Permeability Simulation Using PoreSpy and OpenPNM

In [9]:

```
import numpy as np
import pandas as pd
import openpnm as op
import porespy as ps
import seaborn as sns
import matplotlib.pyplot as plt
import supplementary_code as sc
from skimage import io, color,img_as_ubyte, morphology
import imageio
np.set_printoptions(precision=4)
np.random.seed(10)
%matplotlib inline
```

In [ ]:

```

```

Loading Binary Image

In [10]:

```
Path = "D:/DISSERTATION/CT Data/Berea CT/Berea_2d25um_binary.raw"
```

In [11]:

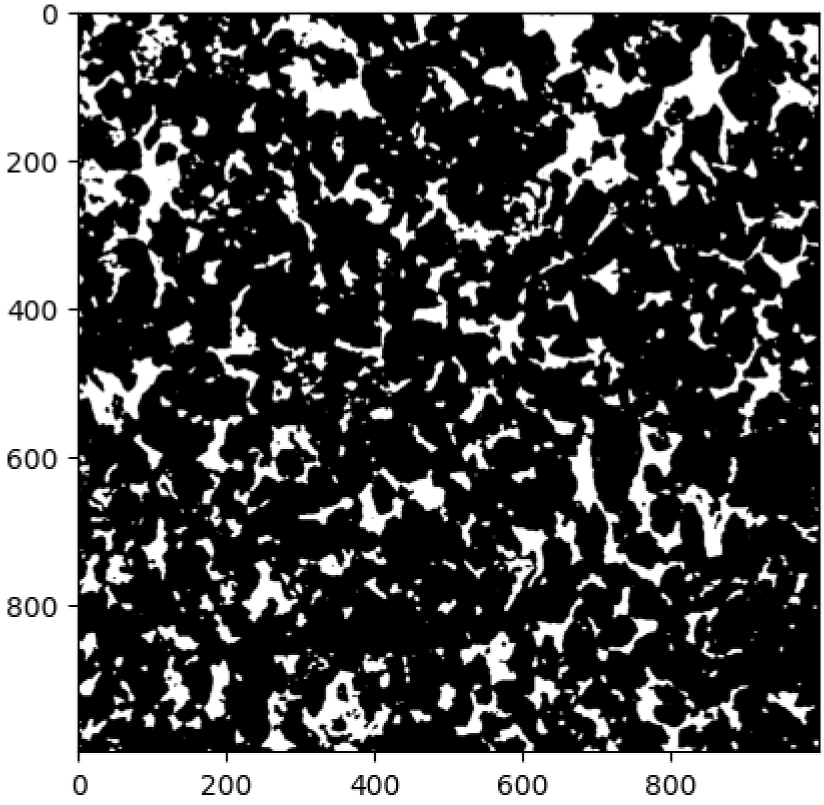
```
resolution = 2.25e-6
name = 'Berea'
# Read input RAW file
raw_file = np.fromfile(Path, dtype=np.uint8)
im = (raw_file.reshape(1000,1000,1000))
```

In [12]:

```
im = im == 0
plt.imshow(im[:,:,23], cmap = 'gray')
```

Out[12]:

```
<matplotlib.image.AxesImage at 0x16145df3450>
```



Porosity, Shape and Dtype

In [31]:

```
print(ps.metrics.porosity(im)*100)
print(im.shape)
print(im.dtype)
```

```
21.671533200000002
(1000, 1000, 1000)
bool
```

Network Extraction PoreSpy Snow2

In [33]:

```
ps_network_snow = ps.networks.snow2(im, voxel_size=resolution)
```

```
0it [00:00, ?it/s]

0it [00:00, ?it/s]

0it [00:00, ?it/s]

0it [00:00, ?it/s]

0it [00:00, ?it/s]

0it [00:00, ?it/s]

0it [00:00, ?it/s]

0it [00:00, ?it/s]

Extracting pore and throat properties:   0%|          | 0/126405 [00:00<?, ?it/s]
```

In [34]:

```
pn = op.io.network_from_porespy(ps_network_snow.network)
net = ps.networks.label_boundaries(network=pn)
```

```
In [35]: print(pn)
```

```
══════════════════════════════════════════════════════════════════════════════
net : <openpnm.network.Network at 0x1eee739c130>
――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――
  #  Properties                                                  Valid Values
――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――
  2  throat.conns                                             205597 / 205597
  3  pore.coords                                              126405 / 126405
  4  pore.region_label                                        126405 / 126405
  5  pore.phase                                               126405 / 126405
  6  throat.phases                                            205597 / 205597
  7  pore.region_volume                                       126405 / 126405
  8  pore.equivalent_diameter                                 126405 / 126405
  9  pore.local_peak                                          126405 / 126405
 10  pore.global_peak                                         126405 / 126405
 11  pore.geometric_centroid                                  126405 / 126405
 12  throat.global_peak                                       205597 / 205597
 13  pore.inscribed_diameter                                  126405 / 126405
 14  pore.extended_diameter                                   126405 / 126405
 15  throat.inscribed_diameter                                205597 / 205597
 16  throat.total_length                                      205597 / 205597
 17  throat.direct_length                                     205597 / 205597
 18  throat.perimeter                                         205597 / 205597
 19  pore.volume                                              126405 / 126405
 20  pore.surface_area                                        126405 / 126405
 21  throat.cross_sectional_area                              205597 / 205597
 22  throat.equivalent_diameter                               205597 / 205597
――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――
  #  Labels                                                 Assigned Locations
――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――
  2  pore.all                                                          126405
  3  throat.all                                                        205597
  4  pore.boundary                                                       9026
  5  pore.xmin                                                           1411
  6  pore.xmax                                                           1462
  7  pore.ymin                                                           1435
  8  pore.ymax                                                           1521
  9  pore.zmin                                                           1426
 10  pore.zmax                                                           1771
 11  pore.left                                                           1411
 12  pore.right                                                          1462
 13  pore.front                                                          1435
 14  pore.back                                                           1521
 15  pore.top                                                            1426
 16  pore.bottom                                                         1771
――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――
```

```
In [ ]:
```

Check network health

Remove isolated pores or cluster of pores from the network by checking it network health.

```
In [40]: h = op.utils.check_network_health(pn)
         print(h)
```

```
――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――
Key                                 Value
――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――
headless_throats                    []
looped_throats                      []
isolated_pores                      []
disconnected_pores                  [0, 4, 5, 6, 7, 12, 16, 20, 25, 35, 51, 87,
129, 130, 131, 137, 145, 157, 167, 168, 180, 184, 200, 205, 215, 218, 229, 233,
242, 251, 252, 260, 268, 270, 272, 282, 304, 311, 314, 343, 355, 357, 391, 410,
413, 426, 427, 434, 437, 440, 453, 477, 487, 494, 511, 520, 537, 566, 592, 599,
608, 627, 634, 647, 651, 658, 727, 736, 767, 777, 803, 855, 858, 867, 869, 883,
894, 898, 899, 918, 930, 956, 957, 997, 1007, 1015, 1030, 1039, 1040, 1044, 106
4, 1080, 1101, 1117, 1118, 1124, 1135, 1136, 1162, 1166, 1178, 1187, 1244, 1271,
1276, 1289, 1333, 1336, 1343, 1349, 1356, 1388, 1389, 1419, 1426, 1429, 1437, 14
45, 1448, 1460, 1471, 1517, 1542, 1544, 1553, 1565, 1582, 1667, 1668, 1680, 169
0, 1701, 1703, 1732, 1735, 1737, 1751, 1755, 1776, 1819, 1838, 1843, 1918, 1946,
1947, 1951, 1966, 1978, 2037, 2044, 2109, 2150, 2152, 2172, 2219, 2232, 2257, 22
60, 2282, 2292, 2336, 2352, 2382, 2393, 2396, 2455, 2468, 2469, 2473, 2474, 247
6, 2480, 2491, 2506, 2560, 2574, 2575, 2580, 2582, 2600, 2646, 2653, 2659, 2751,
```

```
In [41]: op.topotools.trim(network=pn, pores=h['isolated_pores'])
```

```
In [42]: op.topotools.trim(network=pn, pores=h['disconnected_pores'])
```

```
In [43]: h = op.utils.check_network_health(net)
         print(h)
```

```
――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――
Key                                 Value
――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――
headless_throats                    []
looped_throats                      []
isolated_pores                      []
disconnected_pores                  []
duplicate_throats                   []
bidirectional_throats               []
――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――
```

Assign geometry

```
In [44]: pn.add_model_collection(op.models.collections.geometry.spheres_and_cylinders)
         pn.regenerate_models()
```

Assign phase

```
In [45]: # water = op.phase.Water(network=pn)
```

In [46]:

```
phase = op.phase.Phase(network=pn)
phase['pore.viscosity']=1.0
phase.add_model_collection(op.models.collections.physics.basic)
phase.regenerate_models()
```

```
[14:36:49] WARNING  throat.entry_pressure was not run since the following property i
                    'throat.surface_tension'

           WARNING  throat.diffusive_conductance was not run since the following pro
                    missing: 'throat.diffusivity'
```

Apply Stokes Flow

In [47]:

```
inlet = pn.pores('left')
outlet = pn.pores('right')
flow = op.algorithms.StokesFlow(network=pn, phase=phase)
flow.set_value_BC(pores=inlet, values=1)
flow.set_value_BC(pores=outlet, values=0)
flow.run()
phase.update(flow.soln)
```

Calculate effective permeability x - axis

permeability using Darcy’s law:

In [53]:

```
Q = flow.rate(pores=inlet, mode='group')[0]
A = (1000*1000) *resolution**2
L = 1000 * resolution
Delta_P = 1
mu = 1
K = Q * L * mu / (A * Delta_P) # mu and Delta_P were assumed to be 1.
# K = Q * L / A
print(f'The value of K is: {K/0.98e-12*1000:.2f} mD')
# print(K,"mD")
```

```
[14:37:46] WARNING  Attempting to estimate inlet area...will be low

           ERROR    Inlet and outlet faces are different area

           WARNING  Attempting to estimate domain length...could be low if boundary
                    not added

           ERROR    A unique value of length could not be found

The value of K is: 5.28 mD
```

In [ ]:

In [ ]: