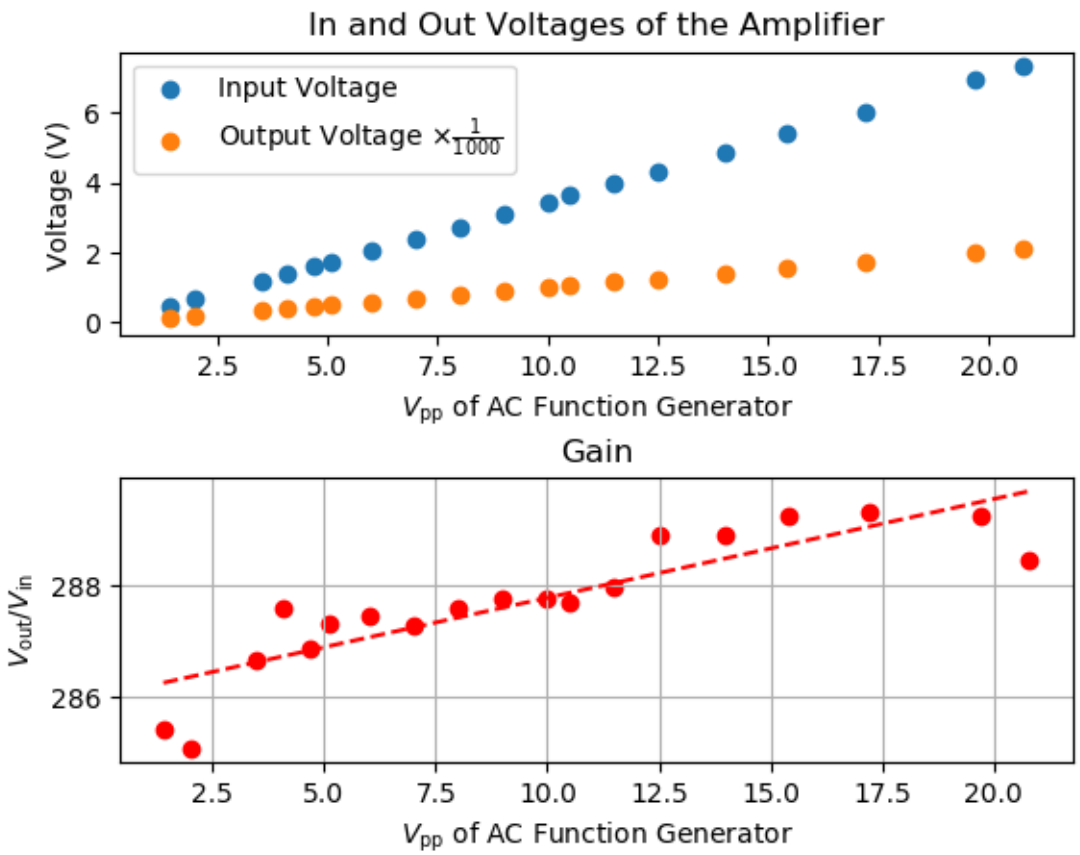
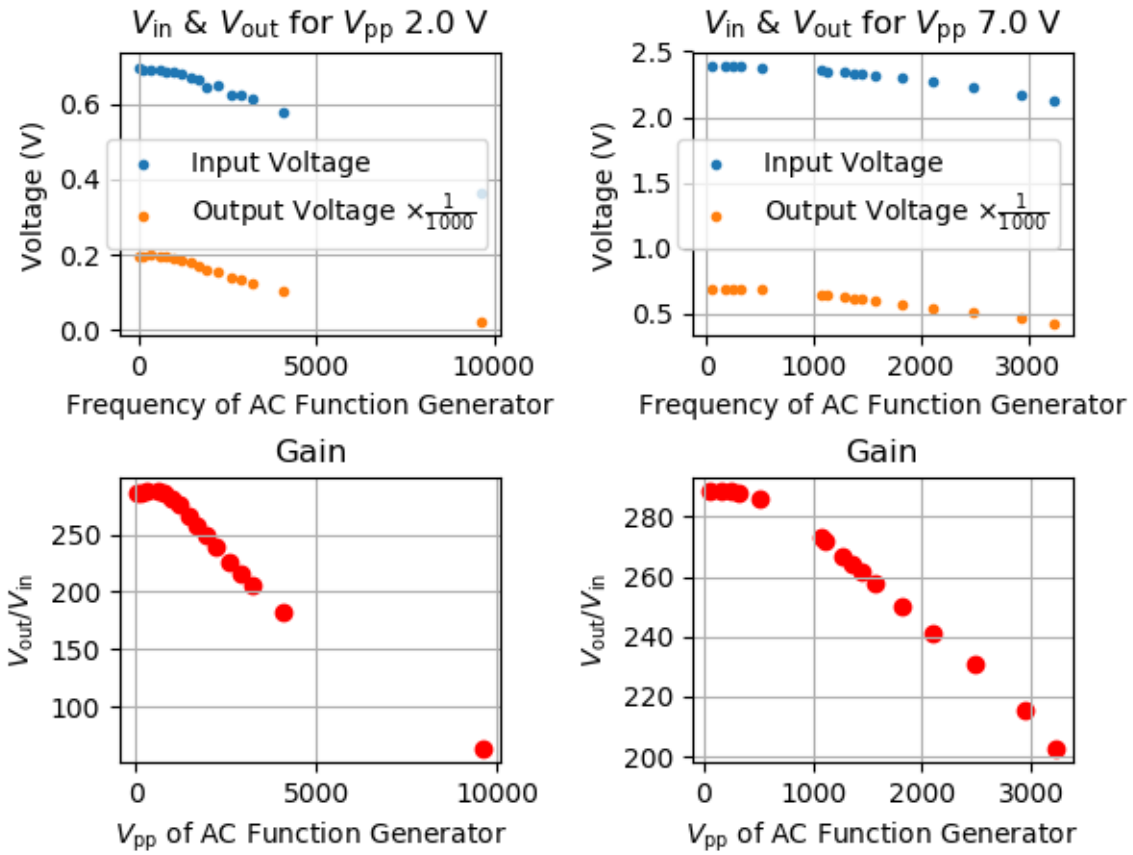
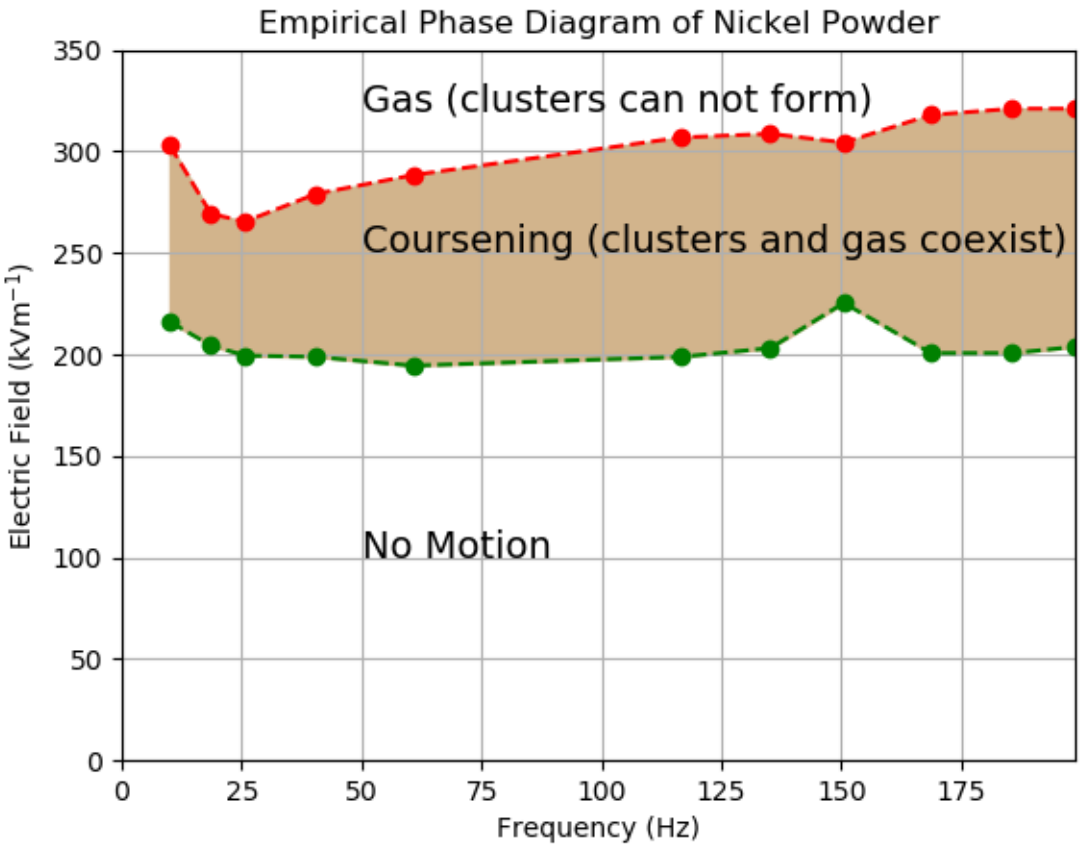


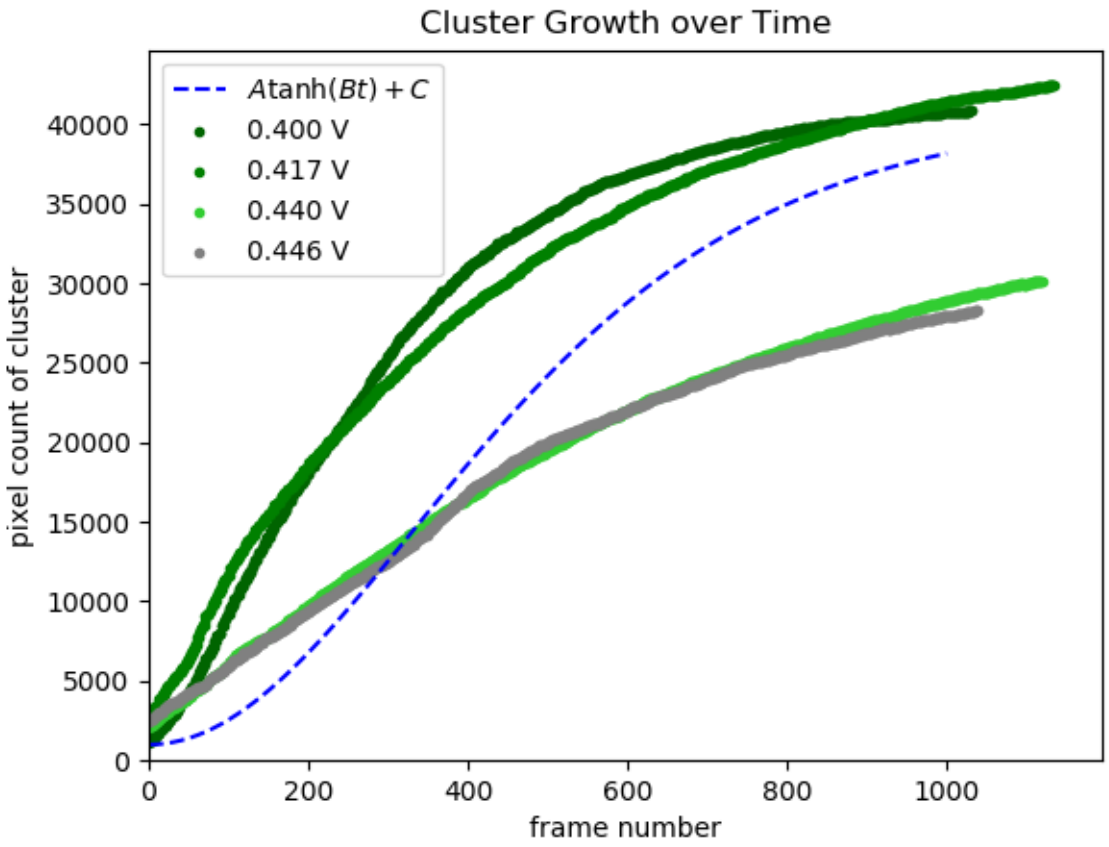

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

def get_trendline(x, y, axes):
    m, b = np.polyfit(x, y, 1)
    xval = np.linspace(np.min(x), np.max(x), 10)
    axes.plot(xval, m*xval+b, color='red', linestyle='--')

gdf = pd.read_csv('va.tsv', sep='\t')
vpp = gdf['Vpp']
vin = gdf['Vin']
vout = gdf['Vout']*1000
gain = vout/gdf['Vin']

fig, ax = plt.subplots(2,1)
plt.subplots_adjust(hspace=0.5)

ax[0].scatter(vpp, vin, label='Input Voltage')
ax[0].scatter(vpp, vout/1000, label=r'Output Voltage $\times \frac{1}{1000}$')
ax[0].set_title('In and Out Voltages of the Amplifier')
ax[0].set_xlabel(r'$V_\mathrm{pp}$ of AC Function Generator')
ax[0].set_ylabel('Voltage (V)')
ax[0].legend()

ax[1].grid(True)
ax[1].scatter(vpp, gain, marker='o', color='red')
ax[1].set_title('Gain')
ax[1].set_xlabel(r'$V_\mathrm{pp}$ of AC Function Generator')
ax[1].set_ylabel(r'$V_\mathrm{out}/V_\mathrm{in}$')
get_trendline(vpp, gain, ax[1])

plt.savefig('./images/va.png')
plt.show()
```



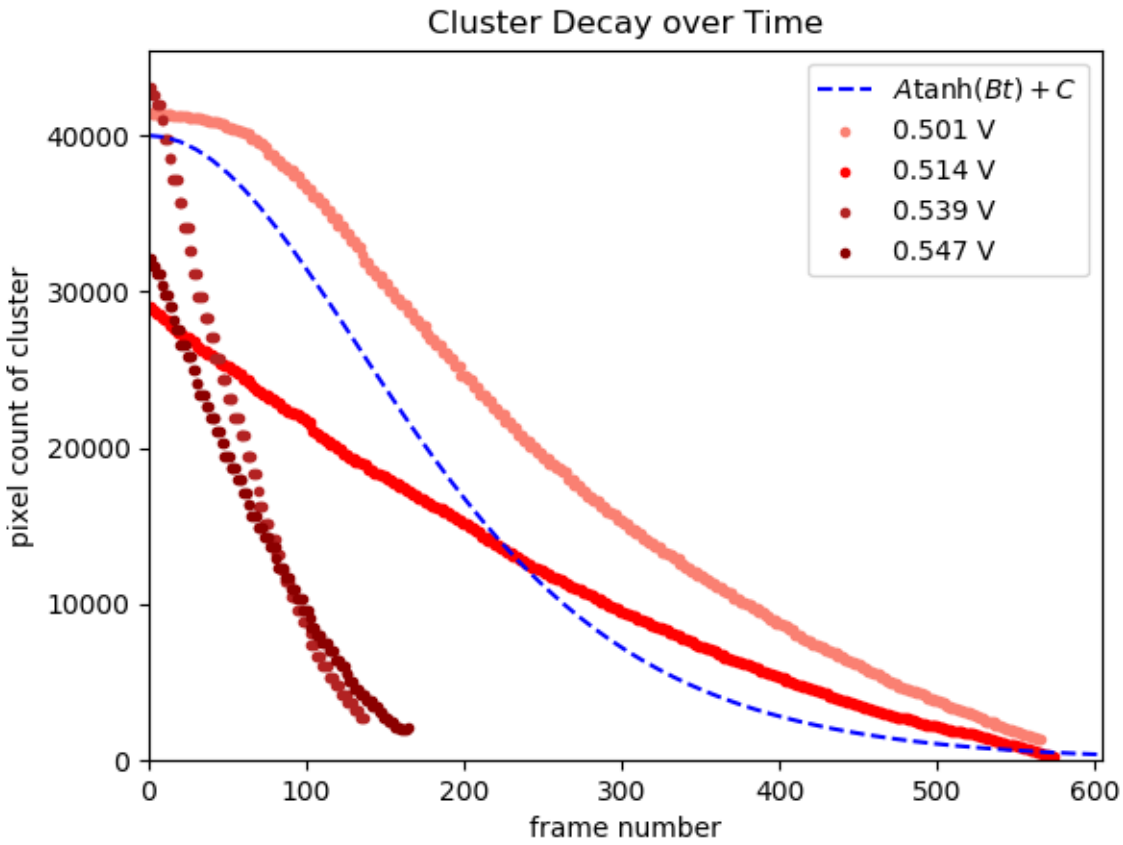

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

fig, ax = plt.subplots(2,2)
plt.subplots_adjust(wspace=0.5, hspace=0.5)

def get_trendline(x, y, axes):
    m, b = np.polyfit(x, y, 1)
    xval = np.linspace(np.min(x), np.max(x), 10)
    axes.plot(xval, m*xval+b, color='red', linestyle='--')

def plots(filename, column, title):
    gdf = pd.read_csv(filename, sep='\t').sort_values('f')
    f = gdf['f']
    vin = gdf['Vin']
    vout = gdf['Vout']*1000
    gain = vout/gdf['Vin']

    ax[0,column].grid(True)
    ax[0,column].scatter(f, vin, label='Input Voltage', marker='.')
    ax[0,column].scatter(f, vout/1000, label=r'Output Voltage $\times \frac{1}{1000}$', marker='.')
    ax[0,column].set_title(title)
    ax[0,column].set_xlabel('Frequency of AC Function Generator')
    ax[0,column].set_ylabel('Voltage (V)')
    ax[0,column].legend()

    ax[1,column].grid(True)
    ax[1,column].scatter(f, gain, marker='o', color='red')
    ax[1,column].set_title('Gain')
    ax[1,column].set_xlabel(r'$V_\mathrm{pp}$ of AC Function Generator')
    ax[1,column].set_ylabel(r'$V_\mathrm{out}/V_\mathrm{in}$')
    #get_trendline(f, gain, ax[1,column])

plots('vf.tsv', 0, r'$V_\mathrm{in}$ & $V_\mathrm{out}$ for $V_\mathrm{pp}$ 2.0 V')
plots('vf2.tsv', 1, r'$V_\mathrm{in}$ & $V_\mathrm{out}$ for $V_\mathrm{pp}$ 7.0 V')

plt.savefig('./images/vf.png')
plt.show()
```




```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

df = pd.read_csv("phased.tsv", sep='\t')
#df = df.append(pd.DataFrame([[0,0,0]], columns=['f', 'v1', 'v2']))
df = df.sort_values('f')
thickness = 1.62*10**(-3)
f = df['f']
E1 = df['v1']/thickness
E2 = df['v2']/thickness

fig, ax = plt.subplots()

ax.grid(True)
ax.plot(f, E1, linestyle='--', marker='o', color='g')
ax.plot(f, E2, linestyle='--', marker='o', color='r')
ax.fill_between(f, E2, E1, color='tan')
ax.set_title('Empirical Phase Diagram of Nickel Powder')
ax.set_xlabel('Frequency (Hz)')
ax.set_ylabel(r'Electric Field (kVm$^{-1}$)')
ax.set_xlim(0, np.max(f))
ax.set_ylim(0, 350)
ax.text(50, 320, 'Gas (clusters can not form)', fontsize=14)
ax.text(50, 250, 'Coursening (clusters and gas coexist)', fontsize=14)
ax.text(50, 100, 'No Motion', fontsize=14)

rho = 8908
a = 75*10**(-6)
g = 9.8
ep0 = 8.854*10**(-12)
k =1.36
tE1 = np.sqrt((rho*a*g)/(3*ep0*k))/1000
print(tE1)
ax.plot(f, (0*f)+tE1, linestyle='--', color='blue')

plt.savefig("images/phased.png")
plt.show()
```




```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

path = "./ratedata/"
g1 = pd.read_csv(path+"g1.tsv", sep='\t')
g2 = pd.read_csv(path+"g2.tsv", sep='\t')
g3 = pd.read_csv(path+"g3.tsv", sep='\t')
g4 = pd.read_csv(path+"g4.tsv", sep='\t')

fig, ax = plt.subplots()
ax.scatter(g3['frame'], g3['area'], marker='.', label='0.400 V', color='darkgreen')
ax.scatter(g4['frame'], g4['area'], marker='.', label='0.417 V', color='green')
ax.scatter(g1['frame'], g1['area'], marker='.', label='0.440 V', color='limegreen')
ax.scatter(g2['frame'], g2['area'], marker='.', label='0.446 V', color='gray')
ax.set_title("Cluster Growth over Time")
ax.set_xlabel("frame number")
ax.set_ylabel("pixel count of cluster")
ax.set_xlim(0)
ax.set_ylim(0)

A=40000
B=2*10**(-3)
C = 1000
label = r'$A\tanh (Bt)+C$'
x = np.linspace(0, 1000, 100)
area = A*(np.tanh(B*x))**2 + C
ax.plot(x, area, linestyle='--', color='blue', label=label)
ax.legend()
plt.savefig("growth.png")
plt.show()
```




```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

path = "./ratedata/"
d1 = pd.read_csv(path+"d1.tsv", sep='\t')
d2 = pd.read_csv(path+"d2.tsv", sep='\t')
d3 = pd.read_csv(path+"d3.tsv", sep='\t')
d4 = pd.read_csv(path+"d4.tsv", sep='\t')[:-120]

fig, ax = plt.subplots()
ax.scatter(d3['frame'], d3['area'], marker='.', label='0.501 V', color='salmon')
ax.scatter(d2['frame'], d2['area'], marker='.', label='0.514 V', color='red')
ax.scatter(d4['frame'], d4['area'], marker='.', label='0.539 V', color='firebrick')
ax.scatter(d1['frame'], d1['area'], marker='.', label='0.547 V', color='darkred')
ax.set_title("Cluster Decay over Time")
ax.set_xlabel("frame number")
ax.set_ylabel("pixel count of cluster")
ax.set_xlim(0)
ax.set_ylim(0)

A=-40000
B=5*10**(-3)
C = 40000
label = r'$A\tanh (Bt)+C$'
x = np.linspace(0, 1000, 100)
area = A*(np.tanh(B*x))**2 + C
ax.plot(x, area, linestyle='--', color='blue', label=label)
ax.legend()

plt.savefig("decay.png")
plt.show()
```

```
import numpy as np
import matplotlib.pyplot as plt
from skimage import (io, measure)

import os

def pixels_of_cluster(filename):
    image = io.imread(filename, as_gray=True)
    image = image[100:-100]
    image = image[:,200:1050]
    image = np.where(image > 0.18, 0, image)
    return np.count_nonzero(image)

path = "./lab2vids/growth4/"
tsvtitle = "g4.tsv"
frames = sorted(os.listdir(path))

with open(tsvtitle, mode='w') as tsv:
    tsv.write('frame\tarea\n')
    pixels = np.array([])
    i=1
    for frame in frames:
        cluster = pixels_of_cluster(path+frame)
        print('{}\t{}'.format(frame,cluster))
        tsv.write('{}\t{}\n'.format(i,cluster))
        i+=1
        pixels = np.append(pixels, cluster)

fig, ax = plt.subplots()
ax.scatter(range(len(pixels)), pixels, marker='.')
plt.show()
```