern.*, vol. 9, no. 1, pp. 62–66, 1979.**

[14] **J. Kennedy and R. Eberhart, "Particle swarm optimization," *Neural Networks, 1995. Proceedings., IEEE Int. Conf.*, vol. 4, pp. 1942–1948 vol.4, 1995.**

[15] **R. C. Eberhart and Y. Shi, "Particle swarm optimization: developments, applications and resources," *Evolutionary Computation, 2001. Proceedings of the 2001 Congress on*, vol. 1. pp. 81–86, 2001.**

[16] **M. A. Balafar, A. R. Ramli, M. I. Saripan, and S. Mashohor, "Review of brain MRI image segmentation methods," *Artif. Intell. Rev.*, vol. 33, no. 3, pp. 261–274, 2010.**

[17] **S. Haykin, *Neural networks, A Comprehensive Foundation*. McMaster University. Ontario, Canada.**

[18] **A. Diplaros, N. Vlassis, and T. Gevers, "A spatially constrained generative model and an EM algorithm for image segmentation.," *IEEE Trans. Neural Netw.*, vol. 18, no. 3, pp. 798–808, 2007.**

[19] **J. Sinkkonen and S. Kaski, "Clustering based on conditional distributions in an auxiliary space.," *Neural Comput.*, vol. 14, no. 1, pp. 217–39, 2002.**

[20] **S. Kaski, J. Sinkkonen, and A. Klami, "Discriminative clustering," *Neurocomputing*, vol. 69, no. 1–3, pp. 18–41, 2005.**

[21] **M. Parvathi, "Segmentation of Medical Image using Clustering and Watershed Algorithms," *Am. J. Appl. Sci.*, vol. 8, no. 12, pp. 1349–1352, 2011.**

[22] **R. Ratan *et al*, "Un-supervised segmentation and quantisation of malignancy from breast MRI images", *Journal of the National Science Foundation of Sri Lanka,* vol. 44, no. 4, pp. 437–442, 2016.**

[23] **Y. Chen, "Medical Image Segmentation Using Independent Component Analysis-Based Kernelized Fuzzy C-Means Clustering," *Mathematical Problems in Engineering,* vol. 2017, 2017.**

[24] **A. Hyvärinen and E. Oja, "Independent component analysis: algorithms and applications," *Neural networks,* vol. 13, no. 4–5, pp. 411–430, 2000.**

[25] **S. M. Holland, "Principal Components analysis (PCA)", *Scientific Paper,* Department of Geology, University of Georgia, Athens, GA 30602-2501. July, 2016.**

[26] **I. Kaya, A. Pehlivanlı, E. Sekizkardeş, and T. Ibrikci, "PCA based clustering for brain tumor segmentation of T1w MRI images," *Comput Meth Prog Bio,* vol. 140, pp. 19–28, 2017.**

[27] **M. N. Ahmed, S. M. Yamany, N. Mohamed, A. A. Farag, and T. Moriarty, "A modified fuzzy C-means algorithm for bias field estimation and segmentation of MRI data," *IEEE Trans. Med. Imaging,* vol. 21, no. 3, pp. 193–199, 2002.**

[28] **R. H. Chunming LI ZhaoHua Ding, J. Chris Gatenby, Dimistris N. Metaxas, and John C. Gore, "A Level Set Method for Image Segmentation in the Presence of Intensity Inhomogene ities with Application to MRI," *Ieee Trans. Image Process.,* vol. 20, no. 7, pp. 2007–2016, 2011.**

[29] **A.Moreira, *"Desenvolvimento e aplicação de ferramentas para o pré-processamento de imagens de ressonância magnética multi-espectral.",* Faculdade de Ciências e Tecnologias da Universidade Nova de Lisboa, Monte da Caparica, Setembro, 2017.**

[30] **C. Lucchinetti, et al., *"Heterogeneity of Multiple Sclerosis Lesions: Implications for the Pathogenesis of Demyelination".* Departments of Neurology, Neuropathology, and Immunology, Mayo Clinic, Rochester, MN; Institute of**

Neuropathology, University of Gottingen, Germany; and Brain Research Institute, University of Vienna, Austria. *American Neurological Association*, February, 2000.

[31] Kalat, J. W. (2009). *Biological psychology*. Belmont, Calif: Wadsworth, Cengage Learning, 2009.

[32] R. Vigário, *"Imagiologia: Parâmetros de Aquisição"*, in *Imagiologia – MIEB – FCT/UNL 2016-2017*, Faculdade de Ciências e Tecnologias da Universidade Nova de Lisboa, , 2016.

[33] J. Fonseca, *"Sistemas Sensoriais: Extração de características de objectos isolados"*, in DEE/FCT/UNL – Sistemas Sensoriais 2016/2017, 2016.

9 Anexos

9.1 *Flow* do algoritmo

Para que a compreensão do algoritmo seja facilitada e, visando a facilidade de compreensão do encadeamento das funções usadas, para possível consulta no futuro, de seguida estão apresentados os diversos encadeamentos para cada passo do algoritmo, bem como as funções desenvolvidas para aplicação da máscara.

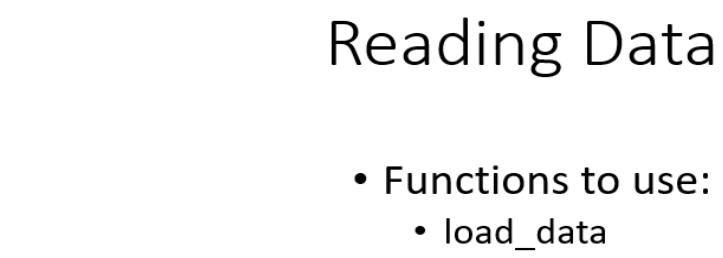


Figura 9.1 – Funções usadas no passo de carregamento de dados para estudo.

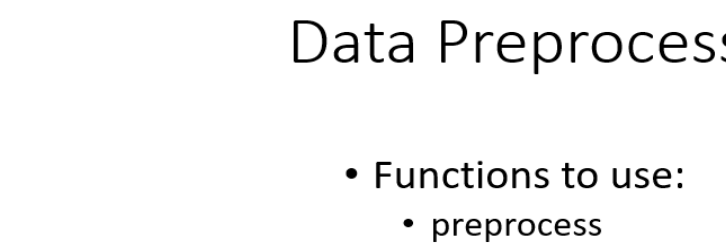


Figura 9.2 – Funções usadas na suavização, ou filtragem das imagens.

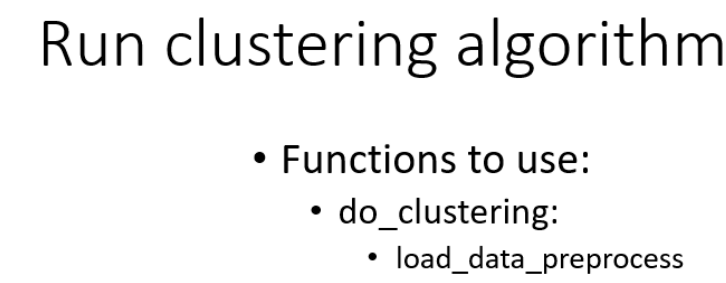


Figura 9.3 – Funções usadas para as múltiplas corridas de SOM, de notar que funções que fazem parte de *toolboxes* do Matlab não estão incluídas no *flow*.

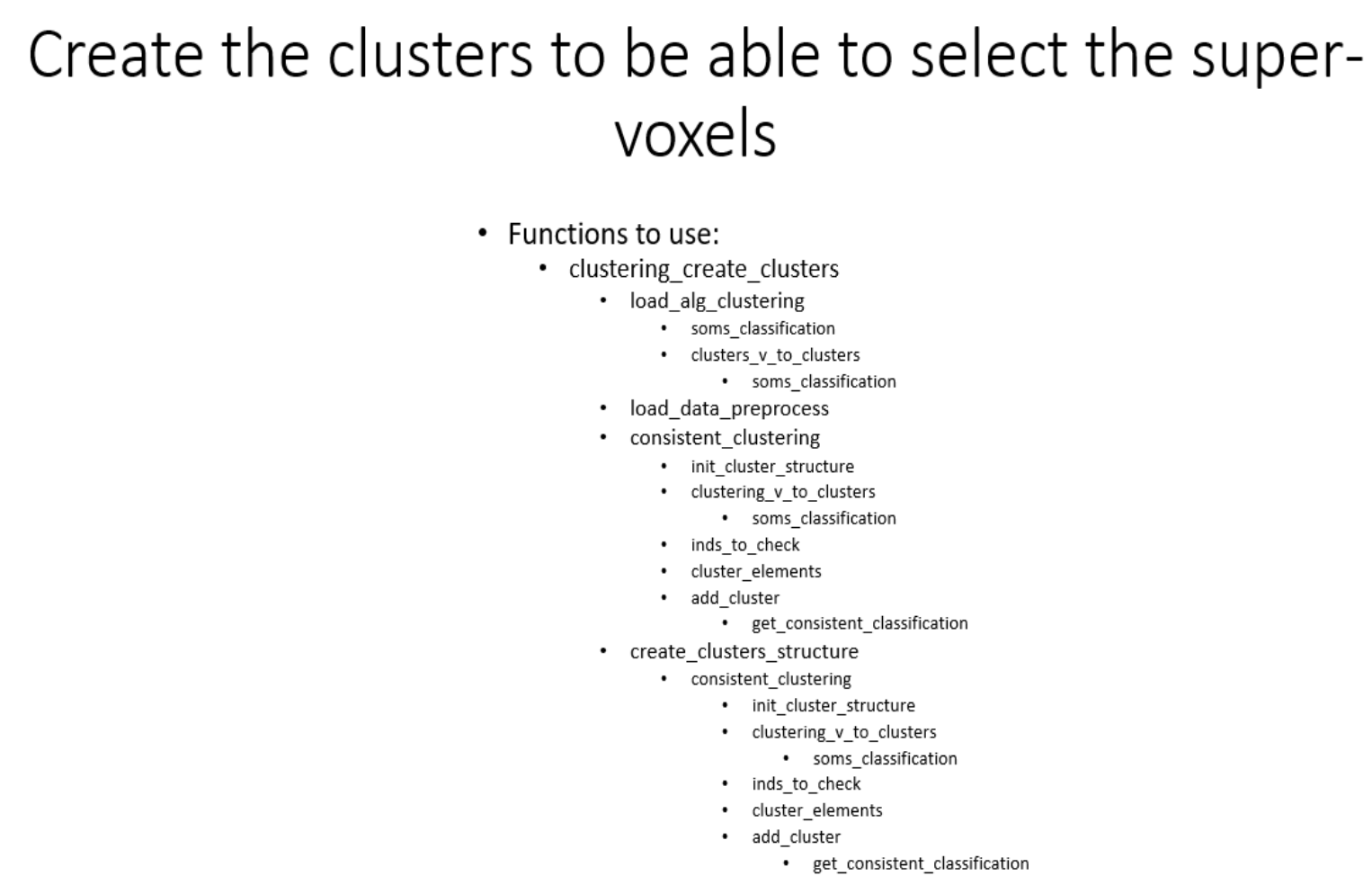


Figura 9.4 – Encadeamento de funções para seleção de super-vóxies, primeira parte.

- create_clusters_structure
  - consistent_clustering
    - init_cluster_structure
    - clustering_v_to_clusters
      - soms_classification
    - inds_to_check
    - cluster_elements
    - add_cluster
      - get_consistent_classification
  - method_clustering
    - init_cluster_structure
    - clustering_get_blocks
    - add_cluster
      - get_consistent_classification
    - clean_clusters
      - distance_function
      - unmerge_clusters
    - recluster
      - clustering_v_to_clusters
      - get_consistent_classification
    - clean_inds
      - distance_function
      - unmerge_clusters
  - cluster_postprocessing
    - sort_clusters

Figura 9.5 – Encadeamento de funções para seleção de super-vóxeis, segunda parte.

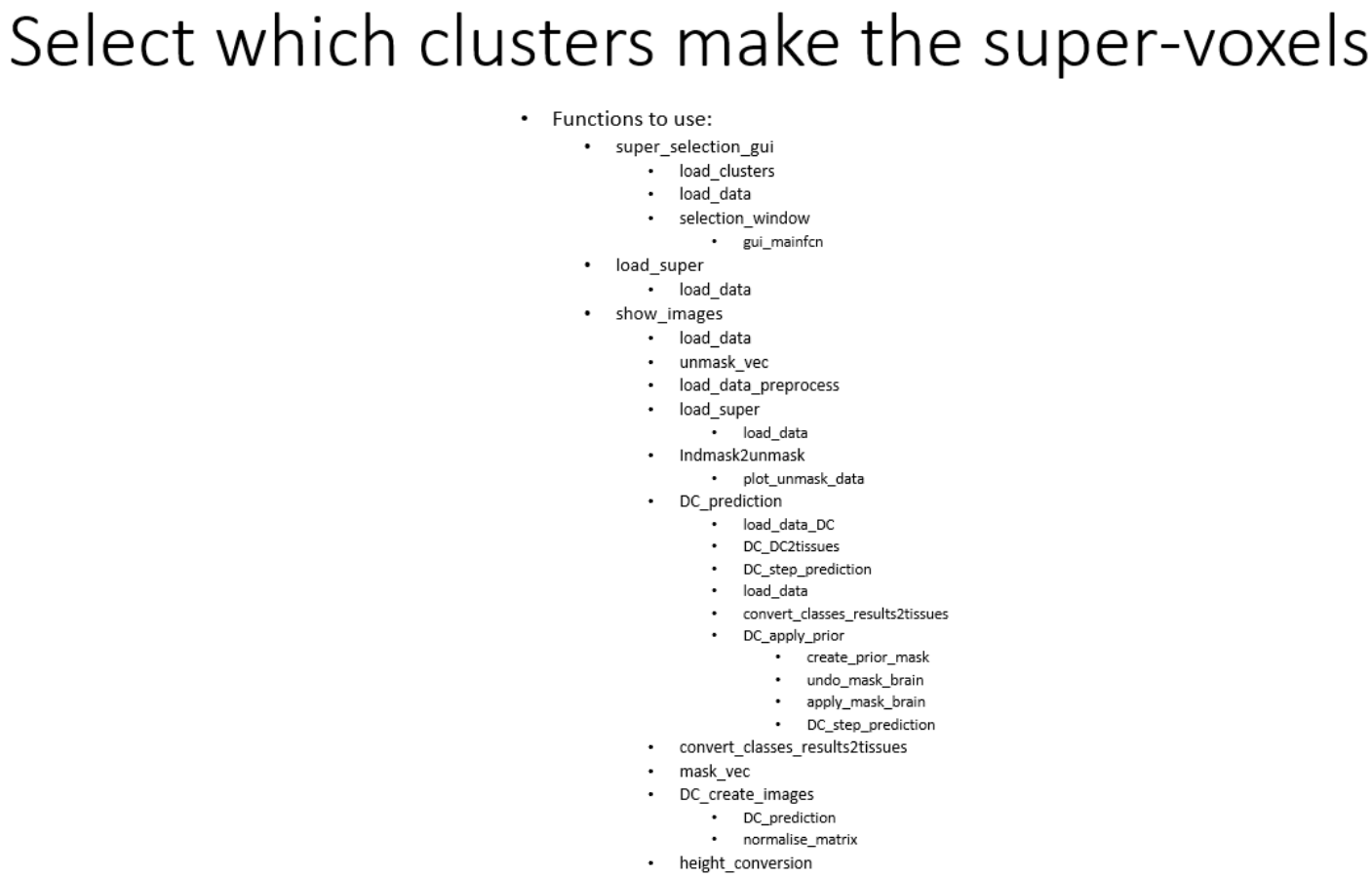


Figura 9.6 – Encadeamento de funções para atribuição de protótipos a classes.

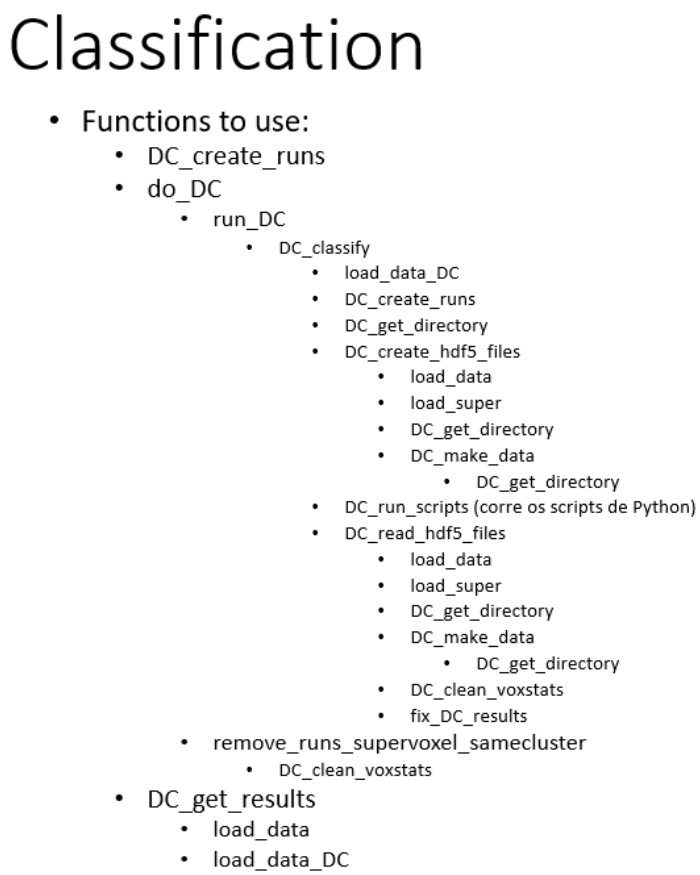


Figura 9.7 – Encadeamento de funções para classificação de pixéis/vóxeis (uso do DC).

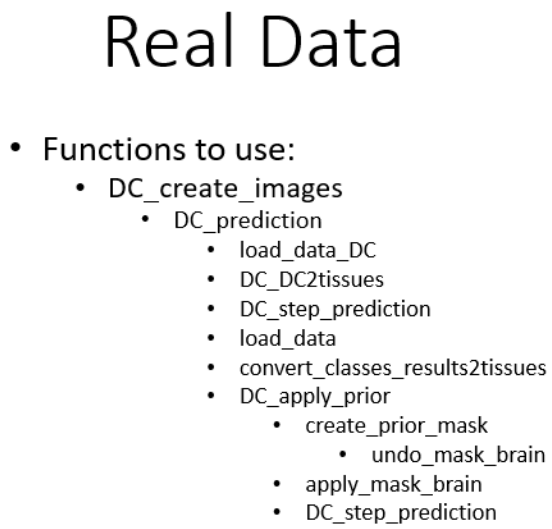


Figura 9.8 – Encadeamento de funções para obtenção de imagens dos resultados do DC.

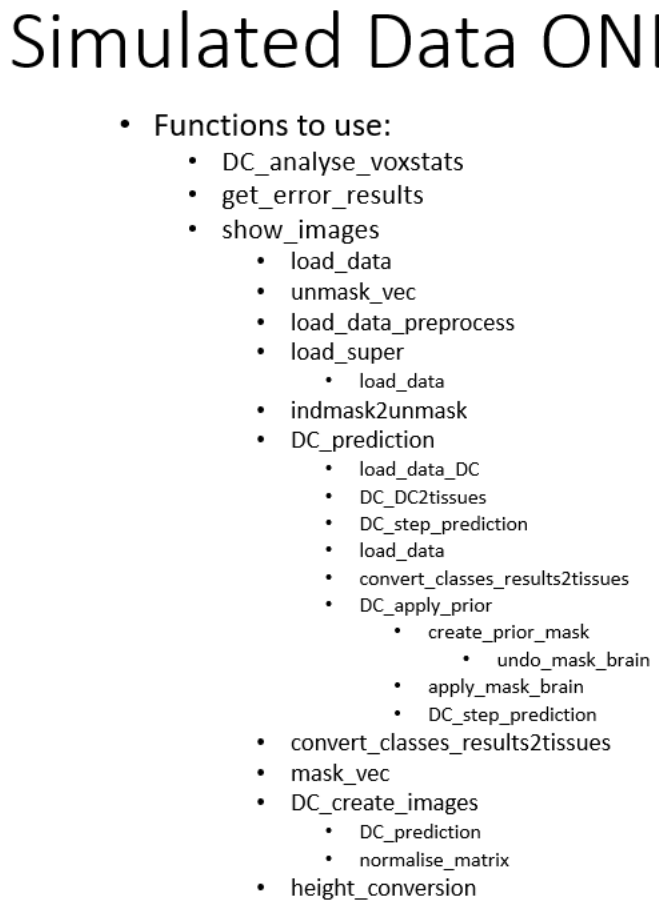


Figura 9.9 – Encadeamento de funções para obtenção de estatísticas.

Crop image matrix to obtain lesion column only

```
binarized_img_DC=(img_DC<1);   %binarize image
figure(10),imagesc(binarized_img_DC),colormap gray,axis image,axis off;
[x,y,h]=size(binarized_img_DC);   %save image dimensions
lesion_matrix_width=y/4;   %size to keep
binarized_img_DC_lesion=binarized_img_DC(1:x,((y-lesion_matrix_width):y),(1:DATA.n_images));   %lesion matrix
figure(11),imagesc(binarized_img_DC_lesion),colormap gray,axis image,axis off;
```

Figura 9.10 – Código da função responsável por isolar e binarizar a lesão, a partir do primeiro resultado de DC.

Crop lesion column to obtain lesion images for each sequence and mask dilation

```
[x_lesion,y_lesion,h_lesion]=size(binarized_img_DC_lesion);   %save image matrix dimensions
dilate_struct=strel('disk',1);   %create structure for image dilation, in this case a disk was the most appropriate
for i=1:DATA.n_images
        binarized_img_DC_lesion_sequence=binarized_img_DC_lesion(1:x_lesion,(1:y_lesion),i);   %obtain lesion column for each sequence
        dilated_sequences=imdilate(binarized_img_DC_lesion_sequence,dilate_struct);   %dilation applying disk structure
        figure(12);
        subplot(1,3,i);
        imagesc(dilated_sequences),colormap gray,axis image, axis off;
end
z_index=numel(DATA.z); %variable to save the number of slices in DATA.z
for j=1:z_index
        dilated_sequences_sliced{j}=dilated_sequences(1 + (j-1)*256:(j)*256,1:256);   %crop dilated sequences in smaller matrixes
        figure(13);
        subplot(8,1,j);
        imagesc(dilated_sequences_sliced{j}), colormap gray, axis image, axis off;
end
```

Figura 9.11 – Código da função que isola a lesão em matrizes de menores dimensões, e que dilata a imagem binarizada para a criação da máscara.

Apply lesion mask to original images

```
for sequence=1:DATA.n_images
        v=sequence;
        for orig=DATA.z(1):DATA.z(end)   %Obtain the images of the heights displayed in DATA.z concerning every sequence
                for l=1:z_index
                        img_orig_by_z{l}=DATA.ORIG.volume(:,:,orig,sequence);
                        figure(14);
                        subplot(8,3,v);
                        imagesc(img_orig_by_z{l}), colormap gray, axis image, axis off;
                end
                v=v+3;
        end
end
for w=1:DATA.n_images
        e=w;
        for ordinal=1:z_index
                mask_apllied{ordinal}=dilated_sequences_sliced{ordinal}.*img_orig_by_z{ordinal};   %mask applied by multiplying the dilated mask matrix with the images of the sequences
                figure(15);
                subplot(8,3,e);
                imagesc(mask_apllied{ordinal}), colormap gray, axis image, axis off;
                if ordinal > 1
                        mask_apllied_array=cat(3,mask_apllied{ordinal-1}, mask_apllied{ordinal});
                end
                e=e+3;
        end
        new_volume(:,:,:,w)=mask_apllied_array;   %save the new volume in a 4D cell array
end
```

Figura 9.12 – Código da função que recorta o volume inicial e aplica a máscara previamente obtida nessas imagens.

9.2 Problema de instalação do Python

Como, por vezes, o software não colabora, existiu um problema com a instarão do Pyhton 2.7 e com os editores do mesmo, pelo que aqui fica uma maneira de resolver o problema.

Foi no chamamento dos ficheiros de Python, que apareceram alguns problemas, principalmente a nível de compatibilidade. Dado que, actualmente, a maioria dos utilizadores do sistema operativo Windows utiliza a sua versão mais recente, o Windows 10, a primeira abordagem baseou-se em utilizar o Python 3.6, a versão mais recente e aquela que teoricamente correria com maior fluidez, e compatibilidade, quando instalada em computadores com Windows 10. Sendo que os *scripts* iniciais tinham sido programados em Python 2.7. Imediatamente, gerou-se uma sucessão de erros na consola do Matlab ligados aos ficheiros de Python. Inicialmente, a solução tentada, foi ceder um pouco e, em vez de tentar actualizar todo o código para a versão 3.6, tentou-se instalar a versão 2.7. Esta tentativa não deu certo durante algum tempo, devido a um problema de codificação de caracteres, ou seja, a codificação que a versão 2.7 tinha de base não reconhecia certos caracteres como, por exemplo, plicas **(‘)**, **Figura 9.13.**. A solução encontrada, após bastante pesquisa, foi forçar uma nova codificação geral no código base do Python, forçou-se a codificação ***uint8***, que é, actualmente a codificação *default*, mas que na versão utilizada não estava inserida. Dado que o executável foi instalado juntamente com o *software Anaconda*, o qual providenciou também o editor *Spyder* para edição do código, os erros iniciais (com o Python 2.7 já a funcionar) prenderam-se com *callbacks* ligados a ficheiros base de instalação do *Anaconda* principalmente em *tables.py*. Neste ponto, apesar de bem sucedida a instalação do Python 2.7, o editor *Spyder* começou a apresentar erros de inicialização, deixando totalmente de correr. O mesmo aconteceu com mais editores até ter sido possível instalar o *Pycharm*, ao fim de algumas tentativas. A visualização do código de Pyhton num editor para o efeito (ao invés de no WordPad como vinha a ser feito há algum tempo) permitiu colocar *breakpoints* no código, chegar à base do problema e resolvê-lo.

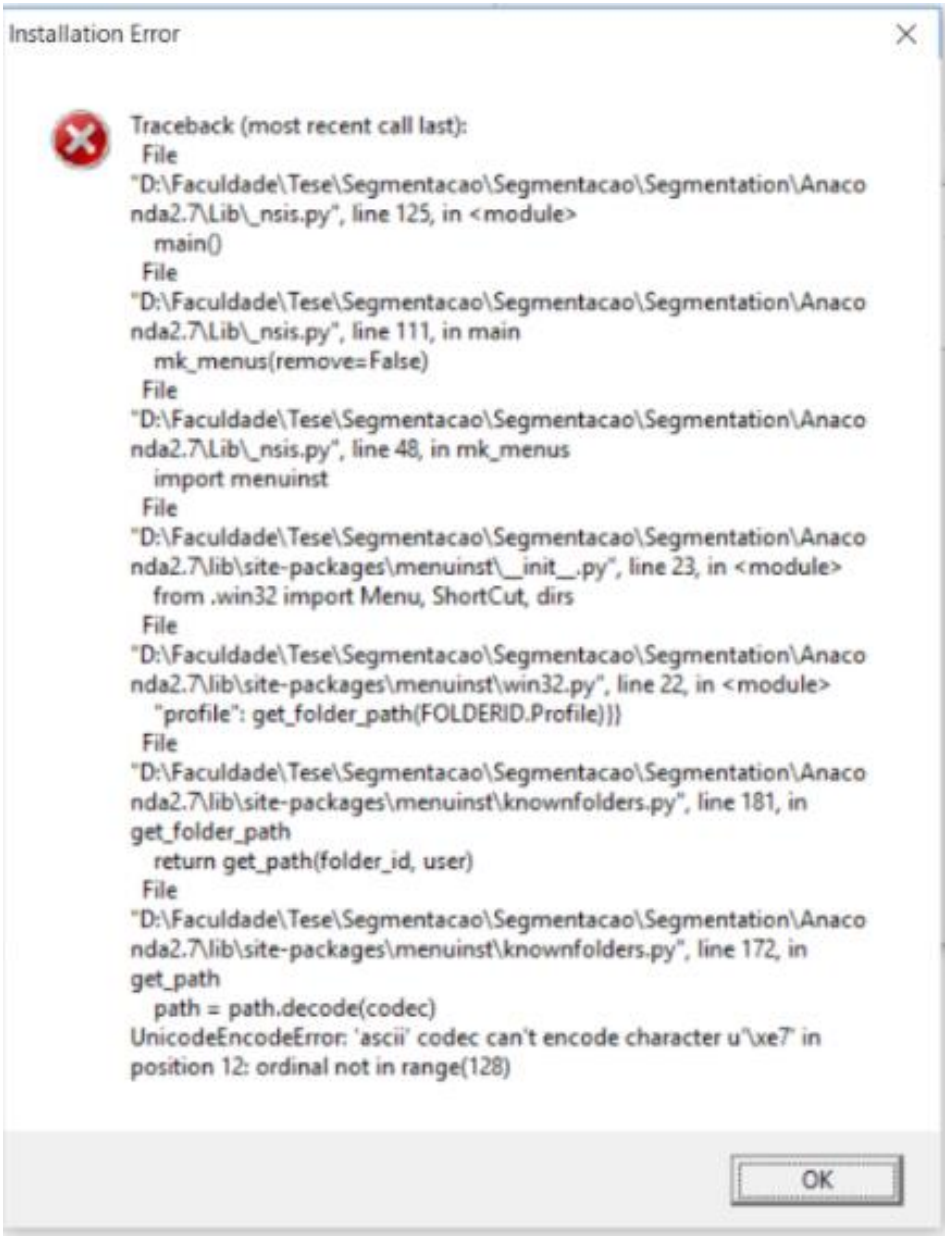


Figura 9.13 – Mensagem resultante do problema de instalação do Python 2.7.