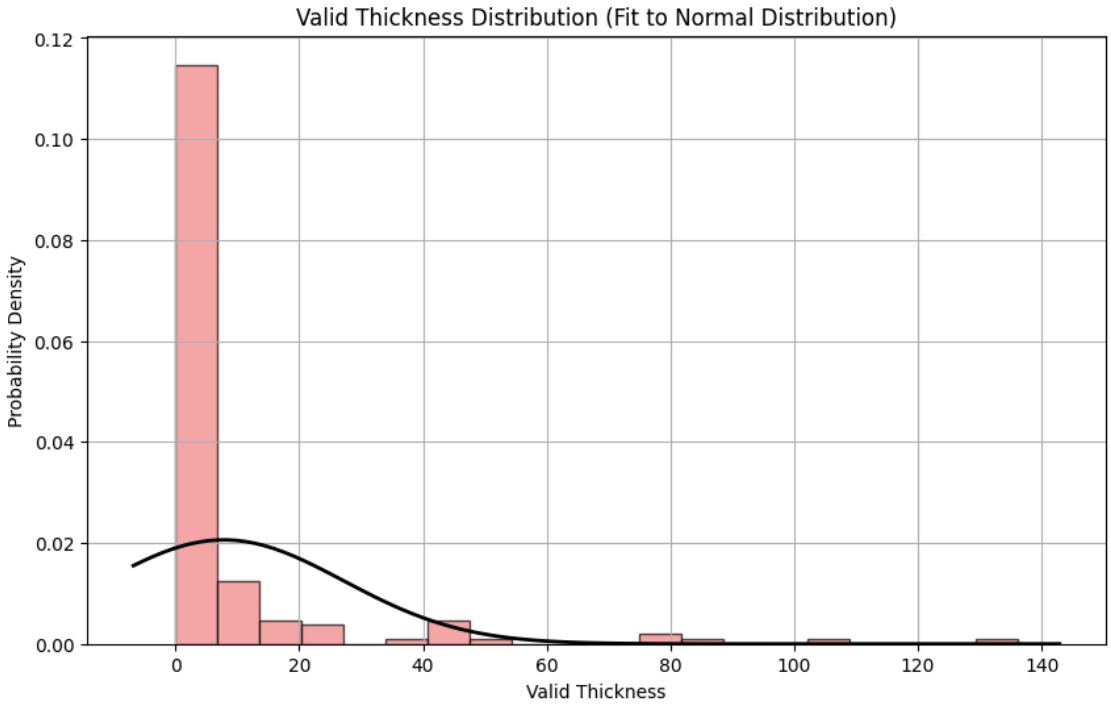
Valid_thickness

May 13, 2024

[4]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import norm

#
dfs = []

#
for i in range(1,15):  #   100
    filename = f"./sensor-data/{i:02d}.tsv"  #       0   99
    df = pd.read_csv(filename, sep='\t')  #
    dfs.append(df)  #

#
merged_df = pd.concat(dfs, ignore_index=True)

#
merged_df = merged_df.iloc[:, :-1]

#
depth = merged_df.iloc[:, 0]
porosity = merged_df.iloc[:, 1]
saturation = merged_df.iloc[:, 2]



#
valid_thickness = []

flag=True

for i in range(len(depth)):
    if abs(porosity[i] * saturation[i]) < 1 and porosity[i] * saturation[i] !=␣
 ↪0:
        if flag:
            first_depth = depth[i]
            flag = False
```

```
    elif flag == False:
        last_depth = depth[i-1]
        valid_thickness.append(last_depth - first_depth)
        flag = True


#
mu, std = norm.fit(valid_thickness)

#
plt.figure(figsize=(10, 6))
plt.hist(valid_thickness, bins=20, density=True, alpha=0.7, color='lightcoral',␣
 ↪edgecolor='black')

#
xmin, xmax = plt.xlim()
x = np.linspace(xmin, xmax, 100)
p = norm.pdf(x, mu, std)
plt.plot(x, p, 'k', linewidth=2)

#
plt.title('Valid Thickness Distribution (Fit to Normal Distribution)')
plt.xlabel('Valid Thickness')
plt.ylabel('Probability Density')
plt.grid(True)
plt.show()
```

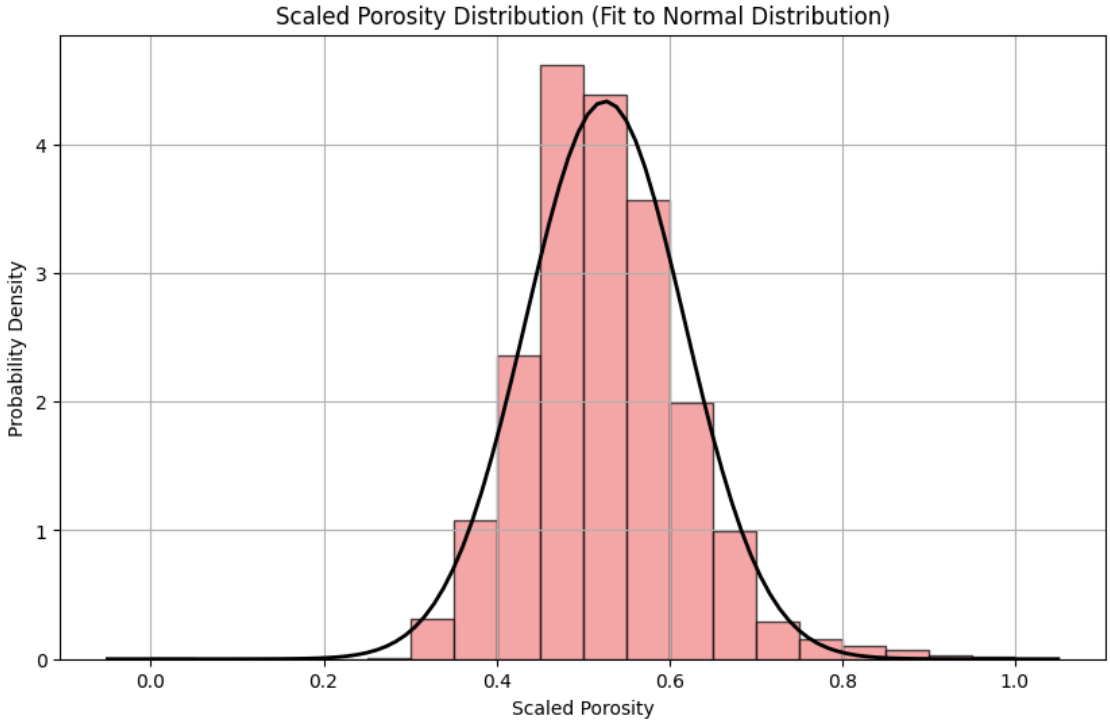


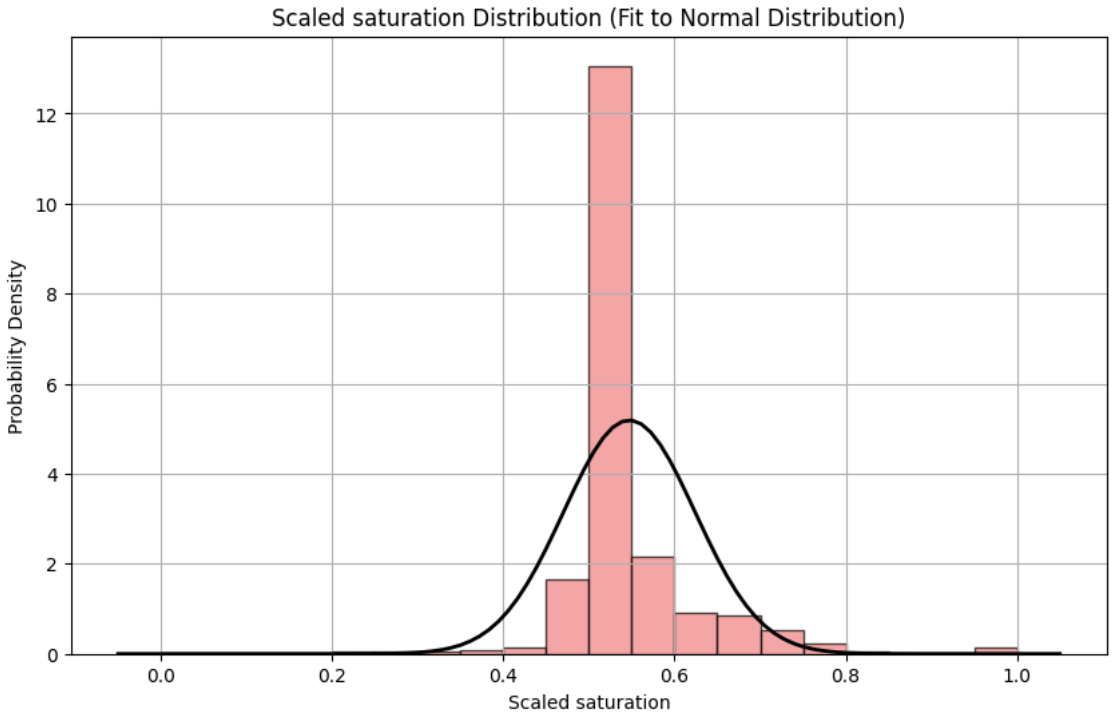
[5]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import norm
from sklearn.preprocessing import MinMaxScaler

#
dfs = []

#
for i in range(1,15):  #   100
    filename = f"./sensor-data/{i:02d}.tsv"  #       0   99
    df = pd.read_csv(filename, sep='\t')  #
    dfs.append(df)  #

#
merged_df = pd.concat(dfs, ignore_index=True)

#
merged_df = merged_df.iloc[:, :-1]

#
depth = merged_df.iloc[:, 0]
porosity = merged_df.iloc[:, 1]
saturation = merged_df.iloc[:, 2]

#     1
filtered_porosity = porosity[abs(porosity) <= 1]
filtered_saturation = saturation[abs(saturation) <= 1]


#
filtered_porosity_2d = filtered_porosity.values.reshape(-1, 1)
filtered_saturation_2d = filtered_saturation.values.reshape(-1, 1)

#  MinMaxScaler
scaler = MinMaxScaler()

#
scaled_porosity = scaler.fit_transform(filtered_porosity_2d)
scaled_saturation = scaler.fit_transform(filtered_saturation_2d)

#
scaled_porosity_1d = scaled_porosity.flatten()
```

```
scaled_saturation_1d = scaled_saturation.flatten()

# #
# print(scaled_porosity_1d)
print(scaled_saturation_1d)


#
mu, std = norm.fit(scaled_porosity_1d)
print(len(scaled_saturation_1d))
#
plt.figure(figsize=(10, 6))
plt.hist(scaled_porosity_1d, bins=20, density=True, alpha=0.7,␣
 ↪color='lightcoral', edgecolor='black')

#
xmin, xmax = plt.xlim()
x = np.linspace(xmin, xmax, 100)
p = norm.pdf(x, mu, std)
plt.plot(x, p, 'k', linewidth=2)

#
plt.title('Scaled Porosity Distribution (Fit to Normal Distribution)')
plt.xlabel('Scaled Porosity')
plt.ylabel('Probability Density')
plt.grid(True)
plt.show()




#
mu, std = norm.fit(scaled_saturation_1d)

#
plt.figure(figsize=(10, 6))
plt.hist(scaled_saturation_1d, bins=20, density=True, alpha=0.7,␣
 ↪color='lightcoral', edgecolor='black')

#
xmin, xmax = plt.xlim()
x = np.linspace(xmin, xmax, 100)
p = norm.pdf(x, mu, std)
plt.plot(x, p, 'k', linewidth=2)

#
plt.title('Scaled saturation Distribution (Fit to Normal Distribution)')
```

```
plt.xlabel('Scaled saturation')
plt.ylabel('Probability Density')
plt.grid(True)
plt.show()
```

```
[0.53934972 0.53605253 0.52674595 … 0.52674595 0.52674595 0.52674595]
15969
```




Scaled saturation Distribution (Fit to Normal Distribution)
Probability Density
0
2
4
6
8
10
12
0.0
0.2
0.4
0.6
0.8
1.0
Scaled saturation

[ ]: