

DO_Commissioning

September 1, 2020

1 This notebook contains the code for plotting the Dissolved Oxygen Commissioning plots

1.0.1 Two new dissolved oxygen instruments were purchased from Scripps - these new instruments were subsequently taken on in2020_e01 for commissioning. The data below was collected on the voyage where all 3 instruments (2 new, 1 old) were setup in the same laboratory space and operated in parallel.

Imports and style sets:

```
In [2]: import pandas as pd
import matplotlib.pyplot as plt
import matplotlib as mpl
import seaborn as sns
import seawater as sw
import scipy.stats as sci_st

sns.set(style="whitegrid") # I like this
mpl.rc('font', family='serif') # Cast serif as the font
mpl.rc('figure', figsize=[8, 5]) # Set fig size to something more fitting for A4 word
```

1.0.2 Variables for locations of datafiles (in same order as headings)

```
In [3]: INDEPENDENT_IODATE_DATA = 'https://raw.githubusercontent.com/kendall-s/do_commissioning/master/iodate_data.csv'
DEP_1_DEEP_REPLICATES_SINGLE_NISKINS_DATA = 'https://raw.githubusercontent.com/kendall-s/do_commissioning/master/deep_replicates_single_niskins_data.csv'
DEP_1_DEEP_REPLICATES_SHARED_NISKINS_DATA = 'https://raw.githubusercontent.com/kendall-s/do_commissioning/master/deep_replicates_shared_niskins_data.csv'
ATMOSPHERIC_DIFF_INSTRUMENTS_DATA = 'https://raw.githubusercontent.com/kendall-s/do_commissioning/master/atmospheric_diff_instruments_data.csv'
ATMOSPHERIC_ONE_INSTRUMENT_DATA = 'https://raw.githubusercontent.com/kendall-s/do_commissioning/master/atmospheric_one_instrument_data.csv'
PROFILE_COMPARISON_DATA = 'https://raw.githubusercontent.com/kendall-s/do_commissioning/master/profile_comparison_data.csv'
DEP_2_DEEP_REPLICATES_DATA = 'https://raw.githubusercontent.com/kendall-s/do_commissioning/master/deep_replicates_data.csv'

COMBINED_DATA = 'https://raw.githubusercontent.com/kendall-s/do_commissioning/master/combined_data.csv'
```

1.1 3.1 Independent Iodate Standards

```
In [4]: iodate_df = pd.read_csv(INDEPENDENT_IODATE_DATA)
```

```
In [5]: iodate_df.head()
```

```
Out[5]:
```

	Instrument	Bottle	FlaskVol	RawTitre	Titre20	O2ml/L	ThioTemp	DrawTemp	\
0	New A	200	142.19	0.51524	0.51529	4.964	19.46	20	
1	New A	200	142.19	0.51495	0.51496	4.960	19.86	20	
2	New A	200	142.19	0.51513	0.51514	4.962	19.86	20	
3	New A	200	142.19	0.51485	0.51486	4.959	19.91	20	
4	New B	200	142.19	0.61870	0.61877	4.960	19.40	20	

	EndVolts	TitreTime	O2µmol/L
0	2.450	155700	221.5632
1	2.412	160520	221.4292
2	2.427	161045	221.5186
3	2.424	161648	221.3846
4	2.358	160025	221.5632

1.1.1 3.1.1 Iodate Standards across Instruments Boxplot

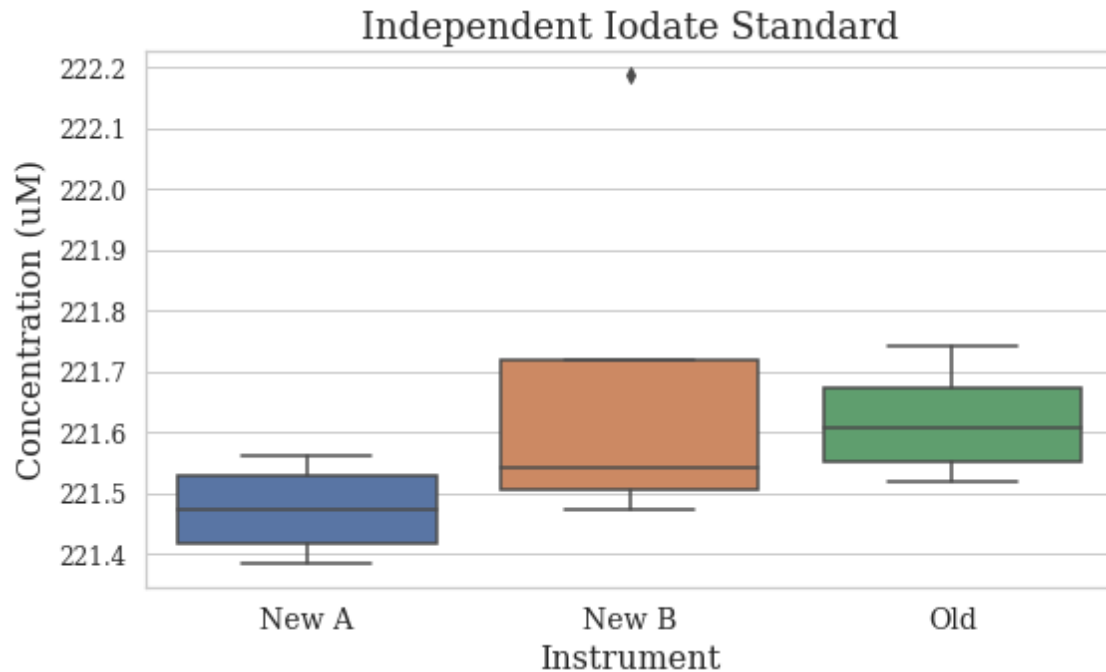
```
In [6]: sns.boxplot(iodate_df['Instrument'], iodate_df['O2µmol/L'])

plt.xlabel("Instrument", fontsize=16)
plt.ylabel("Concentration (uM)", fontsize=16)
plt.xticks(fontsize=14)
plt.yticks(fontsize=12)
plt.title('Independent Iodate Standard', fontsize=18)

plt.tight_layout()

# Comment or include next two lines if wanting to scale chart for all
#data_mean = iodate_df['O2µmol/L'].mean()
#plt.ylim(data_mean-2, data_mean+2)

plt.savefig('independent_iodate_standards.svg', format='svg')
```



1.1.2 3.1.2 Iodate Standards Descriptive Statistics

```
In [7]: iodate_df.groupby(['Instrument'])['O2μmol/L'].describe()
```

```
Out [7]:
```

	count	mean	std	min	25%	50%	\
Instrument							
New A	4.0	221.47390	0.081538	221.3846	221.418050	221.47390	
New B	4.0	221.68605	0.336945	221.4739	221.507425	221.54090	
Old	4.0	221.61905	0.099029	221.5186	221.552050	221.60785	

	75%	max
Instrument		
New A	221.529750	221.5632
New B	221.719525	222.1885
Old	221.674850	221.7419

1.2 3.2 Repeated Deep Sample Measurements: 1

1.2.1 3.2.1 Samples from One Niskin per Instrument

```
In [8]: deep_reps_single_df = pd.read_csv(DEP_1_DEEP_REPLICATES_SINGLE_NISKINS_DATA)
```

```
In [9]: deep_reps_single_df.head()
```

```
Out [9]:
```

	Instrument	Bottle	FlaskVol	RawTitre	Titre20	O2ml/L	ThioTemp	DrawTemp	\
0	New A	143	145.25	0.44525	0.44526	4.195	19.88	6.7	
1	New A	144	146.25	0.44776	0.44777	4.190	19.84	6.9	
2	New A	148	143.04	0.43883	0.43884	4.199	19.90	7.1	
3	New A	150	140.86	0.43107	0.43108	4.189	19.84	7.2	
4	New A	152	145.03	0.44393	0.44395	4.189	19.82	7.4	

	EndVolts	TitreTime	O2µmol/L	RP
0	2.423	162322	187.3082	4
1	2.471	162637	187.0403	4
2	2.405	163229	187.4869	4
3	2.389	163558	187.0403	4
4	2.393	163904	187.0403	4

3.2.1.1 Samples from One Niskin Boxplot

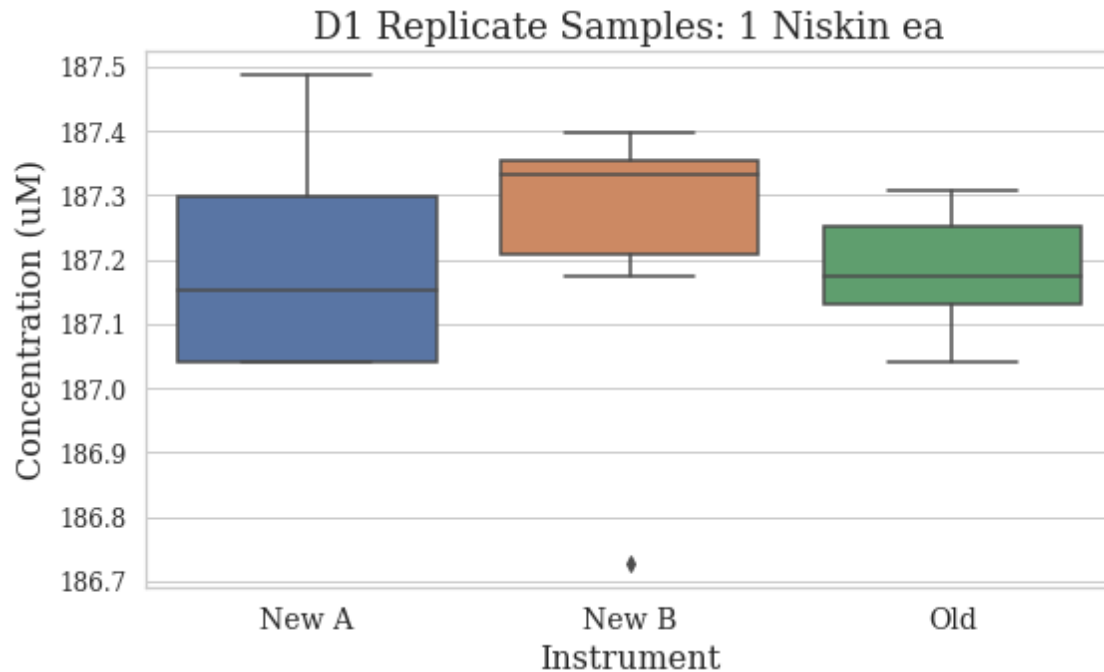
```
In [10]: sns.boxplot(deep_reps_single_df['Instrument'], deep_reps_single_df['O2µmol/L'])

plt.xlabel("Instrument", fontsize=16)
plt.ylabel("Concentration (uM)", fontsize=16)
plt.xticks(fontsize=14)
plt.yticks(fontsize=12)
plt.title('D1 Replicate Samples: 1 Niskin ea', fontsize=18)

plt.tight_layout()

# Comment or include next two lines if wanting to scale chart for all
#data_mean = deep_reps_single_df['O2µmol/L'].mean()
#plt.ylim(data_mean-2, data_mean+2)

plt.savefig('replicate_deep_samples_1_single.svg', format='svg')
```



3.2.1.2 Descriptive Statistics

```
In [11]: deep_reps_single_df.groupby(['Instrument'])['O2µmol/L'].describe()
```

```
Out[11]:
```

	count	mean	std	min	25%	50%	\
Instrument							
New A	6.0	187.196600	0.186819	187.0403	187.040300	187.15195	
New B	6.0	187.218917	0.252660	186.7276	187.207775	187.33055	
Old	6.0	187.181700	0.099515	187.0403	187.129600	187.17425	

	75%	max
Instrument		
New A	187.297050	187.4869
New B	187.352900	187.3976
Old	187.252425	187.3082

3.2.1.3 T-Test Comparison of Means

```
In [12]: deep_reps_single_newa = deep_reps_single_df.loc[deep_reps_single_df['Instrument'] ==
            deep_reps_single_newb = deep_reps_single_df.loc[deep_reps_single_df['Instrument'] ==
            deep_reps_single_old = deep_reps_single_df.loc[deep_reps_single_df['Instrument'] == 'Old']
```

Compare New A to Old instrument

```
In [13]: result = sci_st.ttest_ind(deep_reps_single_newa['02µmol/L'], deep_reps_single_old['02µmol/L'])
print(f'Comparison of the New A instrument to Old, p-value: {result[1]}')
if result[1] < 0.05:
    print('Significance !')
```

Comparison of the New A instrument to Old, p-value: 0.8665430859682957

Compare New B to Old instrument

```
In [14]: result = sci_st.ttest_ind(deep_reps_single_newb['02µmol/L'], deep_reps_single_old['02µmol/L'])
print(f'Comparison of the New B instrument to Old, p-value: {result[1]}')
if result[1] < 0.05:
    print('Significance !')
```

Comparison of the New B instrument to Old, p-value: 0.7440286422028863

1.2.2 3.2.2 Samples from Two Niskins for all Instruments

```
In [15]: deep_reps_shared_df = pd.read_csv(DEP_1_DEEP_REPLICATES_SHARED_NISKINS_DATA)
```

```
In [16]: deep_reps_shared_df.head()
```

```
Out[16]:
```

	Instrument	Bottle	FlaskVol	RawTitre	Titre20	02ml/L	ThioTemp	DrawTemp	\
0	New A	161	144.68	0.44360	0.44343	4.195	21.84	7.0	
1	New A	167	144.39	0.44349	0.44331	4.202	21.88	7.4	
2	New A	211	144.58	0.44272	0.44254	4.189	21.95	7.2	
3	New A	217	141.62	0.43344	0.43327	4.188	21.94	7.7	
4	New B	162	145.50	0.53636	0.53638	4.199	19.81	7.2	

	EndVolts	TitreTime	02µmol/L	RP
0	2.074	232544	187.2636	5
1	2.132	232834	187.6209	5
2	2.137	233205	187.0403	6
3	2.083	233501	186.9956	6
4	2.270	233544	187.5762	5

3.2.2.1 Samples from Shared Niskins Boxplot

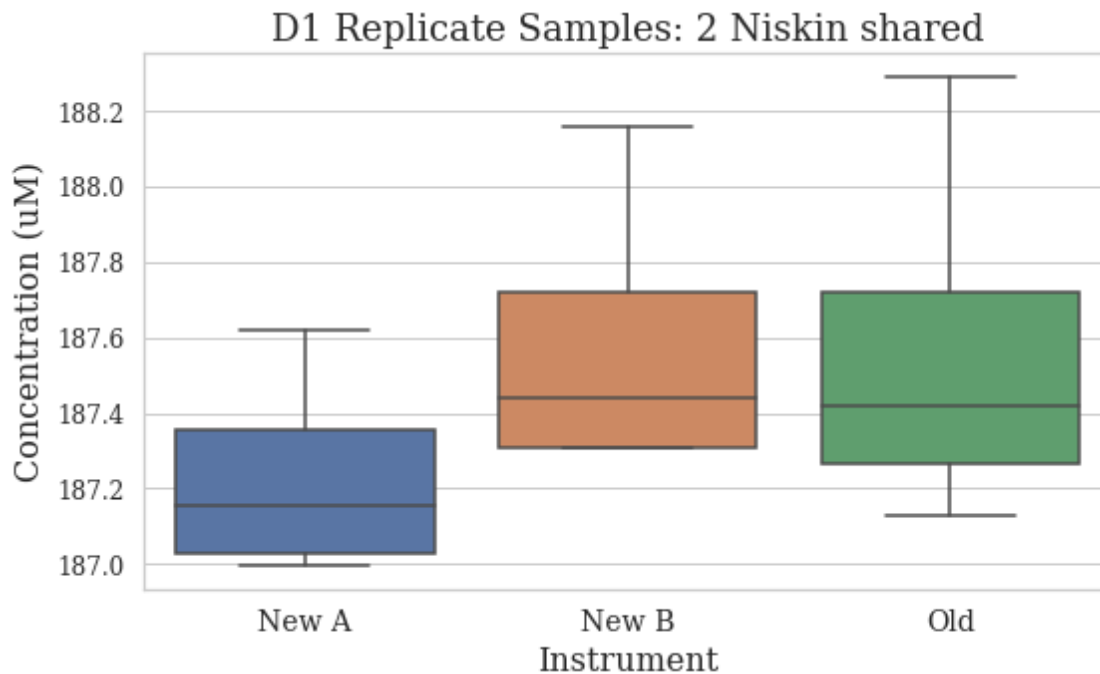
```
In [17]: sns.boxplot(deep_reps_shared_df['Instrument'], deep_reps_shared_df['02µmol/L'])

plt.xlabel("Instrument", fontsize=16)
plt.ylabel("Concentration (uM)", fontsize=16)
plt.xticks(fontsize=14)
plt.yticks(fontsize=12)
plt.title('D1 Replicate Samples: 2 Niskin shared', fontsize=18)

plt.tight_layout()
```

```
# Comment or include next two lines if wanting to scale chart for all
#data_mean = deep_reps_shared_df['O2mol/L'].mean()
#plt.ylim(data_mean-2, data_mean+2)

plt.savefig('replicate_deep_samples_1_shared.svg', format='svg')
```



3.2.2.2 Descriptive Statistics

```
In [18]: deep_reps_shared_df.groupby(['Instrument'])['O2mol/L'].describe()
```

```
Out[18]:
```

	count	mean	std	min	25%	50%	\
Instrument							
New A	4.0	187.230100	0.285693	186.9956	187.029125	187.15195	
New B	4.0	187.587350	0.400103	187.3082	187.308200	187.44220	
Old	4.0	187.565025	0.511021	187.1296	187.263550	187.41985	

	75%	max
Instrument		
New A	187.352925	187.6209
New B	187.721350	188.1568
Old	187.721325	188.2908

1.3 3.3 Atmospheric Saturated Sample: All Instruments

```
In [19]: atmospheric_all_df = pd.read_csv(ATMOSPHERIC_DIFF_INSTRUMENTS_DATA)
```

```
In [20]: atmospheric_all_df.head()
```

```
Out[20]:
```

	Instrument	Bottle	FlaskVol	RawTitre	Titre20	O2ml/L	ThioTemp	DrawTemp	\
0	New A	257	138.61	0.62271	0.62247	6.157	21.86	21.5	
1	New A	260	140.86	0.63292	0.63285	6.158	20.50	21.5	
2	New A	263	137.65	0.61726	0.61720	6.148	20.49	21.5	
3	New A	266	142.77	0.64096	0.64089	6.152	20.48	21.5	
4	New B	258	138.45	0.74709	0.74689	6.154	21.31	21.5	

	EndVolts	TitreTime	O2µmol/L
0	2.030	2101	274.8438
1	2.009	2318	274.9331
2	2.053	2617	274.4418
3	2.017	2934	274.6205
4	2.232	2951	274.8885

1.3.1 3.3.1 Atmospheric Saturated Sample Boxplot (auto-scale)

Plot just initial box plot scaled to the measurements

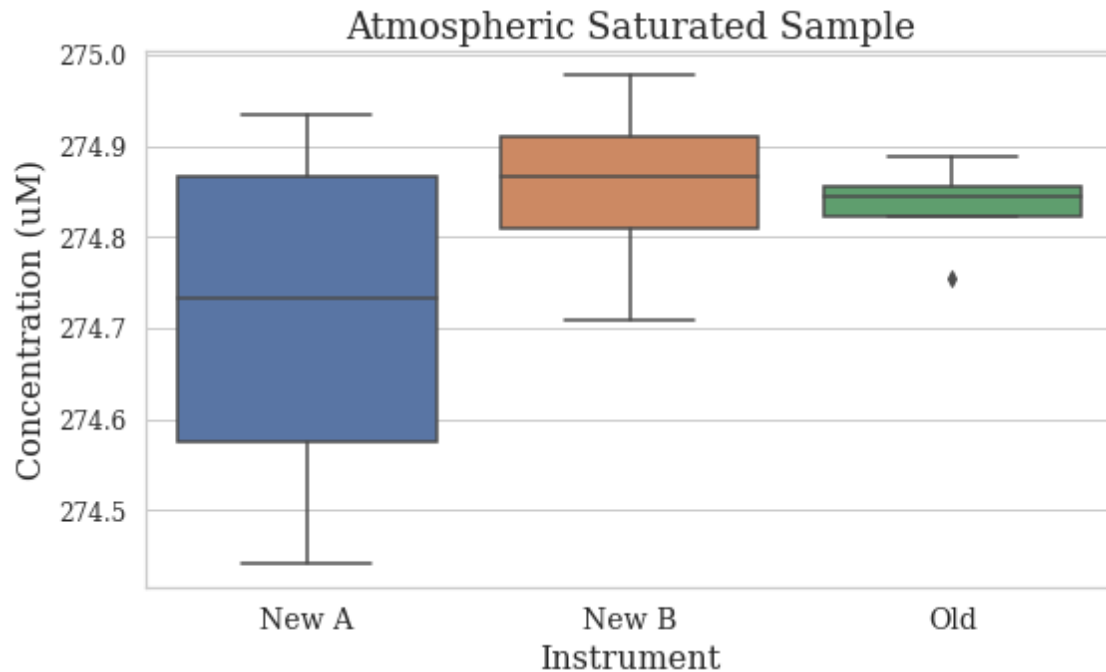
```
In [21]: sns.boxplot(atmospheric_all_df['Instrument'], atmospheric_all_df['O2µmol/L'])

plt.xlabel("Instrument", fontsize=16)
plt.ylabel("Concentration (uM)", fontsize=16)
plt.xticks(fontsize=14)
plt.yticks(fontsize=12)
plt.title('Atmospheric Saturated Sample', fontsize=18)

plt.tight_layout()

# Comment or include next two lines if wanting to scale chart for all
#data_mean = atmospheric_all_df['O2µmol/L'].mean()
#plt.ylim(data_mean-2, data_mean+2)

plt.savefig('atmospheric_diff_instruments.svg', format='svg')
```

1.3.2 3.3.2 Atmospheric Saturated Sample Boxplot (QC Control Lines)

Plot with lines to indicate various QC limits

```
In [22]: sns.boxplot(atmospheric_all_df['Instrument'], atmospheric_all_df['O2mol/L'])

plt.xlabel("Instrument", fontsize=16)
plt.ylabel("Concentration (uM)", fontsize=16)
plt.xticks(fontsize=14)
plt.yticks(fontsize=12)
plt.title('Atmospheric Saturated Sample w/QC Bars', fontsize=18)

# Calculate the theoretical saturation
MEASURED_SALINITY = 0
MEASURED_TEMPERATURE = 21.5
max_saturation = sw.satO2(MEASURED_SALINITY, MEASURED_TEMPERATURE)
max_saturation_mol = max_saturation * 44.66

max_saturation_mol_1pct = max_saturation_mol * 0.01

# Plot the calculated saturation as a line on the chart
plt.plot([-0.5, 2.5], [max_saturation_mol, max_saturation_mol], color="#32a858")

# Plot the 1% upper and lower lines
plt.plot([-0.5, 2.5], [max_saturation_mol-max_saturation_mol_1pct, max_saturation_mol_1pct], color="#32a858")
```

```

plt.plot([-0.5, 2.5], [max_saturation_mol+max_saturation_mol_1pct, max_saturation_mol_1pct], color="#2c9fa3")

# Plot the +- 1uM upper and lower lines
plt.plot([-0.5, 2.5], [max_saturation_mol-1, max_saturation_mol-1], color="#2c9fa3")
plt.plot([-0.5, 2.5], [max_saturation_mol+1, max_saturation_mol+1], color="#2c9fa3")

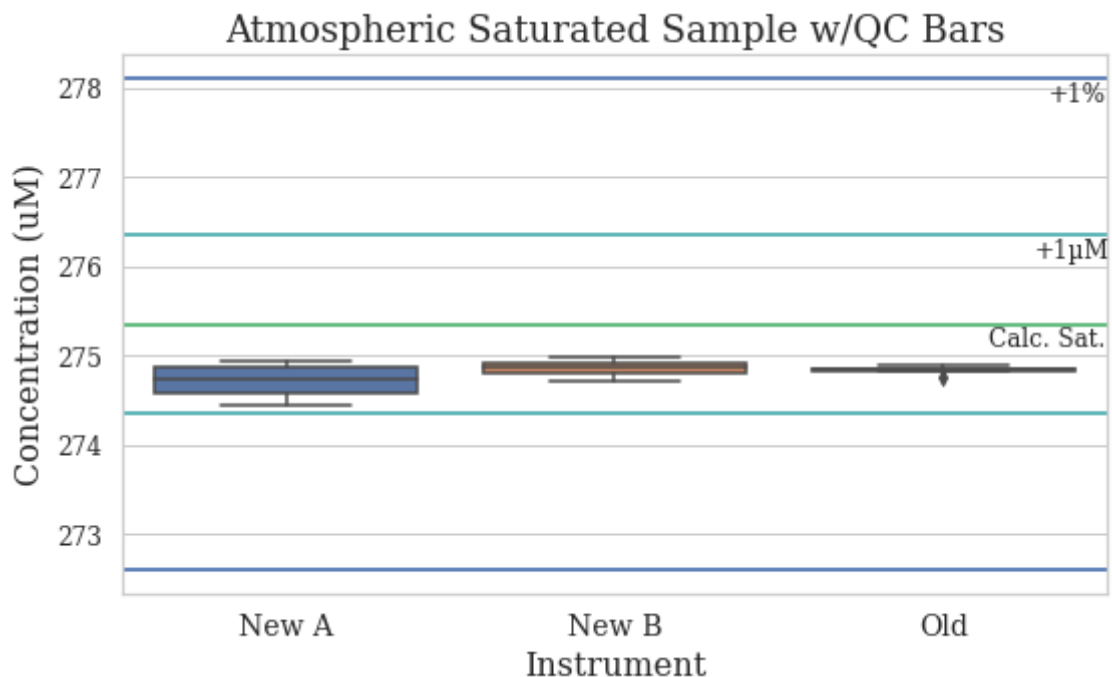
# Plot the annotations
plt.annotate('+1uM', xy=(2.28, max_saturation_mol+1-0.25))
plt.annotate('+1%', xy=(2.32, max_saturation_mol+max_saturation_mol_1pct-0.25))
plt.annotate('Calc. Sat.', xy=(2.14, max_saturation_mol-0.25))

plt.tight_layout()

# Comment or include next two lines if wanting to scale chart for all
#data_mean = atmospheric_all_df['O2umol/L'].mean()
#plt.ylim(data_mean-2, data_mean+2)

plt.savefig('atmospheric_diff_instruments_with_bars.svg', format='svg')

```



1.3.3 3.3.3 Descriptive Statistics

In [23]: `atmospheric_all_df.groupby(['Instrument'])['O2umol/L'].describe()`

Out[23]:

	count	mean	std	min	25%	50%	\
Instrument							

New A	4.0	274.709800	0.221824	274.4418	274.575825	274.73215
New B	4.0	274.854975	0.111670	274.7098	274.810300	274.86615
Old	4.0	274.832650	0.056200	274.7545	274.821475	274.84380

	75%	max
Instrument		
New A	274.866125	274.9331
New B	274.910825	274.9778
Old	274.854975	274.8885

1.4 3.4 Atmospheric Saturated Sample: One Instrument

```
In [24]: atmospheric_one_df = pd.read_csv(ATMOSPHERIC_ONE_INSTRUMENT_DATA)
```

```
In [25]: atmospheric_one_df.head()
```

```
Out [25]:
```

	Instrument	Bottle	FlaskVol	RawTitre	Titre20	O2ml/L	ThioTemp	DrawTemp	\
0	New B	210	143.84	0.78354	0.78332	6.208	21.38	21.5	
1	New B	211	144.58	0.78712	0.78686	6.204	21.60	21.5	
2	New B	214	146.67	0.79938	0.79910	6.210	21.70	21.5	
3	New B	216	144.35	0.78570	0.78541	6.203	21.77	21.5	
4	New B	217	141.62	0.77087	0.77058	6.204	21.83	21.5	

	EndVolts	TitreTime	O2µmol/L
0	2.320	145535	277.3895
1	2.327	145744	277.2108
2	2.344	150012	277.4788
3	2.306	150236	277.1662
4	2.287	150440	277.2108

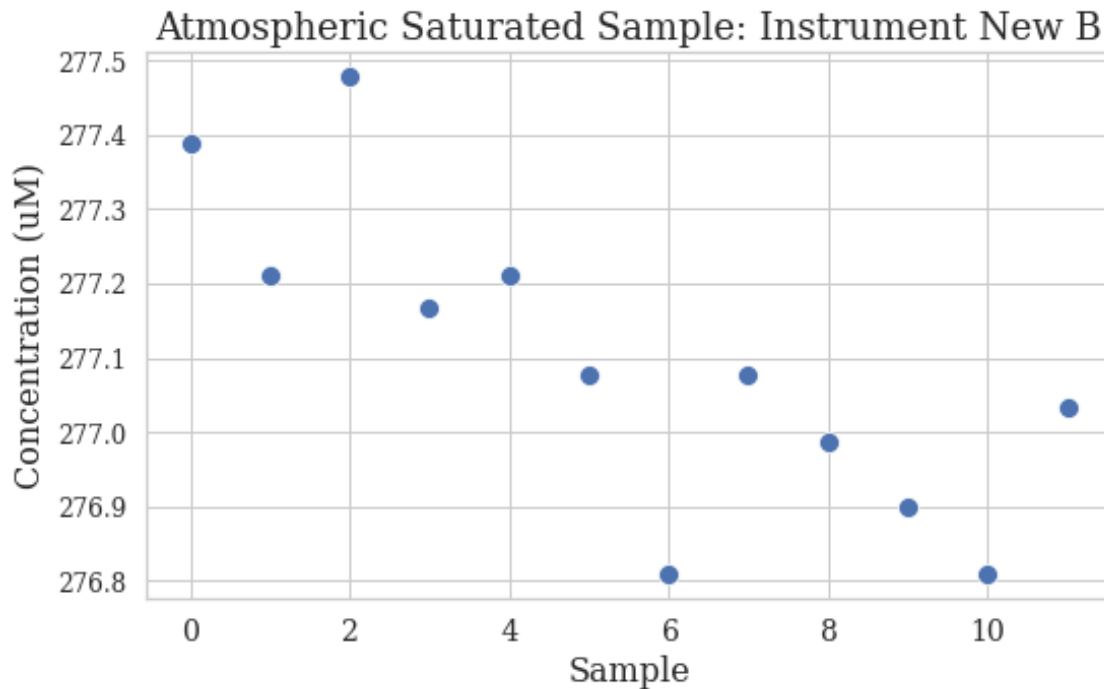
1.4.1 3.4.1 Atmospheric Saturated Sample: Instrument New B (auto-scale)

```
In [26]: sns.lineplot(atmospheric_one_df.index, atmospheric_one_df['O2µmol/L'], lw=0, marker="

plt.xlabel("Sample", fontsize=16)
plt.ylabel("Concentration (uM)", fontsize=16)
plt.xticks(fontsize=14)
plt.yticks(fontsize=12)
plt.title('Atmospheric Saturated Sample: Instrument New B', fontsize=18)

plt.tight_layout()

plt.savefig('atmospheric_one_instrument.svg', format='svg')
```



1.4.2 3.4.2 Atmospheric Saturated Sample: Instrument New B (QC Control Limits)

In [27]: `sns.lineplot(atmospheric_one_df.index, atmospheric_one_df['O2mol/L'], lw=0, marker="o",`

```
plt.xlabel("Sample", fontsize=16)
plt.ylabel("Concentration (uM)", fontsize=16)
plt.xticks(fontsize=14)
plt.yticks(fontsize=12)
plt.title('Atmospheric Saturated Sample: Instrument New B', fontsize=18)

plt.tight_layout()

# Calculate the theoretical saturation
MEASURED_SALINITY = 0
MEASURED_TEMPERATURE = 21.5
max_saturation = sw.satO2(MEASURED_SALINITY, MEASURED_TEMPERATURE)
max_saturation_mol = max_saturation * 44.66

max_saturation_mol_1pct = max_saturation_mol * 0.01

# Plot the calculated saturation as a line on the chart
plt.plot([-1, 13], [max_saturation_mol, max_saturation_mol], color="#32a858")

# Plot the 1% upper and lower lines
```

```

plt.plot([-1, 13], [max_saturation_mol-max_saturation_mol_1pct, max_saturation_mol-max_saturation_mol_1pct], color="#2c9fa3")
plt.plot([-1, 13], [max_saturation_mol+max_saturation_mol_1pct, max_saturation_mol+max_saturation_mol_1pct], color="#2c9fa3")

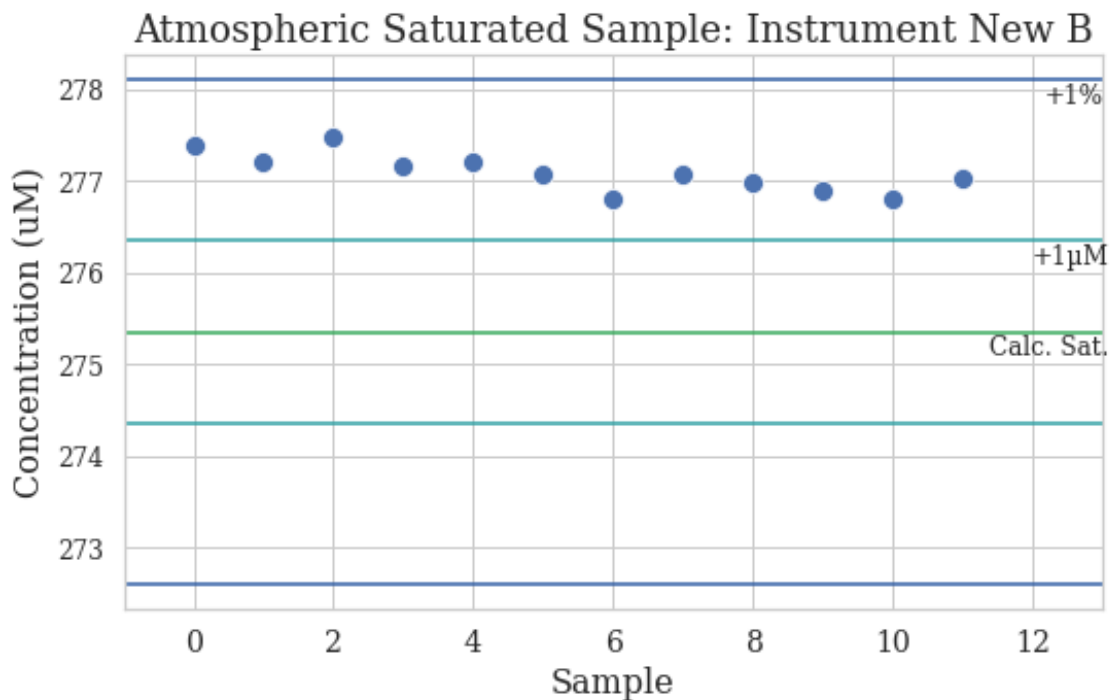
# Plot the +- 1uM upper and lower lines
plt.plot([-1, 13], [max_saturation_mol-1, max_saturation_mol-1], color="#2c9fa3")
plt.plot([-1, 13], [max_saturation_mol+1, max_saturation_mol+1], color="#2c9fa3")

# Plot the annotations
plt.annotate('+1µM', xy=(12, max_saturation_mol+1-0.25))
plt.annotate('+1%', xy=(12.18, max_saturation_mol+max_saturation_mol_1pct-0.25))
plt.annotate('Calc. Sat.', xy=(11.4, max_saturation_mol-0.25))

plt.xlim(-1, 13)

plt.savefig('atmospheric_one_instrument_withBars.svg', format='svg')

```



1.4.3 3.4.3 Atmospheric Saturated Sample: Instrument New B Boxplot (QC Control Limits)

```

In [28]: sns.boxplot(atmospheric_one_df['Instrument'], atmospheric_one_df['O2µmol/L'])

plt.xlabel("Sample", fontsize=16)
plt.ylabel("Concentration (uM)", fontsize=16)
plt.xticks(fontsize=14)
plt.yticks(fontsize=12)

```

```

plt.title('Atmospheric Saturated Sample: Instrument New B', fontsize=18)

plt.tight_layout()

# Calculate the theoretical saturation
MEASURED_SALINITY = 0
MEASURED_TEMPERATURE = 21.5
max_saturation = sw.satO2(MEASURED_SALINITY, MEASURED_TEMPERATURE)
max_saturation_mol = max_saturation * 44.66

max_saturation_mol_1pct = max_saturation_mol * 0.01

# Plot the calculated saturation as a line on the chart
plt.plot([-0.5, 0.5], [max_saturation_mol, max_saturation_mol], color="#32a858")

# Plot the 1% upper and lower lines
plt.plot([-0.5, 0.5], [max_saturation_mol-max_saturation_mol_1pct, max_saturation_mol_1pct], color="#32a858")
plt.plot([-0.5, 0.5], [max_saturation_mol+max_saturation_mol_1pct, max_saturation_mol_1pct], color="#32a858")

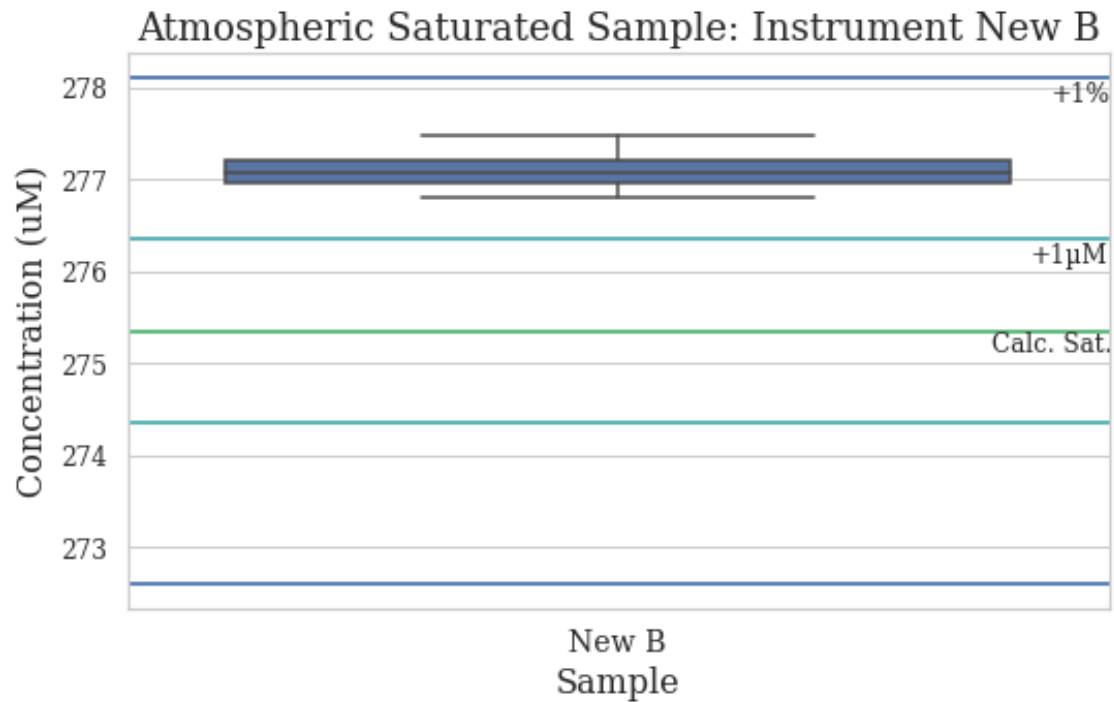
# Plot the +- 1uM upper and lower lines
plt.plot([-0.5, 0.5], [max_saturation_mol-1, max_saturation_mol+1], color="#2c9fa3")
plt.plot([-0.5, 0.5], [max_saturation_mol-1, max_saturation_mol+1], color="#2c9fa3")

# Plot the annotations
plt.annotate('+1uM', xy=(0.42, max_saturation_mol+1-0.25))
plt.annotate('+1%', xy=(0.44, max_saturation_mol+max_saturation_mol_1pct-0.25))
plt.annotate('Calc. Sat.', xy=(0.38, max_saturation_mol-0.23))

# Comment or include next two lines if wanting to scale chart for all
#data_mean = atmospheric_one_df['O2umol/L'].mean()
#plt.ylim(data_mean-2, data_mean+2)

plt.savefig('atmospheric_one_instrument_with_bars-boxplot-version.svg', format='svg')

```



1.4.4 3.4.4 Descriptive Statistics

In [29]: `atmospheric_one_df['O2µmol/L'].describe()`

```
Out[29]: count      12.000000
         mean      277.095450
         std       0.209874
         min      276.808900
         25%      276.965175
         50%      277.076800
         75%      277.210800
         max      277.478800
         Name: O2µmol/L, dtype: float64
```

1.5 3.5 Water Profile Comparison

In [30]: `profile_comparison_df = pd.read_csv(PROFILE_COMPARISON_DATA)`

In [31]: `profile_comparison_df.head()`

```
Out[31]:   Instrument  Bottle  FlaskVol  RawTitre  Titre20  O2ml/L  ThioTemp  DrawTemp  \
0      New A      278      142.48    0.59621  0.59623    5.734    19.83      14.0
1      New A      276      140.04    0.58568  0.58551    5.730    21.38      14.1
```

2	New A	627	145.60	0.60911	0.60893	5.729	21.45	14.1
3	New A	625	143.75	0.60031	0.60012	5.720	21.51	13.9
4	New A	623	140.11	0.58475	0.58456	5.718	21.53	13.9

	EndVolts	TitreTime	O2µmol/L	RP	Pressure
0	2.282	201954	255.9969	24	5
1	2.276	202345	255.8182	23	5
2	2.299	202606	255.7735	22	5
3	2.292	203001	255.3716	21	40
4	2.258	205345	255.2823	20	40

1.5.1 3.5.1 Water Profile Plot

```
In [32]: instrument_new_a_df = profile_comparison_df.loc[profile_comparison_df['Instrument'] ==
instrument_old_df = profile_comparison_df.loc[profile_comparison_df['Instrument'] ==

plt.plot(instrument_new_a_df['O2µmol/L'], instrument_new_a_df['Pressure'], lw=0, marker="o")
plt.plot(instrument_old_df['O2µmol/L'], instrument_old_df['Pressure'], lw=0, marker="x")

plt.xlabel("Concentration (uM)", fontsize=16)
plt.ylabel("Pressure (db)", fontsize=16)
plt.xticks(fontsize=14)
plt.yticks(fontsize=12)
plt.title('Profile Comparison', fontsize=18)

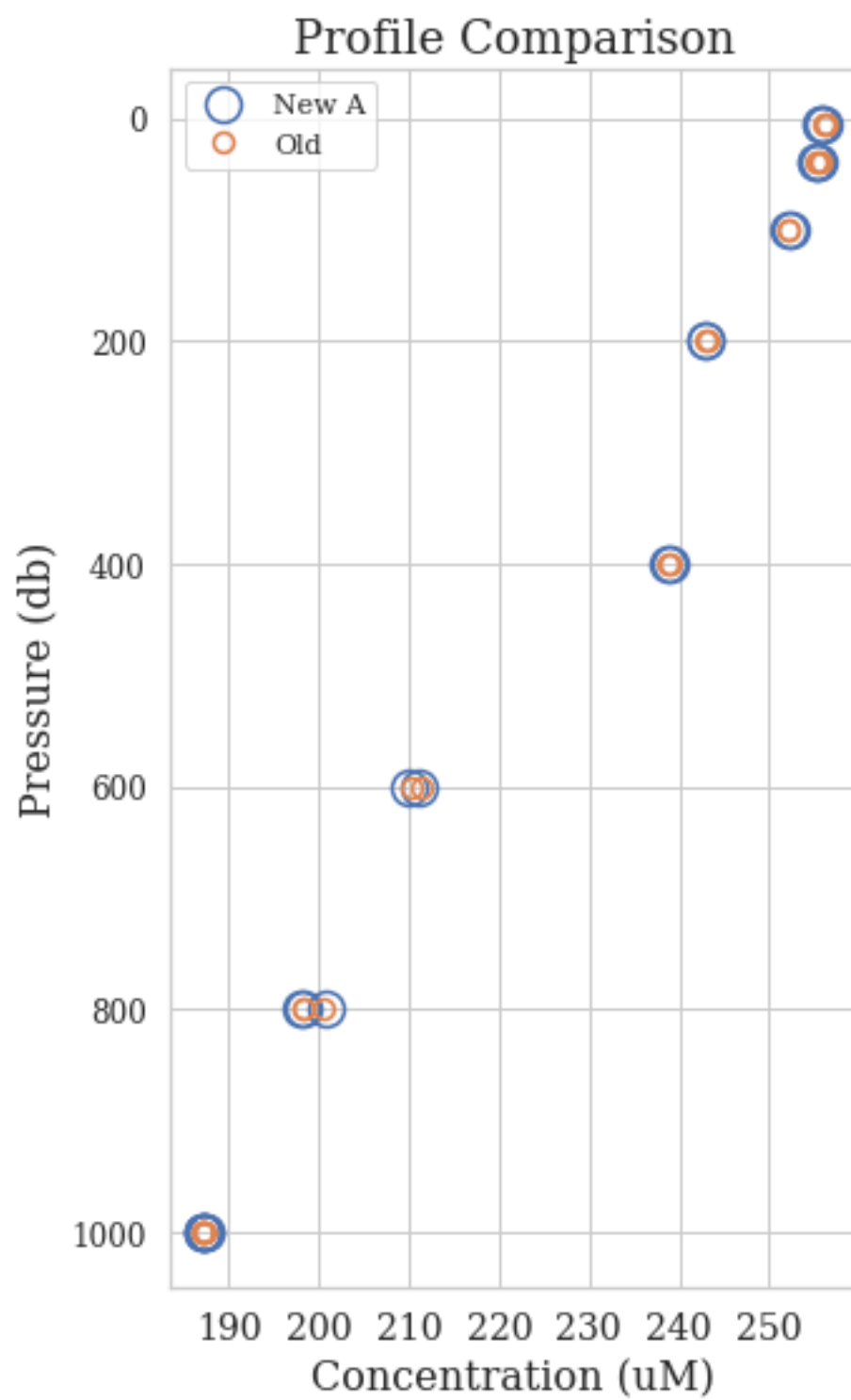
plt.legend()

plt.gca().invert_yaxis()

fig = mpl.pyplot.gcf()
fig.set_size_inches(5, 8)

plt.tight_layout()

plt.savefig('profile_comparison.svg', format='svg')
```

1.5.2 3.5.2 T-Test Comparison of means

```
In [33]: results_dict = {}
        depths_to_test = [5, 40, 800, 1000]
        for depth in depths_to_test:
            instrument_a_depth_subset = instrument_new_a_df.loc[instrument_new_a_df['Pressure']
            instrument_old_depth_subset = instrument_old_df.loc[instrument_old_df['Pressure']
            result = sci_st.ttest_ind(instrument_a_depth_subset['O2µmol/L'], instrument_old_d
            results_dict[depth] = result[1]

        print("For the tested depths, the t-test p-value is shown")
        for key in results_dict:
            print(f'At depth: {key} the p-value is: {results_dict[key]}')
```

For the tested depths, the t-test p-value is shown

At depth: 5 the p-value is: 0.10839798708992938

At depth: 40 the p-value is: 0.3026985172966608

At depth: 800 the p-value is: 0.9008494336784751

At depth: 1000 the p-value is: 0.8665430859682957

1.6 3.6 Repeated Deep Sample Measurements: 2

```
In [34]: deep_reps_two_df = pd.read_csv(DEP_2_DEEP_REPLICATES_DATA)
```

```
In [35]: deep_reps_two_df.head()
```

```
Out[35]:
```

	Instrument	Bottle	FlaskVol	RawTitre	Titre20	O2ml/L	ThioTemp	DrawTemp	\
0	New A	675	135.77	0.41383	0.41379	4.174	20.46	6.3	
1	New A	678	139.77	0.42626	0.42625	4.175	20.12	6.3	
2	New A	681	141.76	0.43149	0.43148	4.166	20.18	6.5	
3	New A	684	138.61	0.42143	0.42141	4.163	20.24	6.5	
4	New A	687	141.65	0.43127	0.43125	4.167	20.27	6.5	

	EndVolts	TitreTime	O2µmol/L	RP
0	2.085	5547	186.3257	2
1	2.443	5948	186.4150	5
2	2.452	10147	186.0131	8
3	2.407	10343	185.8344	11
4	2.439	10624	186.0577	14

1.6.1 3.6.1 Deployment 2 Replicates Boxplot

```
In [36]: sns.boxplot(deep_reps_two_df['Instrument'], deep_reps_two_df['O2µmol/L'])

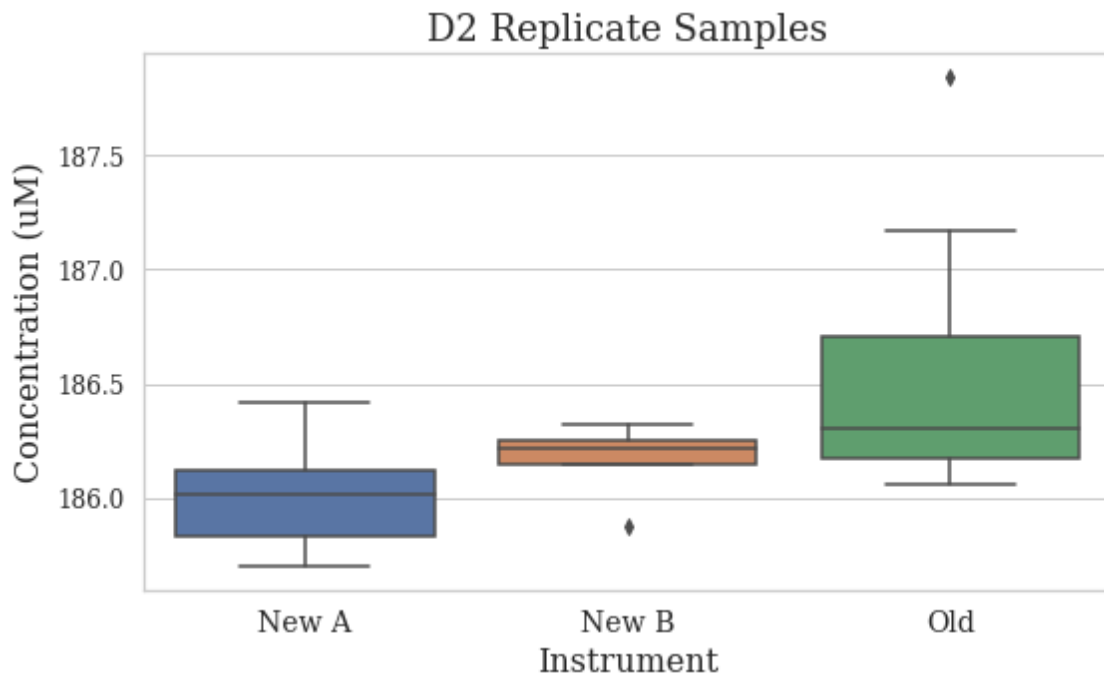
plt.xlabel("Instrument", fontsize=16)
plt.ylabel("Concentration (uM)", fontsize=16)
```

```
plt.xticks(fontsize=14)
plt.yticks(fontsize=12)
plt.title('D2 Replicate Samples', fontsize=18)

plt.tight_layout()

# Comment or include next two lines if wanting to scale chart for all
#data_mean = deep_reps_two_df['O2µmol/L'].mean()
#plt.ylim(data_mean-2, data_mean+2)

plt.savefig('replicate_deep_samples_2.svg', format='svg')
```



1.6.2 3.6.2 Descriptive Statistics

```
In [37]: deep_reps_two_df.groupby(['Instrument'])['O2µmol/L'].median()
```

```
Out[37]: Instrument
New A    186.01310
New B    186.21405
Old      186.30335
Name: O2µmol/L, dtype: float64
```

1.6.3 3.6.3 T-Test Comparison of Means

```
In [38]: # Subset by instrument into new dataframes just for simplicity in next cells
deep_reps_two_newa = deep_reps_two_df.loc[deep_reps_two_df['Instrument'] == 'New A']
```

```

deep_reps_two_newb = deep_reps_two_df.loc[deep_reps_two_df['Instrument'] == 'New B']
deep_reps_two_old = deep_reps_two_df.loc[deep_reps_two_df['Instrument'] == 'Old']

```

Compare New A to the Old instrument

```

In [39]: result = sci_st.ttest_ind(deep_reps_two_newa['02µmol/L'], deep_reps_two_old['02µmol/L'])
print(f'Comparison of the New A instrument to Old, p-value: {result[1]}')
if result[1] < 0.05:
    print('Significance!')

```

Comparison of the New A instrument to Old, p-value: 0.039304940051647204

Significance!

Compare New B to Old instrument

```

In [40]: result = sci_st.ttest_ind(deep_reps_two_newb['02µmol/L'], deep_reps_two_old['02µmol/L'])
print(f'Comparison of the New B instrument to Old, p-value: {result[1]}')
if result[1] < 0.05:
    print('Significance!')

```

Comparison of the New B instrument to Old, p-value: 0.11189671028378533

Compare New A to New B

```

In [48]: result = sci_st.ttest_ind(deep_reps_two_newa['02µmol/L'], deep_reps_two_newb['02µmol/L'])
print(f'Comparison of the New A instrument to New B, p-value: {result[1]}')
if result[1] < 0.05:
    print('Significance!')

```

Comparison of the New A instrument to New B, p-value: 0.13788994603790747

1.7 3.7 Experimental Meta Analysis

```

In [41]: combined_df = pd.read_csv(COMBINED_DATA)

```

```

In [42]: combined_df.head()

```

```

Out[42]:  Experiment Instrument  Bottle  FlaskVol  RawTitre  Titre20  02ml/L  \
0      Iodate      New A      200      142.19    0.51524  0.51529  4.964
1      Iodate      New A      200      142.19    0.51495  0.51496  4.960
2      Iodate      New A      200      142.19    0.51513  0.51514  4.962
3      Iodate      New A      200      142.19    0.51485  0.51486  4.959
4      Iodate      New B      200      142.19    0.61870  0.61877  4.960

```

	ThioTemp	DrawTemp	EndVolts	TitreTime	O2µmol/L	RP	Pressure
0	19.46	20.0	2.450	155700	221.5632	NaN	NaN
1	19.86	20.0	2.412	160520	221.4292	NaN	NaN
2	19.86	20.0	2.427	161045	221.5186	NaN	NaN
3	19.91	20.0	2.424	161648	221.3846	NaN	NaN
4	19.40	20.0	2.358	160025	221.5632	NaN	NaN

1.7.1 3.7.1 Calculation of Mean Standard Deviation

```
In [43]: # Take out the profile experiment as it will skew the data, unless pressure is included
excl_profile = combined_df.loc[combined_df['Experiment'] != "Profile_Comp"]
# Group by Instrument then Experiment type then show descriptive stats
excl_profile.groupby(['Instrument', 'Experiment'])['O2µmol/L'].describe()
```

```
Out [43]:
```

		count	mean	std	min	\
New A	Atmos_All	4.0	274.709800	0.221824	274.4418	
	Dep1_Reps_Shared	4.0	187.230100	0.285693	186.9956	
	Dep1_Reps_Single	6.0	187.196600	0.186819	187.0403	
	Dep2_Reps	8.0	186.024225	0.245500	185.7004	
	Iodate	4.0	221.473900	0.081538	221.3846	
New B	Atmos_All	4.0	274.854975	0.111670	274.7098	
	Atmos_NewB	12.0	277.095450	0.209874	276.8089	
	Dep1_Reps_Shared	4.0	187.587350	0.400103	187.3082	
	Dep1_Reps_Single	6.0	187.218917	0.252660	186.7276	
	Dep2_Reps	8.0	186.180538	0.136615	185.8791	
Old	Iodate	4.0	221.686050	0.336945	221.4739	
	Atmos_All	4.0	274.832650	0.056200	274.7545	
	Dep1_Reps_Shared	4.0	187.565025	0.511021	187.1296	
	Dep1_Reps_Single	6.0	187.181700	0.099515	187.0403	
	Dep2_Reps	8.0	186.565750	0.627521	186.0577	
	Iodate	4.0	221.619050	0.099029	221.5186	

		25%	50%	75%	max
New A	Atmos_All	274.575825	274.73215	274.866125	274.9331
	Dep1_Reps_Shared	187.029125	187.15195	187.352925	187.6209
	Dep1_Reps_Single	187.040300	187.15195	187.297050	187.4869
	Dep2_Reps	185.834400	186.01310	186.124700	186.4150
	Iodate	221.418050	221.47390	221.529750	221.5632
New B	Atmos_All	274.810300	274.86615	274.910825	274.9778
	Atmos_NewB	276.965175	277.07680	277.210800	277.4788
	Dep1_Reps_Shared	187.308200	187.44220	187.721350	188.1568
	Dep1_Reps_Single	187.207775	187.33055	187.352900	187.3976
	Dep2_Reps	186.147000	186.21405	186.247550	186.3257
Old	Iodate	221.507425	221.54090	221.719525	222.1885
	Atmos_All	274.821475	274.84380	274.854975	274.8885
	Dep1_Reps_Shared	187.263550	187.41985	187.721325	188.2908

Dep1_Reps_Single	187.129600	187.17425	187.252425	187.3082
Dep2_Reps	186.169375	186.30335	186.705325	187.8442
Iodate	221.552050	221.60785	221.674850	221.7419

```
In [44]: # Get final standard deviation
excl_profile.groupby(['Instrument', 'Experiment'])['O2µmol/L'].std().groupby(['Instrument', 'Experiment'])
```

```
Out[44]: Instrument
New A      0.204275
New B      0.241311
Old        0.278657
Name: O2µmol/L, dtype: float64
```

1.7.2 3.7.2 F-Test for Difference in Variances

```
In [45]: # Take out the profile experiment as it will skew the data, unless pressure is included
excl_profile = combined_df.loc[combined_df['Experiment'] != "Profile_Comp"]
excl_atmosB = excl_profile.loc[excl_profile['Experiment'] != "Atmos_NewB"]

# Group by Instrument then Experiment type then show descriptive stats
grouped_st_devs = excl_atmosB.groupby(['Instrument', 'Experiment'])['O2µmol/L'].std()
```

```
In [46]: grouped_st_devs
```

```
Out[46]: Instrument  Experiment
New A              Atmos_All      0.221824
              Dep1_Reps_Shared    0.285693
              Dep1_Reps_Single    0.186819
              Dep2_Reps           0.245500
              Iodate              0.081538
New B              Atmos_All      0.111670
              Dep1_Reps_Shared    0.400103
              Dep1_Reps_Single    0.252660
              Dep2_Reps           0.136615
              Iodate              0.336945
Old              Atmos_All      0.056200
              Dep1_Reps_Shared    0.511021
              Dep1_Reps_Single    0.099515
              Dep2_Reps           0.627521
              Iodate              0.099029
Name: O2µmol/L, dtype: float64
```

```
In [47]: sci_st.f_oneway(grouped_st_devs['New A'].values, grouped_st_devs['New B'].values, grouped_st_devs['Old'].values)
```

```
Out[47]: F_onewayResult(statistic=0.22295493881859532, pvalue=0.8033921150555711)
```

```
In [ ]:
```