Skin Depth in Conductors

KDSMIL001 PHY2004W: PHYLAB 2

13 October 2020

Contents

1 Introduction

When an alternating current is travelling on a conductor a phenomenon known as Skin Effect is observed, where it appears that the only part of the conductor that is actually conducting the current is the outermost part, or the "skin". This extends to conductors present in time-varying magnetic fields, where we find this relation

$$B(z) = B(0)e^{-\frac{z}{\delta}} \tag{1.1}$$

for a uniform alternating magnetic field at the surface of an infinite sheet of conductor. In this case z is the distance from the conductor surface and δ is a length known as the skin depth, given by $\delta = \sqrt{\frac{2}{\mu\sigma\omega}}$ where μ is the permeability of the conductor, σ is the conductivity, ω is the angular frequency of the field B, and $B(0)$ is the amplitude of the field without any conductor in place. In this experiment we are using a copper tube so we have cylindrical symmetry, resulting in the relation

$$B(d) = B(0)\frac{1}{\sqrt{1 + (Rd/\delta^2)^2}} \tag{1.2}$$

where R is the inner radius of the tube and $R + d$ is the outer radius. This $B(d)$ represents the magnetic field present within a tube of wall thickness d. It's important to note here that the original field $B(0)$ is taken as being uniform, as well as the resultant field $B(d)$. This means that measuring the field at any point within the tube will return the same result.

2 Aim

The aim of this experiment is to investigate the shielding effect of a copper tube in a time-varying magnetic field and to find the conductivity of the copper tube using Equation 1.2 by varying the frequency of the magnetic field and fitting the equation to the experimental data.

3 Apparatus

All measurement devices used have an uncertainty rating of 2%.

- Signal generator producing a sinusoidal AC current
- Audio amplifier
- Ammeter
- Helmholtz coil with 80 winds on each coil

- Axial search coil with 200 winds
- Oscilloscope
- Copper tube with inner radius $R = (20.0 \pm 0.1) \times 10^{-3}\,\mathrm{m}$ and wall thickness $d = (1.0 \pm 0.1) \times 10^{-3}\,\mathrm{m}$.

In order to produce the time-varying magnetic field we used a signal generator feeding into an audio amplifier, to control the amplitude of the current, which in turn fed into a Helmholtz coil, which is two identical coils connected in series, separated by a gap. This is used in order to create a more stable magnetic field than if we were to use just one. On this circuit is an ammeter connected in series in order to monitor the amplitude of the current (in rms) to make sure it doesn't exceed 1 A or else the coils would heat up and distort the sinusoidal current.
Along with this we had an axial search coil, aligned in the same way as the two larger coils, in the middle of the two and connected to an oscilloscope to monitor the emf induced in the axial coil. Lastly we had a copper tube that would fit over the axial coil.

4 Method

With a sinusoidal AC current being supplied by the signal generator and the amplitude of this current being kept below 1 A by adjusting the audio amplifier, we varied the frequency of the supplied signal from 100 Hz to 5 kHz, measuring the induced voltage at each frequency with and without the copper tube around the axial coil. It is vital that the source is AC as a DC current in the coil will produce a magnetic field, but it won't be time-varying. The time-varying nature of the magnetic field is crucial as, by Faraday's Law, the emf induced in a conductor by a magnetic field is proportional to the time derivative of the magnetic field. With no time-variance, no emf is induced and so no field can be set up in the shielding material to oppose the original field. Looking at Equation 1.2, we can see that the ratio between the reduced magnetic field $B(d)$ and the magnetic field without any shielding $B(0)$ should give us

$$\frac{B(d)}{B(0)} = \frac{1}{\sqrt{1 + (Rd/\frac{2}{\mu\sigma\omega})^2}} \tag{4.1}$$

Taking our measurements of the induced voltage with and without shielding and finding their ratio should give us the same result. This is a consequence of Faraday's Law which says

$$\epsilon = -N_a \frac{d}{dt} \int \vec{B} \cdot d\vec{A} \approx -N_a A \frac{dB}{dt} \tag{4.2}$$

where N_a is the number of winds in a search coil and A is the cross-sectional area of that coil. Since we're interested in amplitudes and the amplitude of our supplied B

does not change with respect to time, we find the relation

$$|\epsilon| \approx N_a A |B| \tag{4.3}$$

for the amplitudes of induced emf and magnetic field. From this we find

$$\frac{B(d)}{B(0)} \approx \frac{N_a A \epsilon(d)}{N_a A \epsilon(0)} = \frac{\epsilon(d)}{\epsilon(0)}$$

so we did not need to do any conversions, we only needed to compare induced emfs. Plotting the ratio of emfs above against angular frequency ω, we then used the value for the permeability of copper, $\mu = 1.256629 \times 10^{-6}\,\mathrm{H/m}$, and used `scipy.optimize.curve_fit` to fit Equation 4.1 to the plotted data, thus finding the only unknown parameter σ, the conductivity. To determine uncertainties we used the Jackknife Monte Carlo method of determining uncertainties of fitted parameters. The uncertainty on our data points is found from the 2% uncertainty of the measurement device, propagated using the following formula.

$$u\left(\frac{x}{y}\right) = \left|\frac{x}{y}\right| \sqrt{\left(\frac{u(x)}{x}\right)^2 + \left(\frac{u(y)}{y}\right)^2}$$

5 Results

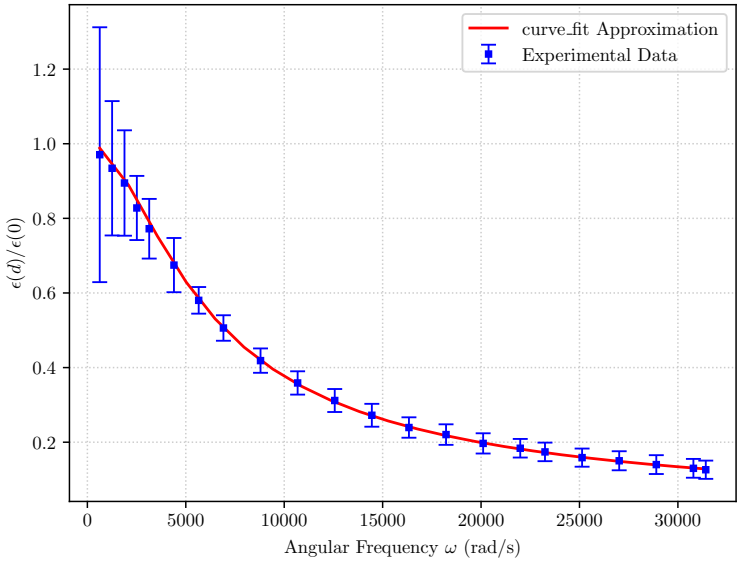

Figure 5.1: Plot of the ratio between shielded emf and non-shielded emf induced in an axial search coil placed within a Helmholtz coil, against the angular frequency of the magnetic field inducing the emf. The theoretical model is fitted to the data with `curve_fit` to find the conductivity of the copper tube shielding the axial coil.

The fitting to the data shown in Figure 5.1 gave us a value for the conductivity of copper:

$$\sigma = (1958.9 \pm 1.2) \times 10^4$$

6 Discussion and Recommendations

Thanks to the magic of Google, we are able to find the agreed upon value for the conductivity of copper:

$$\sigma = 5.96 \times 10^7$$

Comparing this value to ours, we see that they are within an order of magnitude of each other, but our value does not agree with the actual value within experimental uncertainty. There are a few possible reasons for this discrepancy. We only took 23 data points and this isn't really an appropriate amount of data to have when trying to accurately determine a property of a material such as conductivity. A possible source of uncertainty could be the measurement of the thickness of the copper tube. It could be that our measurement was off by a significant amount, which would likely affect the result. The uncertainty on the data points in Figure 5.1 is also quite large at some points. This could be reduced with more accurate equipment, although it would likely not solve the problem of our σ being roughly a third of the expected value.

7 Conclusion

To conclude, we found a value for σ, the conductivity of copper, by examining the shielding effect of a copper tube on a roughly uniform time-varying magnetic field and fitting an appropriate equation to data. The value we found does not agree with the expected value but we have outlined some possible reasons for this.

8 Appendix

Appendix 1: Code for determining σ and plotting the data

```
from matplotlib import pyplot as plt
import numpy as np
from scipy.optimize import curve_fit
from numpy import cos, pi, sin, sqrt, exp, random
import matplotlib
# matplotlib.use('pgf')
matplotlib.rcParams.update({
    'pgf.texsystem': 'pdflatex',
    'font.family': 'serif',
    'text.usetex': True,
    'pgf.rcfonts': False,
})
file = open(r'PHY2004W Practicals and Reports\Skin Effect Prac\Report
    \Data\Skin_Data_Text.txt', 'r')
header = file.readline()
lines = file.readlines()
N = len(lines)
i=0
data = np.zeros((3,N))
# Reading the file, getting the data into the data array
for line in lines:
    line = line.strip()
    columns = line.split()
    data[0][i] = float(columns[0])
    data[1][i] = float(columns[1])
    data[2][i] = float(columns[2])
    i += 1
file.close()
# Some constants
R=20e-3
RUn=0.1e-3
d=1e-3
dUn=0.1e-3
mu=1.256629e-6
# Converting to SI and amplitudes
data[0]*=2*pi
data[1]*=0.5*1e-3
data[2]*=0.5*1e-3
# Ratio of mag fields = ratio of emfs
yData=data[2]/data[1]
# Uncertainties on these measured values
omegaUn=0.02*2000*pi
emfUn=sqrt(0.02**2 + (0.0001/(2*sqrt(3)))**2)*0.5
yDataUn=np.zeros(N)
for m,ms in enumerate(yData):
    yDataUn[m]=ms*sqrt((emfUn/data[1][m])**2 + (emfUn/data[2][m])**2)
```

```
# Function for curve_fit
def func(omega,sigma):
    return 1/(sqrt(1+((R*d)/((2)/(mu*sigma*omega)))**2))
# Array used to plot the optimised function and initial guess
xModel=np.linspace(data[0][0],data[0][-1],N,endpoint=True)
p0=[1e7]
numParams=len(p0)
# Jackknife Curve Fitting
jackknifeData = np.zeros((4, N, N-1))
popts=np.zeros((numParams,N))
for c in range(N):
    # Removing random values from each array
    r = random.randint(0, N)
    jackknifeData[0][c] = np.delete(data[0], r)
    jackknifeData[1][c] = np.delete(yData, r)
    jackknifeData[2][c] = np.delete(yDataUn, r)
    jackknifeData[3][c] = np.delete(xModel, r)
    # Fitting N times
    popt, pcov = curve_fit(func, jackknifeData[0][c], jackknifeData
        [1][c], p0, sigma=jackknifeData[2][c], absolute_sigma=True)
    popts[0][c]=popt
# Means and standard uncertainties of the parameters
pOptimals=np.zeros((2,numParams))
for j, js in enumerate(popts):
    pMean=np.mean(js)
    pOptimals[0][j]=pMean
    sumI=0
    for i in js:
        sumI+=(i-pMean)**2
    # Jackknife method for determining uncertainties of fitted
        parameters
    pOptimals[1][j]=sqrt(float(((N-1)/N)*sumI))/sqrt(N-1)
# Printing optimal parameters
print('sigma =',pOptimals[0][0],'+/-',pOptimals[1][0])
# Plotting the function with the optimal parameters
yfit=func(xModel,*pOptimals[0])
plt.errorbar(data[0],yData,yerr=yDataUn,fmt='bs',ms=3,elinewidth=1,
    capsize=4,label='Experimental Data')
plt.plot(xModel,yfit,'r',label='curve\_fit Approximation')
plt.xlabel('Angular Frequency $\omega$ (rad/s)')
plt.ylabel('$\epsilon(d)/\epsilon(0)$')
plt.grid(color='#CCCCCC', linestyle=':')
plt.legend()
plt.show()
# plt.savefig(r'PHY2004W Practicals and Reports\Skin Effect Prac\
    Report\Data\SkinEffect.pgf')
```