Title：Python implementation of MRI microstructure mapping toolbox

Author：Jijun Liang

Degree：MSc Computing and IT Management

Supervisor Name：Paddy Slator

Organization Name：School of Computer Science and Informatics, Cardiff University； Cardiff University Brain Research Imaging Center(CUBRIC)

Date：26/08/2024

1. Preamble

1.1 Table of contents

Table of Contents

1.2 Abstract

The core goal of this project is to migrate MRI microstructure mapping technology from the MATLAB platform to the Python platform, creating a robust, open source, and

user-friendly package to improve the broad availability and utility of the technology. Python was chosen as the target platform because of its wide range of applications, rich library support, and good community activity, making it an ideal choice for scientific research and application development. Through this project, we have successfully transformed InSpect technology from MATLAB to Python, which is a good fit for InSpect technology. Using Python to implement machine science instead of the traditional optimization method used in MATLAB not only optimizes the operation efficiency of the algorithm, but also enhances its application flexibility in the field of medical image processing. In addition, the open-source nature allows researchers worldwide to freely use and modify the software package, further driving innovation and development of medical imaging technology.

1.3 Acknowledgement

I would like to express my sincere gratitude to everyone who supported me throughout the journey of completing this dissertation. First and foremost, I am deeply thankful to my supervisor, Paddy Slator, whose guidance, encouragement, and invaluable insights have been instrumental in shaping this research. Your patience and expertise have truly enriched my understanding and approach to the subject matter. I am also grateful to the faculty and staff of Cardiff University for providing an excellent academic environment and resources that facilitated this research endeavor. The opportunities for learning and growth offered by the university have been crucial in my academic development. Special thanks go to my family and friends for their unwavering support, understanding, and motivation throughout this challenging yet rewarding experience. Your belief in me kept me focused and determined to accomplish this milestone. Lastly, I extend my appreciation to all the researchers and developers whose work laid the foundation for the technologies and methodologies utilized in this dissertation. Your contributions have been indispensable. This dissertation would not have been possible without the contributions and support of each individual mentioned

above. Thank you all for being part of this journey and for your invaluable contributions.

1.4 Privacy or commercial limitations on the use of the information contained in the dissertation

Since InSpect is open source, both the papers and the code, there are no privacy or commercial restrictions. All research results and technical details are publicly accessible, ensuring transparency and broad accessibility of research. This openness promotes knowledge sharing, accelerates technological innovation, and enhances the credibility of research.

2. Introduction

Magnetic resonance imaging (MRI) technology plays a crucial role in the modern medical field, especially in the diagnosis and monitoring of diseases. As technology advances, more and more MRI applications are incorporating machine learning techniques to parse and process image data to generate microstructural maps that reflect tissue microstructure and function, such as axon diameter, cell size, and blood flow. The microstructure map captures multiple MRI images through different scanner Settings. Each MRI image has a different contrast, and a model is fitted to these multiple MRI images to infer a series of microscopic structures. Advances in this technology not only enhance the detailed resolution of imaging, but also deepen our understanding of the internal structure of biological tissues. The goal of this project is to transform the MATLAB implementation of InSpect, an existing microstructure mapping technology, into a robust, open source, and user-friendly Python package. This conversion will make it easier for the technology to be widely applied and modified, especially in medical imaging research worldwide. Python was chosen as the implementation language because of its wide use, strong library support, and good extensibility, making it particularly suitable for developing scientific tools and applications. By developing such a tool, we can not only improve the accessibility and utility of MRI microstructure mapping technology, but also advance scientific research in the field of medical imaging. In addition, the project will enable developers to collaborate and communicate with the global research community, in particular the Cardiff University Brain Research Imaging Centre (CUBRIC). This interaction will greatly facilitate the rapid iteration and optimization of the technology, while also providing project participants with valuable practical experience and research skills improvement opportunities. In summary, through technology transformation and community collaboration, the project aims to lower the barriers to application of advanced MRI technology, promote its use worldwide, and further promote scientific and technological progress and innovation in the field of medical imaging.

To further understand MRI imaging techniques, we need to understand what a T2-D experiment is, which is actually talking about a magnetic resonance imaging (MRI) technique that combines T2 relaxation time and diffusion coefficient. To make it easier to understand, we can break down this technique into its two core components: T2 relaxation time and diffusion imaging and explore how they work together. T2 relaxation time is a parameter used in MRI to describe the rate at which the water molecular signal decays. In the human body, water molecules are excited by a magnetic field and then gradually return to their equilibrium state. During this process, the magnetization vector of the water molecules becomes out of sync, which leads to the attenuation of the MRI signal. Different biological tissues (such as muscle, fat, or tumors) have different T2 relaxation times due to differences in their physical and chemical environment. Therefore, by measuring T2 relaxation time, doctors can distinguish between different tissue types and diagnose certain disease states. Diffusion imaging is used to observe the movement of water molecules through tissues. In areas with dense cells (such as active tumor areas), the movement of water molecules is more restricted, resulting in a lower diffusion coefficient. In contrast, in regions with lower cell density or looser cell structure, water molecules will diffuse more freely and have a higher diffusion coefficient. By measuring diffusion coefficients in different areas, doctors can obtain information about tissue structure and possible pathological changes.

By capturing both T2 relaxation and diffusion data, the T2-D assay provides physicians with a more comprehensive means of analysis. This combination allows for a better interpretation of the physical and biochemical properties of the tissue, thus providing more accurate information for disease diagnosis.

As shown in the figure, Fig.1 shows InSpect's experimental effect [1]. It is shown that the examination can infer the microstructure map more accurately than the standard method on the simulated data. At a noise level of SNR=100 (signal-to-noise ratio), InSpect captures the kernel features of the map. Our project can use free open source and more efficient python code to realize the simulation of MRI images at various signal-to-noise ratio levels and use random forest network to learn the features contained in it. This is due to its being faster and potentially more stable than the standard optimization techniques used by InSpect.

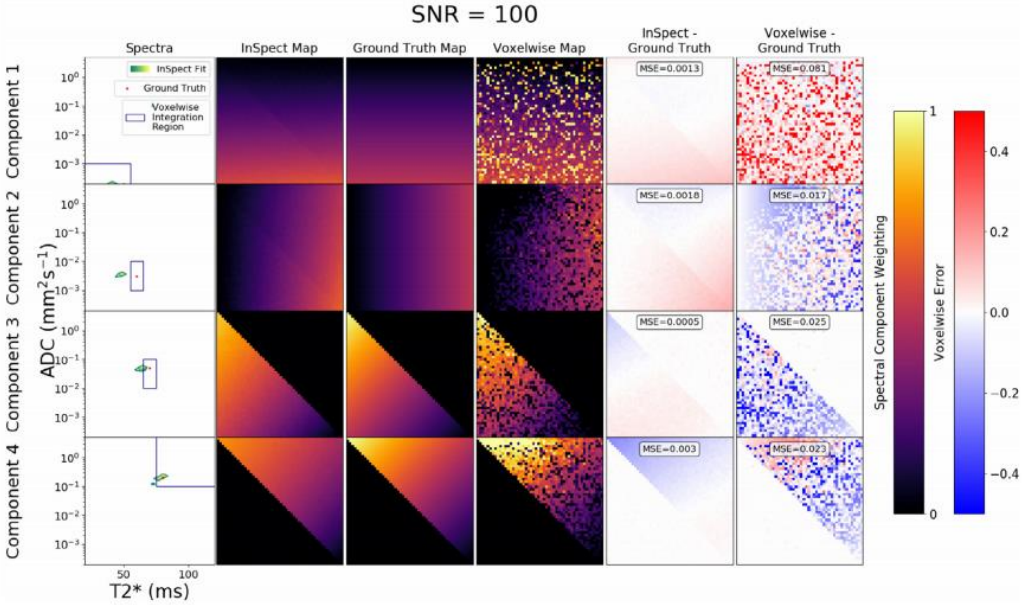


Figure 1. Figure reproduced from Slator et al. 2021 under the Creative Commons CC-BY license. InSpect is applied to a simulated image with SNR=100 noise level, It breaks down an MRI image into a set of components, each with specific MRI properties (indicated by the leftmost "spectrum"). For each component, it also extrapolates a distribution map showing the spatial distribution of each component.

3. Aim and Objectives

The main research goal of this project is to develop a powerful and user-friendly Python package for migrating existing MRI microstructure mapping techniques from MATLAB to the Python platform. The software package will be applied in the field of medical imaging, especially for the study and analysis of the distribution of multiple spectral components in images. With this technique, researchers can more accurately identify and quantify microscopic features of cell structure such as axon diameter, cell size, and blood flow velocity, thus providing more detailed data support in the diagnosis and treatment of diseases. In addition, the project aims to optimize and reconstruct MRI image data through machine learning techniques. Through the training and application of machine learning algorithms, the image processing process can be improved, the automation level of image analysis can be improved, and the accuracy and efficiency of image analysis can be improved. This includes, but is not limited to, automated image segmentation, feature extraction, and structure recognition. Through the process of simulation, processing, and learning, the project will not only help drive innovation and application of MRI technology but will also promote the development of medical imaging analytical methods, providing researchers with a powerful tool to more deeply understand and utilize MRI image data for scientific research and clinical diagnosis. Ultimately, this open-source software package will bring significant technology improvements and applications to the global medical imaging research community.

4. Background material

In the study of modern medical imaging techniques, the importance and universality of magnetic resonance imaging (MRI) has been widely discussed in several studies. For example, through MRI technology, researchers can observe and analyze the internal structure and function of the human body without invasion, which has been emphasized in many papers. For example, Smith et al. (2020) [2] pointed out in his study that MRI plays an irreplaceable role in the diagnosis of neurodegenerative diseases. In addition, the increasing use of MRI technology in tumor localization and treatment monitoring, as mentioned in (Jones 2021) [3], highlights its central position in the field of medical imaging. In the field of microstructure mapping, the application of MRI technology has also received extensive attention. In particular, when it comes to analyzing the distribution of spectral components in images, Lee et al. (2019) [4] detailed in their paper how MRI techniques can be used to analyze the distribution of water molecules in brain tissue to infer the state of brain disease. Accurate acquisition and analysis of these microscopic data is essential for early diagnosis and treatment planning. However, despite the great potential of MRI technology, it still faces some limitations and challenges in practical applications. According to Chaban et al. (2024) [5], traditional MRI analytical tools rely on expensive software licenses and expertise, which limits their application in low-resource environments. In addition, as noted by Slator et al. (2021) [6], existing microstructure mapping techniques often require complex post-processing steps, which not only increases processing time, but may also affect the reproducibility of results. Based on the above literature review, this project proposes to transfer MRI microstructure mapping technology from MATLAB to Python as an open-source platform in order to lower the threshold of technology application and expand its availability on a global scale. Through the development of open-source software packages, we can overcome the limitations of traditional MRI technology and increase its penetration and usefulness in medical research and clinical diagnosis worldwide. This will not only democratize technology, but also provide researchers around the world with a powerful tool for more accurate and efficient disease diagnosis and monitoring.

5. Problem

MRI microstructure mapping technology, traditionally implemented on the MATLAB platform, is limited by high software costs, restricted accessibility, and complex post-processing requirements. These limitations hinder the broad application and adaptability of the technology, particularly in low-resource environments. To overcome these challenges, there is a need to migrate the technology to an open-source, more versatile platform like Python. This migration will enhance the technology's accessibility, reduce costs, and streamline its use in medical imaging research and clinical diagnostics.

6. Approach

In order to solve the problems of high threshold and high cost of MRI microstructure mapping technology in practical application, this project plans to convert the existing MATLAB implementation into an open-source package based on Python. Python was chosen as the development language because of its wide range of applications, strong library support, and excellent community activity, which make Python ideal for research and education applications. At the same time, MATLAB requires a high cost while python, as a free open-source software, provides a lot of application convenience. In this way, we can reduce the cost of use and simplify the operational process, while maintaining or improving the accuracy and efficiency of the technology. This decision, based on an extensive review of existing literature and discussions with domain experts, confirms that open source and Pythonization are the most effective strategies for improving the accessibility and usefulness of the technology. In addition, given Python's strong position in the field of data science and machine learning, this move will also allow MRI data processing to be more closely integrated with advanced machine learning techniques, thus driving further development of medical imaging technology. Through this innovative approach, the project aims to solve the problems of high cost and complex operation, so as to make MRI microstructure mapping technology more popular and contribute to the global health industry.

7. Application of the chosen approach

In order to better complete the task proposed in this paper, we will divide the task into several parts to achieve. The general idea is that we use Ground truth to generate simulated MRI images, and finally use random forest regressors to learn and predict voxel fractions from simulated image data. To better understand the method proposed in this paper, we will further explain the proper terms involved in detail.

7.1 Create and display ground truth maps

First, we use the matplotlib and NumPy libraries to create and display a series of images that show the volume fractions of the four spectral components defined on a 20x20 image. A volume fraction is a unit of measure used to describe the proportion of a component in a mixture or composite material to the overall material volume. In multiphase systems, such as multi-component chemical solutions, biological tissues, or engineered composites, the volume fraction is a key physical quantity because it directly affects the structure, properties, and behavior of the material. For example, in medical imaging analysis, volume scores can help us understand the distribution and concentration of different tissues or substances in the image, allowing accurate evaluation and analysis of disease diagnosis or material properties. For example, in biological tissues, the volume fraction can encode the amount of space taken up by cells in the tissue.

First, we define a tuple called imgdim to represent the size of the image. Here the image is 100x100 pixels and has only one-color channel (viridis). Also create an all-1 array with the same size as imgdim, which can be used as a mask. Define a variable called 'ncomp' with a value of 4, indicating that there are four spectral components. Spectral component refers to the wavelength specific absorption, emission, or reflection properties exhibited by a substance or object in spectral analysis or imaging techniques. In practical applications, each substance has its own unique spectral characteristics, and the performance of these characteristics at different wavelengths can be defined as the spectral components of the substance. For example, in remote sensing, medical imaging or chemical analysis, various substance components can be identified and quantified by analyzing the spectral responses of samples at different wavelengths. The analysis of spectral components is the basis of many scientific research and technical applications, which not only help scientists understand the composition and state of materials, but also support applications in the fields of environmental monitoring, disease diagnosis, and substance identification. By measuring and resolving spectral components, researchers can build detailed spectral maps of substances that can be used in various analytical and diagnostic processes. Specifically, spectral analysis is an important technique in MRI imaging, which shows in the form of two-dimensional graphs how a tissue or substance behaves on two key MRI properties: ADC (apparent diffusion coefficient) and T2*. This spectrogram not only reveals the microstructure and functional properties of different components, but also provides a fine tool for quantifying tissue changes caused by disease, which greatly improves the application value of MRI in clinical diagnosis and disease monitoring.

Then, as shown in Figure 2 we define a vfimg = np.zeros(imgdim (ncomp,)) to indicate a new multidimensional array 'vfimg' whose shape is extended by the tuple 'imgdim' with a new dimension 'ncomp' (4). Thus, 'vfimg' has the shape '(20, 20, 1, 4)' and is used to store the volume fraction of each component at each location. The first and second components of 'vfimg' are then filled with linear gradients from 0 to 0.5, respectively. This is done by generating a vector of a linear space by 'np.linspace' and then extending it to the desired dimension by 'np.tile'. Then calculate the volume fraction of the third component 'vfimg[:, :, :, 2] = np.abs(1 - np.sum(vfimg[:, :, :, 0:2], axis=3, keepdims=True)).squeeze(-1)'， First, calculate the sum of the first two components, and then subtract this sum from 1 to get the remaining fraction. 'keepdims=True' keeps the dimensions unchanged, and 'squeeze(-1)' is used to remove the last dimension because it has only one element. At the same time, we copy the data from the third component to the fourth component. And modify the array of the third and fourth components to contain only their lower and upper trigonometric parts, using the 'np.tril' and 'np.triu' functions. Finally, the volume fraction of all components for each pixel position is normalized, ensuring that their sum is 1 'vfimg /= np.sum(vfimg, axis=3, keepdims=True)'.

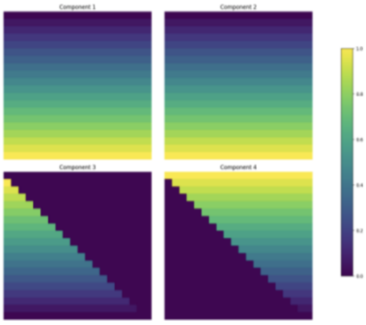
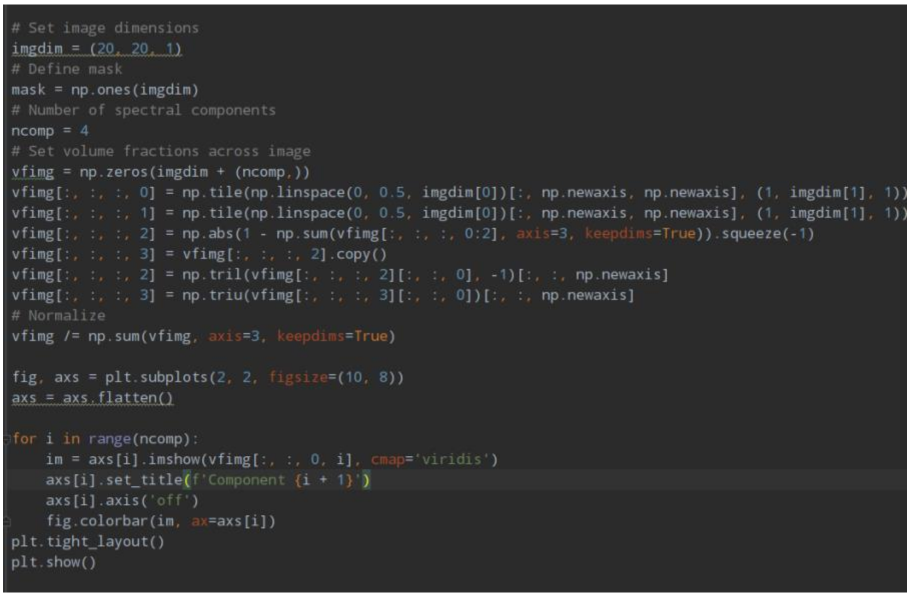


Figure 2. Generated component image vfimg

To display the simulated images, we used the matplotlib library function to create a graph and a 2x2 grid of subgraphs, each of which displays the volume fraction of a component.

The Kernel code is shown in Figure 3:



```
# Set image dimensions
imgdim = (20, 20, 1)
# Define mask
mask = np.ones(imgdim)
# Number of spectral components
ncomp = 4
# Set volume fractions across image
vfimg = np.zeros(imgdim + (ncomp,))
vfimg[:, :, :, 0] = np.tile(np.linspace(0, 0.5, imgdim[0])[:, np.newaxis, np.newaxis], (1, imgdim[1], 1))
vfimg[:, :, :, 1] = np.tile(np.linspace(0, 0.5, imgdim[0])[:, np.newaxis, np.newaxis], (1, imgdim[1], 1))
vfimg[:, :, :, 2] = np.abs(1 - np.sum(vfimg[:, :, :, 0:2], axis=3, keepdims=True)).squeeze(-1)
vfimg[:, :, :, 3] = vfimg[:, :, :, 2].copy()
vfimg[:, :, :, 2] = np.tril(vfimg[:, :, :, 2][:, :, 0], -1)[:, :, np.newaxis]
vfimg[:, :, :, 3] = np.triu(vfimg[:, :, :, 3][:, :, 0])[:, :, np.newaxis]
# Normalize
vfimg /= np.sum(vfimg, axis=3, keepdims=True)

fig, axs = plt.subplots(2, 2, figsize=(10, 8))
axs = axs.flatten()

for i in range(ncomp):
    im = axs[i].imshow(vfimg[:, :, 0, i], cmap='viridis')
    axs[i].set_title(f'Component {i + 1}')
    axs[i].axis('off')
    fig.colorbar(im, ax=axs[i])
plt.tight_layout()
plt.show()
```

Figure 3． Kernel code 1

7.2 Simulate the MRI Imaging process

By combining the theoretical model with real data, the complex process of MRI Imaging is simulated, including signal generation, noise addition and image display. This is very helpful for understanding how different tissue types appear on MRI Images.

First, before going into the implementation details, we need to understand what the T2-D experiment is, which is talking about a magnetic resonance imaging (MRI) technique that combines T2 relaxation time and diffusion coefficient. To make it easier to understand, we can break down this technique into its two core components: T2 relaxation time and diffusion imaging and explore how they work together. T2 relaxation time is a parameter used in MRI to describe the rate at which the water molecular signal decays. In the human body, water molecules are excited by a magnetic field and then gradually return to their equilibrium state. During this process, the magnetization vector of the water molecules becomes out of sync, which leads to the attenuation of the MRI signal. Different biological tissues (such as muscle, fat, or tumors) have different T2 relaxation times due to differences in their physical and chemical environment. Therefore, by measuring T2 relaxation time, doctors can distinguish between different tissue types and diagnose certain disease states. Diffusion imaging is used to observe the movement of water molecules through tissues. In areas with dense cells (such as active tumor areas), the movement of water molecules is more restricted, resulting in a lower diffusion coefficient. In contrast, in regions with lower cell density or looser cell structure, water molecules will diffuse more freely and have a higher diffusion coefficient. By measuring diffusion coefficients in different areas, doctors can obtain information about tissue structure and possible pathological changes. By capturing both T2 relaxation and diffusion data, the T2-D assay provides physicians with a more comprehensive means of analysis. This combination allows for a better interpretation of the physical and biochemical properties of the tissue, thus providing more accurate information for disease diagnosis.

After understanding what the T2-D test is, we stored the two real data in the placenta_gradechoinv.txt file and simulated the generation of MRI images according to the T2 relaxation time and diffusion coefficient. First, we define a function to load data from a file. Secondly, a function KernelMeas is defined to simulate signal formation during magnetic resonance imaging (MRI). This function calculates the signal strength of the voxel based on the physical parameters. It uses an exponential decay model to calculate signal strength by combining the physical parameters ADC (apparent diffusion coefficient) and T2 (transverse relaxation time) with the diffusion weight and echo time of the given data. This calculation reflects how signals decay due to diffusion and relaxation properties during MRI and is critical for predicting and analyzing the imaging performance of different tissues under specific MRI parameters. The specific calculation formula is as follows：

$$\mathrm{S(b,TE)} = \mathrm{e}^{-\mathrm{b}\cdot\mathrm{ADC}}\mathrm{e}^{-\frac{\mathrm{TE}}{\mathrm{T2}}}$$

Where S(b,TE) is the signal strength at a given diffusion weight b and echo time TE. ADC is the apparent diffusion coefficient, which indicates the ability of a substance to diffuse in an MRI scan. T2 is the transverse relaxation time and represents the rate at which the energy of the water molecules in the tissue decays. The specific expansion formula is as follows:

$$\mathrm{S(b,TE)} = \mathrm{e}^{-\mathrm{b}\cdot\mathrm{ADC}^{(1)}}\cdot\mathrm{e}^{-\frac{\mathrm{TE}}{\mathrm{T}_2^{(1)}}} + \mathrm{e}^{-\mathrm{b}\cdot\mathrm{ADC}^{(2)}}\cdot\mathrm{e}^{-\frac{\mathrm{TE}}{\mathrm{T}_2^{(2)}}} + \mathrm{e}^{-\mathrm{b}\cdot\mathrm{ADC}^{3)}}\cdot\mathrm{e}^{-\frac{\mathrm{TE}}{\mathrm{T}_2^{(3)}}} + \mathrm{e}^{-\mathrm{b}\cdot\mathrm{ADC}^{(4)}}\cdot\mathrm{e}^{-\frac{\mathrm{TE}}{\mathrm{T}_2^{(4)}}}$$

Of which:

$$ADC^{(1)} = 0.0002, T_2^{(1)} = 50ms$$

$$ADC^{(2)} = 0.003, T_2^{(2)} = 60ms$$

$$ADC^{(3)} = 0.05, T_2^{(3)} = 70ms$$

$$ADC^{(4)} = 0.3, T_2^{(4)} = 80ms$$

An add noise function is then defined for adding noise to the MRI signal, which is an important step in simulating real imaging conditions. Noise simulation can help researchers understand and predict data quality problems that may be encountered in real-world imaging. The standard deviation of the noise is calculated from the given signal-to-noise ratio (SNR), ensuring that the magnitude of the noise is proportional to the strength of the signal. Next, you define the function process_data to simulate the core of the entire MRI image. It traverses each voxel of the image, computes the corresponding MRI signal, and adds noise to simulate the actual imaging conditions. The signal-to-noise ratio is initially set to 50, which is a key parameter for subsequent noise-adding steps. And initializes simimg an externally pre-allocated array to store the simulated image data. After traversing each voxel, three cycles are used to traverse each voxel of the image, where imgdim contains the dimensional information of the image. And calculate the voxel signal f = vfimg[x, y, z, :]: extract the volume fraction of the current voxel in each spectral component, and traverse each spectral component. kernel['params'] = spectral_comp[:, i]: Sets the physical parameters of the current spectral component (such as ADC and T2 values). S += f[i] * KernelMeas(kernel, gradechoinv): The signal contribution of the current spectral component is calculated and added to the total signal. After the signal is normalized, noise S = add_noise(S, SNR) is added, and the add_noise function is called to add noise to the signal to simulate the signal uncertainty in the actual imaging. Finally, the calculated signal is assigned to the image data array simimg[x, y, z, :] = S: The processed signal is stored in a pre-allocated image array corresponding to the currently traversed voxel. return simimg: After processing is complete, the image array filled with simulated MRI data is returned as shown in Figure 4.

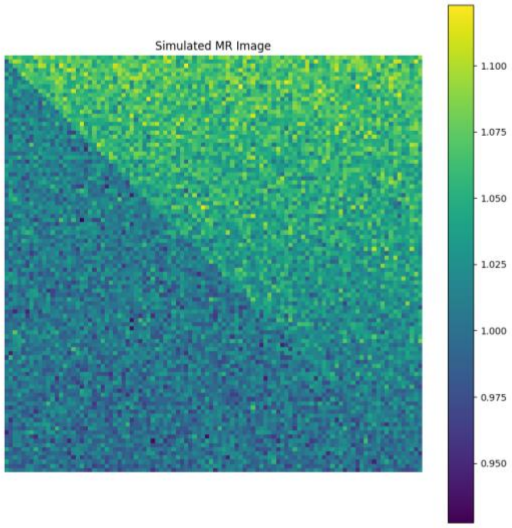
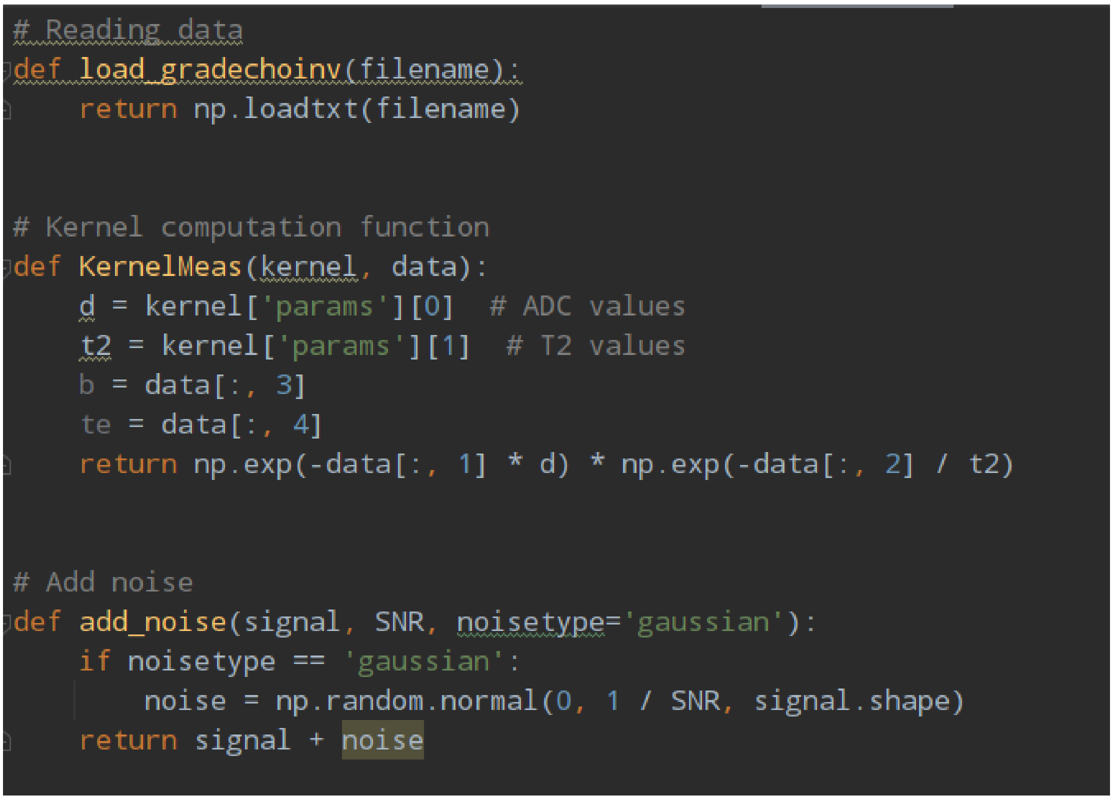


Figure 4. Simulated MRI images

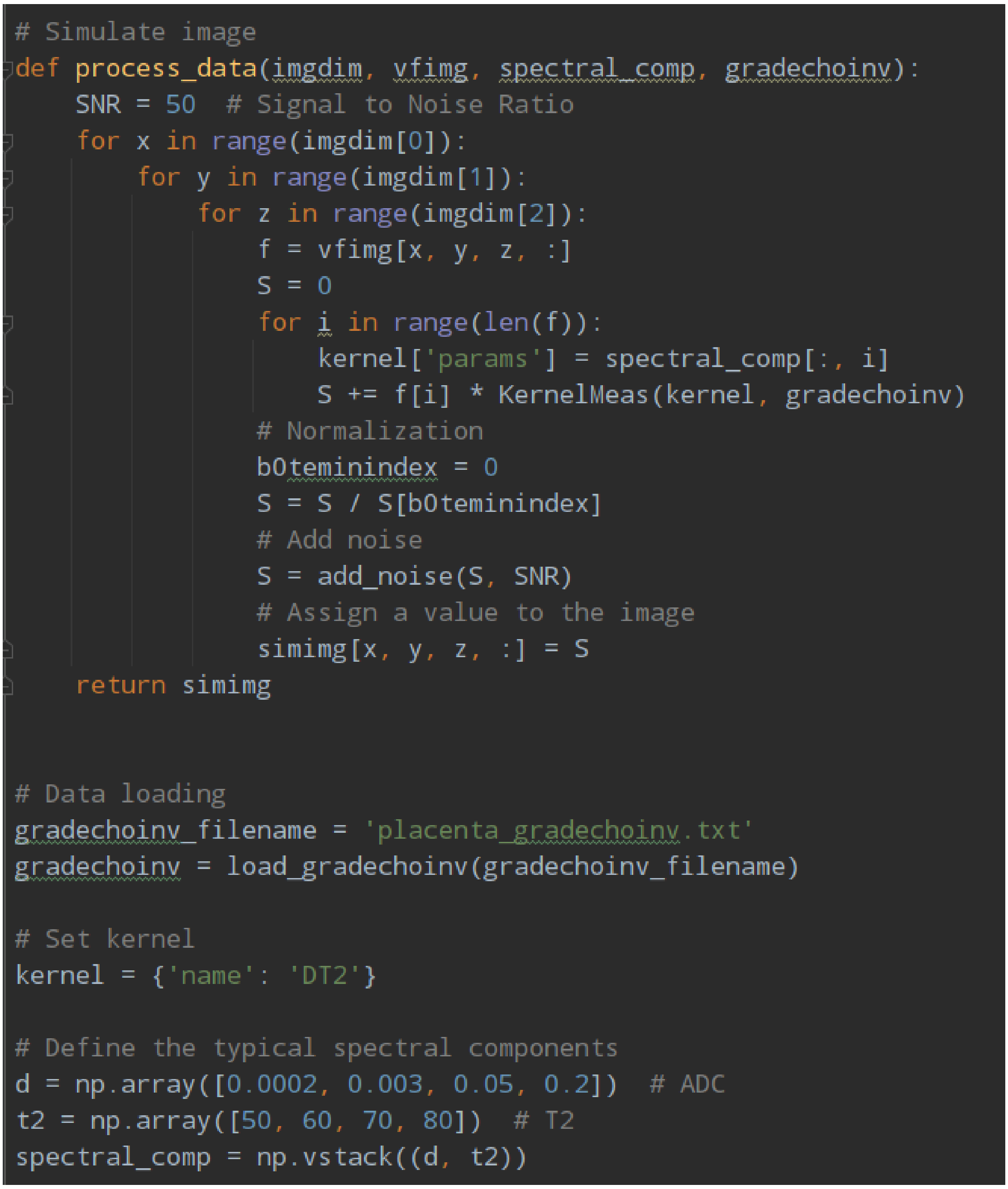
The Kernel code is shown in Figure 5:



```python
# Reading data
def load_gradechoinv(filename):
    return np.loadtxt(filename)


# Kernel computation function
def KernelMeas(kernel, data):
    d = kernel['params'][0]  # ADC values
    t2 = kernel['params'][1]  # T2 values
    b = data[:, 3]
    te = data[:, 4]
    return np.exp(-data[:, 1] * d) * np.exp(-data[:, 2] / t2)


# Add noise
def add_noise(signal, SNR, noisetype='gaussian'):
    if noisetype == 'gaussian':
        noise = np.random.normal(0, 1 / SNR, signal.shape)
    return signal + noise
```



```python
# Simulate image
def process_data(imgdim, vfimg, spectral_comp, gradechoinv):
    SNR = 50  # Signal to Noise Ratio
    for x in range(imgdim[0]):
        for y in range(imgdim[1]):
            for z in range(imgdim[2]):
                f = vfimg[x, y, z, :]
                S = 0
                for i in range(len(f)):
                    kernel['params'] = spectral_comp[:, i]
                    S += f[i] * KernelMeas(kernel, gradechoinv)
                # Normalization
                b0teminindex = 0
                S = S / S[b0teminindex]
                # Add noise
                S = add_noise(S, SNR)
                # Assign a value to the image
                simimg[x, y, z, :] = S
    return simimg


# Data loading
gradechoinv_filename = 'placenta_gradechoinv.txt'
gradechoinv = load_gradechoinv(gradechoinv_filename)

# Set kernel
kernel = {'name': 'DT2'}

# Define the typical spectral components
d = np.array([0.0002, 0.003, 0.05, 0.2])  # ADC
t2 = np.array([50, 60, 70, 80])  # T2
spectral_comp = np.vstack((d, t2))
```

Figure 5. Kernel code 2

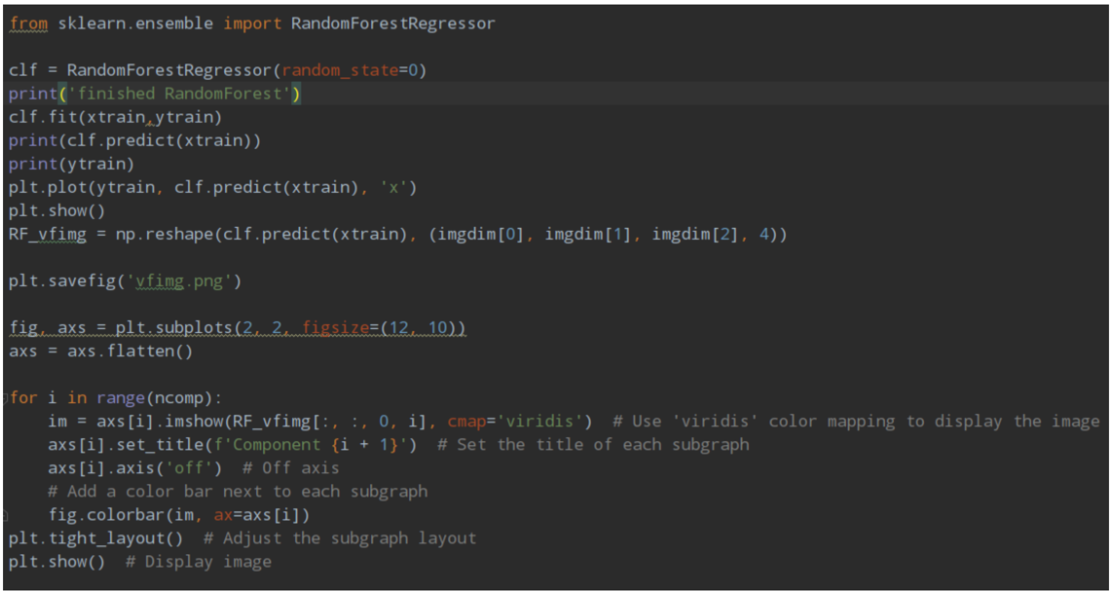
7.3 Machine learning model training and prediction

Finally, we used Random Forest Regressor to learn and predict voxel fractions (vfimg) from simulated image data. Random forest is a decision tree-based ensemble learning method that improves accuracy and controls overfitting by training on multiple decision trees and averaging their predictions.

First, we transform the data, simimg is simulated image data that contains simulated MRI data and reshapes it into a two-dimensional array that can be used as input to a machine learning model. Each row represents one pixel of simulated data. vfimg is the spectral component fraction of each voxel, also reshaped into a two-dimensional array. Each row corresponds to the voxel fraction of the four spectral components of a pixel point. Eventually everything is done and can be reshaped back to get the map and compare this machine learning estimated map to the ground truth.

Then initialize Random Forest Regressor and import the Random Forest Regressor class from the sklearn.ensemble library. Create a Random Forest Regressor instance clf with random_state=0 to ensure repeatability of the results. Random Forest Regressor was trained using the fit method. Here, xtrain contains the input feature (simulated MRI data), while ytrain contains the target variable (spectral component fraction of the voxel). The trained model is used to make predictions on the training data xtrain, which can help evaluate the model's performance on the training set. Finally, a scatter plot is drawn using matplotlib to compare the actual voxel fractions (ytrain) with the predicted voxel fractions. This helps visualize the accuracy and error of the model's predictions. This part of the code reformats the prediction results into the original four-dimensional array format, and then displays the prediction images for each spectral component through subgraphs. Each subgraph has a color bar that helps explain how the color in the image maps to the voxel fractions. With these steps, Random Forest Regressor is used here to understand and predict material composition in MRI images.

The Kernel code is shown in Figure 6:



```python
from sklearn.ensemble import RandomForestRegressor

clf = RandomForestRegressor(random_state=0)
print('finished RandomForest')
clf.fit(xtrain,ytrain)
print(clf.predict(xtrain))
print(ytrain)
plt.plot(ytrain, clf.predict(xtrain), 'x')
plt.show()
RF_vfimg = np.reshape(clf.predict(xtrain), (imgdim[0], imgdim[1], imgdim[2], 4))

plt.savefig('vfimg.png')

fig, axs = plt.subplots(2, 2, figsize=(12, 10))
axs = axs.flatten()

for i in range(ncomp):
    im = axs[i].imshow(RF_vfimg[:, :, 0, i], cmap='viridis')  # Use 'viridis' color mapping to display the image
    axs[i].set_title(f'Component {i + 1}')  # Set the title of each subgraph
    axs[i].axis('off')  # Off axis
    # Add a color bar next to each subgraph
    fig.colorbar(im, ax=axs[i])
plt.tight_layout()  # Adjust the subgraph layout
plt.show()  # Display image
```

Figure 6. Kernel code 3

8. Products

The core product of this project is a full-featured MRI microstructure mapping technology package, developed in Python for the global medical imaging research community. The product has been designed with user friendliness and high accessibility of the technology in mind, aiming to provide medical researchers with a powerful and easy-to-use tool to facilitate the application and development of medical imaging technology. The product has the following characteristics:

1. Open Source: The software package is fully open source, allowing users to freely use, modify, and distribute, thus encouraging technology sharing and innovation worldwide.

2. User-friendly: Provides an intuitive user interface and detailed documentation that makes it easy for even non-programming experts to install, configure, and use the software.

3. Versatility: Supports the processing and analysis of a variety of MRI imaging data, including but not limited to the calculation of voxel fractions, recognition of spectral components, and image reconstruction.

4. Efficiency: Ensures efficient software operation through optimized algorithms,

especially in large data set processing and complex computing tasks.

5. Scalability: The architecture is designed to support future technology expansion and the addition of new capabilities to adapt to growing scientific research needs and emerging medical imaging technologies.

The main application scenarios are as follows:

1. Disease diagnosis: MRI microstructure mapping technology is used to accurately analyze the internal microstructure of the human body, which helps doctors achieve higher accuracy in diagnosing neurodegenerative diseases and tumors.

Scientific research tools: As an important research tool for researchers in the field of medical imaging, it supports detailed data analysis and exploration of new theories.

We provide ongoing technical support and maintain an active user community that facilitates communication and collaboration among users. In addition, the team will work with global research and medical institutions to collect feedback and continuously optimize the product to ensure that it meets the high standards of clinical and research needs. Overall, the product provides important technical support for the global medical imaging research field by providing high-quality MRI microstructure mapping solutions and promotes the further development and popularization of medical imaging technology.

9. Analysis

The final generated experimental effect is shown in the figure, Fig.5 is the generated image of 4 components. The resulting four-dimensional array 'vfimg', where the first three dimensions correspond to spatial coordinates and the fourth dimension corresponds to different spectral components of organizational properties. This setup is typical in multi-parameter imaging (such as MRI), where each voxel may have different properties, represented by these spectral components. Both components 1 and 2 are linear gradients, increasing from 0 to 0.5 over the entire image height. Components 3 and 4 are derived from the remaining volume fraction after considering components 1 and 2 but modified to form a triangular pattern (lower triangle and upper triangle). Each subgraph shows one component in a 20x20 grid, using a "viridis" color map that effectively shows gradients and patterns in the data, often used in scientific visualization to clearly represent magnitude.

The function "KernelMeas" calculates the expected signal based on tissue properties, simulating a typical MRI data generation process. In the absence of real-world data, this is crucial for synthetic data simulations used to train machine learning models. The model predicts the volume fraction (ytrain) from the simulated imaging data (' xtrain '). This regression is common in quantitative imaging studies, where the goal is to learn the mapping from observable MRI signals to underlying tissue properties. After the training is completed, the predicted results are visually compared with real data to assess the accuracy of the model. This visualization is critical to understanding model performance and diagnosing problems such as overfitting or bias.

By displaying the real value and predicted value of the model, the matching degree between the predicted value and the actual value is shown. The ideal predicted value would follow a diagonal line from the top left to the bottom right. The final visual reconstruction volume fraction of the predicted component. Using the prediction results of the model, the reconstructed volume fraction images are displayed. It can be seen from Fig.7 that the prediction of the model is accurate, and these images should preferably be similar to the original volume fraction images. By visually comparing the original component image to the reconstructed image, researchers can assess the effectiveness of the model in capturing complex relationships in the data. This type of analysis is useful in areas such as diagnostic imaging, where understanding the composition of tissue can help in disease diagnosis and treatment planning. Illustrates how complex simulations can be combined with machine learning in medical imaging to develop predictive models that are interpretable and useful in a clinical setting.

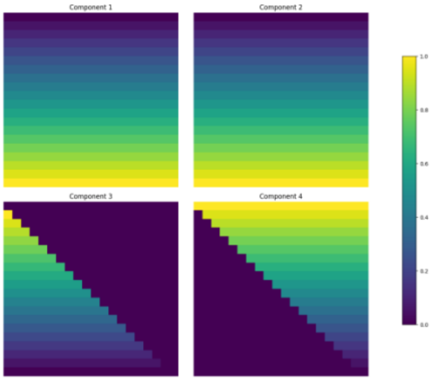


Figure 7. 4 Simulated Ground Truth volume fraction map for components

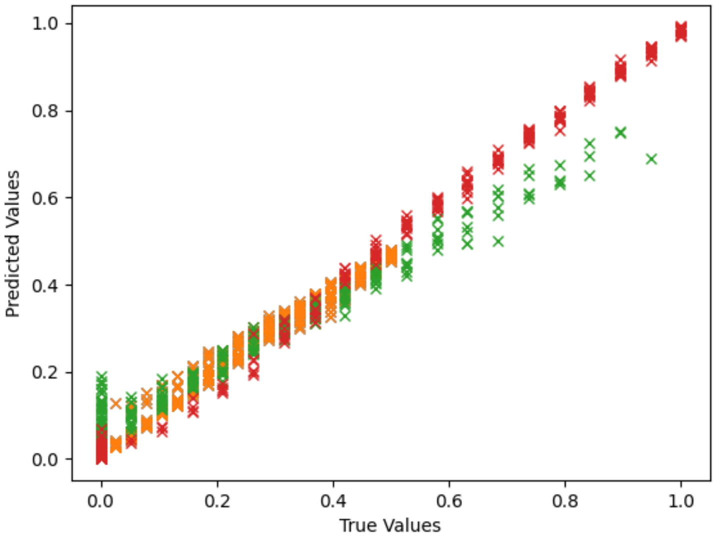


Figure 8. Prediction results and standard values of Random Forest EACH COLOUR REPRESENTS ONE COMPONENT

10. Conclusions

This project prototypes a new python implementation of the InSpect microstructure mapping toolbox. The new implementation uses machine learning instead of the traditional learning of the original InSpect. My machine learning python implementation can accurately estimate the ground truth component map in simulated data, as shown in the figures. After the completion of the project, the platform conversion of MRI microstructure mapping technology has been successfully realized and remarkable progress has been made in several key aspects. First, the migration of technology has made this tool easier to use and scale, and in particular has contributed to the popularization of medical imaging technology on a global scale. Second, integrated machine learning capabilities optimize the image processing process, improving the automation and accuracy of imaging analysis. In addition, through collaboration with the global research community, the project enhances interdisciplinary communication and technology sharing, laying the foundation for future innovation. In short, this project not only improves the practical application efficiency of MRI imaging technology, but also brings a positive impact on the global medical research field.

11. Reflection/Learning

During the implementation of the project, team members worked across disciplines to gain insight into the complexities of software development, algorithm optimization, and medical imaging. During the project, the team faced many challenges, such as migration compatibility of code, technical difficulties in algorithm optimization, and coordination between team members with different backgrounds. These experiences not only exercised the technical abilities of the team, but also improved their problem-solving strategies and the efficiency of teamwork. In addition, interaction with the global research community, particularly the collaboration with Cardiff University's Brain Research Imaging Centre, has greatly broadened the team's horizons and enhanced the in-depth understanding and application of medical imaging techniques. The successful implementation of the project not only improves the individual abilities of the participants, but also has a positive impact on the development of the entire field of medical imaging.

12. References

[1] Slator, P.J., Hutter, J., Marinescu, R.V. et al., 2021. Data-Driven multi-Contrast spectral microstructure imaging with InSpect: INtegrated SPECTral component estimation and mapping. Medical Image Analysis, 71, p.102045.

[2] Smith, S.S., Stewart, M.E., Davies, B.M. et al., 2021. The prevalence of asymptomatic and symptomatic spinal cord compression on magnetic resonance imaging: a systematic review and meta-analysis. Global Spine Journal, 11(4), pp.597-607.

[3] Jones, L.I., Marshall, A., Elangovan, P. et al., 2022. Evaluating the effectiveness of abbreviated breast MRI (abMRI) interpretation training for mammogram readers: a multi-centre study assessing diagnostic performance, using an enriched dataset. Breast Cancer Research, 24(1), p.55.

[4] Lee, H., Nakamura, K., Narayanan, S. et al., 2019. Estimating and accounting for the effect of MRI scanner changes on longitudinal whole-brain volume change measurements. Neuroimage, 184, pp.555-565.

[5] Chaban, Y.V., Vosshenrich, J., McKee, H. et al., 2024. Environmental sustainability and MRI: challenges, opportunities, and a call for action. Journal of Magnetic Resonance Imaging, 59(4), pp.1149-1167.

[6] Slator, P.J., Palombo, M., Miller, K.L. et al., 2021. Combined diffusion-relaxometry microstructure imaging: Current status and future prospects. Magnetic Resonance in Medicine, 86(6), pp.2987-3011.