Permeability Simulation Using PoreSpy and OpenPNM

In [71]:
```
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
```

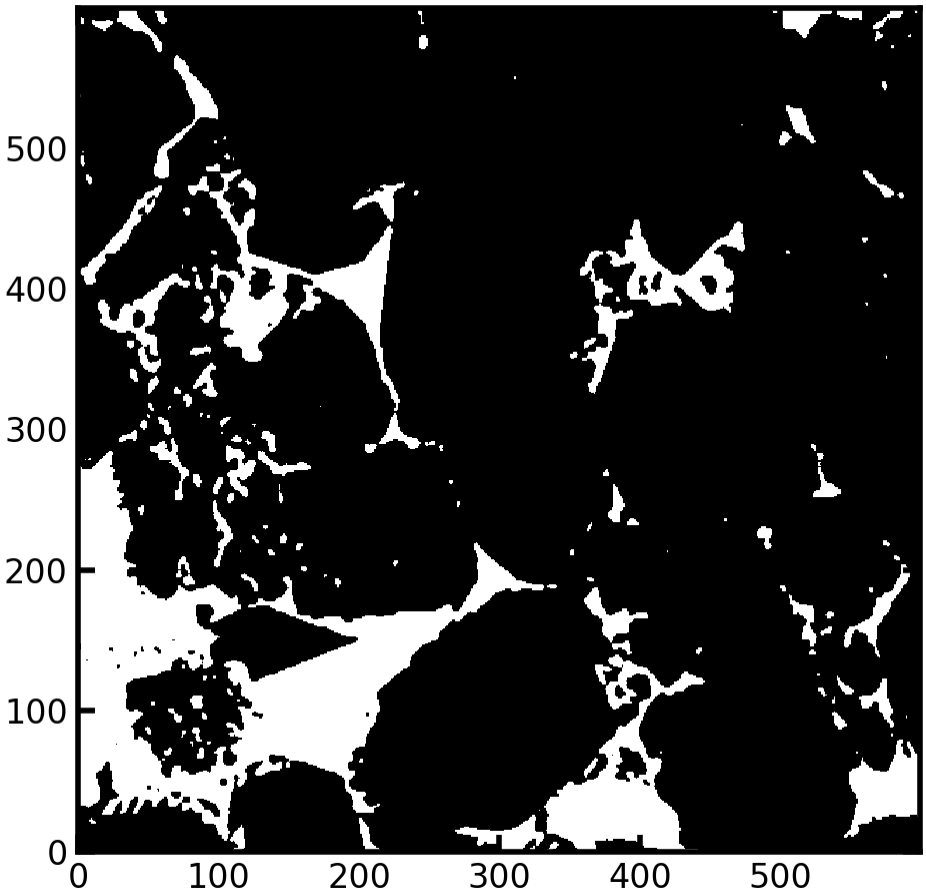
Loading Binary Image

In [73]:
```
resolution = 0.711586e-6
name = 'DS'
im = io.imread("D:/DISSERTATION/CT Data/DS 25/DS_SST_Binary.tif")
```

Confirm image and check image porosity

In [100…
```
io.imshow(im[:,:,100], cmap = 'gray')
```

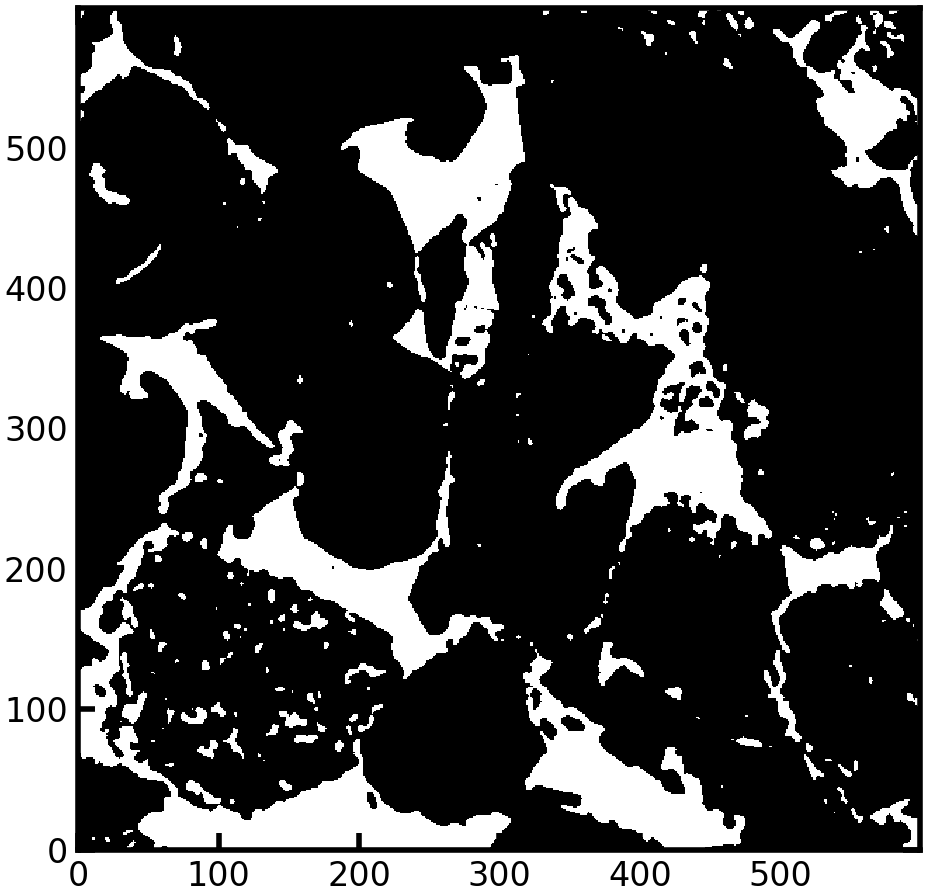
Out[100…
```
<matplotlib.image.AxesImage at 0x28fd66dd990>
```



In [75]:
```
im = im == 0
```

In [76]:
```
io.imshow(im[:,:,1], cmap = 'gray')
```

Out[76]:
```
<matplotlib.image.AxesImage at 0x28fcc952690>
```



Porosity, Shape and Dtype

In [77]:
```
print("Thotal porosity is ",ps.metrics.porosity(im)*100)
print(im.shape)
print(im.dtype)
```

```
Thotal porosity is  17.60148472222222
(600, 600, 600)
bool
```

Network Extraction PoreSpy Snow2

In [78]:
```
snow_output = ps.networks.snow2(im, voxel_size=resolution)
```

In [79]:
```
pn = op.io.network_from_porespy(snow_output.network)
```

In [80]:
```
geo = op.models.collections.geometry.spheres_and_cylinders
```

In [81]:
```
pn.add_model_collection(geo)
pn.regenerate_models()
print(pn)
```

```
net : <openpnm.network.Network at 0x28fcfec7a10>
```

#	Properties	Valid Values
2	throat.conns	27843 / 27843
3	pore.coords	17240 / 17240
4	pore.region_label	17240 / 17240
5	pore.phase	17240 / 17240
6	throat.phases	27843 / 27843
7	pore.region_volume	17240 / 17240
8	pore.equivalent_diameter	17240 / 17240
9	pore.local_peak	17240 / 17240
10	pore.global_peak	17240 / 17240
11	pore.geometric_centroid	17240 / 17240
12	throat.global_peak	27843 / 27843
13	pore.inscribed_diameter	17240 / 17240
14	pore.extended_diameter	17240 / 17240
15	throat.inscribed_diameter	27843 / 27843
16	throat.total_length	27843 / 27843
17	throat.direct_length	27843 / 27843
18	throat.perimeter	27843 / 27843
19	pore.volume	17240 / 17240
20	pore.surface_area	17240 / 17240
21	throat.cross_sectional_area	27843 / 27843
22	throat.equivalent_diameter	27843 / 27843
23	pore.coordination_number	17240 / 17240
24	pore.max_size	17240 / 17240
25	throat.spacing	27843 / 27843
26	pore.seed	17240 / 17240
27	pore.diameter	17240 / 17240
28	throat.max_size	27843 / 27843
29	throat.diameter	27843 / 27843
30	throat.hydraulic_size_factors	27843 / 27843
31	throat.diffusive_size_factors	27843 / 27843
32	throat.lens_volume	27843 / 27843
33	throat.length	27843 / 27843
34	throat.total_volume	27843 / 27843
35	throat.volume	27843 / 27843

#	Labels	Assigned Locations
2	pore.all	17240
3	throat.all	27843
4	pore.boundary	1647
5	pore.xmin	263
6	pore.xmax	212
7	pore.ymin	253
8	pore.ymax	273
9	pore.zmin	392
10	pore.zmax	254

In [88]:
```
h = op.utils.check_network_health(pn)
```

In [89]:
```
op.topotools.trim(network=pn, pores=h['isolated_pores'])
op.topotools.trim(network=pn, pores=h['disconnected_pores'])
```

In [90]:
```
print(h)
```

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

In [91]:

```
net = ps.networks.label_boundaries(network=pn)
```

In [92]:

```
print(net)
```

net : <openpnm.network.Network at 0x28fcfec7a10>

#	Properties	Valid Values
2	throat.conns	26312 / 26312
3	pore.coords	12706 / 12706
4	pore.region_label	12706 / 12706
5	pore.phase	12706 / 12706
6	throat.phases	26312 / 26312
7	pore.region_volume	12706 / 12706
8	pore.equivalent_diameter	12706 / 12706
9	pore.local_peak	12706 / 12706
10	pore.global_peak	12706 / 12706
11	pore.geometric_centroid	12706 / 12706
12	throat.global_peak	26312 / 26312
13	pore.inscribed_diameter	12706 / 12706
14	pore.extended_diameter	12706 / 12706
15	throat.inscribed_diameter	26312 / 26312
16	throat.total_length	26312 / 26312
17	throat.direct_length	26312 / 26312
18	throat.perimeter	26312 / 26312
19	pore.volume	12706 / 12706
20	pore.surface_area	12706 / 12706
21	throat.cross_sectional_area	26312 / 26312
22	throat.equivalent_diameter	26312 / 26312
23	pore.coordination_number	12706 / 12706
24	pore.max_size	12706 / 12706
25	throat.spacing	26312 / 26312
26	pore.seed	12706 / 12706
27	pore.diameter	12706 / 12706
28	throat.max_size	26312 / 26312
29	throat.diameter	26312 / 26312
30	throat.hydraulic_size_factors	26312 / 26312
31	throat.diffusive_size_factors	26312 / 26312
32	throat.lens_volume	26312 / 26312
33	throat.length	26312 / 26312
34	throat.total_volume	26312 / 26312
35	throat.volume	26312 / 26312

#	Labels	Assigned Locations
2	pore.all	12706
3	throat.all	26312
4	pore.boundary	1358
5	pore.xmin	220
6	pore.xmax	160
7	pore.ymin	201
8	pore.ymax	236
9	pore.zmin	317
10	pore.zmax	224
11	pore.left	220
12	pore.right	160
13	pore.front	201
14	pore.back	236
15	pore.top	317
16	pore.bottom	224

Permeability

Defining Phase

In [101…

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

```
[14:07:23] WARNING  throat.entry_pressure was not run since the following property is r
                    'throat.surface_tension'
           WARNING  throat.diffusive_conductance was not run since the following proper
                    missing: 'throat.diffusivity'
```

Apply Stokes flow

In [102…

```
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 permeability

In [105…

```
# NBVAL_IGNORE_OUTPUT
Q = flow.rate(pores=inlet, mode='group')[0]
A = 600*600*resolution**2
L = 600*resolution
# 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')
```

```
The value of K is: 0.33 mD
```

In [ ]: