TECHNISCHE UNIVERSITÄT DARMSTADT

Clinical Applications of Brain Imaging, Stimulation, and Modeling

Exercise 2

Talha Bhatti (2667250)

Contents

List of Figures

1 Questions

1.1 Describe how different parts of the cerebral cortex differ in the number and thickness of cortical layers. (5 points)

More than 90% of the cortical area is covered by neocortex/isocortex, which has a relatively uniform 6-layered structure. However, the primary sensory cortex has a dominant layer IV for input, whereas the motor cortex has dominant output layers III and V. The rest of the cortical surface is occupied by Allocortex, mainly consisting of the hippocampus and the olfactory system, has a traditional 3-layered structure.

1.2 Assuming that concentration of Ca2+ ions outside the cell is 2,000 times greater than inside, what is the Nernst equilibrium potential for Ca2+? (5 points)

We can determine the equilibrium potential for Ca^{2+} using the following equation:

$$E_{Ca^{2+}} = \frac{RT}{zF} \ln \left(\frac{[Ca^{2+}]_{out}}{[Ca^{2+}]_{in}} \right)$$

Where:

$$R = 8.314 J.K^{-1}.mol^{-1}$$

$$T = 310K$$

$$z = 2$$

$$F = 9.65x10^4 C.mol^{-1}$$

$$\frac{[Ca^{2+}]_{out}}{[Ca^{2+}]_{in}} = 2000$$

Plugging these values into the equation, we get:

$$E_{Ca^{2+}} = \frac{(8.314)(310)}{(2)(96485)} \ln(2000)$$

$$E_{Ca^{2+}} \approx 13 \times 7.6 \approx 98.8\,\mathrm{mV}$$

1.3 Given normal intra- and extracellular concentrations of Na+, K+, Cl-, Ca2+ ions, for which of them is the overall electrochemical gradient inwards, and for which outwards? (4 points)

i Na^{+}**:** intracellular concentration $<$ extracellular. Negatively charged inside the cell attracting Na^{+} inwards. Which means the overall electrochemical gradient is inwards.

ii K^{+}**:** intracellular concentration $>$ extracellular. It is negatively charged inside the cell, so the electrical gradient is inwards. However, the concentration gradient drives it out of the cell, making the overall electrochemical gradient outwards.

iii Ca^{2+}**:** intracellular concentration $<<$ extracellular. The electrical gradient is also inwards because the inside of the cell is negatively charged. Overall, the electrochemical gradient is inwards.

iv Cl^{-}**:** intracellular Cl concentration $<$ extracellular. The electrical gradient is inwards because the Cl^{-} charge is attracted inside the cell. But the higher concentration outside the cell dominates, making the overall electrochemical gradient outwards.

1.4 If at rest, some Na+ and K+ channels are open, how are the different concentrations of these ions inside and outside the cell membrane maintained? (3 points)

It is done using the $Na^{+}/K^{+}pump$, which actively moves 3 Na^{+} ions out and 2 K^{+} ions in, maintaining higher Na^{+} concentration outside and higher K^{+} concentration inside. This maintains the ionic gradients necessary for resting potentials.

1.5 What is the function of voltage-gates Na+ channels? (3 points)

The voltage-gates Na^{+} channels are used for the initiation and propagation of action potentials in an excitable cell. When threshold is reached, influx of positive ions takes place by opening Na^{+} channels. This process generates action potential.

1.6 The action potentials of a given cell are characterised by 3 main attributes: amplitude, shape, frequency. Which of these can be influenced by the strength of the signaling input? (3 points)

Since action potentials are all-or-none events, i:e they fire only if the threshold is exceeded, their magnitude is always the same. So, the amplitude doesn't change. Whereas, the shape of the action potential depends on the cell's properties. So, the only attribute of action potentials affected by signaling input strength is the frequency, as a stronger signal causes the cell membrane to reach the threshold often, generating more action potential.

1.7 Name 3 neurotransmitters that are commonly excitatory and 3 that are typically inhibitory (6 points)

Three common excitatory neurotransmitters are:

i Glutamate

ii ACh(acetylcholine)

iii Histamine

Whereas, 3 commonly inhibitory neurotransmitters are:

i GABA (Gamma-aminobutyric acid)

ii Glycine

iii Melatonin

1.8 Which type of imaging is typically used first on patients suspected to have a stroke, and why? (5 points)

Generally, in case of a suspected stroke, Computerized Tomography (CT) is used for initial assessment. It is done by rotating the X-ray source around the patient's head to take multiple tomographic images from different angles. Then 3D reconstruction is performed from multiple tomographic sources. It is relatively quick with the whole scan lasting for 2 - 10 minutes. It can detect hemorrhagic stroke with relatively high frequency, however, may struggle with the more common ischemic stroke, for which advanced techniques like MRI can be used.

1.9 Which type of imaging is seen as more reliable in stroke detection? (2 points)

MRI is typically more reliable for stroke detection, especially for ischemic strokes which have higher uncertainty than CT scans.

1.10 What are contrast agents in MRI and CT, and what are the risks associated with their use? (6 points)

CT scans utilize a radiocontrast, typically iodine and rarely barium sulfate. These agents absorb X-rays, highlighting the regions in which they accumulate. These contrasts can, however, result in allergic reactions, often causing nausea and headaches. On rare occasions, these contrasts can cause kidney damage.

On the other hand, MRI mainly uses Gadolinium-based contrast agents which alter the local magnetic field for better soft tissue visualization. Although gadolinium-based agents are relatively safer than CT contrast agents, they can still cause allergic reactions. Sometimes, they might trigger fibrosis at high doses.

1.11 Which atoms are typically excited in magnetic resonance imaging? (3 points)

MRI uses hydrogen atoms since they are abundant in water and fat. Moreover, most hydrogen atoms have single proton nuclei, which are highly sensitive to the magnetic fields used in MRI.

1.12 Which are the most common image types in MRI and how do they differ? (5 points)

The most common image types in MRI are as follows:

i **T1-weighted:** It is based on longitudinal relaxation with short echo time (TE) and repetition time (TR). In T1-weighted images, fat appears bright due to larger longitudinal and transverse magnetization, while water appears dark. It is mostly used to visualize fatty tissues.

ii **T2-weighted:** It is based on transverse (spin-spin) relaxation with longer TE and TR. In T2-weighted images, fat appears dark and water is bright. It is useful to detect inflammations from fluids.

Two more types of MRI images are Flair and Susceptibility-weighted imaging (SWI). Flair has a very long TE and TR and it is highly sensitive to abnormalities in CSF. SWI is sensitive to magnetic susceptibility differences in tissues. It is good for detecting blood and calcium.

2 Python Exercises

2.1 Among the extracted files, find the file ‘anat.nii.gz’ and load it with the function nibabel.load(). From the loaded image, get both the image data and the header and save them to new variables. (check nibabel online documentation for help). What is the size of the image data in x, y, z dimensions? How many voxels does the image have and what is their size? (3 points)

```
import os
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.patches import Rectangle
import nibabel as nib

data_folder = "NI-edu-master/NI-edu/fMRI-introduction/week_1"
file = os.path.join(data_folder, ’anat.nii.gz’)

imgdata = nib.load(file)
imgdata.dataobj
np.set_printoptions(precision=2, suppress=True)

print(imgdata.affine)

header = imgdata.header

print(header)

img = imgdata.get_fdata()

img_size = header.get_data_shape()
data_type = header.get_data_dtype()
total_voxels = img.size
voxel_size = header.get_zooms()
```

```python
print("Size of image:", img_size)
print("Data type:", data_type)
print("Total number of voxels:", total_voxels)
print("Voxel size:", voxel_size)
```

```
# Output
[[  -1.     -0.03   -0.01  122.24]
 [  -0.03    1.      0.07 -117.46]
 [  -0.     -0.07    1.    -51.11]
 [   0.      0.      0.      1.  ]]
<class ’nibabel.nifti1.Nifti1Header’> object, endian=’<’
sizeof_hdr      : 348
data_type       : np.bytes_(b’’)
db_name         : np.bytes_(b’’)
extents         : 0
session_error   : 0
regular         : np.bytes_(b’r’)
dim_info        : 0
dim             : [  3 240 240 220   1   1   1   1]
intent_p1       : 0.0
intent_p2       : 0.0
intent_p3       : 0.0
intent_code     : none
datatype        : float32
bitpix          : 32
slice_start     : 0
pixdim          : [-1.    1.    1.    1.    0.01  0.    0.    0.  ]
vox_offset      : 0.0
scl_slope       : nan
scl_inter       : nan
slice_end       : 0
slice_code      : unknown
xyzt_units      : 10
cal_max         : 0.0
cal_min         : 0.0
slice_duration  : 0.0
toffset         : 0.0
glmax           : 0
glmin           : 0
descrip         : np.bytes_(b’5.0.10’)
aux_file        : np.bytes_(b’ni-edu’)
qform_code      : scanner
sform_code      : scanner
```

```
quatern_b       : -0.017374652
quatern_c       : 0.9993026
quatern_d       : -0.03294419
qoffset_x       : 122.24499
qoffset_y       : -117.456505
qoffset_z       : -51.112476
srow_x          : [ -1.    -0.03  -0.01 122.24]
srow_y          : [  -0.03    1.      0.07 -117.46]
srow_z          : [ -0.    -0.07   1.    -51.11]
intent_name     : np.bytes_(b'')
magic           : np.bytes_(b'n+1')
Size of image: (240, 240, 220)
Data type: float32
Total number of voxels: 12672000
Voxel size: (np.float32(1.0), np.float32(1.0), np.float32(1.0))
```

2.2 From the image data, create the following 2D plots using the matplotlib function imshow(): a. Sagittal view (slice along x axis) at approximately the middle. (5 points) b. Coronal view (slice along y axis) approximately the middle. (5 points) c. Axial view (slice along z axis) at approximately the middle. (5 points)

```python
x_mid = img.shape[0] // 2
sagittal_view = img[x_mid,:,:].T

y_mid = img.shape[1] // 2
coronal_view = img[:,y_mid,:].T

z_mid = img.shape[2] // 2
axial_view = img[:,:,z_mid].T

plt.figure(figsize=(6, 6))
plt.imshow(sagittal_view, cmap='gray', origin='lower', aspect='auto')
plt.colorbar(label='Intensity')
plt.title("Sagittal View")
plt.xlabel("Y-axis (mm)")
plt.ylabel("Z-axis (mm)")
plt.show()

plt.figure(figsize=(6, 6))
plt.imshow(coronal_view, cmap='gray', origin='lower', aspect='auto')
plt.colorbar(label='Intensity')
plt.title("Coronal View")
plt.xlabel("X-axis (mm)")
plt.ylabel("Z-axis (mm)")
plt.show()

plt.figure(figsize=(6, 6))
plt.imshow(axial_view, cmap=’gray’, origin=’lower’, aspect=’auto’)
plt.colorbar(label=’Intensity’)
plt.title("Axial View")
plt.xlabel("X-axis (mm)")
plt.ylabel("Y-axis (mm)")
plt.show()
```

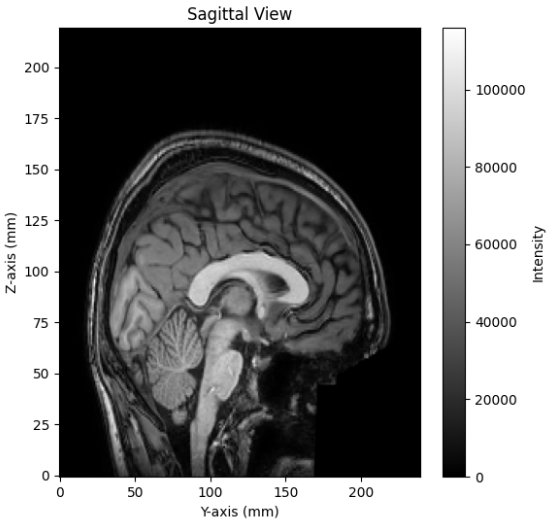

Figure 1: Sagittal View at approximately the middle

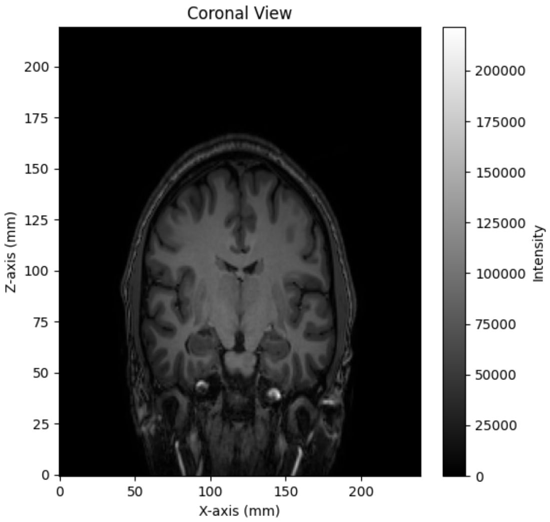

Figure 2: Coronal View at approximately the middle

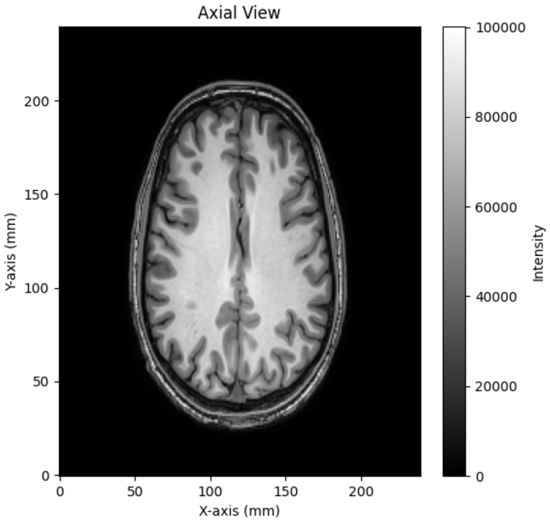

Figure 3: Axial View at approximately the middle

2.3 Plot all these three plots next to each other (horizontally) in a single figure using matplotlib.pyplot.subplots() and title the subplots as ‘sagittal’, ‘coronal’, and ‘axial’. (4 points)

```
ig, axes = plt.subplots(1, 3, figsize=(15, 6))

axes[0].imshow(sagittal_view, cmap=’gray’, origin=’lower’, aspect=’auto’)
axes[0].set_title("Sagittal View")
axes[0].set_xlabel("Y-axis (mm)")
axes[0].set_ylabel("Z-axis (mm)")
plt.colorbar(axes[0].imshow(sagittal_view, cmap=’gray’, origin=’lower’, aspect
    =’auto’), ax=axes[0], label="Intensity")

axes[1].imshow(coronal_view, cmap=’gray’, origin=’lower’, aspect=’auto’)
axes[1].set_title("Coronal View")
axes[1].set_xlabel("X-axis (mm)")
axes[1].set_ylabel("Z-axis (mm)")
plt.colorbar(axes[1].imshow(coronal_view, cmap=’gray’, origin=’lower’, aspect=
    ’auto’), ax=axes[1], label="Intensity")

axes[2].imshow(axial_view, cmap=’gray’, origin=’lower’, aspect=’auto’)
axes[2].set_title("Axial View")
axes[2].set_xlabel("X-axis (mm)")
axes[2].set_ylabel("Y-axis (mm)")
plt.colorbar(axes[2].imshow(axial_view, cmap=’gray’, origin=’lower’, aspect=’
    auto’), ax=axes[2], label="Intensity")
plt.tight_layout()
plt.show()
```

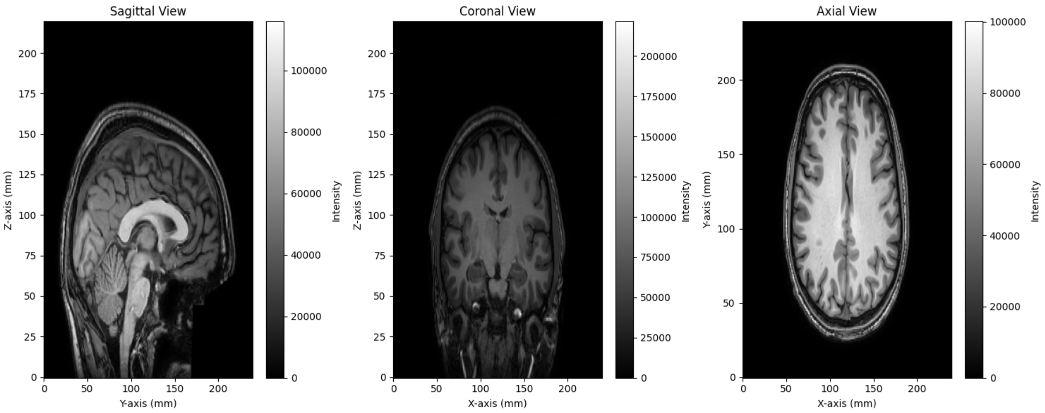

Figure 4: Sagittal (left), Coronal (middle), Axial (right) views of the brain

2.4 Add the following plots: a. Coronal view of the brain in which the cerebellum can be seen. (3 points) b. Axial view of the brain in which the cerebellum can be seen. (3 points)

```python
y_mid_low = img.shape[1] // 4
coronal_cerebellum_view = img[:,y_mid_low,:].T

z_mid_low = img.shape[2] // 4
axial_cerebellum_view = img[:,:,z_mid_low].T

plt.figure(figsize=(6, 6))
plt.imshow(coronal_cerebellum_view, cmap=’gray’, origin=’lower’, aspect=’auto’
    )
plt.colorbar(label=’Intensity’)
plt.title("Coronal View with Cerebellum")
plt.xlabel("Y-axis (mm)")
plt.ylabel("Z-axis (mm)")
plt.show()

plt.figure(figsize=(6, 6))
plt.imshow(axial_cerebellum_view, cmap=’gray’, origin=’lower’, aspect=’auto’)
plt.colorbar(label=’Intensity’)
plt.title("Axial View with Cerebellum")
plt.xlabel("X-axis (mm)")
plt.ylabel("Y-axis (mm)")
plt.show()
```

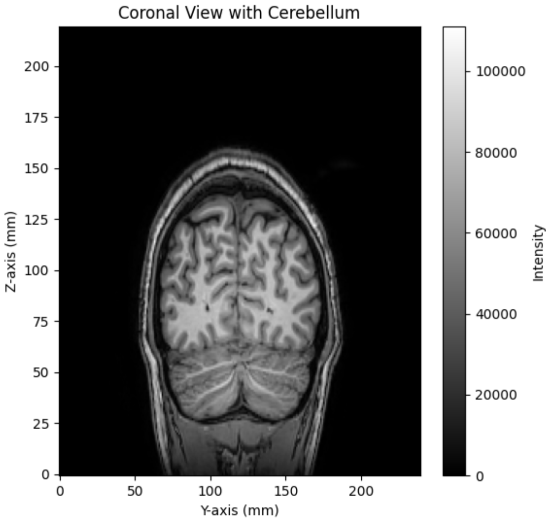

Figure 5: Coronal view in which the cerebellum can be seen

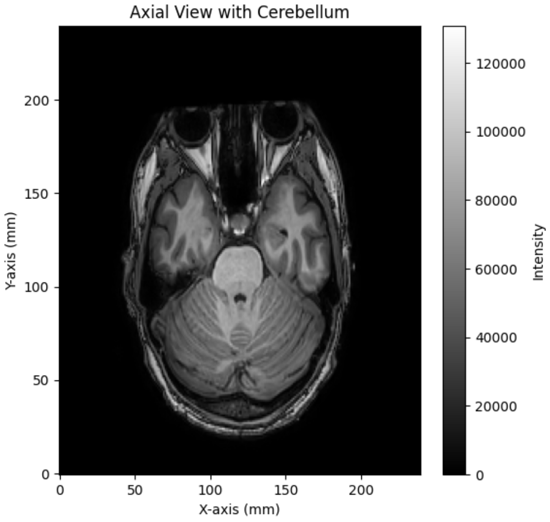

Figure 6: Axial view in which the cerebellum can be seen

2.5 Is this a T1- or T2-weighted scan? Why do you think so? (3 bonus points)

This is a T1-weighted scan because the fat appears brighter and the fluids appears darker in the scan.

2.6 Using numpy, load the data from the file 'sim_brain.npy' (on Moodle). This only has the image data, no metadata. 4. With fig, ax = plt.subplots(), create figure and axis objects (with a single subplot). Create an axial plot of the data at z-index 100. With the patches.rectangle() function from matplotlib, draw red rectangles around the major brain areas that look like they contain lesions. (8 points)

```
sim_brain_data = np.load("sim_brain.npy")
data_shape = sim_brain_data.shape
print("Data shape:", data_shape)

z_index = 100
axial_slice = sim_brain_data[:, :, z_index]

fig, ax = plt.subplots(figsize=(8, 8))
ax.imshow(axial_slice, cmap="gray", origin="lower", aspect="auto")

lession_points = [
    (75, 110,15, 15), #via visual inspection
    (120, 97, 30, 17),
    (74, 59, 18, 10),
    (131, 70, 8, 8),
]

for idx in lession_points:
    x, y, w, h = idx
    rect = Rectangle((x, y), w, h, linewidth=2, edgecolor="red", facecolor="
    none")
    ax.add_patch(rect)

ax.set_title("Axial View with Bounding Boxes around Lesions")
ax.set_xlabel("X-axis")
ax.set_ylabel("Y-axis")
plt.show()
```

```
# Output
Data shape: (181, 217, 181)
```

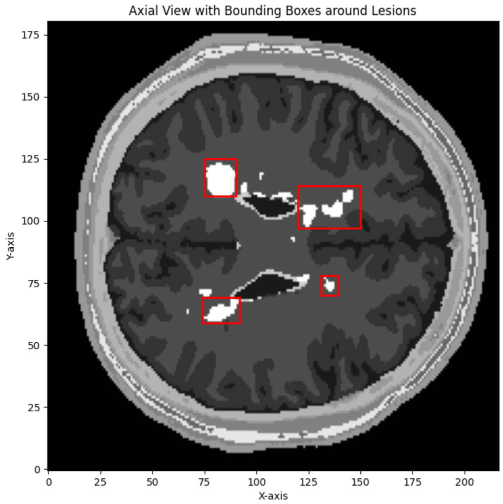

Figure 7: Axial View with Bounding Boxes around Lesions

2.7 Find the file ‘func.nii.gz’ from the extracted folder (see task 1) and load it in the same way as the anatomical dataset. Get the dimensions and units of the data and describe them. (3 points)

```
data_folder = "NI-edu-master/NI-edu/fMRI-introduction/week_1"
file = os.path.join(data_folder, ’func.nii.gz’)

imgdata = nib.load(file)
imgdata.dataobj
np.set_printoptions(precision=2, suppress=True)

print(imgdata.affine)

header = imgdata.header

print(header)

img = imgdata.get_fdata()

img_size = header.get_data_shape()
data_type = header.get_data_dtype()
total_voxels = img.size
voxel_size = header.get_zooms()

print("Size of image:", img_size)
print("Data type:", data_type)
print("Total number of voxels:", total_voxels)
print("Voxel size:", voxel_size)
```

```
# Output
```

```
[[ -2.7   -0.08   0.04 106.96]
 [ -0.08   2.46  -1.22 -74.14]
 [ -0.     1.11   2.71 -68.38]
 [  0.     0.     0.     1.  ]]
<class ’nibabel.nifti1.Nifti1Header’> object, endian=’<’
sizeof_hdr      : 348
data_type       : np.bytes_(b’’)
db_name         : np.bytes_(b’’)
extents         : 0
session_error   : 0
regular         : np.bytes_(b’r’)
dim_info        : 0
dim             : [ 4 80 80 44 50  1  1  1]
intent_p1       : 0.0
intent_p2       : 0.0
intent_p3       : 0.0
intent_code     : none
datatype        : float32
bitpix          : 32
slice_start     : 0
pixdim          : [-1.    2.7   2.7   2.97  0.7   0.    0.    0.  ]
vox_offset      : 0.0
scl_slope       : nan
scl_inter       : nan
slice_end       : 0
slice_code      : unknown
xyzt_units      : 10
cal_max         : 0.0
cal_min         : 0.0
slice_duration  : 0.0
toffset         : 0.0
glmax           : 0
glmin           : 0
descrip         : np.bytes_(b’FSL5.0’)
aux_file        : np.bytes_(b’ni-edu’)
qform_code      : scanner
sform_code      : scanner
quatern_b       : 0.015037585
quatern_c       : -0.9775214
quatern_d       : -0.21027614
qoffset_x       : 106.95868
qoffset_y       : -74.141106
qoffset_z       : -68.38321
```

```
srow_x          : [ -2.7   -0.08   0.04 106.96]
srow_y          : [ -0.08   2.46  -1.22 -74.14]
srow_z          : [ -0.     1.11   2.71 -68.38]
intent_name     : np.bytes_(b’’)
magic           : np.bytes_(b’n+1’)
Size of image: (80, 80, 44, 50)
Data type: float32
Total number of voxels: 14080000
Voxel size: (np.float32(2.7), np.float32(2.7), np.float32(2.97), np.float32(0.7))
```

2.8 Create a figure with subplots and plot the data, axial view sliced at the middle, at 8 evenly-spaced time points, starting from t=0. Here, do not use grey tones, but a multi-color colormap. All plots should have the same mapping of colors to specific values (8 points).

```
z_mid = img.shape[2] // 2
axial_view = img[:,:,z_mid,:].T

time_dimension = img.shape[3]
print("Total time points: ",time_dimension)

time_points = np.linspace(0, time_dimension - 1, 8, dtype=int)

fig, axes = plt.subplots(2, 4, figsize=(16, 8))

intensity_min = axial_view.min()
intensity_max = axial_view.max()

for i, t in enumerate(time_points):
    ax = axes.flat[i]
    im = ax.imshow(img[:, :, z_mid, t].T, cmap=’viridis’, origin=’lower’, vmin
    =intensity_min, vmax=intensity_max)
    ax.set_title(f"Time Point: t={t}")
    ax.set_xlabel("X-axis")
    ax.set_ylabel("Y-axis")

cbar = fig.colorbar(im, ax=axes.ravel().tolist(), shrink=0.9, label="Intensity
    ")

plt.show()
```

```
#Output
Total time points:  50
```

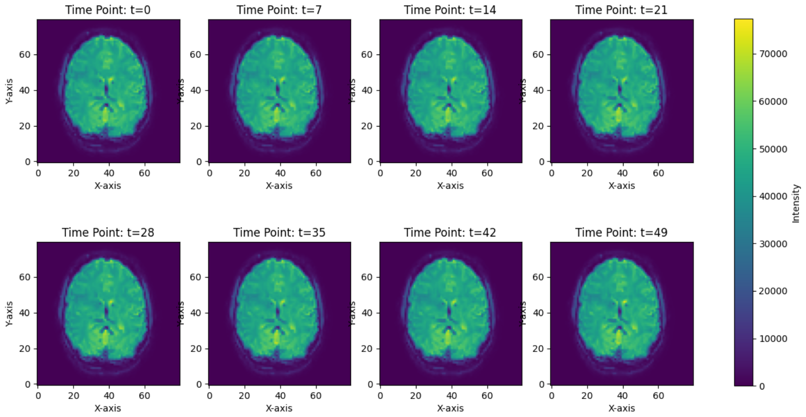

Figure 8: Axial View sliced at the middle at 8 evenly-spaced time points

2.9 From the data slice you plotted for the first subplot (t=0), identify the voxel with the highest value. From the whole dataset, extract the time series for this voxel and plot it as a function of t. Label your x-axis correctly and correctly mark time points as x-axis ticks. (4 points)

```
t0_slice = img[:, :, z_mid, 0]
highest_voxel_index = np.unravel_index(np.argmax(t0_slice), t0_slice.shape)
print("Voxel index with the highest value: ", highest_voxel_index)


time_series = img[highest_voxel_index[0], highest_voxel_index[1], z_mid, :]

plt.figure(figsize=(10, 6))
plt.plot(range(time_dimension), time_series, marker='o', label="Voxel Time
    Series")
plt.title("Time Series for Voxel with the Highest Value at t=0", fontsize=14)
plt.xlabel("Time Point (t)", fontsize=12)
plt.ylabel("Voxel Intensity", fontsize=12)
plt.xticks(range(0, time_dimension, max(1, time_dimension // 10)))
plt.legend()
plt.grid()
plt.tight_layout()
plt.show()
```

```
#Output
Voxel index with the highest value:  (np.int64(41), np.int64(46))
```

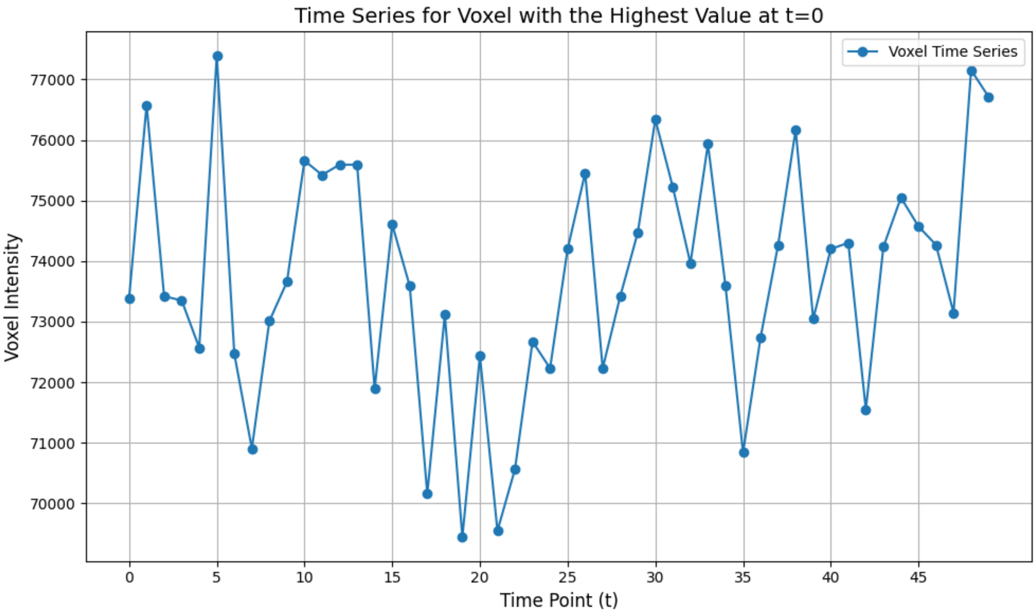

Figure 9: Time Series for Voxel with the Highest Value at $t = 0$