-
-
-
-
-
-

Installing and using Nibabel Python Library

```
from numpy import concatenate, zeros

from matplotlib.pyplot import subplots, tight_layout, show

import nibabel as nib
```

Load images and get data

```
img_nibabel = nib.load("data/T1_mask.nii")

type(img_nibabel)
```

```
<class 'nibabel.nifti1.Nifti1Image'>
```

```
meta_info = img_nibabel.header

print(meta_info)
```

```
<class 'nibabel.nifti1.Nifti1Header'> object, endian='<'
sizeof_hdr      : 348
data_type       : b''
db_name         : b''
extents         : 0
session_error   : 0
regular         : b'r'
dim_info        : 0
dim             : [  3 128 128  70   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.   2.   2.   2.2  0.   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         : b'5.0.11'
aux_file        : b''
qform_code      : scanner
sform_code      : scanner
quatern_b       : 0.0
quatern_c       : 1.0
quatern_d       : 0.0
qoffset_x       : 125.5061
qoffset_y       : -109.38977
qoffset_z       : -86.742615
srow_x          : [ -2.       0.       0.     125.5061]
srow_y          : [  0.         2.         0.      -109.38977]
srow_z          : [  0.         0.         2.2     -86.742615]
intent_name     : b''
magic           : b'n+1'
```

```
print(meta_info.get_xyzt_units())
```

```
('mm', 'sec')
```

```
img1 = img_nibabel.get_fdata()

print(type(img1), img1.shape)
```

```
<class 'numpy.memmap'> (128, 128, 70)
```

```
img_nibabel = nib.load("data/b0_mask.nii")

img2 = img_nibabel.get_fdata()
```

```
img1.shape
```

```
(128, 128, 70)
```

plotting Images

```
img_slice = 30
```

```
fig, ax = subplots(ncols=2, figsize=(15, 5))

f1 = ax[0].imshow(img1[:, :, img_slice], cmap="gray")
f2 = ax[1].imshow(img2[:, :, img_slice], cmap="gray")

fig.colorbar(f1, ax=ax[0])
```

```
<matplotlib.colorbar.Colorbar object at 0x7f44438bf670>
```

```
fig.colorbar(f2, ax=ax[1]);

ax[0].set_xlabel('T1 Image', fontsize=16);
ax[1].set_xlabel('B0 Image', fontsize=16);

show()
```

Data pre-processing

```
img1_slice = img1[:, :, img_slice]
img2_slice = img2[:, :, img_slice]
```

Remove zeros

```
fig, ax = subplots(ncols=2, figsize=(20, 5))

ax[0].hist(img1_slice.flatten(), bins=50)
```

```
(array([1.2374e+04, 3.0000e+00, 6.0000e+00, 1.0000e+01, 1.1000e+01,
       2.4000e+01, 3.2000e+01, 2.5000e+01, 4.3000e+01, 3.5000e+01,
       3.8000e+01, 3.3000e+01, 2.9000e+01, 3.0000e+01, 2.6000e+01,
       4.7000e+01, 4.6000e+01, 6.6000e+01, 5.3000e+01, 6.6000e+01,
       6.8000e+01, 6.4000e+01, 9.7000e+01, 9.5000e+01, 1.2100e+02,
       1.2700e+02, 1.3100e+02, 1.2600e+02, 1.5600e+02, 1.3700e+02,
       1.4000e+02, 1.2900e+02, 1.1900e+02, 1.1900e+02, 1.3100e+02,
       1.4600e+02, 1.5400e+02, 1.4300e+02, 1.7000e+02, 1.2800e+02,
       1.1100e+02, 1.1100e+02, 1.0900e+02, 1.0500e+02, 1.0500e+02,
       1.2600e+02, 9.9000e+01, 5.5000e+01, 4.2000e+01, 2.3000e+01]),
           array([  0.        ,   9.6287677 ,  19.2575354 ,  28.8863031
           ,
        38.5150708 ,  48.1438385 ,  57.7726062 ,  67.4013739 ,
        77.0301416 ,  86.6589093 ,  96.287677  , 105.9164447 ,
       115.5452124 , 125.1739801 , 134.8027478 , 144.4315155 ,
       154.0602832 , 163.6890509 , 173.3178186 , 182.9465863 ,
```

```
        192.575354  , 202.2041217 , 211.8328894 , 221.4616571 ,
        231.0904248 , 240.7191925 , 250.34796021, 259.97672791,
        269.60549561, 279.23426331, 288.86303101, 298.49179871,
        308.12056641, 317.74933411, 327.37810181, 337.00686951,
        346.63563721, 356.26440491, 365.89317261, 375.52194031,
        385.15070801, 394.77947571, 404.40824341, 414.03701111,
        423.66577881, 433.29454651, 442.92331421, 452.55208191,
        462.18084961, 471.80961731, 481.43838501]), <BarContainer object
            of 50 artists>)
```

```
ax[1].hist(img2_slice.flatten(), bins=50);

show()
```

```
mask = (img1_slice>0) & (img2_slice>0)

img1_nz = img1_slice[mask]
img2_nz = img2_slice[mask]

fig, ax = subplots(nrows=1, ncols=2, figsize=(20, 5))

ax[0].hist(img1_nz, bins=50)
```

```
(array([  7.,   9.,  11.,  18.,  35.,  19.,  39.,  33.,  32.,  41.,
    29.,
         25.,  27.,  29.,  52.,  58.,  52.,  48.,  66.,  71.,  57.,
            96.,
         88., 119., 124., 122., 125., 149., 136., 121., 137., 112.,
            113.,
        135., 116., 151., 156., 135., 159., 113., 108., 100., 107.,
           104.,
        109., 114.,  88.,  53.,  38.,  23.]), array([ 17.50374222,
           26.78243507,  36.06112793,  45.33982079,
         54.61851364,  63.8972065 ,  73.17589935,  82.45459221,
         91.73328506, 101.01197792, 110.29067078, 119.56936363,
        128.84805649, 138.12674934, 147.4054422 , 156.68413506,
        165.96282791, 175.24152077, 184.52021362, 193.79890648,
        203.07759933, 212.35629219, 221.63498505, 230.9136779 ,
        240.19237076, 249.47106361, 258.74975647, 268.02844933,
        277.30714218, 286.58583504, 295.86452789, 305.14322075,
        314.4219136 , 323.70060646, 332.97929932, 342.25799217,
        351.53668503, 360.81537788, 370.09407074, 379.3727636 ,
        388.65145645, 397.93014931, 407.20884216, 416.48753502,
        425.76622787, 435.04492073, 444.32361359, 453.60230644,
        462.8809993 , 472.15969215, 481.43838501]), <BarContainer object
            of 50 artists>)
```

```
ax[1].hist(img2_nz, bins=50);

show()
```

Scaling

Standard Scaler: removing the mean and scaling to unit variance

```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

img1_scaled = scaler.fit_transform(img1_nz.reshape(-1, 1))
img2_scaled = scaler.fit_transform(img2_nz.reshape(-1, 1))
```

```
img1_scaled.shape
```

```
(4009, 1)
```

Visualise and Concatenate

Seaborn: https://seaborn.pydata.org

c.f. pair grid example https://seaborn.pydata.org/examples/pair_grid_with_kde.html

kdeplot documentation https://seaborn.pydata.org/generated/seaborn.kdeplot.html

```
fig, ax = subplots(1, 3, figsize=(20, 6))

# Scatter plot
ax[0].scatter(img1_nz, img2_nz)

# 2D Histogram
ax[1].hist2d(img1_nz, img2_nz, bins=50, vmax=10);

from seaborn import kdeplot

# Density Plot
kdeplot(x=img1_nz, y=img2_nz, ax=ax[2]);
show()
```

Scaled Images

```
fig, ax = subplots(1, 3, figsize=(20, 6))

# Scatter plot
ax[0].scatter(img1_scaled[:,0], img2_scaled[:,0])

# 2D Histogram
ax[1].hist2d(img1_scaled[:,0], img2_scaled[:,0], bins=50, vmax=10);
```

```

from seaborn import kdeplot

# Density Plot
kdeplot(x=img1_scaled[:,0], y=img2_scaled[:,0], ax=ax[2]);
show()
```

```
all_img_scaled = concatenate([img1_scaled, img2_scaled], axis=1)

all_img_scaled.shape
```

```
(4009, 2)
```

Segmenting images with Gaussian Mixtures

GMM clustering

```
from sklearn.mixture import GaussianMixture
```

```
n_components = 3

RANDOM_STATE = 12345

gmm = GaussianMixture(n_components=n_components,
                      random_state=RANDOM_STATE)

all_img_labels = gmm.fit_predict(all_img_scaled)

all_img_labels[0]
```

```
2
```

```
fig, ax = subplots(figsize=(8, 8))

ax.scatter(img1_nz, img2_nz, c=all_img_labels, s=100)

ax.set_xlabel('Image 1', fontsize=16)
ax.set_ylabel('Image 2', fontsize=16);

show()
```

```
all_img_labels_mapped = zeros(img1_slice.shape)

all_img_labels_mapped[mask] = all_img_labels
```

```
fig, ax = subplots(figsize=(20, 10))

```

```
ax.imshow(all_img_labels_mapped);

show()
```

Keypoints

-
-
-