IDENTIFYING A SUPERCONDUCTOR SAMPLE

PEDRO ROJAS FERNÁNDEZ

UNDERSTANDING THE PROBLEM

We must determine the identity of the superconducting sample.

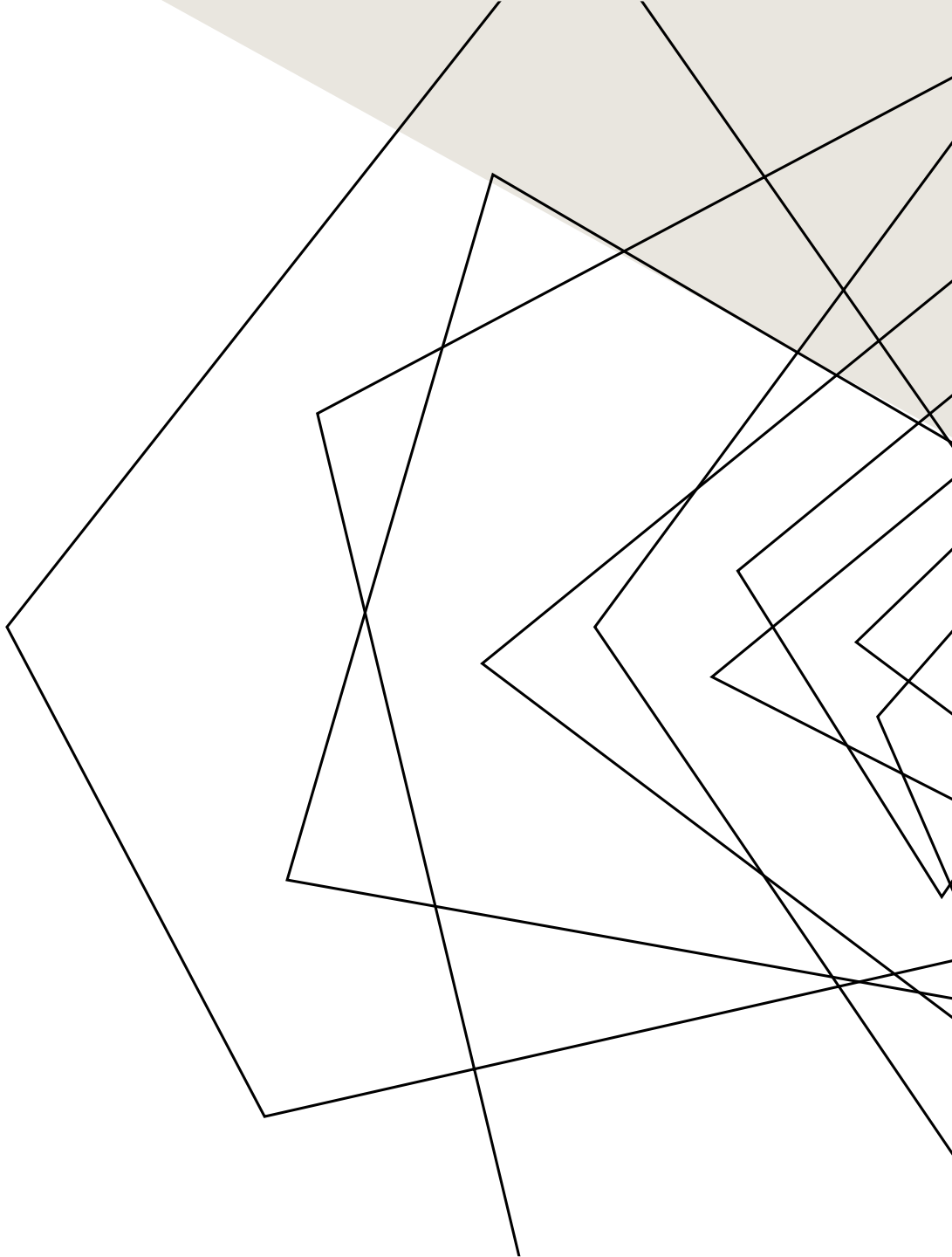
There are two main parameters to achieve our goal:

- Critic Temperature T_C
- Onset temperature T_{ON}

With these values, one can compare with external references to identify the sample.

A .txt file was provided with values for voltage, current and temperature taken from the experiment. However, the file contains noise.

To obtain our parameters, we must fit our data under a sigmoid function.



STRATEGY

- Clean the data by eliminating columns with less than 3 values.
- Compute the resistance by the Ohm's Law.
- Preview the data in a graph to see the behavior of resistance vs temperature.
- According to the graph, plot making cuts in the data if its necessary, and considering uncertainties. The cuts should improve the fitting and the uncertainty or error, should be calculated considering the last count of the Keithley 2000 by error propagation.
- The fit should made with a sigmoid function as suggested. We should identify the Critic Temperature as the center value of the fitted curve. As most of the references show a sharpened resistance vs temperature curve.
- The Onset Temperature should be identified as the temperature at which the resistance begins to increase. This value could be calculated from the fitted curve.
- Compute, trough the error, how close of far are our estimations from the literature.

LET'S CODE

```
#First import the necessary libraries
import ROOT
import pandas as pd
import numpy as np
```

```
Welcome to JupyROOT 6.30/04
```

Cleaning data

```
file = "warmingdata.txt"
vrows = [] #Valid rows (with 3 values)
with open(file,'r') as file:
    for lines in file:
        rows = lines.split() #Create a list where each row has a value as a string
        if len(rows) == 3:
            vrows.append(list(map(float, rows))) #Create a row which is a list with float elements
data = pd.DataFrame(vrows, columns=['Current','Voltage','Temperature'])
display(data)
```

	Current	Voltage	Temperature
0	1.000583	5.052302e-07	83.353296
1	1.000527	3.823206e-07	83.432066
2	1.000586	9.862932e-07	83.513228
3	1.000596	9.595965e-07	83.606945
4	1.000565	6.656858e-07	83.741711
...	...	...	...
320	1.000512	1.377246e-03	144.377174
321	1.000492	1.377535e-03	144.512303
322	1.000434	1.378797e-03	144.645051
323	1.000508	1.380333e-03	144.780610
324	1.000482	1.380783e-03	144.917115

325 rows × 3 columns

```
#Lets group the columns in different arrays
temperature = data["Temperature"].values #K
voltage = data["Voltage"].values #V
current = data["Current"].values*(1e-1) #A
type(temperature)
```

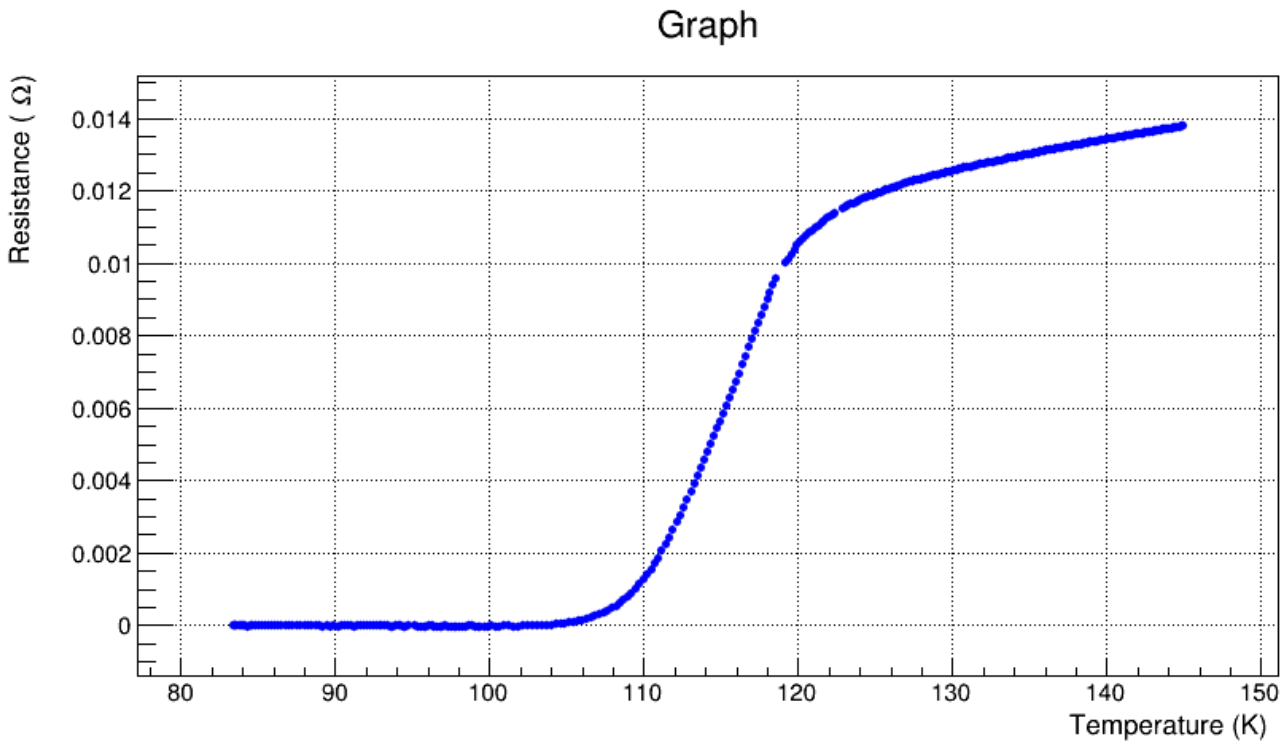
```
numpy.ndarray
```

```
resistance=voltage/current
type(resistance)
```

LET'S CODE

Preview of the data

```
#Making a preview of the data
graph = ROOT.TGraph(len(data), temperature, resistance)

# Create a canvas for the graph
canvas = ROOT.TCanvas("canvas", "Gráfico sin errores", 900, 500)

# Graph style
graph.SetMarkerStyle(20)
graph.SetMarkerSize(0.5)
graph.SetMarkerColor(ROOT.kBlue)

# Configure axis
graph.GetXaxis().SetTitle("Temperature (K)")
graph.GetYaxis().SetTitle("Resistance (#Omega)")
graph.GetXaxis().SetTitleSize(0.04)
graph.GetYaxis().SetTitleSize(0.04)
canvas.SetGrid()

# Draw the graph in the canva
graph.Draw("AP")  # "AP" points and lines

# Show the canvas
canvas.Draw()
```



LET'S CODE

Ploting and fitting

```
#As we can see there is a temperature window where the temperature remains constant:
#Lets define that window in the interval from 100 to 125 K
temperature_filtered = []
voltage_filtered = []
current_filtered = []
resistance_filtered = []
for temp, volt, curr, res in zip(temperature,voltage,current,resistance):
    if 100 <= temp <= 125:
        temperature_filtered.append(temp)
        voltage_filtered.append(volt)
        current_filtered.append(curr)
        resistance_filtered.append(res)
```

```
#Lets consider the last count of the Keithley 2000
#For voltage 100.0000mV range the resolution is 0.1uV
    #Computing the uncertainty
voltage_error = (0.1e-6)*np.ones_like(voltage_filtered)

#For current 100.0000mA range the resolution is 100nA
     #Computing the uncertainty
current_error = (1e-6)*np.ones_like(current_filtered)

#For temperature -200 to +760 the resolution is 0.5
temperature_error= (0.5)*np.ones_like(temperature_filtered)

#Computing the error propagation for the resistance
resistance_error = resistance_filtered*np.sqrt(((voltage_error/voltage_filtered)**2) + ((current_error/current_filtered)**2))
print("voltage_error:",voltage_error,"\n current_error:",current_error,"\n resistance_error:",resistance_error)
```

```
temperature_filtered = np.array(temperature_filtered)
voltage_filtered = np.array(voltage_filtered)
current_filtered = np.array(current_filtered)
resistance_filtered = np.array(resistance_filtered)
type(resistance_filtered)
```

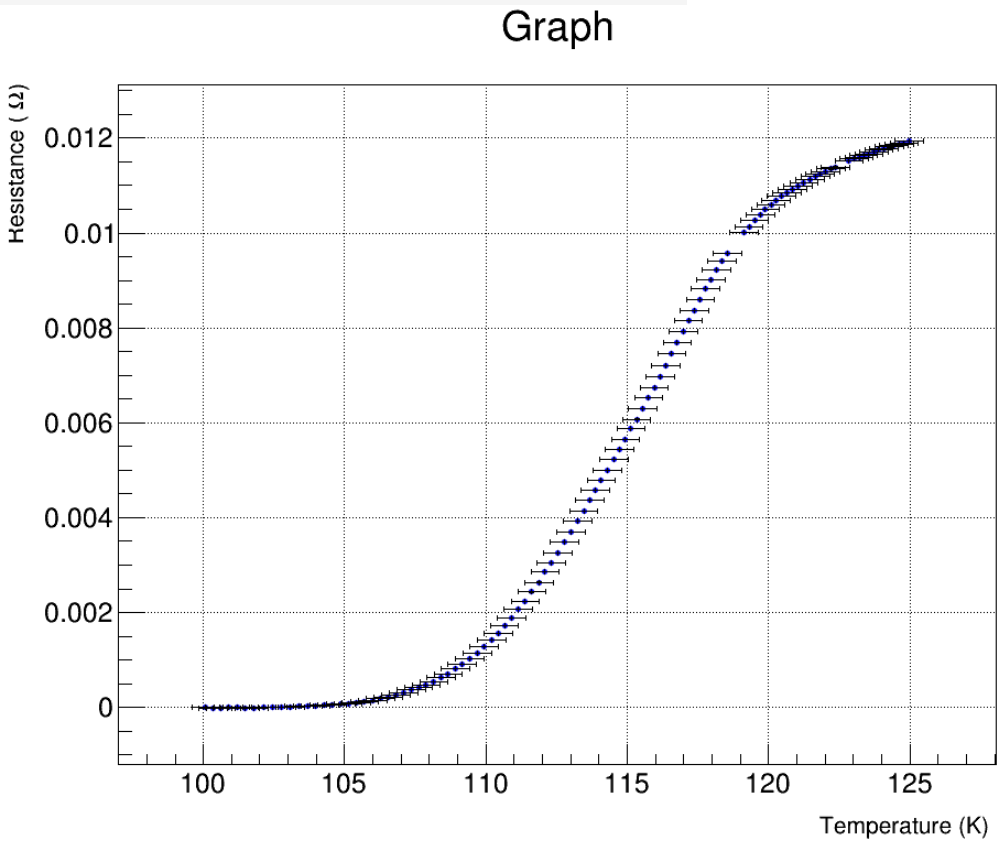
```
numpy.ndarray
```

```
#Finding the minimum and maximun values
min_temp = np.min(temperature_filtered)
max_temp = np.max(temperature_filtered)
min_voltage = np.min(voltage_filtered)
max_voltage = np.max(voltage_filtered)
min_current = np.min(current_filtered)
max_current = np.max(current_filtered)
min_resistance = np.min(resistance_filtered)
max_resistance = np.max(resistance_filtered)
print("T",min_temp,max_temp,"V",min_voltage,max_voltage,"A",min_current,max_current,"R",min_resistance,max_resistance)
```

```
T 100.083911 124.962156 V -1.51517576e-06 0.00119410162 A 0.10004401000000002 0.100064365 R -1.5144252865492506e-05 0.011934903261898288
```

LET'S CODE

```
#Using ROOT.TGraphErrors(num_points, x_values, y_values, x_error, y_error)

graph = ROOT.TGraphErrors(len(temperature_filtered), temperature_filtered, resistance_filtered, temperature_error, resistance_error)

canvas = ROOT.TCanvas('canvas_name','Resistance vs Temperature', 1000, 800)

# Draw the graph with error bars
graph.Draw('AP')

graph.SetMarkerStyle(20)
graph.SetMarkerSize(0.5)
graph.SetMarkerColor(ROOT.kBlue)

# Set axis labels
graph.GetXaxis().SetTitle('Temperature (K)')
graph.GetYaxis().SetTitle('Resistance (#Omega)')
graph.GetXaxis().SetTitleSize(0.03)
graph.GetYaxis().SetTitleSize(0.03)
graph.GetXaxis().SetTitleOffset(1.5)

canvas.SetGrid()

# Show the canvas
canvas.Draw()
```

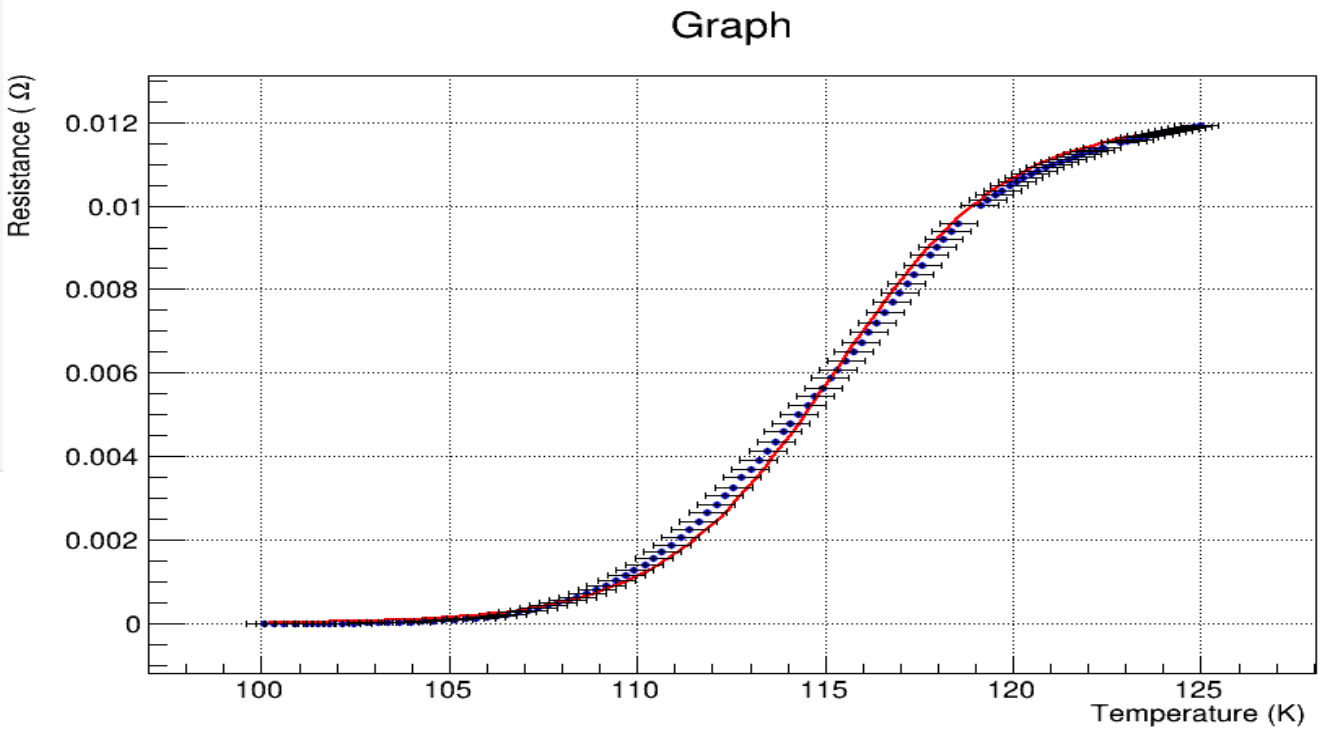


LET'S CODE

Fitting the curve

```
#sigmoid L / (1+exp(-k(x-x_0)))
#x = temperature
#p = [L,x_0,k]

#graph = ROOT.TGraph(len(data), temperature, resistance)
graph = ROOT.TGraphErrors(len(temperature_filtered), temperature_filtered, resistance_filtered, temperature_error, resistance_error)

def sigmoid(x,p):
    return p[0] / (1 + np.exp(-p[2]*(x[0] - p[1])))

sigmoid_func = ROOT.TF1("Sigmoid_fit", sigmoid, min_temp, max_temp, 3)
sigmoid_func.SetParameters(0.0115537, 114.46, 0.430845)

graph.Fit(sigmoid_func,"R")

canvas = ROOT.TCanvas('fitting','Resistance vs Temperature', 900, 600)

graph.SetMarkerStyle(20)
graph.SetMarkerSize(0.5)
graph.SetMarkerColor(ROOT.kBlue)

# Set axis labels
graph.GetXaxis().SetTitle('Temperature (K)')
graph.GetYaxis().SetTitle('Resistance (#Omega)')
canvas.SetGrid()

# Show the canvas
graph.Draw("AP")
canvas.Draw()
```

```
****************************************
Minimizer is Minuit2 / Migrad
Chi2                      =       422.626
NDf                       =           104
Edm                       =   1.12346e-07
NCalls                    =            84
p0                        =    0.0120371   +/-   2.13791e-05
p1                        =      115.265   +/-   0.0586025
p2                        =     0.433127   +/-   0.00465656
```



LET’S CODE

Critic Temperature

```
#Here the Tc corresponds to P[1]
tc = sigmoid_func.GetParameter(1)
tc_error = sigmoid_func.GetParError(1)
print("Tc =",tc,"K" "\nTc_error =",tc_error, "K")
```

```
Tc = 115.2650697379087 K
Tc_error = 0.05860249658556038 K
```

Onset Temperature

```
onset_threshold = 0.01 * (max_resistance - min_resistance)  #1% threshold
onset_temperature = sigmoid_func.GetX(onset_threshold)
onset_temperature_error = sigmoid_func.GetParError(1)

print("TON = ", onset_temperature, "K" "\nTON_error = ", onset_temperature_error, "K")
```

```
TON =  104.63886861739324 K
TON_error =  0.05860053284021219 K
```

COMPARING WITH THE REFERENCES

These superconductors have the basic composition of (Bi–Pb)–Sr–Ca–Cu–O that is commonly abbreviated as BSCCO. If the cations have the ratio of 2212, with (Bi, Pb) taken as one cation type, the transition temperature is between 80 and 90 K.[5] If the cation ratio is 2223, the transition temperature is about 110 K.[5,7] Therefore, the 2223 phase is the more desirable. [1]

In summary, we have identified $Bi_{2.1}CaSrCuO_x$ (2111), $Bi_{2.1}CaSr_2CaCu_2O_x$ (2122) and $Bi_{2.1}Ca_2Sr_2Cu_3O_x$ (2223) as superconducting phases in a homologous series with optimized zero resistance at 80 K, 91 K and 105 K (extrapolated) with [2]

strongly limit the high-power applications of these materials. The non-toxic Bi-based [$Bi_2Sr_2Ca_2Cu_3O_{10+\delta}$ (Bi-2223), $T_c \approx 110$ K] system also has a transition temperature exceeding 100 K, but the very layered [3]

COMPARING WITH THE REFERENCES

[4]

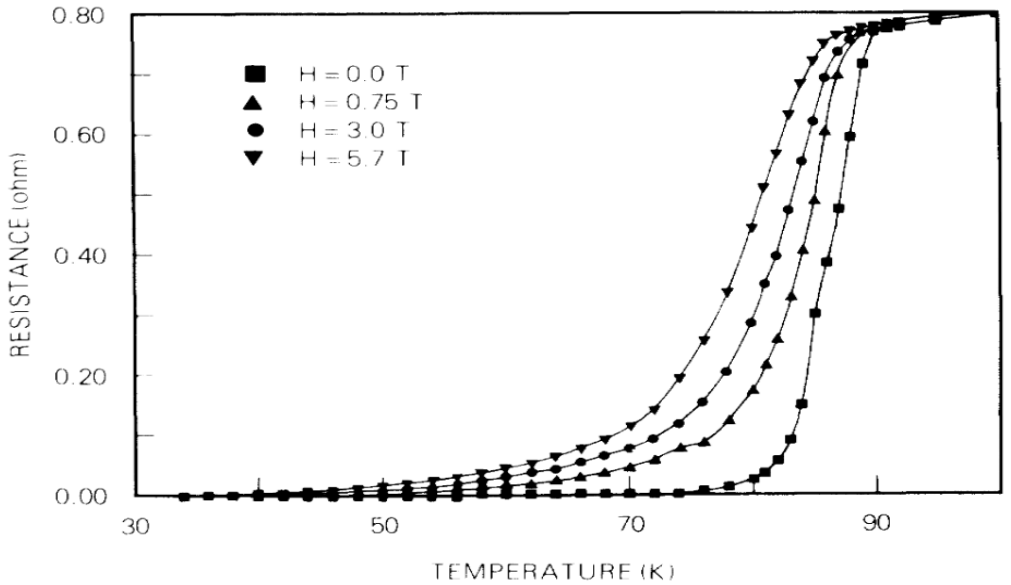


FIG. 1. Temperature dependence of resistance determined in a simple liquid-nitrogen Dewar.

The above results demonstrate unambiguously that superconductivity occurs in the Y-Ba-Cu-O system with a transition between 80 and 93 K. We have determined

COMPARING WITH THE REFERENCES

[5]

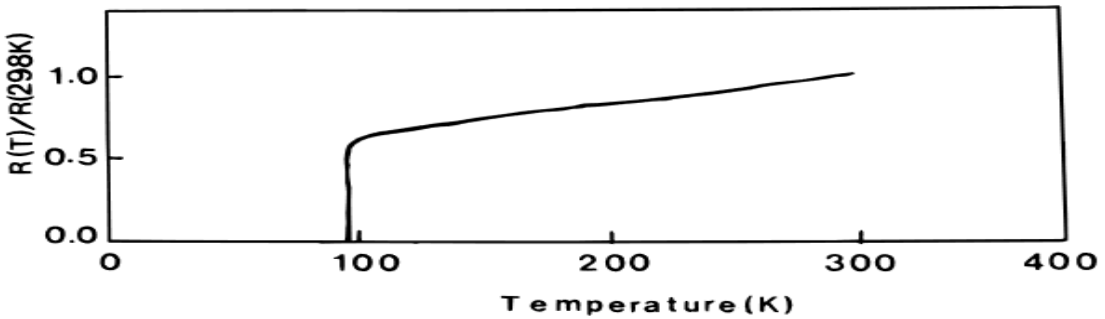


Figure 8. Temperature dependence of resistivity for the material calcined at 920 °C for 2 h.

from the powders calcined at 920 °C for 2 h. The resistivity of the sample decreases gradually with decreasing temperature. The onset and zero temperature were 97.3 and 94 K, respectively. As can be seen in Figure 8, the transition range is narrow and equals $\Delta T_c = 3.3$ K, indicating that the material shows a very

[6]

a monodomain state of the ferromagnetic interlayer. Since YBCO is a high-temperature superconductor with $T_c \simeq 90$ K, our devices can operate comfortably at liquid-nitrogen temperatures. Furthermore, since we are not using the

CONCLUSION

Identifying the superconductor sample

References are shown in the presentation.

The most closest value of critic temperature correspond to the Bisco-2223, with a critic temperature of aproximately 110K. The onset temperature is close too.

```
#Error of the calculated critical temperature and the one found in the literature:
tc_lit = 110.0
Error = (np.abs(tc - tc_lit)/tc_lit) * 100
print("Error in Critical Temperature", Error, "%")
```

```
Error in Critical Temperature 4.7863010942674835 %
```

```
#Error of the calculated onset temperature and the one found in the literature:
Error2 = (np.abs(onset_temperature - tc_lit)/tc_lit) * 100
print("Error in Onset Temperature", Error2, "%")
```

```
Error in Onset Temperature 4.87375580236978 %
```

Both parameters are useful to identify the sample as Bisco-2223. Since YBCO superconductor has a critical temperature below 90K. The error is higher, as we can see:

```
tc_lit2 = 90.0
Error3 = (np.abs(tc - tc_lit2)/tc_lit2) * 100
print("Error in Critical Temperature", Error3, "%")
```

```
Error in Critical Temperature 28.07214578188248 %
```

```
#Error of the calculated onset temperature and the one found in the literature:
Error2 = (np.abs(onset_temperature - tc_lit2)/tc_lit2) * 100
print("Error in Onset Temperature", Error2, "%")
```

```
Error in Onset Temperature 16.26540957488138 %
```

THE SAMPLE MOSTLY CORRESPOND TO A BISCO-2223

REFERENCES

[1]

C. L. Briant, E. L. Hall, K. W. Lay, y I. E. Tkaczyk, “Microstructural evolution of the BSCCO-2223 during powder-in-tube processing”, *J. Mater. Res.*, vol. 9, núm. 11, pp. 2789–2808, nov. 1994, doi: 10.1557/JMR.1994.2789.

[2]

J. L. Tallon *et al.*, “High-Tc superconducting phases in the series Bi2.1(Ca, Sr)n+lCunO2n+4+δ”, *Nature*, vol. 333, núm. 6169, pp. 153–156, may 1988, doi: 10.1038/333153a0.

[3]

Y. Zhang *et al.*, “Unprecedented High Irreversibility Line in Nontoxic Cuprate Superconductor (Cu,C)Ba2Ca3Cu4O11+delta”, *Sci. Adv.*, vol. 4, núm. 9, p. eaau0192, sep. 2018, doi: 10.1126/sciadv.aau0192.

[4]

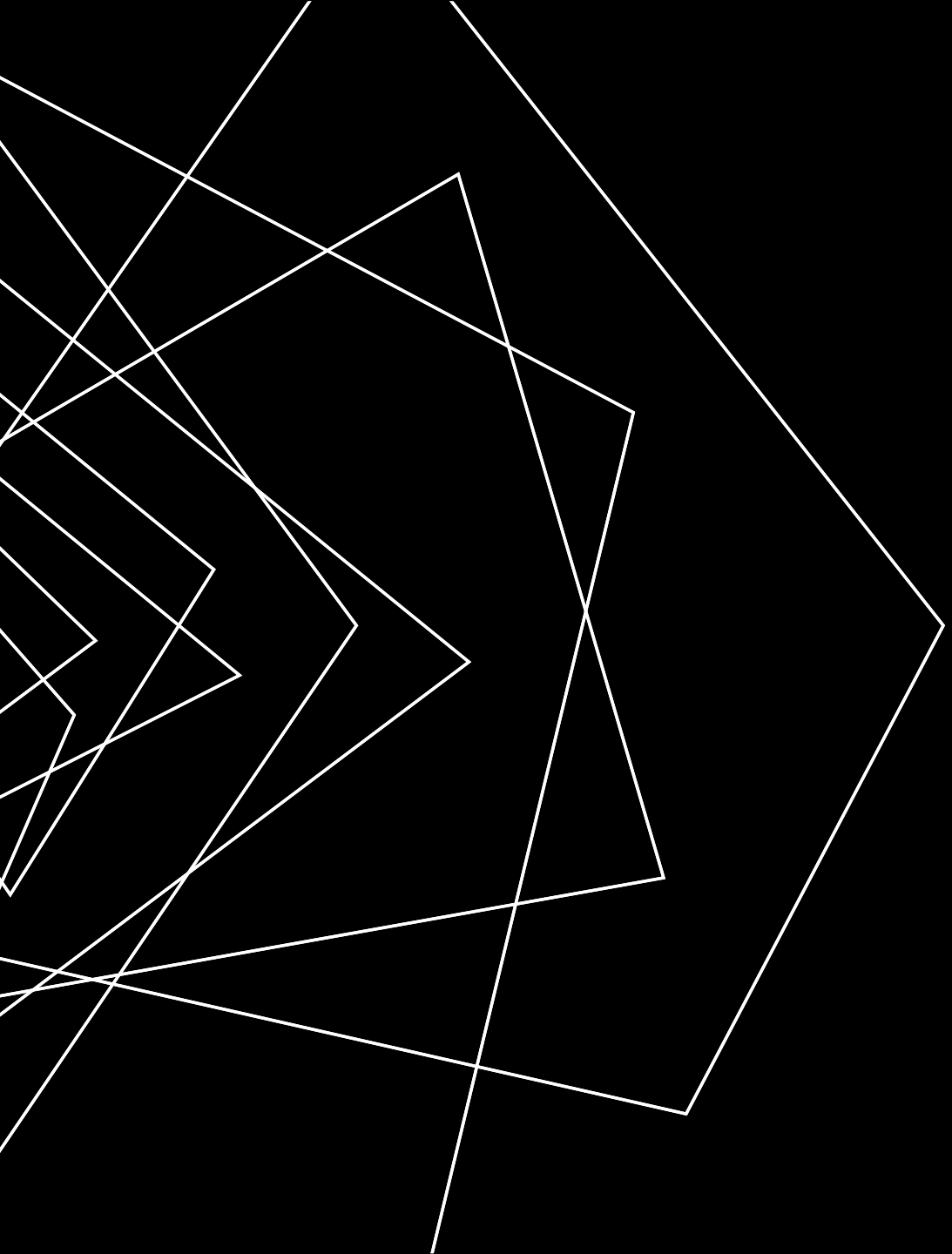
M. K. Wu *et al.*, “Superconductivity at 93 K in a new mixed-phase Y-Ba-Cu-O compound system at ambient pressure”, *Phys. Rev. Lett.*, vol. 58, núm. 9, pp. 908–910, mar. 1987, doi: 10.1103/PhysRevLett.58.908.

[5]

Y.-K. Sun y I.-H. Oh, “Preparation of Ultrafine $YBa_2Cu_3O_{7-x}$ Superconductor Powders by the Poly(vinyl alcohol)-Assisted Sol–Gel Method”, *Ind. Eng. Chem. Res.*, vol. 35, núm. 11, pp. 4296–4300, ene. 1996, doi: 10.1021/ie950527y.

[6]

R. de A. Prada, T. Golod, O. M. Kapran, E. A. Borodianskyi, C. Bernhard, y V. M. Krasnov, “YBa2Cu3O7/LaXMnO3 (X: Ca, Sr) based Superconductor/Ferromagnet/Superconductor junctions with memory functionality”, *Phys. Rev. B*, vol. 99, núm. 21, p. 214510, jun. 2019, doi: 10.1103/PhysRevB.99.214510.



THANK YOU