06/16/25
[illegible]

Fitting Data GNUPlot and Matplotlib

#t (s)	Angle (mra	Error (mrad)
0	-14.7	3.6
1	8.6	3.6
2.1	28.8	3
3.1	46.7	3.4
4.2	47.4	3.5
5.2	36.5	3.4
6.2	37	10.3
7.2	5.1	3.4
8.2	-11.2	3.4
9.1	-22.4	3.5
10	-35.5	3.6
11	-33.6	3.9
12	-21.1	3.9
12.9	-15	4.2
13.8	-1.6	2.7
14.9	19.5	3.2
15.9	27.5	2.8
17	32.6	3.5
17.9	27.5	2.7
18.9	20.2	3.3
20	13.8	3.4
21	-1.3	4.2
22	-24.5	6.7
23	-25	3.3
24	-25	3.1
25	-20.2	3.6
26	-9.9	3.2
27	5.8	3.2
28	14.7	3
29	21.8	3.5
30	29.8	2.7
31	21.4	4.1
32	24.6	2.7
32.9	25.8	12
33.8	0.6	2.9
34.7	-16.6	3.2
35.7	-24	3.7
36.6	-24.6	3.8
37.7	-19.8	3.5

Data that we want to fit.

I Stole this from Dr. Hauger.

It Should look like a damed sinusoid.

Fitting Data GNUplot and Matplotlib

06/16/25
Wesley Cooke

GNUplot:

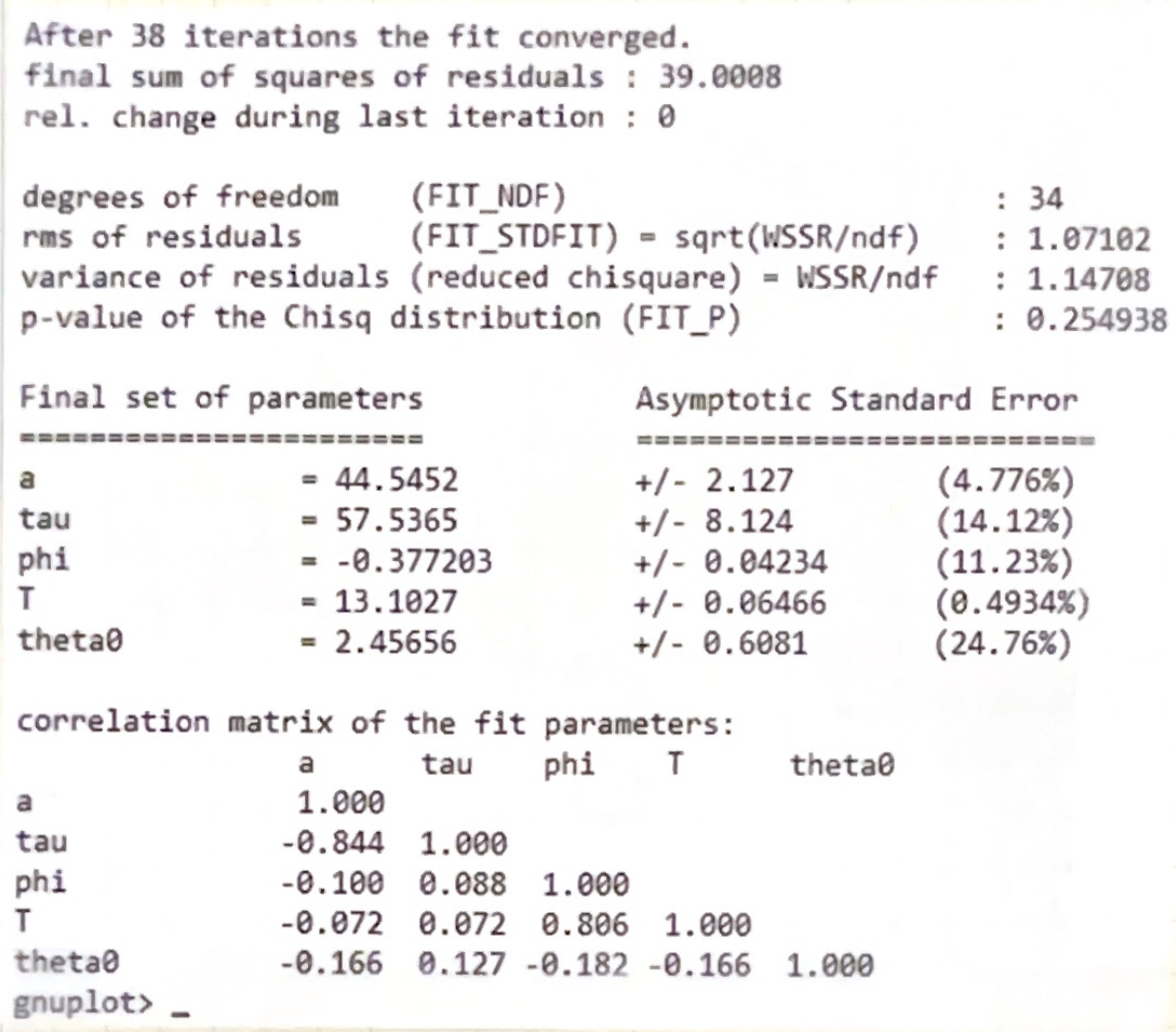
```
set title "HMC Data" font "Consolas, 20"          #Title with Consolas 20 pt font
set xlabel "Time (s)" font "Consolas, 12"         #Use Consolas 12 pt font for axis
labels
set ylabel "Angle (mrad)" font "Consolas, 12"
set y2label "Residuals" font "Consolas, 12"       #This is for the right hand axis, y2
#set xrange [0:40]                                #Set range, x
set yrange [-80:60]                               #Set range for left hand axis, y
set y2range [-20:120]                             #Sets range for right hand axis, y2
set y2tics border
set x2zeroaxis lt -1                              #Uses -1 for zero axis for the y2
plot. -1 is the usual origin axis width
set xtics 5 font "Consolas, 12"                   #Sets xtics at space of 5 and
numeric labels using Consolas, 12 pt font
set ytics 20 font "Consolas, 12"
set y2tics 20 font "Consolas, 12"
set tics out                                      #Sets tic marks pointing out from
axes
set grid xtics ytics                              #Grid spacing is same as tic
spacings

#Fit the data using an exponentially decaying sinusoid with phase shift
theta(x) = theta0 + a*exp(-x/tau)*sin(2.0*pi*x/T + phi)
a = 40
tau = 5
phi = -0.5
T = 15
theta0 = 10
fit theta(x) "TestData.txt" using 1:2:3 via a, tau, phi, T, theta0

#Plot data and error bars, equation of best fit theta(x) and residuals.
#Residuals are computed as the difference between theta(x) where x is from column $1
and the measured data point from column $2.
#These are plotted on axis x1 and the mirror axis y2, thus x1y2
plot "TestData.txt" with yerrorbars, theta(x), "TestData.txt" using 1:(theta($1) -
$2):3 axes x1y2 with yerrorbars
```

Save this to a ".p" file and to run it, use the command "load "filename.p""

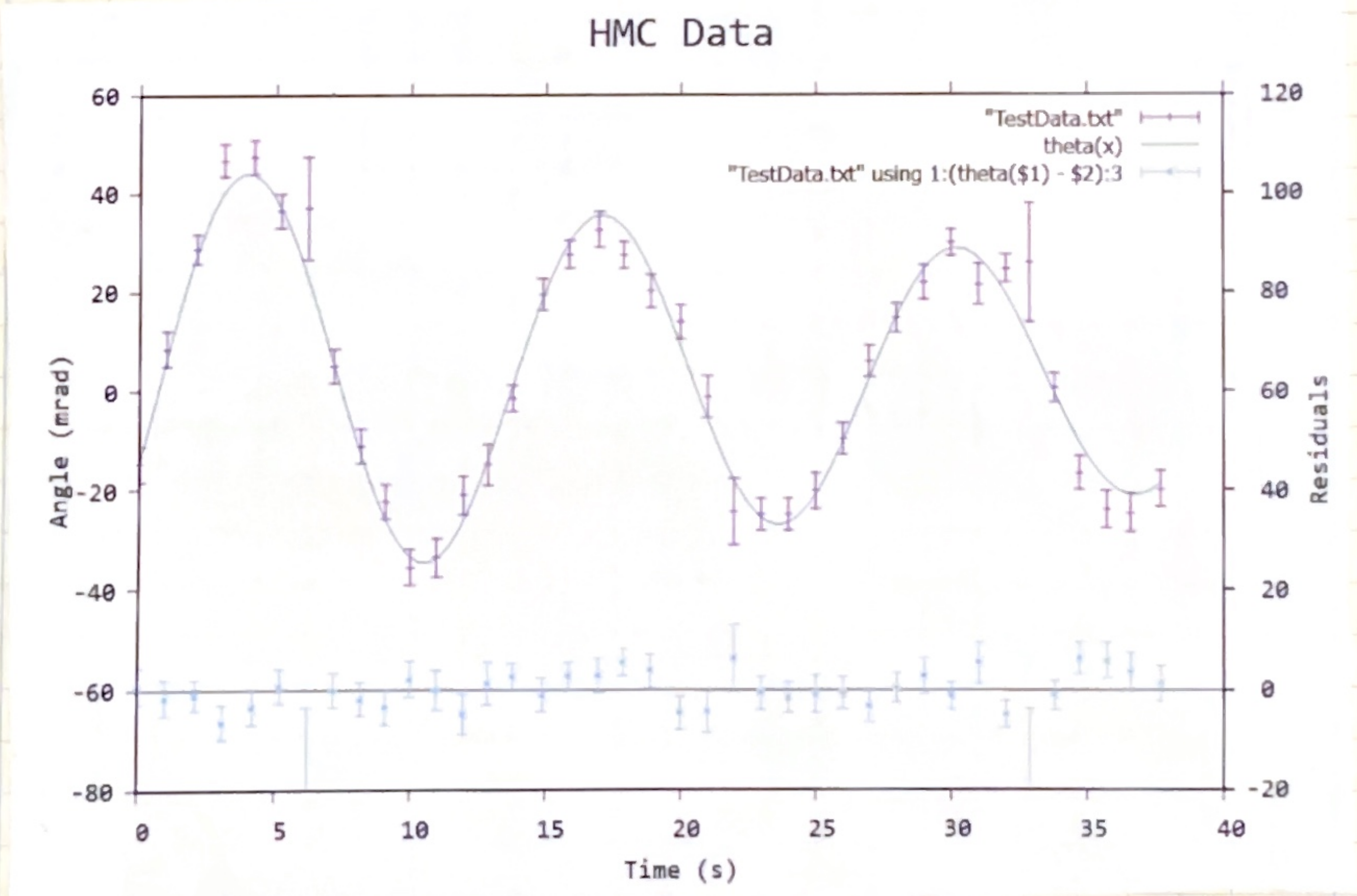
06/16/25
Wesley
code

Fitting Data GNUPlot and Matplotlib

The output of GNU Plot will be the following.



```
After 38 iterations the fit converged.
final sum of squares of residuals : 39.0008
rel. change during last iteration : 0

degrees of freedom    (FIT_NDF)                        : 34
rms of residuals      (FIT_STDFIT) = sqrt(WSSR/ndf)    : 1.07102
variance of residuals (reduced chisquare) = WSSR/ndf   : 1.14708
p-value of the Chisq distribution (FIT_P)              : 0.254938

Final set of parameters            Asymptotic Standard Error
=======================            ==========================
a               = 44.5452          +/- 2.127        (4.776%)
tau             = 57.5365          +/- 8.124        (14.12%)
phi             = -0.377203        +/- 0.04234      (11.23%)
T               = 13.1027          +/- 0.06466      (0.4934%)
theta0          = 2.45656          +/- 0.6081       (24.76%)

correlation matrix of the fit parameters:
                a      tau    phi    T      theta0
a               1.000
tau            -0.844  1.000
phi            -0.100  0.088  1.000
T              -0.072  0.072  0.806  1.000
theta0         -0.166  0.127 -0.182 -0.166  1.000
gnuplot> _
```



Fitting Data GNUPlot and Matplotlib

06/16/25
Wesley Cooke

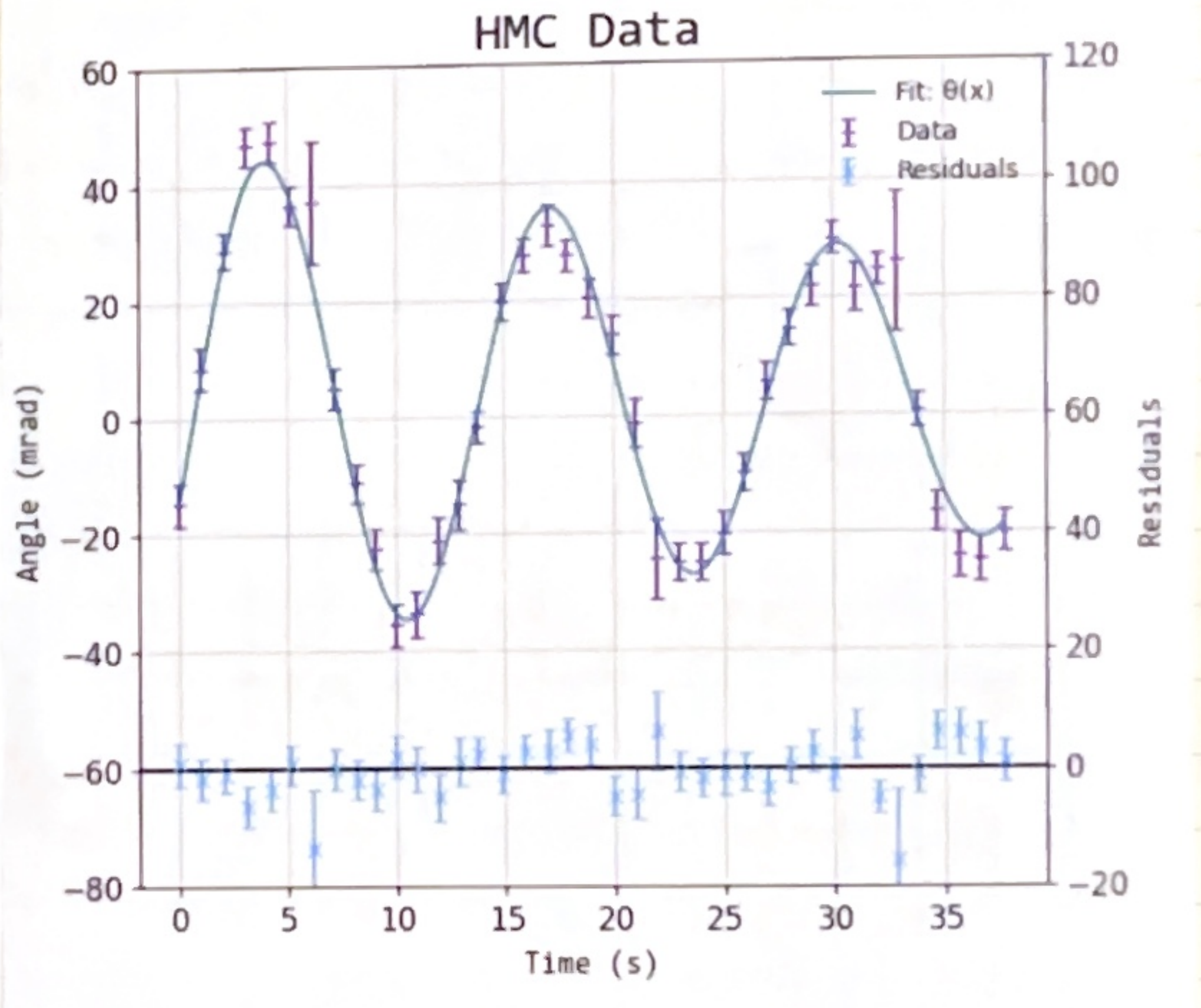
Mat Plot lib:

The goal here is to recreate the GNU Plot ~~progra~~ graph. So there are extra lines of code here to match GNU Plot.

In the end, use the software you are most familiar with and the one you can figure out how to use more quickly.

```python
import pandas as pd                         # pip install pandas
import numpy as np                          # pip install numpy
import matplotlib.pyplot as plt             # pip install matplotlib
from scipy.optimize import curve_fit # pip install scipy

data_path = "C:\\Users\\Wesley\\Downloads\\TestData.csv"
# Load CSV data (assumes headers: time, angle, error)
df = pd.read_csv(data_path)

# Extract columns
x    = df["#t (s)"].values
y    = df[" Angle (mrad)"].values
yerr = df["Error (mrad)"].values

# Define model function: exponentially decaying sinusoid
def theta(x, a, tau, phi, T, theta0):
    return theta0 + a * np.exp(-x / tau) * np.sin(2 * np.pi * x / T + phi)

# Initial guess for parameters
p0 = [40, 5, -0.5, 15, 10]

# Fit the model
params, _ = curve_fit(theta, x, y, sigma=yerr, p0=p0)
a, tau, phi, T, theta0 = params

# Generate fitted curve
x_fit = np.linspace(np.min(x), np.max(x), 500)
y_fit = theta(x_fit, a, tau, phi, T, theta0)

# Residuals: difference between model and data
residuals = theta(x, a, tau, phi, T, theta0) - y

# Create plot with dual y-axes
fig, ax1 = plt.subplots(figsize=(6.5, 5.5))

# FMT stands for format string.
# From the documentation: fmt = '[marker][line][color]'
# Primary axis: data and fit

ax1.errorbar(x, y, yerr=yerr, fmt='_', label='Data', capsize=3, color='#9400d3')
ax1.plot(x_fit, y_fit, '-', label='Fit: θ(x)', color='#06a076')
ax1.set_title("HMC Data", fontname="Consolas", fontsize=20)
ax1.set_xlabel("Time (s)", fontname="Consolas", fontsize=12)
ax1.set_ylabel("Angle (mrad)", fontname="Consolas", fontsize=12)
ax1.set_ylim([-80, 60])
ax1.set_xticks(np.arange(0, np.max(x)+1, 5))
ax1.set_yticks(np.arange(-80, 61, 20))
ax1.tick_params(axis='x', direction='out', labelsize=12)
ax1.tick_params(axis='y', direction='out', labelsize=12)
ax1.grid(True)

# Secondary axis: residuals
ax2 = ax1.twinx() # twin x-axis
ax2.errorbar(x, residuals, yerr=yerr, fmt='x', color='#56b4e9', label='Residuals',
```

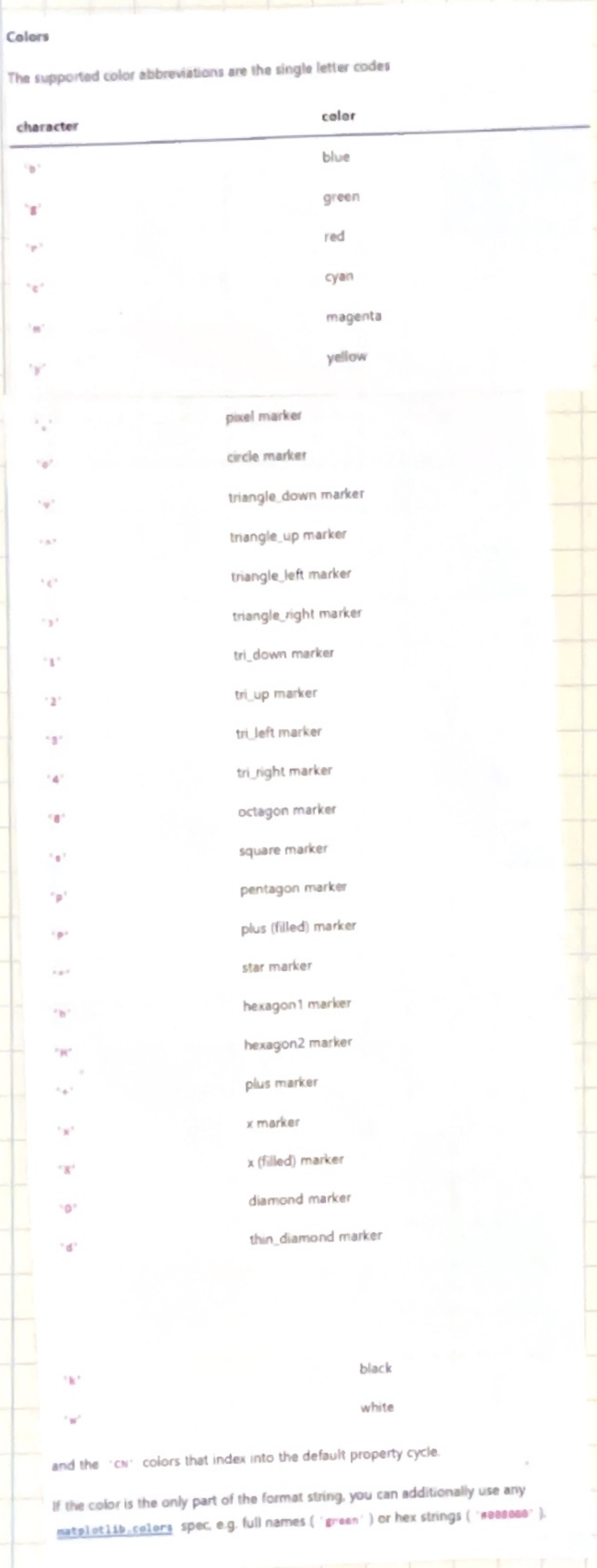
```python
ax2.set_ylabel("Residuals", fontname="Consolas", fontsize=12)
ax2.set_ylim([-20, 120])
ax2.set_yticks(np.arange(-20, 121, 20))
ax2.tick_params(axis='y', direction='out', labelsize=12)
ax2.axhline(y=0, color='k', linestyle='-')

# Combined legend
# bbox_to_anchor=(1, 1)
# This sets the anchor point of the legend box to the coordinate (1, 1).

# bbox_transform=ax1.transAxes
# This tells Matplotlib to interpret the (1, 1) anchor point in the coordinate syst

fig.legend(loc='upper right', bbox_to_anchor=(1, 1), bbox_transform=ax1.transAxes,

plt.tight_layout()
plt.savefig("Test.png")
plt.show()
```



06/16/25
Wesley [illegible]

Fitting Data GNUPlot and Matplotlib

More info on the "fmt" arguement:



Colors

The supported color abbreviations are the single letter codes

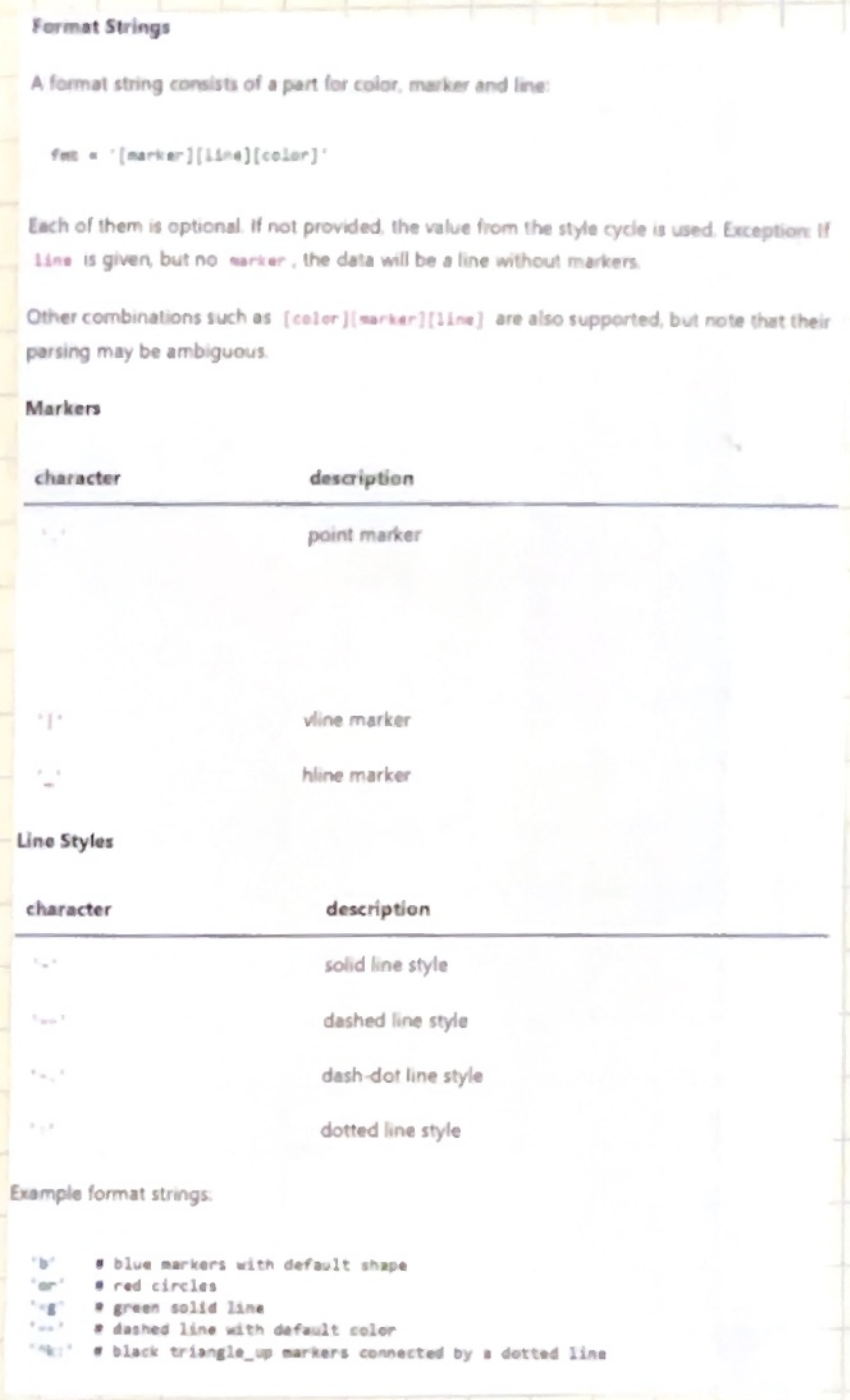
character	color
`'b'`	blue
`'g'`	green
`'r'`	red
`'c'`	cyan
`'m'`	magenta
`'y'`	yellow

`','`	pixel marker
`'o'`	circle marker
`'v'`	triangle_down marker
`'^'`	triangle_up marker
`'<'`	triangle_left marker
`'>'`	triangle_right marker
`'1'`	tri_down marker
`'2'`	tri_up marker
`'3'`	tri_left marker
`'4'`	tri_right marker
`'8'`	octagon marker
`'s'`	square marker
`'p'`	pentagon marker
`'P'`	plus (filled) marker
`'*'`	star marker
`'h'`	hexagon1 marker
`'H'`	hexagon2 marker
`'+'`	plus marker
`'x'`	x marker
`'X'`	x (filled) marker
`'D'`	diamond marker
`'d'`	thin_diamond marker

`'k'`	black
`'w'`	white

and the `'CN'` colors that index into the default property cycle.

If the color is the only part of the format string, you can additionally use any `matplotlib.colors` spec, e.g. full names (`'green'`) or hex strings (`'#008000'`).



Format Strings

A format string consists of a part for color, marker and line:

```
fmt = '[marker][line][color]'
```

Each of them is optional. If not provided, the value from the style cycle is used. Exception: If `line` is given, but no `marker`, the data will be a line without markers.

Other combinations such as `[color][marker][line]` are also supported, but note that their parsing may be ambiguous.

Markers

character	description
`'.'`	point marker
`'|'`	vline marker
`'_'`	hline marker

Line Styles

character	description
`'-'`	solid line style
`'--'`	dashed line style
`'-.'`	dash-dot line style
`':'`	dotted line style

Example format strings:

```
'b'    # blue markers with default shape
'or'   # red circles
'-g'   # green solid line
'--'   # dashed line with default color
'^k:'  # black triangle_up markers connected by a dotted line
```