PHYC40600 Physics with Astronomy and Space Science Lab 2; Measuring Surface Tension through the use of the Dispersion Relation of Capillary Waves

Daragh Hollman
daragh.hollman@ucdconnect.ie
(Dated: November 15, 2023)

This report investigates the process of measuring the surface tension of a liquid using the dispersion relation of capillary waves. The surface tension of deionised water was measured to be $\sigma = (47.3 \pm 2)\,\mathrm{mN\ m^{-1}}$. The surface tension of deionised water was also measured as a function of soap concentration. We observe a strong decaying exponential fit to the data as the surface tension increases from concentrations of 0% soap to 20% soap. The surface tension was also measured as a function of salt concentration, an unexpected strong negative correlation was observed.

I. INTRODUCTION

The surface tension of a liquid is a property which is important to understand for a variety of applications such as fluid dynamics [1], understanding soap solutions, and other similar molecules [2], and in medical physics. Surface tension is used widely in medical and bio physics, including the detection of certain conditions such as jaundice, which can be diagnosed by a decrease in the surface tension of urine [3].

II. THEORY

Molecules in a liquid feel a mutual attraction between each other, particularly with its surrounding neighbours. A liquid molecule at the surface has a reduced number of neighbours and is hence in an energetically unfavourable state [4]. To create another new surface costs energy, and a liquid will act to minimise the surface area of the system [4]. This is the process we refer to as surface tension. The surface tension of a liquid can be measured using many techniques, one such method comes from the dispersion relation of capillary waves, which will be investigated in this report.

A. Diffraction

Diffraction is the bending and spreading out of light as it passes through small openings [5]. When multiple gaps are present, the geometry of such spreading results in constructive and destructive interference along particular angles yielding a diffraction pattern at long distances. This can be expanded to a large number of successive gaps known as a diffraction grating [5]. Maximum intensities (fringes) are observed at the following condition:

$$d \sin\theta = m\Lambda \tag{1}$$

where d is the distance between gaps in the grating, θ is the angle at which fringes are observed, Λ is the wavelength of the light, and m is the order of the fringe, with $m = 0$ being at the centre of the diffraction pattern [5]. However, a diffraction grating is not the only way to create a diffraction pattern.

B. Capillary Waves as a Reflection Grating

Reflection can also be used to emulate diffraction and create a diffraction pattern. This is exploited using capillary waves on the surface of a liquid to create a reflection grating [6]. A simple reflection grating is shown in figure 1, a series of closely spaced parallel surfaces reflect light, with no or little reflection between [6]. Capillary waves are waves on the surface of fluids, with wavelengths under a few millimetres [7], and thus act as a reflection grating by only reflecting the light at the peak of the capillary waves creating small gaps in the apparent surface [6]. As shown in figure 2, we regain equation 1 due to a path length difference before and after the arrival of the light rays. In terms of the angles in figure 2, we have:

$$d(\sin\psi - \sin\phi) = m\Lambda \tag{2}$$

where ψ is the angle from the surface normal to the reflected ray, and ϕ is the angle between the reflected ray and the mth order fringe [6]. This can be rewritten in terms of the diffraction angle δ to be:

$$\Lambda = d \sin\frac{\delta}{2}\left(\sin\left(\theta + \frac{\delta}{2}\right) + \sin\left(\theta - \frac{\delta}{2}\right)\right) \tag{3}$$

which can again be rewritten in terms of the wave number of the capillary waves [6]:

$$k = \frac{2\pi}{\Lambda} \sin\frac{\delta}{2}\left(\sin\left(\theta + \frac{\delta}{2}\right) + \sin\left(\theta - \frac{\delta}{2}\right)\right) \tag{4}$$

The dispersion relation can be derived to relate the surface tension of a liquid to the angular frequency and the wave number of these capillary waves [7]. This is shown in the following equation:

$$\sigma = \frac{\rho\omega^2}{k^3} \tag{5}$$

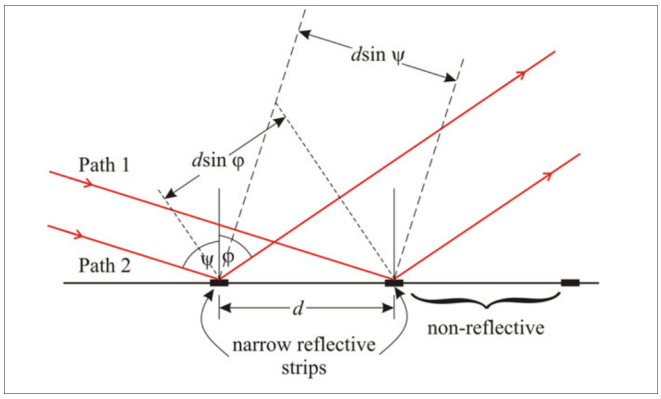


FIG. 1: An example reflection grating [6].

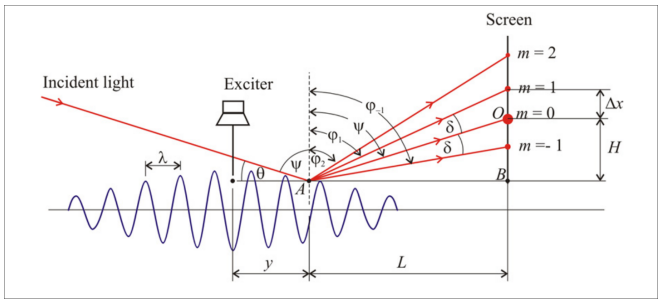


FIG. 2: A sketch of the experiment apparatus showing how the capillary waves are excited and act as a reflection grating [6].

where σ is the surface tension of the liquid, ρ is the density of the liquid, and ω is the angular frequency of the capillary waves.

Therefore, if waves with known angular frequency can be generated in a liquid, and wave number of capillary waves in that liquid can be determined from the diffraction angle - itself determined from the distance between fringes - the surface tension of that liquid can be determined.

III. METHODOLOGY

A. Apparatus

The apparatus was set up following an experiment sketch shown in figure 2 [6]. A small, (Volume $\approx 260\,\text{ml}$), plastic rectangular container was used to hold each liquid tested. A small speaker was used to generate capillary waves in the liquid, placed such that only the tip of the needle was touching the surface of the liquid. This was controlled using a signal generator to be able to vary both the frequency and amplitude as needed. A helium-neon laser ($\lambda = 633\,\text{nm}$) was shone on the surface of the liquid and reflected off the peaks of the capillary waves and onto a wall ($5.6 \pm 0.05\,\text{m}$) away, where a diffraction pattern could be observed.

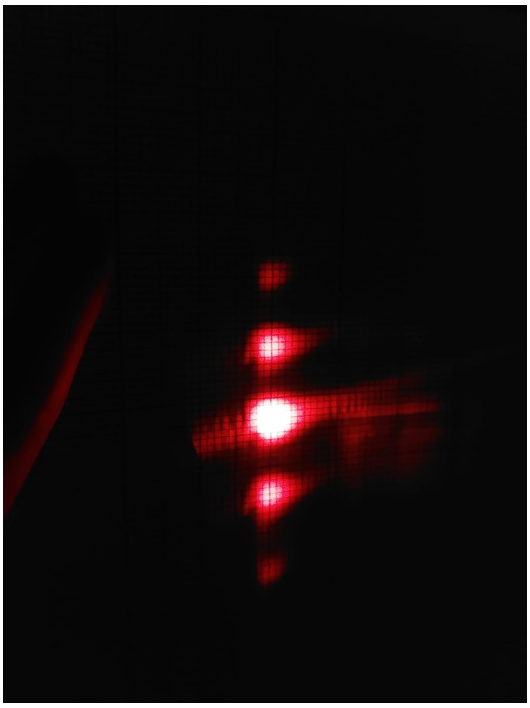


FIG. 3: An example diffraction pattern. Deionised water measured at 190 Hz with fringe spacing $(20 \pm 1)\,\text{mm}$

B. Experimental Methods

The experimental method taken closely followed work done by Nikolic et al. [6]. For each measurement, a diffraction pattern was observed on a sheet of graph paper on the wall, and example pattern is shown in figure 3. This graph paper had resolution of $2 \times 2\,\text{mm}^2$ per grid box, restricting the accuracy of the measurements. Another limiting factor was the size of the fringes, it was difficult to determine the centre of the fringe in all images. A photo of each diffraction patter was taken for each measurement, and the distance between maxima was measured using the graph paper in the images. This was done for a range of frequencies to collect many data points per sample. One sample of deionised water was tested, then measurements were taken on a variety of soap concentrations, lastly measurements were also taken on a variety of NaCl salt concentrations.

C. Analytical Methods

These data were processed to determine the product of the angular frequency squared and the density, $\omega^2\rho$, the wave number was also calculated and these were compared for each data point. In each case, a linear relationship was apparent and characterised using an orthogonal distance regression from Python package `scipy`. By the dispersion relation in equation 5, the slope of this line was an estimate of the surface tension of the sample. In cases where more than one concentration was tested, the

surface tensions were plotted as a function of the concentration to investigate trends.

IV. RESULTS AND ANALYSIS

The method above was carried out for three studies. The first for solely deionised water. The second, a study of how the surface tension changes as a function of soap concentration, and the third, a similar study for the variation with salt concentration.

A sample of the data taken in each measurement is recorded in the first two columns of All data taken for this experiment can be found on GitHub: `https://github.com/daraghhollman/UCD_4thYear_Labs/tree/main/surfaceTension/data`. In all measurements, the uncertainty on the frequency was considered negligible and ignored, and the uncertainty on the spacing was assumed to be 1 mm - only higher for cases where the centre of each fringe was difficult to determine. All included uncertainties were propagated through each calculation to the final data points.

A. Deionised Water

A 250 ml sample of deionised water was used to to measure its surface tension. The container was first rinsed to remove any contaminants or residue from previous measurements and then dried. Following the procedure outlined above, the fringe spacing was measured for 12 frequencies between 100 Hz and 220 Hz. The frequencies and the fringe spacings were used to calculate the wave numbers and angular frequencies for each measurement, which were then recorded in table I. Following the relation in equation 5, these data were plotted (see Figure 4) with the product of the angular frequency squared times the density of the water, against the wave number. A linear orthogonal distance regression fit to the data and the surface tension - the slope of this fit - was determined to be $\sigma = (47.3 \pm 2)\,\mathrm{mN\ m^{-1}}$.

We expect the surface tension of deionised water to follow prior experimental values of $72.2\,\mathrm{mN\ m^{-1}}$ [8] and $72.74\,\mathrm{mN\ m^{-1}}$ at 20 °C. These are far outside the uncertainty of our measurement which could be for a few reasons. Primarily, this is likely due to the high sensitivity in measurements of the height, which are unaccounted for in the data uncertainties. We note that there is also an effect of the temperature on the surface tension of the liquid and our measurements were taken in a room cooler than room temperature, however, cooler temperatures yield larger surface tensions [9] and wouldn't contribute to bringing the values closer together.

f (Hz)	Δx (m)	δ (rad)	k ($\mathrm{m^{-1}}$)	$\omega^2\rho \times 10^{-9}$ ($\mathrm{Hz^2\ kg\ m^{-3}}$)
100	14	0.0025	1990	0.394
110	14	0.0025	1990	0.477
120	15	0.0027	2130	0.567
130	15	0.0027	2130	0.666
140	16	0.0029	2280	0.772
150	18	0.0032	2560	0.887
160	20	0.0036	2840	1.010
170	21	0.0037	2980	1.139
180	21	0.0037	2980	1.277
190	22	0.0039	3130	1.423
200	24	0.0043	3410	1.576
210	25	0.0045	3550	1.738
220	24	0.0043	3410	1.907

TABLE I: Data and calculated parameters for de-ionised water, plotted in figure 4.

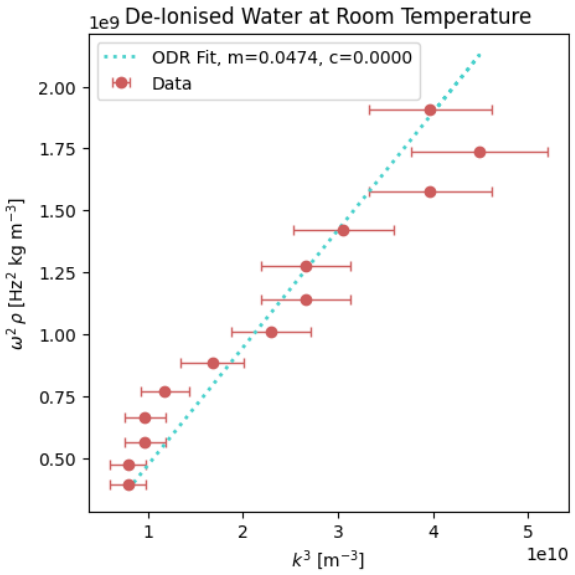


FIG. 4: Dispersion relation plot for deionised water, data (red) plotted from table I. An orthogonal distance regression is determined and plotted for a linear fit, $y = mx + c$, (blue).

B. Soap Concentrations

The surface tension of a soap solution was tested for various concentration. Starting with 250 *textml* of deionised water, concentrated soap - RBS 25 [10] - was added in volume steps of 10, 20, 30, 50, and 70 ml. Data was taken for 9 frequencies between 50 Hz and 120 Hz for each concentration of soap. These data were processed in the same way as the deionised water and a plot for all concentrations is shown in Figure 5. For each concentration the surface tension was recorded and the surface tension as a function of soap concentration was plotted in Figure 6.

These data appear to represent an exponential decay, but also potentially a decreasing linear function. Least

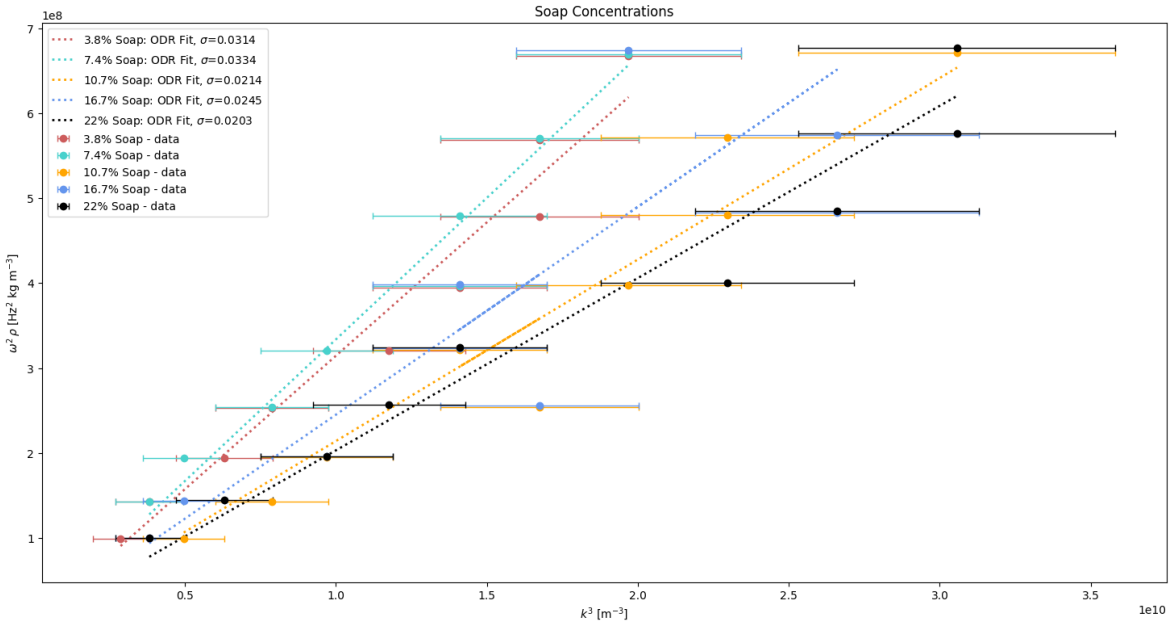


FIG. 5: A collection of dispersion relation plots for varying soap concentrations. An orthogonal distance regression is determined and plotted for a linear fitting function.

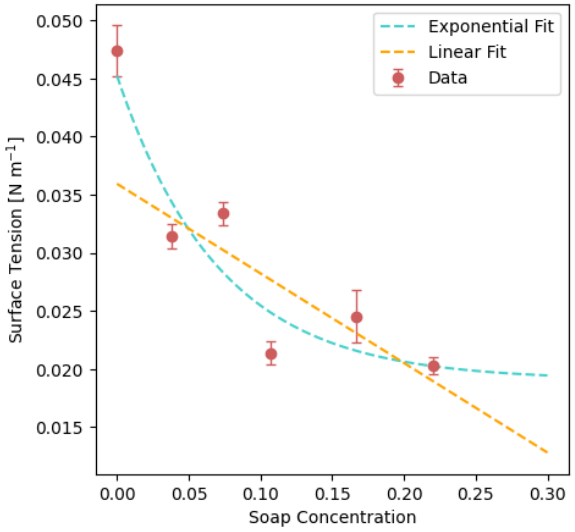


FIG. 6: Surface tension for each soap concentration plotted in red. Two least squares fits were performed on the data, an exponential fit (blue) and a linear fit(orange).

square fits were plotted for both functions as can be seen in figure 6. The exponential fit had form:

$$f(x) = ae^{-bx} + c \tag{6}$$

with parameters $a = (0.026 \pm 0.008)\,\mathrm{N\ m^{-1}}$, $b = (14 \pm 9)$, and $c = (0.019 \pm 0.005)\,\mathrm{N\ m^{-1}}$. The linear fit had the form:

$$f(x) = ax + b \tag{7}$$

with parameters $a = (-0.08 \pm 0.03)\,\mathrm{N\ m^{-1}}$, and $b = (0.036 \pm 0.004)\,\mathrm{N\ m^{-1}}$. Based on the uncertainty of the fitting parameters, we see the linear solution to be a better fit, however we do expect the surface tension to initially drop quickly and then to plateau as a function of concentration, as this is shown in other experimental measurements [11]. We also expect this shape due to the nature of soap molecules having a hydrophobic end and a hydrophilic end [12]. The hydrophobic end of the molecule forces the soap to move to the surface of the liquid causing a reduction in surface tension. There will be a critical point, where the surface no longer has space for more surface molecules, at which point the soap will predominantly form micelles [13], and have little effect on the surface tension.

C. Salt Concentrations

The same was repeated for concentrations of NaCl salt, again starting with 250 ml of deionised water and adding salt in mass steps of 5, 10, 15, 20, 25, 30, and 40 g. When calculating the density of the salt solution, the volume was assumed to stay constant as the effect from adding the salt was negligible. The dispersion relation was again plotted for each concentration of salt as shown in figure 7. Similarly to the soap concentrations these surface tensions were then plotted as a function of salt concentrations, see figure 8, showing a strong linear fit with parameters $a = (-0.04 \pm 0.02)\,\mathrm{N\ m^{-1}}$, and $b = (0.025 \pm 0.001)\,\mathrm{N\ m^{-1}}$. From other measurements, we expect the surface tension to increase with salt concentration [9], however, this is not the trend we observe. We expect this trend may be due to soap contamination leftover in the container having unexpected results in the

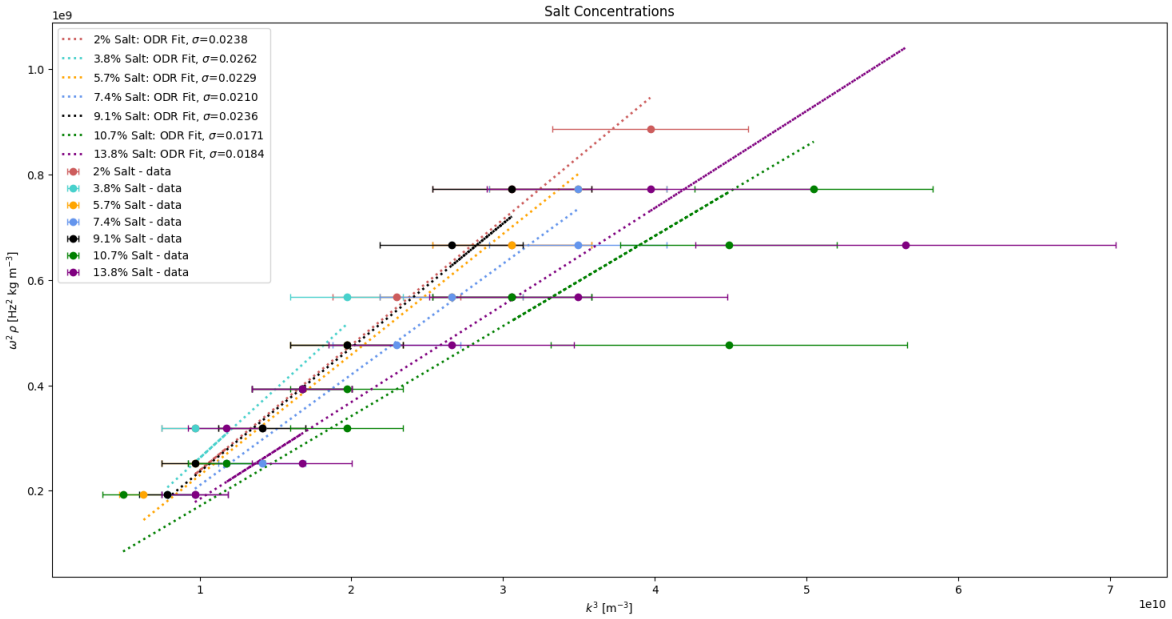


FIG. 7: A collection of dispersion relation plots for varying soap concentrations. An orthogonal distance regression is determined and plotted for a linear fitting function.

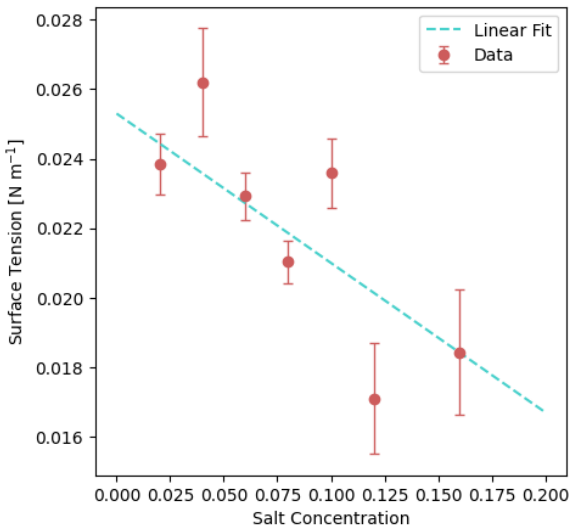


FIG. 8: Surface tension for each salt concentration plotted in red. A least squares fit was performed on the data using a linear fit (orange).

data, however this effect would need to be tested further to confirm.

V. CONCLUSION

The aims of this report were to investigate the process of measuring the surface tension of a liquid using the dispersion relation of capillary waves. The surface tension of deionised water was measured to be $\sigma = (47.3 \pm 2)\,\mathrm{mN\ m^{-1}}$, comparing poorly to theory expected values of $72.2\,\mathrm{mN\ m^{-1}}$ [8] and $72.74\,\mathrm{mN\ m^{-1}}$ at $20\,^{\circ}\mathrm{C}$. This discrepancy is likely due to the high sensitivity in measurements of the height, which are unaccounted for in the data uncertainties. The surface tension of deionised water was also measured as a function of soap concentration. We observe a strong decaying exponential fit to the data as the surface tension increases from concentrations of 0% soap to 20% soap. This relates closely to what is expected from the theory, with other measurements showing similar behaviour [11]. Lastly the surface tension was also measured as a function of salt concentration. A strong negative correlation was observed but goes against what has been shown in prior measurements. We expect this effect to be caused by soap contaminants in the sample although further measurements should be made to try replicate this.

[1] Arnold Krawietz. Surface tension and reaction stresses of a linear incompressible second gradient fluid. *Continuum mechanics and thermodynamics*, 34(4):1027–1050, 2022.

[2] M. M. Bandi. Tension grips the flow. *Journal of fluid mechanics*, 846:1–4, 2018.

[3] Anahita Fathi-Azarbayjani and Abolghasem Jouyban. Surface tension in human pathophysiology and its application as a medical diagnostic tool. *Bioimpacts*, 5(1):29–44, 2015.

[4] John W. M. Bush, Surface Tension Module - Lecture 1: The definition and scaling of surface tension, Department of Mathematics, MIT. Available at `http://web.mit.edu/1.63/www/Lec-notes/Surfacetension/Lecture1.pdf`, accessed on 15/11/23.

[5] R. Nave, Diffraction, Hyperphysics. Available at `http://hyperphysics.phy-astr.gsu.edu/hbase/phyopt/diffracon.html#c1`, accessed on 15/11/23.

[6] D. Nikolic and Lj Neši. Determination of surface tension coefficient of liquids by diffraction of light on capillary waves. *European journal of physics*, 33(6):1677–1685, 2012.

[7] Feng Zhu, Runcai Miao, Chunlong Xu, and Zanzan Cao. Measurement of the dispersion relation of capillary waves by laser diffraction. *American journal of physics*, 75(10):896–898, 2007.

[8] W et al. Heller. Surface tension measurements by means of the "microcone tensiometer. *Journal of colloid and interface science*, 1966.

[9] A guideline of the surface tension of seawater, 2019. The International Association for the Properties of Water and Steam, IAPWS G14-19.

[10] RBS 25 solution concentrate - Specification Sheet, Sigma-Aldrich, available at `https://www.sigmaaldrich.com/IE/en/specification-sheet/SIAL/83460`, accessed on 15/11/23.

[11] Mark Bomberg, Marcin Pazera, Jian Zhang, and Fariborz Haghighat. Weather resistive barriers: Laboratory testing of moisture flow1. 11 2023.

[12] A. S. C. Lawrence. Soap micelles. *Transactions of the Faraday Society*, 31:189–195, 1935.

[13] J. Furukawa. Comparison of block-copolymer with soap in micelle formation. *Colloid and polymer science*, 271(9):852–859, 1993.

Appendix A: Python Notebook

In [ ]:
```
import numpy as np
import matplotlib.pyplot as plt


from uncertainties import ufloat
import uncertainties.unumpy as unp

from scipy.optimize import curve_fit
import scipy.odr as odr

from glob import glob
```

In [ ]:
```
laserSpotHeight = [155, 157]
platformHeightFromGround = [108, 107]
containerHeight = [3.3, 3.6]

heights = [(h1 - (h2 + h3))/100 for h1, h2, h3 in zip(laserSpotHeight, platformHeightFromGround, containerHeight)]
print(f"Height measurements: {heights}")

H = ufloat(np.mean(heights), np.std(heights)) # meters
print(H)

L = ufloat(5.6, 0.05) # meters

lightWavelength = 633e-9 # m
waterDensity = 998.2 # kg m^-3
soapDensity = 1000 * 64.4/60

grazingAngle = unp.arctan2(H, L)

rootPath = "/home/daraghhollman/Main/ucd_4thYearLabs/surfaceTension/data/"
```

```
Height measurements: [0.43700000000000006, 0.4640000000000001]
0.451+/-0.014
```

In [ ]:
```
def LoadData(path):
    data = np.loadtxt(path, skiprows=1)

    frequencies = data[:,0]
    spacings = data[:,1]/1000 # change mm to m
    spacingUncertanties = data[:,2]/1000

    spacings = np.array([ufloat(spacing, uncertainty) for spacing, uncertainty in zip(spacings, spacingUncertanties)])

    return (frequencies, spacings)
```

In [ ]:
```
def GetAngularFrequencies(frequencies):
    angularFrequencies = 2 * np.pi * frequencies
    return angularFrequencies
```

In [ ]:
```
def GetDiffractionAngles(spacings):
    # Input in meters will give an output in degrees

    diffractionAngles = unp.arctan2(spacings, L)

    return diffractionAngles
```

In [ ]:
```
def GetWaveNumbers(diffractionAngles):
    innerBracket = unp.sin(grazingAngle + diffractionAngles / 2) + unp.sin(grazingAngle - diffractionAngles / 2)

    waveNumbers = 2 * (np.pi / lightWavelength) * unp.sin(diffractionAngles / 2) * innerBracket

    return waveNumbers
```

In [ ]:
```
def LinearFunc(p, x):
    m, c = p
    return m * x + c

def PerformODR(function, x, y, xErr):
    model = odr.Model(function)

    data = odr.Data(x, y, wd= 1/xErr)

    odrOutput = odr.ODR(data, model, beta0=[0.03, 0])

    return odrOutput.run()
```

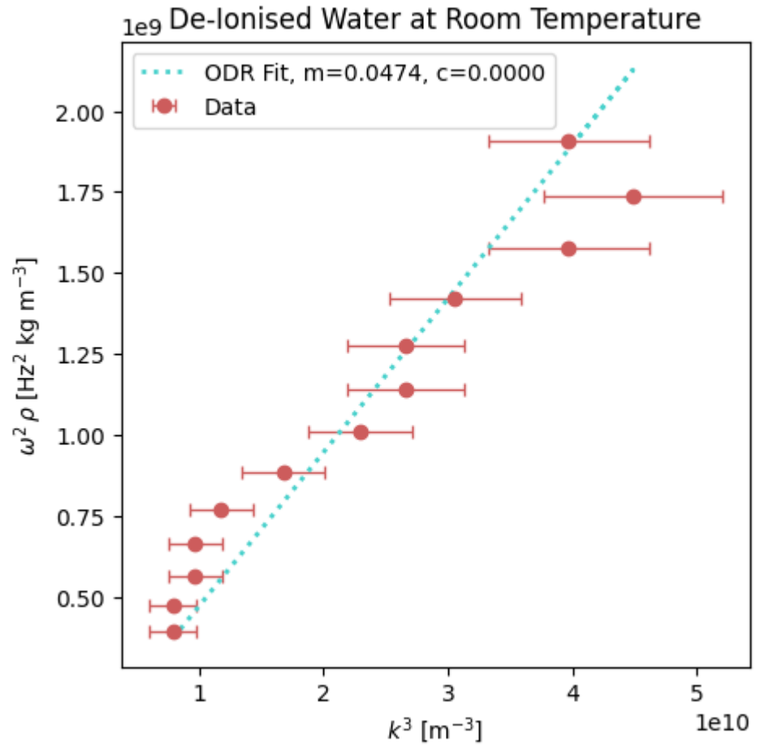
In [ ]:
```
frequencies, spacings = LoadData("/home/daraghhollman/Main/ucd_4thYearLabs/surfaceTension/data/water.txt")

diffractionAngles = GetDiffractionAngles(spacings)
waveNumbers = GetWaveNumbers(diffractionAngles)

angularFrequencies = GetAngularFrequencies(frequencies)

print(f"Diffraction Angles: {unp.nominal_values(diffractionAngles)}")
print(f"Wave Numbers: {unp.nominal_values(waveNumbers)}")
print(f"Angular Frequencies Product: {angularFrequencies**2 * waterDensity}")
```

```
Diffraction Angles: [0.00249999 0.00249999 0.00267857 0.00267857 0.00285714 0.00321427
 0.00357141 0.00374998 0.00374998 0.00392855 0.00428569 0.00446426
 0.00428569]
Wave Numbers: [1989.8521551  1989.8521551  2131.98346613 2131.98346613 2274.11457321
 2558.37612113 2842.63669011 2984.7665735  2984.7665735  3126.89617137
 3411.15445615 3553.28311587 3411.15445615]
Angular Frequencies Product: [3.94073565e+08 4.76829013e+08 5.67465933e+08 6.65984324e+08
 7.72384186e+08 8.86665520e+08 1.00882833e+09 1.13887260e+09
 1.27679835e+09 1.42260557e+09 1.57629426e+09 1.73786442e+09
 1.90731605e+09]
```

In [ ]:
```
def DetermineSurfaceTension(dataPath, density, plotTitle="", showPlot=False, verbose=False):

    # Load and process data
    frequencies, spacings = LoadData(dataPath)

    angularFrequencies = GetAngularFrequencies(frequencies)
    diffractionAngles = GetDiffractionAngles(spacings)

    waveNumbers = GetWaveNumbers(diffractionAngles)

    # Fitting and plotting
    xValues = unp.nominal_values(waveNumbers**3)
    xErr = unp.std_devs(waveNumbers**3)

    yValues = unp.nominal_values(density * angularFrequencies**2)
    yErr = unp.std_devs(density * angularFrequencies**2)

    regression = PerformODR(LinearFunc, xValues, yValues, xErr)

    if verbose: regression.pprint()

    if showPlot:
        fig, ax = plt.subplots(figsize=(5,5))

        ax.errorbar(xValues, yValues, xerr=xErr, fmt="o", color="indianred", capsize=3, linewidth=1, label="Data")

        xRange = np.linspace(np.min(xValues), np.max(xValues), 10)
        ax.plot(xValues, LinearFunc(regression.beta, xValues), color="mediumturquoise", lw=2, ls="dotted", label=f"ODR Fit, m={regression.beta[0]:.4f}, c={regression.beta[1]:.

        ax.set_xlabel("$k^3$ [m$^{-3}$]")
        ax.set_ylabel(r"$\omega^2 \,\rho$ [Hz$^2$ kg m$^{-3}$]")
        ax.legend()
        ax.set_title(plotTitle)

    return (regression.beta[0], regression.sd_beta[0])
```

In [ ]:
```
waterTension = DetermineSurfaceTension(rootPath + "/water.txt", waterDensity, plotTitle="De-Ionised Water at Room Temperature", showPlot=True)
print(waterTension)
```

```
(0.047393156096055974, 0.0021747090378043798)
```

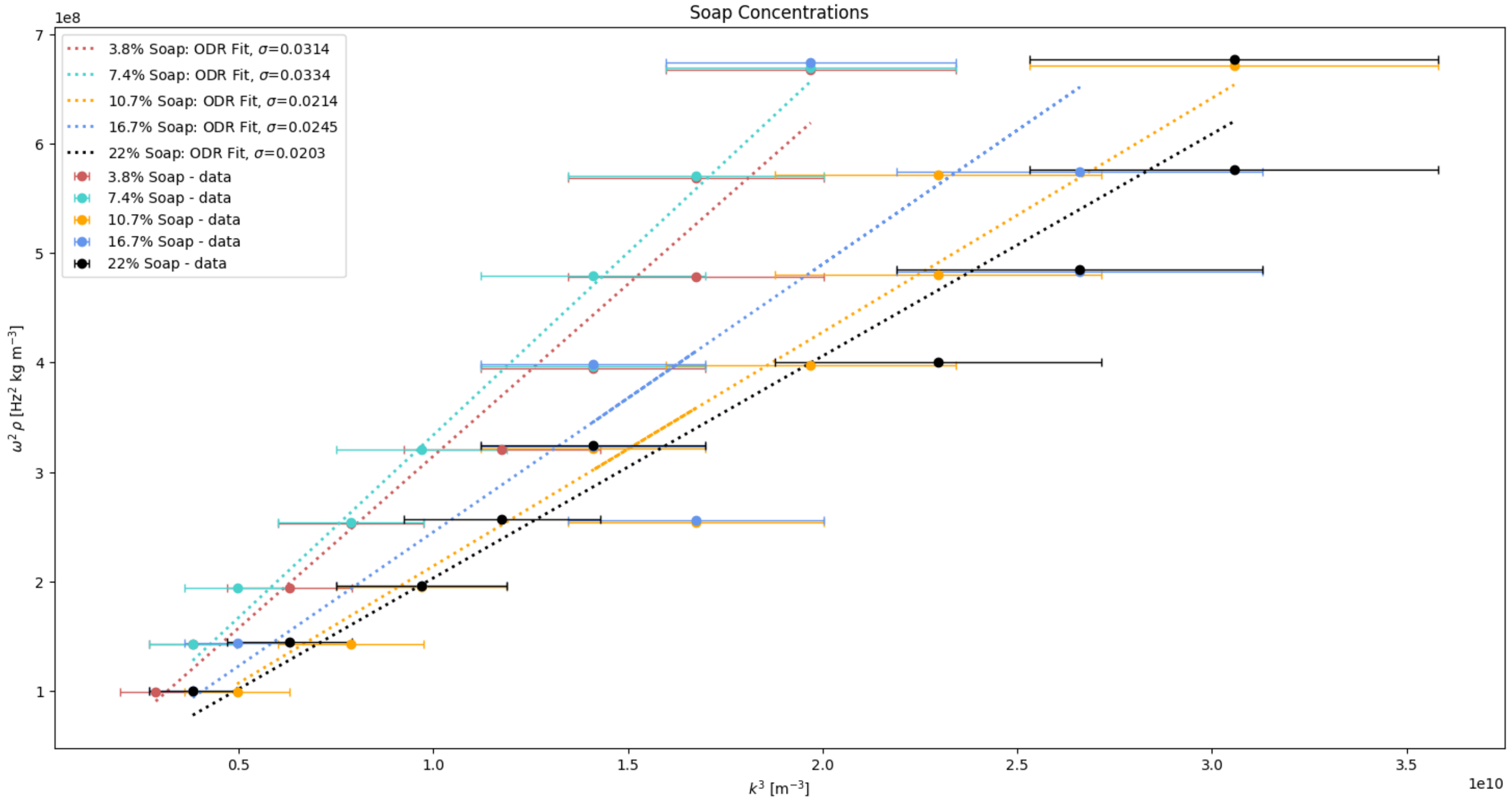

De-Ionised Water at Room Temperature
1e9
ODR Fit, m=0.0474, c=0.0000
Data
2.00
1.75
1.50
1.25
1.00
0.75
0.50
$\omega^2 \rho$ [Hz^2 kg m^{-3}]
1
2
3
4
5
1e10
k^3 [m^{-3}]

Soap

In [ ]:
```
# Multiplot
def Multiplot(dataPath, densities, labels, colours, plotTitle, verbose=False):
    dataPaths = glob(dataPath)
    dataPaths.sort()
    print(dataPaths)

    dataList = []
    for dataPath in dataPaths:
        dataList.append(LoadData(dataPath))

    fig, ax = plt.subplots(figsize=(18,9))


    for data, density, colour, label in zip(dataList,densities, colours ,labels):
        # Load and process data
        frequencies, spacings = data

        angularFrequencies = GetAngularFrequencies(frequencies)
        diffractionAngles = GetDiffractionAngles(spacings)

        waveNumbers = GetWaveNumbers(diffractionAngles)

        # Fitting and plotting
        xValues = unp.nominal_values(waveNumbers**3)
        xErr = unp.std_devs(waveNumbers**3)

        yValues = unp.nominal_values(density * angularFrequencies**2)
        yErr = unp.std_devs(density * angularFrequencies**2)

        regression = PerformODR(LinearFunc, xValues, yValues, xErr)

        if verbose: regression.pprint()

        ax.errorbar(xValues, yValues, xerr=xErr, fmt="o", color=colour, capsize=3, linewidth=1, label=label + " - data")

        xRange = np.linspace(np.min(xValues), np.max(xValues), 10)
        ax.plot(xValues, LinearFunc(regression.beta, xValues), color=colour, lw=2, ls="dotted", label=label + f": ODR Fit, $\sigma$={regression.beta[0]:.4f}")

    ax.set_xlabel("$k^3$ [m$^{-3}$]")
    ax.set_ylabel(r"$\omega^2 \,\rho$ [Hz$^2$ kg m$^{-3}$]")
    ax.legend()
    ax.set_title(plotTitle)
```

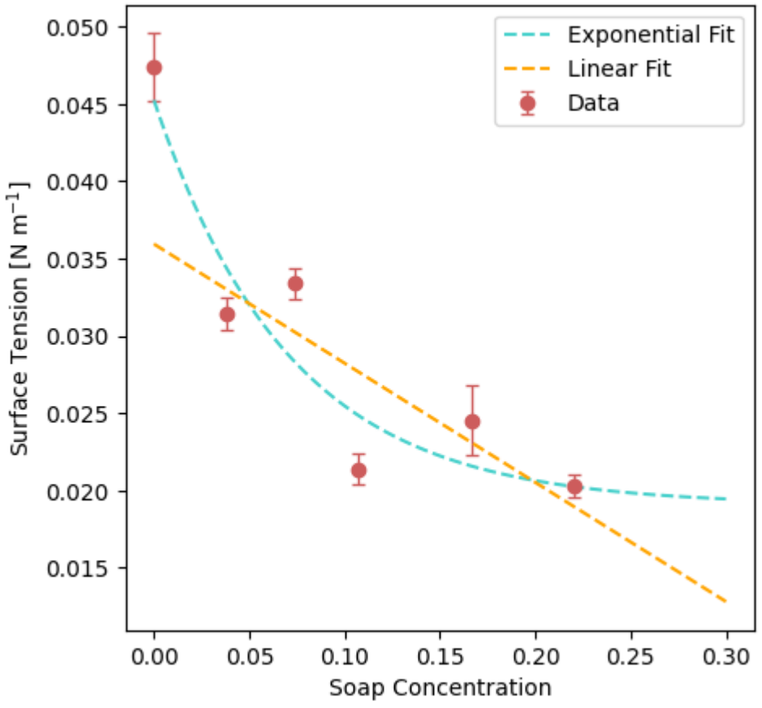
In [ ]:
```
soapRatios = np.array([0.038, 0.074, 0.107, 0.167, 0.22])
soapDensities = soapRatios * soapDensity + ([1]*5 - soapRatios) * waterDensity

colours = ["indianred", "mediumturquoise", "orange", "cornflowerblue", "black"]

Multiplot(rootPath + "/soap_250_*.txt", soapDensities, [r"3.8% Soap", r"7.4% Soap", r"10.7% Soap", r"16.7% Soap", r"22% Soap"], colours, "Soap Concentrations")
```

```
['/home/daraghhollman/Main/ucd_4thYearLabs/surfaceTension/data/soap_250_10.txt', '/home/daraghhollman/Main/ucd_4thYearLabs/surfaceTension/data/soap_250_20.txt', '/home/daraghhollman/Main/ucd_4thYearLabs/surfaceTension/data/soap_250_30.txt', '/home/daraghhollman/Main/ucd_4thYearLabs/surfaceTension/data/soap_250_50.txt', '/home/daraghhollman/Main/ucd_4thYearLabs/surfaceTension/data/soap_250_70.txt']
```


Soap Concentrations
1e8
3.8% Soap: ODR Fit, σ=0.0314
7.4% Soap: ODR Fit, σ=0.0334
10.7% Soap: ODR Fit, σ=0.0214
16.7% Soap: ODR Fit, σ=0.0245
22% Soap: ODR Fit, σ=0.0203
3.8% Soap - data
7.4% Soap - data
10.7% Soap - data
16.7% Soap - data
22% Soap - data
$\omega^2 \rho$ [Hz^2 kg m^{-3}]
k^3 [m^{-3}]
1e10

In [ ]:
```
soapTension_250_10 = DetermineSurfaceTension(rootPath + "/soap_250_10.txt", soapDensities[0], plotTitle=r"0.04% soap at Room Temperature", showPlot=False)
soapTension_250_20 = DetermineSurfaceTension(rootPath + "/soap_250_20.txt", soapDensities[1], plotTitle=r"0.08% soap at Room Temperature", showPlot=False)
soapTension_250_30 = DetermineSurfaceTension(rootPath + "/soap_250_30.txt", soapDensities[2], plotTitle=r"0.12% soap at Room Temperature", showPlot=False)
soapTension_250_50 = DetermineSurfaceTension(rootPath + "/soap_250_50.txt", soapDensities[3], plotTitle=r"0.20% soap at Room Temperature", showPlot=False)
soapTension_250_70 = DetermineSurfaceTension(rootPath + "/soap_250_70.txt", soapDensities[4], plotTitle=r"0.28% soap at Room Temperature", showPlot=False)

soapRatios = np.array([0, 0.038, 0.074, 0.107, 0.167, 0.22])

soapTensions = np.array([waterTension, soapTension_250_10, soapTension_250_20, soapTension_250_30, soapTension_250_50, soapTension_250_70])

# FITTING
def ExpCurve(x, A, B, C):
    return A * np.exp(B * x) + C

def LinearCurve(x, m, c):
    return m * x + c

# Exponential fit
expPars, expCov = curve_fit(ExpCurve, soapRatios, soapTensions[:, 0], [0.05, -5, 0], sigma=soapTensions[:, 1])

# Linear fit
linearPars, linearCov = curve_fit(LinearCurve, soapRatios, soapTensions[:, 0], sigma=soapTensions[:, 1])

# PLOTTING

fig, ax = plt.subplots(figsize=(5,5))

for ratio, tension in zip(soapRatios, soapTensions):
    if ratio == soapRatios[0]:
        label = "Data"
    else: label = ""
    ax.errorbar(ratio, tension[0], yerr=tension[1], fmt="o", color="indianred", capsize=3, linewidth=1, label=label)

xRange = np.linspace(0, 0.3, 100)
ax.plot(xRange, ExpCurve(xRange, expPars[0], expPars[1], expPars[2]), color="mediumturquoise", ls="--", label="Exponential Fit")
ax.plot(xRange, LinearCurve(xRange, linearPars[0], linearPars[1]), color="orange", ls="--", label="Linear Fit")

print(expPars)
print(np.sqrt(np.diag(expCov)))
print()
print(linearPars)
print(np.sqrt(np.diag(linearCov)))

ax.set_xlabel("Soap Concentration")
ax.set_ylabel("Surface Tension [N m$^{-1}$]")
ax.legend()
```

Out[ ]:
```
[  0.0261634  -14.07895824   0.01906944]
[7.61451096e-03 9.24500755e+00 4.81135435e-03]

[-0.07705898  0.03593315]
[0.02739957 0.00395944]
<matplotlib.legend.Legend at 0x7ff44538dd50>
```

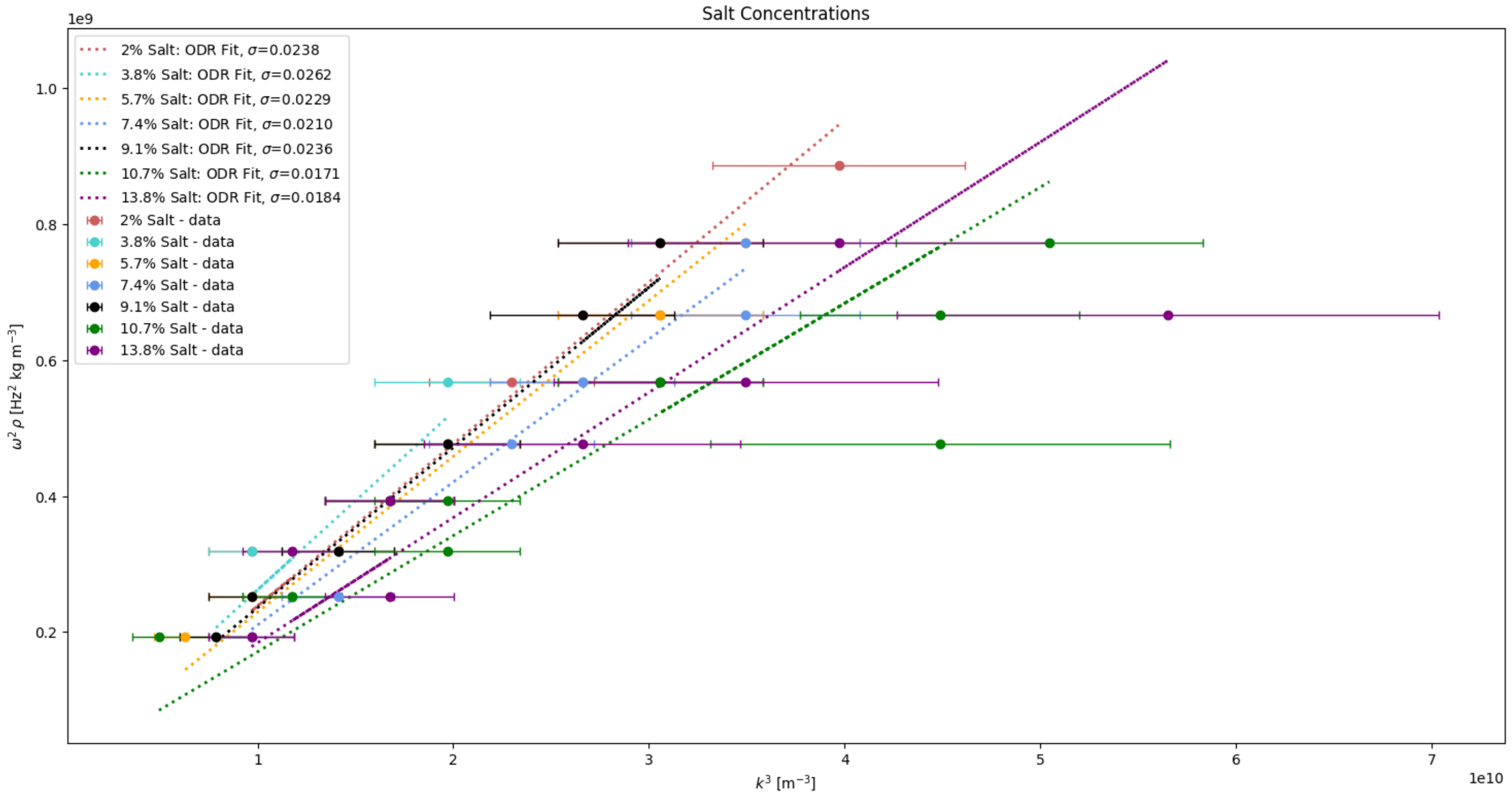


Salt

In [ ]:

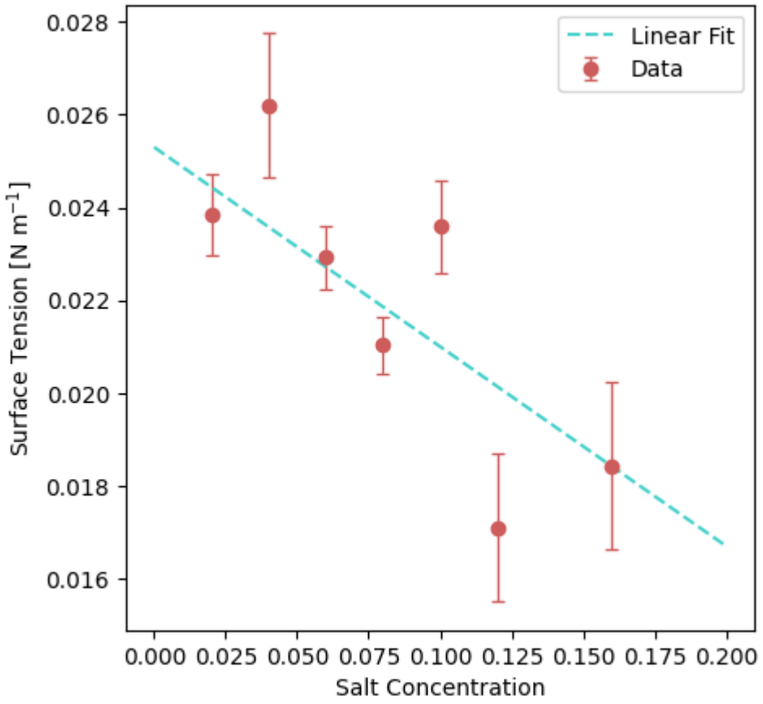
```
saltMasses = [0.005, 0.010, 0.015, 0.020, 0.025, 0.030, 0.040]
saltDensities = [(0.250 * waterDensity + saltMass) / 0.250 for saltMass in saltMasses] # assuming volume is constant

colours = ["indianred", "mediumturquoise", "orange", "cornflowerblue", "black", "green", "purple"]

Multiplot(rootPath + "/salt_250_*.txt", saltDensities, [r"2% Salt", r"3.8% Salt", r"5.7% Salt", r"7.4% Salt", r"9.1% Salt", r"10.7% Salt", r"13.8% Salt"], colours, "Salt Conce
```

```
['/home/daraghhollman/Main/ucd_4thYearLabs/surfaceTension/data/salt_250_05.txt', '/home/daraghhollman/Main/ucd_4thYearLabs/surfaceTension/data/salt_250_10.txt', '/home/daraghh
ollman/Main/ucd_4thYearLabs/surfaceTension/data/salt_250_15.txt', '/home/daraghhollman/Main/ucd_4thYearLabs/surfaceTension/data/salt_250_20.txt', '/home/daraghhollman/Main/ucd
_4thYearLabs/surfaceTension/data/salt_250_25.txt', '/home/daraghhollman/Main/ucd_4thYearLabs/surfaceTension/data/salt_250_30.txt', '/home/daraghhollman/Main/ucd_4thYearLabs/su
rfaceTension/data/salt_250_40.txt']
```


Salt Concentrations
1e9
2% Salt: ODR Fit, σ=0.0238
3.8% Salt: ODR Fit, σ=0.0262
5.7% Salt: ODR Fit, σ=0.0229
7.4% Salt: ODR Fit, σ=0.0210
9.1% Salt: ODR Fit, σ=0.0236
10.7% Salt: ODR Fit, σ=0.0171
13.8% Salt: ODR Fit, σ=0.0184
2% Salt - data
3.8% Salt - data
5.7% Salt - data
7.4% Salt - data
9.1% Salt - data
10.7% Salt - data
13.8% Salt - data
$\omega^2 \rho$ [Hz^2 kg m^{-3}]
1.0
0.8
0.6
0.4
0.2
1
2
3
4
5
6
7
k^3 [m^{-3}]
1e10

In [ ]:
```
saltTension_250_05 = DetermineSurfaceTension(rootPath + "/salt_250_05.txt", saltDensities[0], showPlot=False)
saltTension_250_10 = DetermineSurfaceTension(rootPath + "/salt_250_10.txt", saltDensities[1], showPlot=False)
saltTension_250_15 = DetermineSurfaceTension(rootPath + "/salt_250_15.txt", saltDensities[2], showPlot=False)
saltTension_250_20 = DetermineSurfaceTension(rootPath + "/salt_250_20.txt", saltDensities[3], showPlot=False)
saltTension_250_25 = DetermineSurfaceTension(rootPath + "/salt_250_25.txt", saltDensities[4], showPlot=False)
saltTension_250_30 = DetermineSurfaceTension(rootPath + "/salt_250_30.txt", saltDensities[5], showPlot=False)
saltTension_250_40 = DetermineSurfaceTension(rootPath + "/salt_250_40.txt", saltDensities[6], showPlot=False)

saltTensions = np.array([saltTension_250_05, saltTension_250_10, saltTension_250_15, saltTension_250_20, saltTension_250_25, saltTension_250_30, saltTension_250_40])

saltRatios = np.array([0.02, 0.04, 0.06, 0.08, 0.10, 0.12, 0.16])

# FITTING
pars, cov = curve_fit(LinearCurve, saltRatios, saltTensions[:, 0], sigma=saltTensions[:, 1])

print(pars)
print(np.sqrt(np.diag(cov)))

# PLOTTING

fig, ax = plt.subplots(figsize=(5,5))

for ratio, tension in zip(saltRatios, saltTensions):
    if ratio == saltRatios[0]:
        label = "Data"
    else: label = ""

    ax.errorbar(ratio, tension[0], yerr=tension[1], fmt="o", color="indianred", capsize=3, linewidth=1, label=label)

xRange = np.linspace(0, 0.2, 100)
ax.plot(xRange, LinearCurve(xRange, pars[0], pars[1]), color="mediumturquoise", ls="--", label="Linear Fit")

ax.set_xlabel("Salt Concentration")
ax.set_ylabel("Surface Tension [N m$^{-1}$]")
ax.legend()
```

Out[ ]:
```
[-0.04299809  0.02529889]
[0.01915801 0.00148575]
<matplotlib.legend.Legend at 0x7ff444d39990>
```


Linear Fit
Data
Surface Tension [N m^{-1}]
0.028
0.026
0.024
0.022
0.020
0.018
0.016
0.000 0.025 0.050 0.075 0.100 0.125 0.150 0.175 0.200
Salt Concentration

In [ ]: