Preamble

[ ] ↳ 2 cells hidden

Data

Data Import

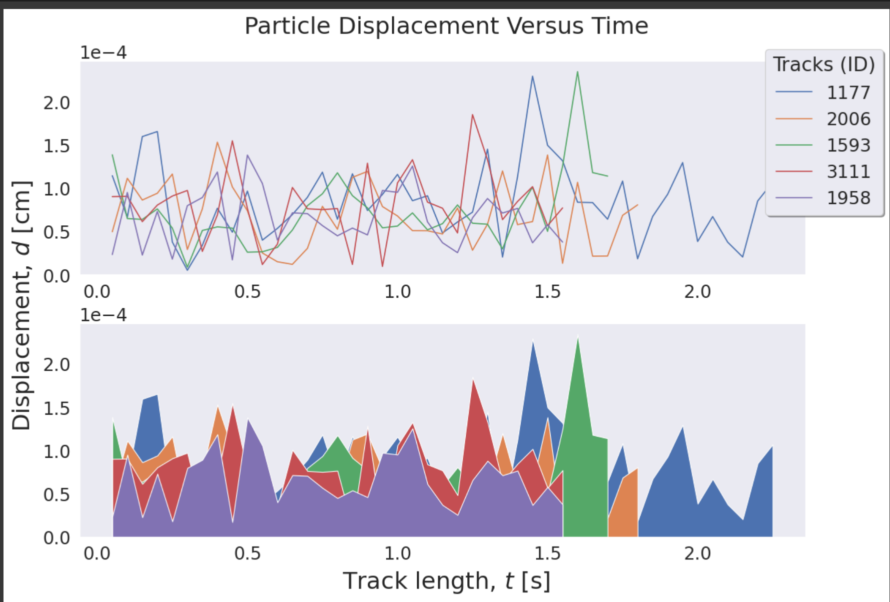
[ ] ↳ 3 cells hidden

Loops and Functions

[ ] ↳ 4 cells hidden

Diffusion Coefficient

```
#data_list = list(zip())

fig, (ax1, ax2) = plt.subplots(2, figsize = (12, 8), sharex=False, sharey=True, layout='constrained')

## List of tracks sorted by size. Length of list determines the readability of the plots
######## Slice the list length by changing the integer i; here [0:i]                     and here [0:i]
zipped_list = zip(np.arange(tracks_by_size.shape[0], dtype='int')[0:5], tracks_by_size.astype('int')[0:5])


# plotting loop
for i,j in zipped_list:  # np.arange(tracks_by_size.shape[0]) # np.arange(0, 5)
  t = np.arange(1, data_links['DISPLACEMENT'].loc[data_links['TRACK_ID'] == tracks_by_size[i]].shape[0]+1)*0.05
  d = np.array(data_links['DISPLACEMENT'].loc[data_links['TRACK_ID'] == tracks_by_size[i]])*6.25*1e-4

  ax1.plot(t, d, label=j)
  ax2.stackplot(t, d)

# combining all plots from the loop
ax1.legend(loc='upper center', bbox_to_anchor=(1.025, 1.1), ncol=1, fancybox=True, shadow=True, title='Tracks (ID)')
fig.supxlabel(r'Track length, $t$ [s]');
fig.supylabel(r'Displacement, $d$ [cm]');
fig.suptitle(r'Particle Displacement Versus Time')
plt.ticklabel_format(style='scientific', axis='y', scilimits=(0,0), useMathText=False)
plt.show()
```




```
# units are wrong; need to be corrected in ImageJ before analysis.
# Good guide from JMU: https://www.jmu.edu/microscopy/resources/basic-image-processing-imagej.pdf
# Also read best practices for data analysis and presentation!

tracks_units
#file_tracks.sort_values(by=['TRACK_DURATION'], ascending=False)
```

	LABEL	TRACK_INDEX	TRACK_ID	NUMBER_SPOTS	NUMBER_GAPS	NUMBER_SPLITS	NUMBER_MERGES	NUMBER_COMPLEX	LONGEST_GA
1	Label	Index	ID	N spots	N gaps	N splits	N merges	N complex	Lgst ga
2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na

2 rows × 28 columns

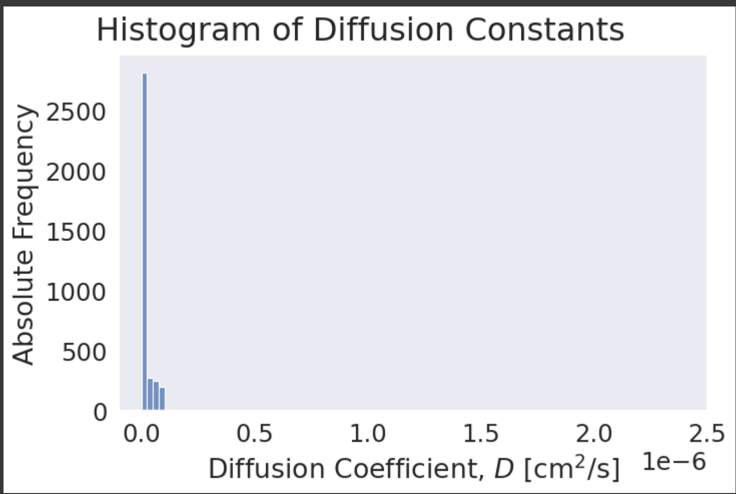
```
# diffusion coefficient

# changing data units
index = file_tracks.index.tolist()
df = pd.DataFrame(file_tracks.TRACK_ID)
df['TRACK_DURATION'] = file_tracks.TRACK_DURATION    #*0.05 # has been calibrated # 50 ms/frame = 0.05 s/frame (20.0 frames/s)
```

```
df['TRACK_DISPLACEMENT'] = file_tracks.TRACK_DISPLACEMENT*6.25  # 512px / 81.92e-6 m = 6.25 px/µm; 0.16 µm/px (?)
df['TRACK_DISPLACEMENT'] = df['TRACK_DISPLACEMENT']*1e-4  # 1 µm = 1e-4 cm

# r-squared and D
df['r2'] = (df.TRACK_DISPLACEMENT ** 2)
df['D'] = ( (df.TRACK_DISPLACEMENT ** 2) / ( 4 * df.TRACK_DURATION) )

# interpretting results: https://www.comsol.com/multiphysics/diffusion-coefficient
# In an aqueous (water) solution, typical diffusion coefficients are in the range of 1e-10 to 1e-9 m2/s
df
```

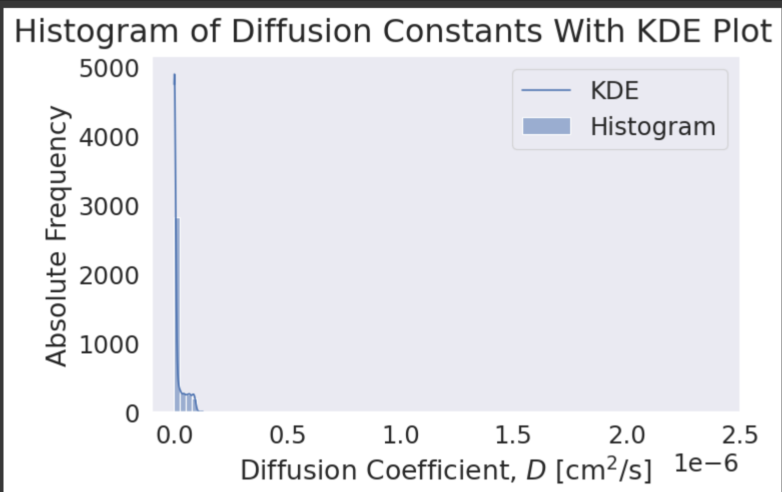
	TRACK_ID	TRACK_DURATION	TRACK_DISPLACEMENT	r2	D
3	0.0	5.0	0.000085	7.213200e-09	3.606600e-10
4	1.0	2.0	0.000078	6.098476e-09	7.623094e-10
5	2.0	1.0	0.000389	1.510451e-07	3.776128e-08
6	3.0	6.0	0.000270	7.276587e-08	3.031911e-09
7	4.0	3.0	0.000096	9.169516e-09	7.641263e-10
...	...	...	...	...	...
3527	3524.0	1.0	0.000318	1.009644e-07	2.524110e-08
3528	3525.0	1.0	0.000541	2.931412e-07	7.328529e-08
3529	3526.0	1.0	0.000398	1.586524e-07	3.966309e-08
3530	3527.0	1.0	0.000049	2.373721e-09	5.934304e-10
3531	3528.0	1.0	0.000168	2.811939e-08	7.029847e-09

3529 rows × 5 columns

```
# histogram using Seaborn + matplotlib

plot = sns.displot(data=df, x="D", kind="hist", kde=False, bins = 5, aspect = 1.5, legend=True)
plot.figure.subplots_adjust(top=0.9);
plt.xlim(-0.1e-6, None)
plot.figure.suptitle("Histogram of Diffusion Constants");
plot.set(xlabel=r'Diffusion Coefficient, $D$ $\left[ \mathrm{cm}{^2}/\mathrm{s}{} \right]$', ylabel='Absolute Frequency', xlim=(None, 2.5e-6));
```




```
# histogram + kernel density estimate (KDE) plot

plot = sns.displot(data=df, x="D", kind="hist", kde=True, bins = 5, aspect = 1.5)
plot.figure.subplots_adjust(top=0.9);
plot.figure.suptitle("Histogram of Diffusion Constants With KDE Plot");
plt.xlim(-0.1e-6, None)
plot.set(xlabel=r'Diffusion Coefficient, $D$ $\left[ \mathrm{cm}{^2}/\mathrm{s}{} \right]$', ylabel='Absolute Frequenc
plt.legend(labels=["KDE","Histogram"]); # kernel density estimate (KDE) plot
```




```
# adding label to the df
df['Data_Series'] = 'pH 9 #04'

```

```
4 D_statistics = df.groupby(['Data_Series'])['D'].describe() #[['mean','std']]
5 D_statistics
```

Data_Series	count	mean	std	min	25%	50%	75%	max
pH 9 #04	3529.0	1.458846e-08	2.529903e-08	1.884294e-13	3.071419e-10	1.523747e-09	1.401826e-08	1.270691e-07

Michaelis-Menten Kinetics

pH 9.0

[ ] ↳ *16 cells hidden*

Michaelis-Menten Kinetics

pH 7.8

[ ] ↳ *5 cells hidden*

Code Snippets From Workshop

[ ] ↳ *6 cells hidden*

Out of Scope

[ ] ↳ *21 cells hidden*