Bulk Modulus

Jonatan Langer

December 10, 2022

1 Introduction

It is the purpose of this report to show how the Bulk modulus of a liquid behaves at different temperatures. To accomplish this goal this report presents fundamental theory related to modeling physical systems, Specifically the theories of linear response and energy bonds. It then uses these techniques to derive a model for both empty and filled bulk transducers. Using this model recorded data is fit and the bulk modulus at these various temperatures are found and presented.

2 Theoretical background

2.1 Linear Response Theory

The linear response function defines the relationship between an input, $\Phi(t)$, and an output, $\gamma(t)$, thereby describing some system. Generally the output of a system is affected by the previous values of the inputs. Therefore the function,

$$\gamma(t) = \int_0^\infty R(t')\dot{\phi}(t-t')dt', \tag{1}$$

relates the input and output by the response function, $R(t')$, by the weighted contribution of all previous inputs, $\phi(t-t')$. Linear response theory assumes that at a small enough time step the relationship between input and output is linear, ie. $R(t')$ is a linear function.

When the input is a harmonic function, $\phi(t) = \phi_\omega e^{i\omega t}$. The time derivative of the input becomes, $\dot{\phi}(t) = i\omega\phi_\omega e^{i\omega t}$. Inserting the harmonic input into equation 1 therefore results in,

$$\gamma(t) = i\omega\phi_\omega e^{i\omega(t)} \int_0^\infty R(t')e^{-i\omega t'}dt'. \tag{2}$$

The harmonic output, $\gamma_\omega = \frac{\gamma(t)}{e^{i\omega t}}$, can be inserted resulting in the very neat equation:

$$\gamma_\omega = \hat{R}(\omega)\phi_\omega, \tag{3}$$

where,

$$\hat{R}(\omega) = i\omega \int_0^\infty R(t')e^{-i\omega t'}dt'. \tag{4}$$

This response function therefore relates a harmonic input with a harmonic output. By working in the frequency domain integration and differentiation become simple linear operators.

In practice response functions can be used to describe the dynamics of many physical phenomena. Some response functions that are commonly used are the Impedance, Compliance, Admittance, and Modulus. In electrical circuits the linear response function most appropriate for a specific input-output combination is summarized in table 1 below.

Response	Input	Output
Impedence Z	Current	Voltage
Compliance J	Voltage	Charge
Admittance Y	Voltage	Current
Modulus G	Charge	Voltage

Table 1: Response functions with their respective relations for the case of an electrical circuit.

As can be seen in table 1 the response functions are all inherently related to each other. Their relationships are described as:

$$Z = \frac{1}{Y} = \frac{1}{i\omega J} = i\omega G \tag{5}$$

2.2 Energy Bonds and Modeling

The idea of linear response can be expanded using the principle of energy bonds. For some physical phenomenon, its description can take the form of energy transfer from one system to another thereby describing its dynamics. This can be done by introducing two variables effort, e, and flow, f, whose product will give the transferred energy per time from one system to another. The flow of a phenomenon will change sign with time reversal and the effort will not. In the mechanical case effort is the force while flow is the resulting velocity. If time were reversed the velocity would change sign but not the force.

The energy bond formalism is powerful because it allows the description of mechanical systems as analogous electrical systems. For example, ohms law, $U = \hat{Z}I$, has effort, $e = U$, and flow, $f = I$ and is related by the Impedence, $\hat{Z}$. Because effort is also analogous to Force and flow is also analogous to velocity ohms law becomes analogous to $F = \hat{Z}v$ which can be thought of as a force of friction dependent on the velocity related again by the Impedance. This is summarized in table 2 below.

Energy Domain	Effort	Flow
Mechanical Translation	Force (F)	Linear Velocity (v)
Electrical	Voltage (U)	Current (I)

Table 2: Energy Bond analogy for mechanical and electrical systems

By applying these analogies to a physical system it is possible to build a circuit with analogous dynamics. We can take the mechanical case of a spring and a dash-pot connected in series as an example, seen in figure 1

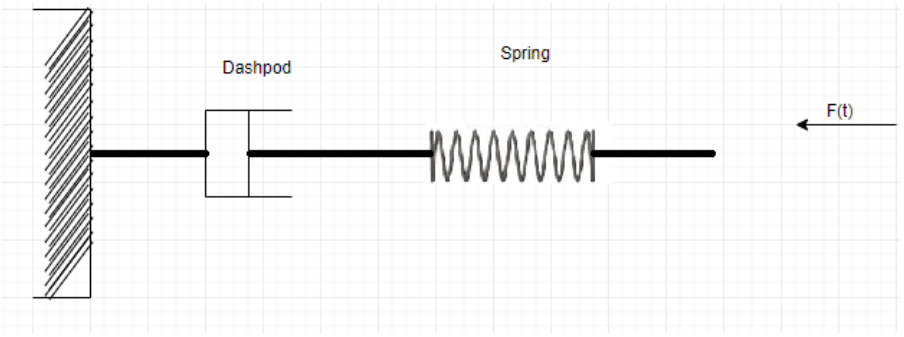

Figure 1: Mechanical System of a spring connected to a dash-pot in series.

The equations for force of a dash pot and spring are,

$$F_{dp} = Rv \tag{6}$$
$$F_s = kx \tag{7}$$

respectively. The spring can therefore be described by a capacitor which is defined as

$$Q = CU \qquad\qquad U = \frac{1}{C}Q, \tag{8}$$

where $\frac{1}{C}$ is analogous to the spring constant k. The dash pot, as shown previously, is analogous to a resistor. In this scenario the Force, F, experienced by the components remains the same while the displacement, x, varies. Thus, in the analogous electrical circuit the components would share the same voltage but experience different currents. According to kirchoffs laws they must thus be connected in parallel. The resulting circuit is shown below in figure 2.

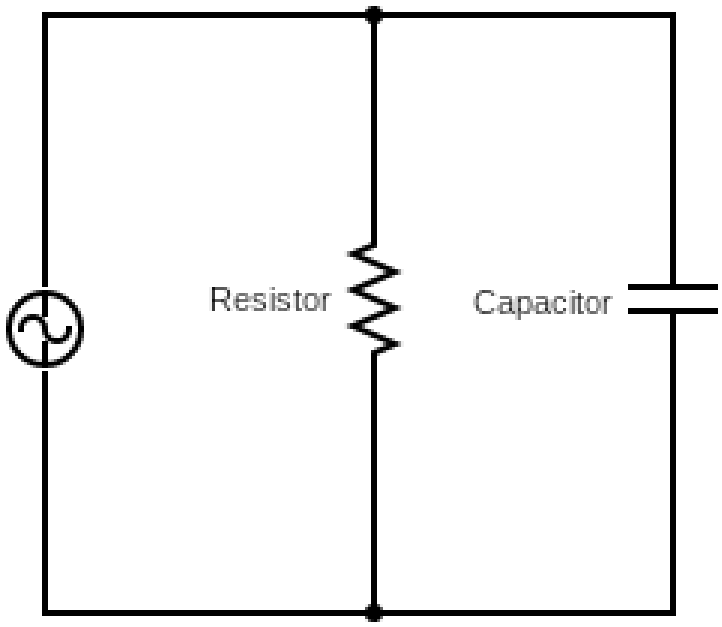

Figure 2: Electrical circuit of a resistor and capacitor in parallel, this circuit is analogous to a dashpot and spring connected in series.

From here the total impedance, and thus all the other response functions, can be derived. This provides a powerful tool to analyze and model more complex mechanical systems, seen in the rest of this report.

3 Experiment

3.1 Description of experiment

The experiment for this report involved studying the bulk modulus of a liquid substance inside of a bulk transducer seen in figure 3 below. The experiment consisted of running a frequency scan of the capacitance at 6 temperatures ranging from 300k to 250K. In order to establish the bulk modulus of a liquid filled transducer we ran the experiment on an empty transducer first to establish a control to compare with then the filled transducer.

Both the empty and filled experiments followed the same protocol. First the bulk transducer was brought to the initial temperature of $300K$ over the course of an hour. Then two sets of logarithmic and linear scans were preformed before another hour long change brought the bulk transducer to its next target temperature this was repeated for all temperatures in the range stated above. In the intervals between frequency scans while the bulk transducer was coming to temperature the measuring device ran a time series scan at a fixed frequency of the capacitance.

3.2 PBG model

Seen in figure 3 above the bulk transducer, or PBG shell, consists of a circuit over a special material called a Piezo-electric ceramic. This material has the interesting behavior of transforming when under an electric

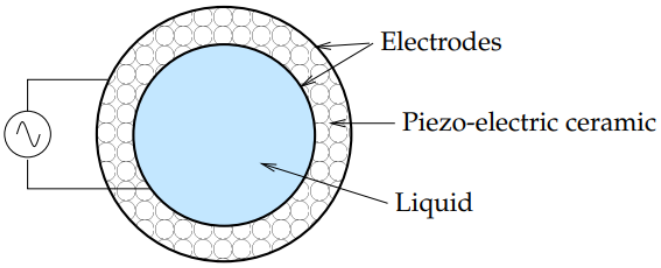

Figure 3: Shematic of bulk transducer from [1]

charge thereby providing a mechanical output to an electrical input.

Using the same principles from section 2 we can arrive at a model for the BPG shell. We start with an analogous electrical circuit for the PBG shell, seen below in figure 4.

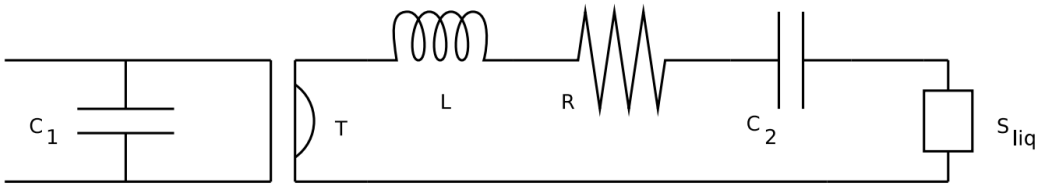

Figure 4: Electrical circuit analogous to the behavior of the PBG shell from [2]

To find the empty capacitance of the model above we consider this circuit shorted over the liquid component. By utilizing Kirchhoff's laws it is easier to begin with the impedence, which can be found as,

$$\frac{1}{Z(\omega)_{total}} = i\omega C_1 + T^2 \frac{1}{\frac{1}{i\omega C_2} + i\omega L + R}. \tag{9}$$

The relationship between impedance and compliance is well understood as $J = \frac{1}{i\omega Z}$ covered in section 1. Inserting this into equation 9 results in

$$J(\omega) = C_1 + \frac{T^2 C_2}{1 - \omega^2 C_2 L + i\omega C_2 R}. \tag{10}$$

We define $C_0 = C_1 + T^2 C_2$ and $C_\infty = C_1$ based on the limiting behavior of the compliance. Further from the resonance frequency of a harmonic oscillator and the definition of the quality factor we have $\omega_0 = \frac{1}{\sqrt{LC_2}}$ and $Q = \frac{1}{R}\sqrt{\frac{L}{C_2}}$. Substituting these factors into equation 10 results in,

$$J(\omega)_{empty} = C_\infty + \frac{C_0 - C_\infty}{1 - \left(\frac{\omega}{\omega_0}\right)^2 + i\frac{\omega}{\omega_0}\frac{1}{Q}}. \tag{11}$$

Equation 11 delivers an equation for the empty bulk transducer's capacitance with only four unknown variables. In a similar fashion the stiffness of the liquid can be added and the compliance is found as,

$$J(\omega)_{full} = C_\infty + \frac{C_0 - C_\infty}{1 - (\frac{\omega}{\omega_0})^2 + i\frac{\omega}{\omega_0}\frac{1}{Q} + \frac{S_{liq}}{L\omega^2}} \tag{12}$$

In this equation we add another unknown variable. However the bulk Modulus $K = -V\frac{\delta p}{\delta V}$ can be found by solving the analytical solution of S_{liq} for K. Where

$$S_{liq}(\omega) = \frac{K}{\frac{4\pi}{3} r_0^3} g(\frac{\omega}{\omega_0}), \tag{13}$$

where $\omega_0 = \sqrt{\frac{K}{\rho}}\frac{1}{r_0}$ and

$$g(x) = -\frac{1}{3}\frac{x^2\sin(x)}{x\cos(x) - \sin(x)} \tag{14}$$

The inductance L can also found through measurements of the bulk transducer. This is due to the inductance being analogous to mass. Therefore the inductance can be written in terms of pressure change.

$$dP = L\frac{d^2V}{dt} \tag{15}$$

Because the sphere of the bulk transducer expands and contracts uniformly the change in volume can be thought of as the change in surface area of the inner shell.

$$dV = A\frac{d^2x}{dt} \tag{16}$$

With pressure being equal to force over area we can find $dp = \frac{m}{A}\frac{d^2x}{dt}$ and thus L can be found as,

$$L = \frac{m}{A^2} \tag{17}$$

Therefore the inductance is measure able and parameters C_0, C_∞, ω_0, and Q can be found from fitting the empty transducer data. Then using equations 12, 13, and 14 we can fit the bulk modulus and density to our data of the full transducer.

4 Results and Analysis

For each temperature the empty model was fit to find the necessary parameters. The fits for the two extreme temperatures are shown below in figure 5 below.

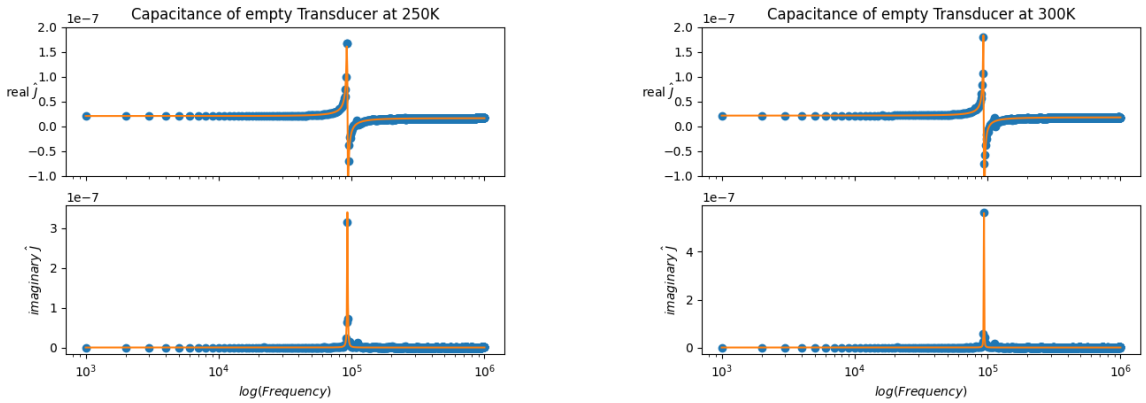

Figure 5: Real and imaginary fits of empty transducer data at 250 Kelvin and 300 Kelvin, visual inspection reveals that the fits seem to represent the data well.

While the fits above seem to represent the data well it can be seen from figure 6 below that there may be something off in the fits conducted.

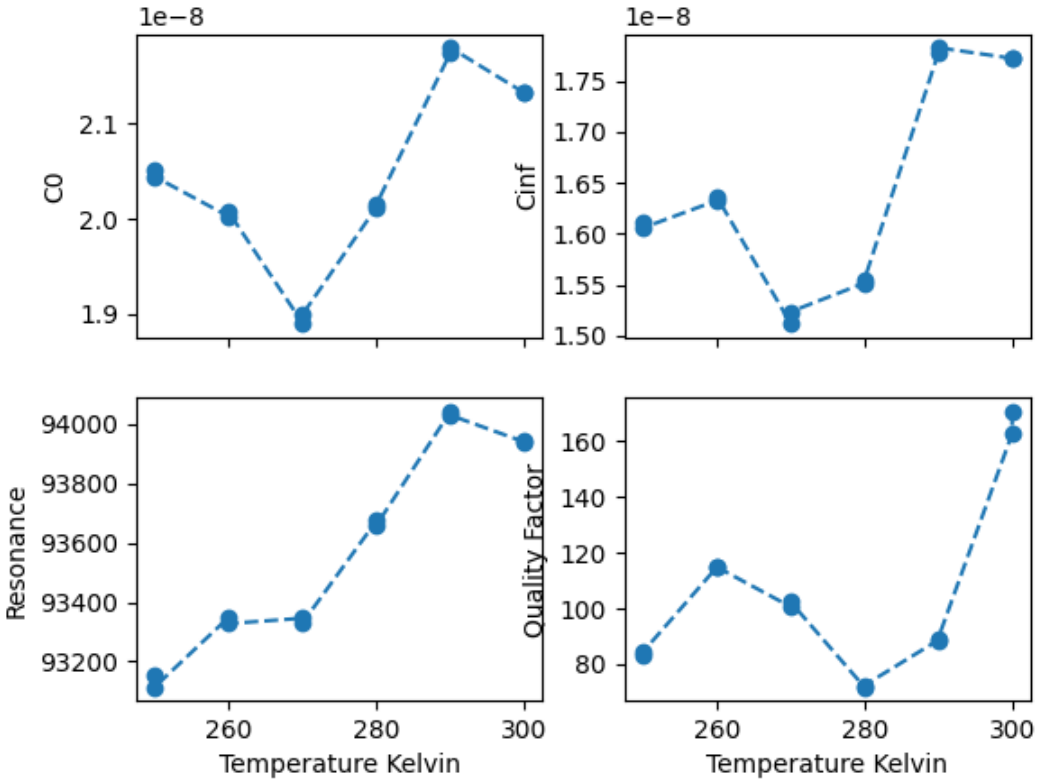

Figure 6: Fit parameters plotted against temperature. It can be seen that a non linear relationship is present in these parameters this may be from errors in the fitting algorithm.

Using these fit parameters the measured full capacitance was fit using equations 11, 13, and 14. In the following plots both Bulk Modulus and density ,ρ, were fit. The fitted data plots are shown in figure 7

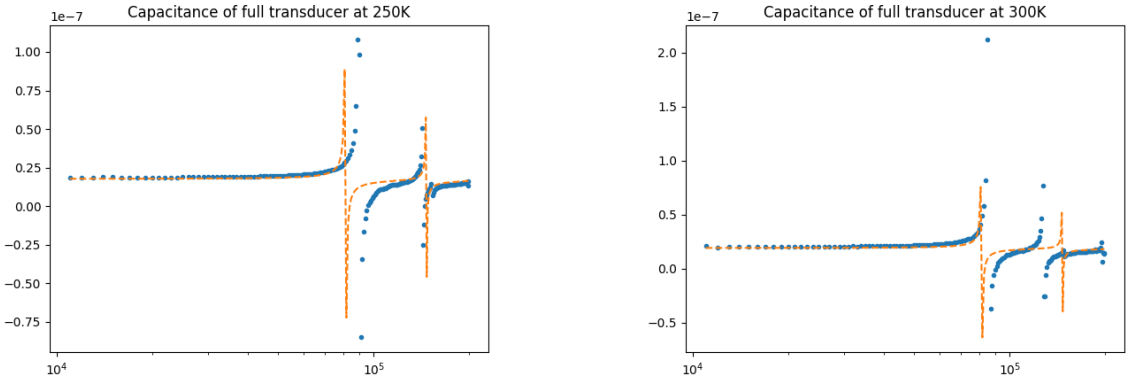

Figure 7: Fit plots of measured capacitance of the full transducer at 250 kelvin and 300 kelvin. It is shown that the fit does not line up exactly with the data in either case.

Fitting the bulk modulus and density at each temperature resulted in the following relationship between these fit parameters and temperature.

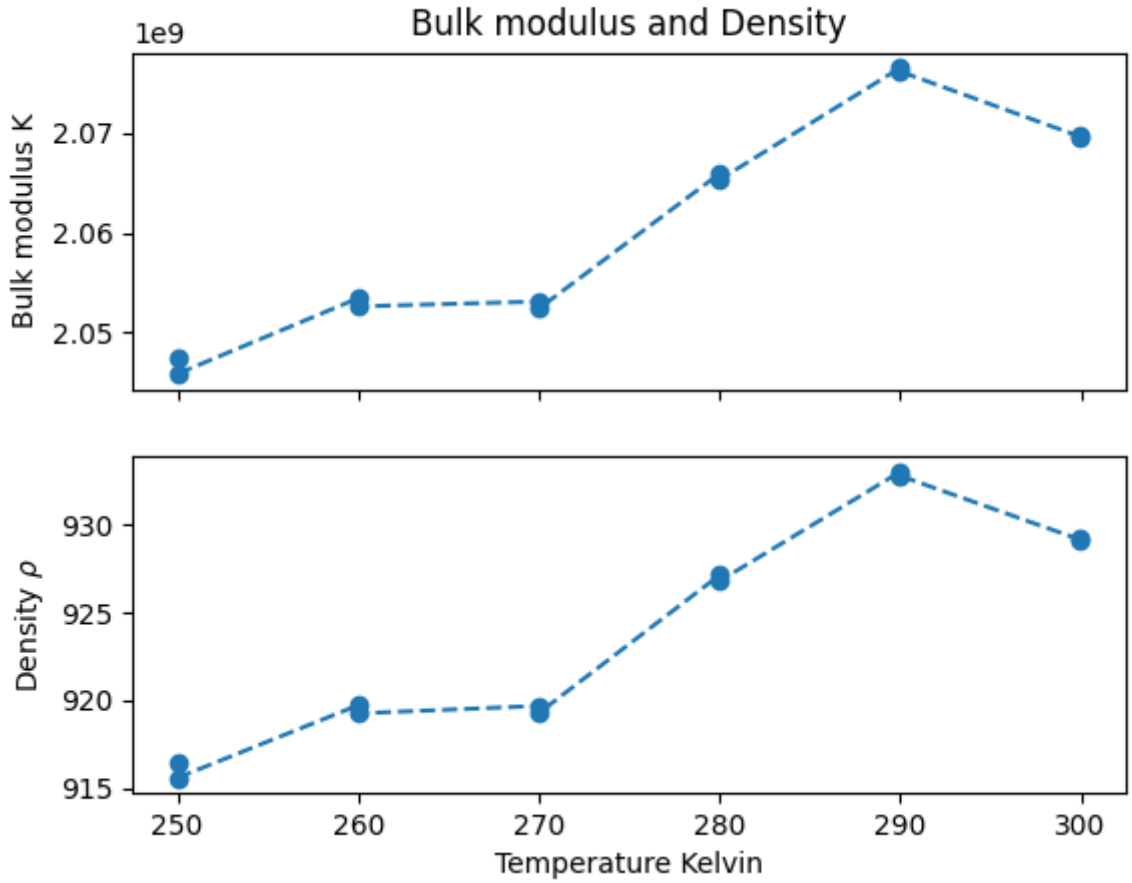

Figure 8: BUlk modulus K and density ρ is plotted against temperature. Here it seems that both increase with temperature

We can see from figure 8 that as temperature decreased both the bulk modulus and density decreased according to our fits.

5 Conclusion

This experiment with the bulk transducer was the culmination of all other experiments done in this course. The technique of modelling through energy bonds and linear response was extremely useful in determining the electrical and mechanical parts of the PBG model. Although the equations do describe the behavior and dynamics of capacitance in these situations limitations in python, both algorithmic and in regards to my own skill level, resulted with parameters that did not quite fit the full transducer. This is unfortunate but can be understood by the extreme sensitivity to initial guesses found in the scipy.curve_fit package. Further work would hopefully fix the runtime errors experienced when attempting to fit parameters. Regardless of these errors the parameters presented in the analysis were the sets that best fit the capacitance visually while remaining near the expected result of $K = 2 \cdot 10^9$.

References

[1] Tina Hecksher. "Relaxation in Supercooled Liquids Linear and Nonlinear, Mechanical and Dielectric Studies of Molecular Liquids". PhD thesis. Roskilde Universitet, p. 305 (cit. on p. 4).

[2] Tage Christensen. "The Transducer Principle . Measuring Bulk Modulus of a Liquid . Physical Techniques". In: (2020), pp. 1–35 (cit. on p. 4).

Appendix A: Code

```
#import importlib
%matplotlib notebook
# %matplotlib --list
import matplotlib.pyplot as plt
import numpy as np
from scipy.optimize import curve_fit
# Load primitive scan_tools from ime_pylib
import scan_tools as st
# set load patp for data
st.set_path(".")

# define the default filename, just for easyines
filename="PBG_q3_empty_bulk_";
filename2 = "PBG_q3_full_bulk_";
# load data
res_sp_empty_lin = st.scan_load(547,578,filename=filename,top_function='measure_c_bridge_lin')
#res_sp_empty_log =st.scan_load(547,578,filename=filename,top_function='measure_c_multi_lcr')

res_sp_full_lin = st.scan_load(583,614,filename=filename2,top_function='measure_c_bridge_lin')
#res_sp_full_log =st.scan_load(583,614,filename=filename,top_function='measure_c_multi_lcr')
print(res_sp_empty_lin[0]._fieldnames)
print(res_sp_full_lin[0]._fieldnames)
#load

#res_sp=st.scan_load(35791,filename=filename)
def fit(f, C, model, p0, fit_method='lin'):

    def convert_from_complex_to_real(y):
        ''' Convert array of complex numbers to array of real numbers '''
        eps = np.finfo(float).eps  # For avoding zeros in logarithms
        if fit_method == 'lin':
            return np.abs(y)**2
        elif fit_method == 'log':
            return np.log10(np.real(y)**2+eps)+np.log10(np.imag(y)**2+eps)
        else:
            raise ValueError(f'Unknown switch = {fit_method}')

    def fit_func(f, *args):
        ''' The (real) fitting function '''
        C = model(f, *args)
        return convert_from_complex_to_real(C)

    from scipy.optimize import curve_fit
    popt, pcov = curve_fit(fit_func, f, convert_from_complex_to_real(C), p0=p0)

    return popt
m = 4.62e-3 #grams
A = 4*np.pi*(9.5e-3**2)
V = (4/3)*np.pi*(9.5e-3**3)
L = 3900#m/(A**2)

def emptymodel(f, c0, cinf, f0, q):
    x = f/f0
    return (cinf + ((c0 - cinf)/(1-(x*x)+1j*x*(1/q))))

def norm_cap(c, c0, cinf):
   return(c-cinf)/(c0-cinf)

def S_liq(f, C, popt):
    x = f/popt[2]
    F = norm_cap(C, popt[0], popt[1])
```

```
    return(L*((popt[2]*2*np.pi)**2)*((1/F)-1-1j*x*(1/popt[3])+(x**2)))

def manyfits(data):
    p0 = 2.9e-8, 1.9e-9, 9.4e4, 90
    popt = fit(data.fr_bridge, data.C_bridge, emptymodel, p0, fit_method='lin')
    C_fit = emptymodel(data.fr_bridge, *popt)
    return C_fit, popt

#print(C_fit)
#print('Fit parameters:', popt)
bulk_modulus = []
rhos = []
temperatures = []
fit_params = []
for empty, full in zip(res_sp_empty_lin, res_sp_full_lin):
    fig, (ax1, ax2) = plt.subplots(2, 1, sharex = True)
    fig2, bulk = plt.subplots()
    f = empty.fr_bridge
    c = empty.C_bridge
    f_full = full.fr_bridge
    C_full = full.C_bridge
    Frequency_Full_filtered = []
    Capacitance_Full_filtered = []

    for index,Cap_full in enumerate(C_full):
        if np.real(Cap_full) < 5e-7 and np.real(Cap_full)>-1e-7:
            if f[index]>1e4 and f[index]<2e5:
                Frequency_Full_filtered.append(f[index])
                Capacitance_Full_filtered.append(Cap_full)

    C_fit_empty, popt_empty = manyfits(empty)
    S_full = S_liq(f_full, C_full, popt_empty)
    c0 = popt_empty[0]
    cinf = popt_empty[1]
    f0 = popt_empty[2]
    q = popt_empty[3]
    fit_params.append(popt_empty)
    Temp_start_empty = empty.T_before
    Temp_end_empty= empty.T_after
    Temp_start_full = full.T_before
    Temp_end_full = full.T_after

    def cm(f, k, rho):
        x = (f)/(f0)
        return cinf + (c0-cinf)/((1+(1j*x*(1/q)-x**2)+(full_s_liq(f, k, rho)/(L*(f0*2*np.pi)**2))))

    def full_s_liq(f, k, rho):
        omega_null = np.sqrt((k/rho)) * (1/9.5e-3)
        x = ([freq*2*np.pi for freq in f])/(omega_null)
        return (-1*(k/(3*(9.5e-3**3)))*(((x**2)*np.sin(x))/((x*np.cos(x))-np.sin(x))))

    popt_full = fit(Frequency_Full_filtered, Capacitance_Full_filtered, cm, (2e9, 900), fit_method =
    C_fit_full = cm(Frequency_Full_filtered, *popt_full)

    ax1.plot(f, np.real(c), 'o')
    ax1.plot(f, np.real(C_fit_empty))
    ax1.set_ylabel(r'real $\hat{J}$', rotation = 0)
    ax2.plot(f, -1*np.imag(c), 'o')
    ax2.plot(f, -1*np.imag(C_fit_empty))
    bulk.plot(Frequency_Full_filtered, np.real(Capacitance_Full_filtered), '.', label = f'data')
    bulk.plot(Frequency_Full_filtered, np.real(C_fit_full), '--', label = f'fit')
```

```
    ax1.set_ylim([-1e-7, 2e-7])
    #ax1.set_xlim([1e4, 1.4e5])
    ax2.set_ylabel(r'$imaginary \ \hat{J}$')
    ax2.set_xlabel(r'$log(Frequency)$')

    ax1.set_xscale('log')
    bulk.set_xscale('log')
    ax1.set_title(f'Capacitance of empty Transducer at {round(Temp_start_empty)}K')
    bulk.set_title(f'Capacitance of full transducer at {round(Temp_start_full)}K')
    fig.savefig(f'EmptyCap_{round(Temp_start_empty)}K.png')
    fig2.savefig(f'FullCap_{round(Temp_start_full)}K.png')
    bulk_modulus.append(popt_full[0])
    rhos.append(popt_full[1])
    temperatures.append(Temp_start_full)
```

```
fig, (bulk, rho) = plt.subplots(2, 1 ,sharex= True)
bulk.plot(temperatures, bulk_modulus, 'o--')
rho.plot(temperatures, rhos, 'o--')
bulk.set_ylabel('Bulk modulus K')
rho.set_xlabel('Temperature Kelvin')
rho.set_ylabel(r'Density $\rho$')
bulk.set_title('Bulk modulus and Density')
fig.savefig('BulkvTemp')
fig, ((ax1, ax2), (ax3, ax4)) = plt.subplots(2, 2, sharex=True)
ax1.plot(temperatures, [i[0] for i in fit_params], 'o--', label = 'C0')
ax2.plot(temperatures, [i[1] for i in fit_params], 'o--', label = 'Cinfinty')
ax3.plot(temperatures, [i[2] for i in fit_params], 'o--', label = 'Resonance')
ax4.plot(temperatures, [i[3] for i in fit_params], 'o--', label = 'Quality Factor')
ax1.set_ylabel('C0')
ax2.set_ylabel('Cinf')
ax3.set_ylabel('Resonance')
ax4.set_ylabel('Quality Factor')
ax3.set_xlabel('Temperature Kelvin')
ax4.set_xlabel('Temperature Kelvin')
fig.savefig('FitsvTemp')
```