02-Matplotlib Exercises

July 2, 2025

1 Assignment - 4

AICTE Faculty ID: 1-3241967546
Faculty Name: Milav Jayeshkuamar Dabgar
Date: July 2, 2025

---

2 Matplotlib Exercises

Welcome to the exercises for reviewing matplotlib! Take your time with these, Matplotlib can be tricky to understand at first. These are relatively simple plots, but they can be hard if this is your first time with matplotlib, feel free to reference the solutions as you go along.

Also don't worry if you find the matplotlib syntax frustrating, we actually won't be using it that often throughout the course, we will switch to using seaborn and pandas built-in visualization capabilities. But, those are built-off of matplotlib, which is why it is still important to get exposure to it!

** * NOTE: ALL THE COMMANDS FOR PLOTTING A FIGURE SHOULD ALL GO IN THE SAME CELL. SEPARATING THEM OUT INTO MULTIPLE CELLS MAY CAUSE NOTHING TO SHOW UP. * **

3 Exercises

Follow the instructions to recreate the plots using this data:

3.1 Data

[1]:
```
import numpy as np
x = np.arange(0,100)
y = x*2
z = x**2
```

** Import matplotlib.pyplot as plt and set %matplotlib inline if you are using the jupyter notebook. What command do you use if you aren't using the jupyter notebook?**

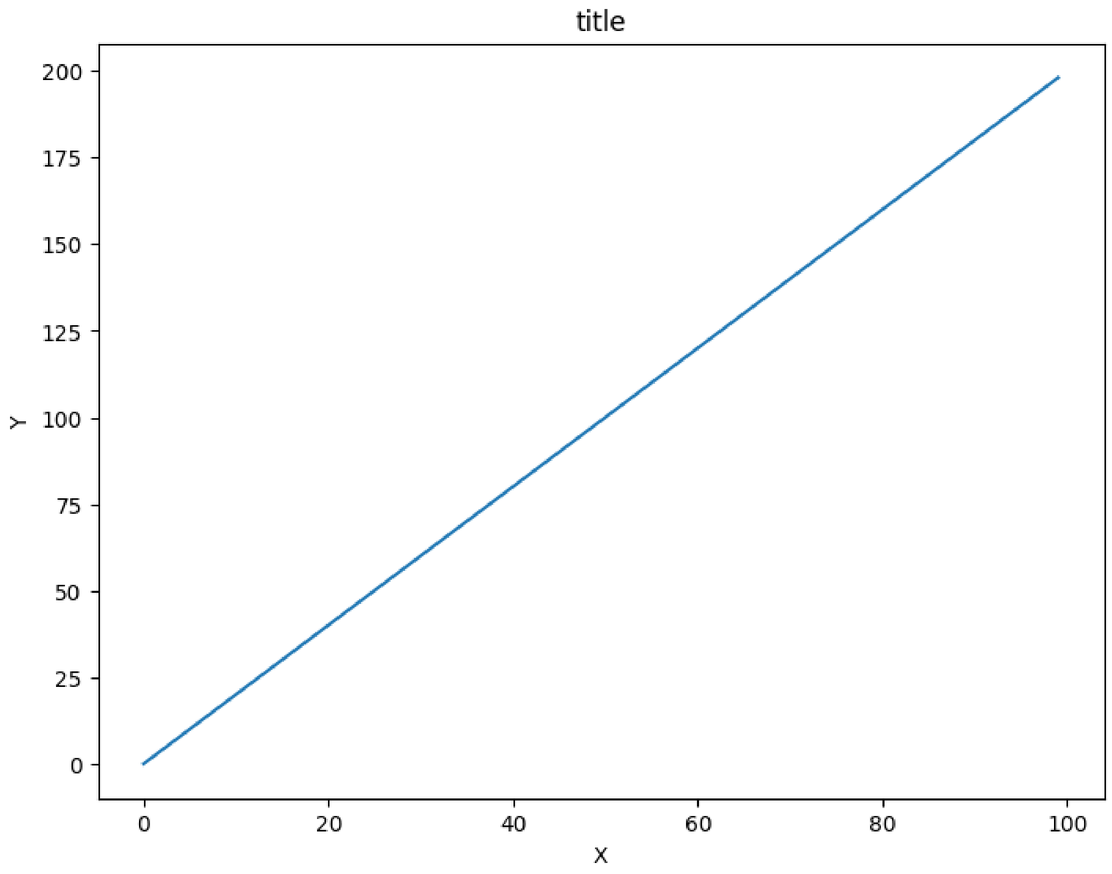
[2]:
```
import matplotlib.pyplot as plt
%matplotlib inline
```

3.2 Exercise 1

** Follow along with these steps: * Create a figure object called fig using plt.figure() * Use add_axes to add an axis to the figure canvas at [0,0,1,1]. Call this new axis ax. * Plot (x,y) on that axes and set the labels and titles to match the plot below:**

[3]:
```python
# Create a figure object
fig = plt.figure()

# Add axes to the figure canvas at [0,0,1,1]
ax = fig.add_axes([0,0,1,1])

# Plot (x,y) on the axes
ax.plot(x, y)

# Set labels and title
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_title('title')
```

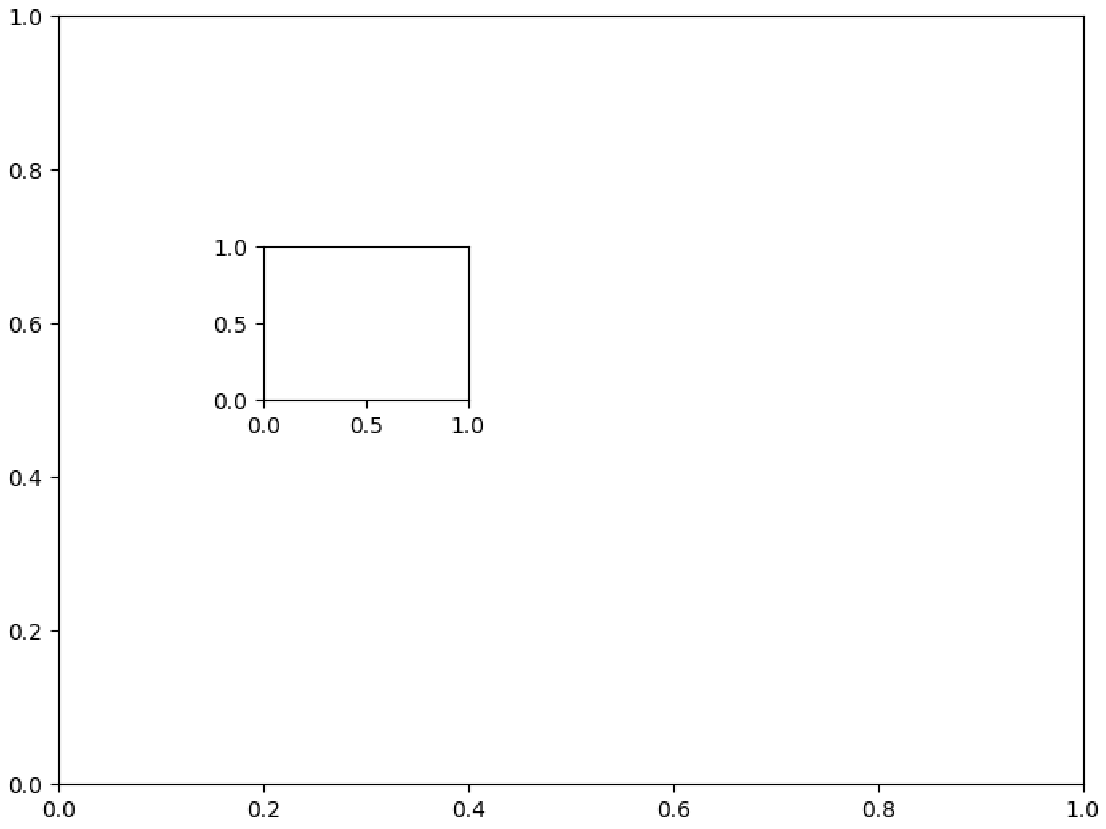
[3]:
```
Text(0.5, 1.0, 'title')
```



3.3 Exercise 2

** Create a figure object and put two axes on it, ax1 and ax2. Located at [0,0,1,1] and [0.2,0.5,.2,.2] respectively.**

[4]:
```
# Create a figure object
fig = plt.figure()

# Add first axis at [0,0,1,1] - full figure
ax1 = fig.add_axes([0,0,1,1])

# Add second axis at [0.2,0.5,.2,.2] - small inset
ax2 = fig.add_axes([0.2,0.5,.2,.2])
```

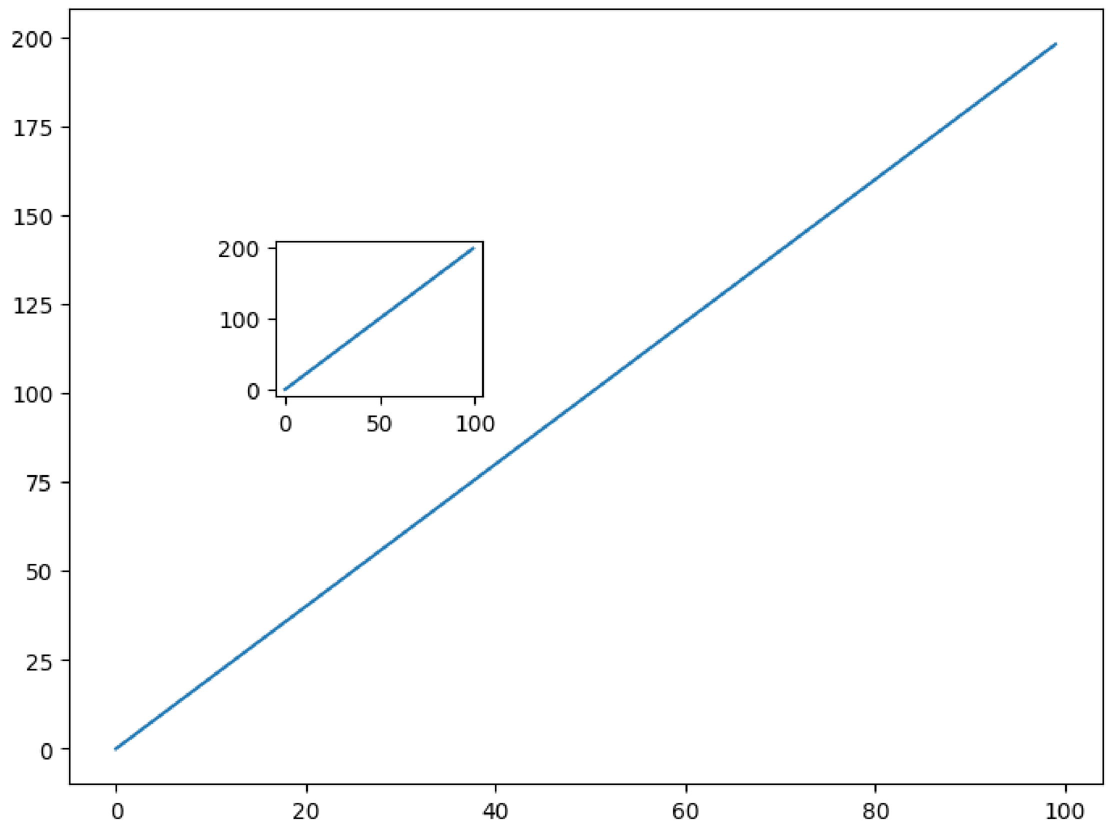


** Now plot (x,y) on both axes. And call your figure object to show it.**

[5]:
```
# Create a figure object
fig = plt.figure()
```

```
# Add first axis at [0,0,1,1] - full figure
ax1 = fig.add_axes([0,0,1,1])

# Add second axis at [0.2,0.5,.2,.2] - small inset
ax2 = fig.add_axes([0.2,0.5,.2,.2])

# Plot (x,y) on both axes
ax1.plot(x, y)
ax2.plot(x, y)
```

[5]:
```
[<matplotlib.lines.Line2D at 0x10f4c0410>]
```

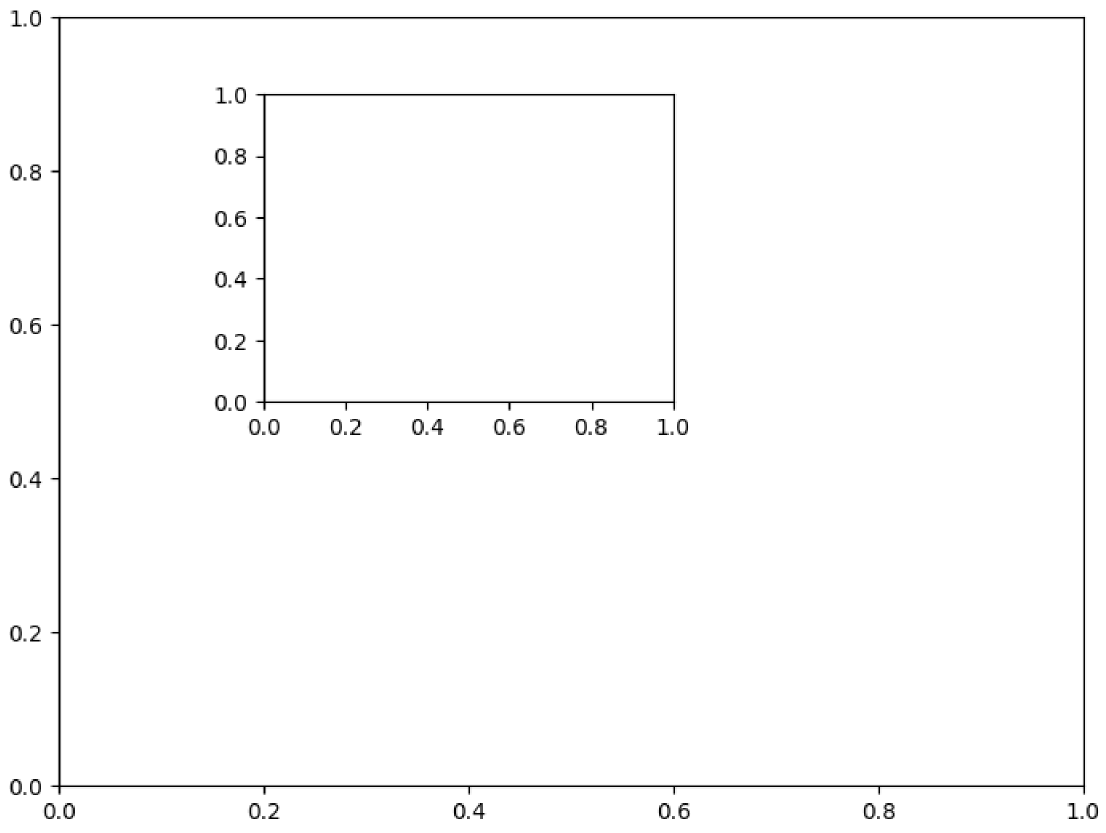


3.4 Exercise 3

** Create the plot below by adding two axes to a figure object at [0,0,1,1] and [0.2,0.5,.4,.4]**

[6]:
```
# Create a figure object
fig = plt.figure()

# Add first axis at [0,0,1,1] - full figure
ax1 = fig.add_axes([0,0,1,1])
```

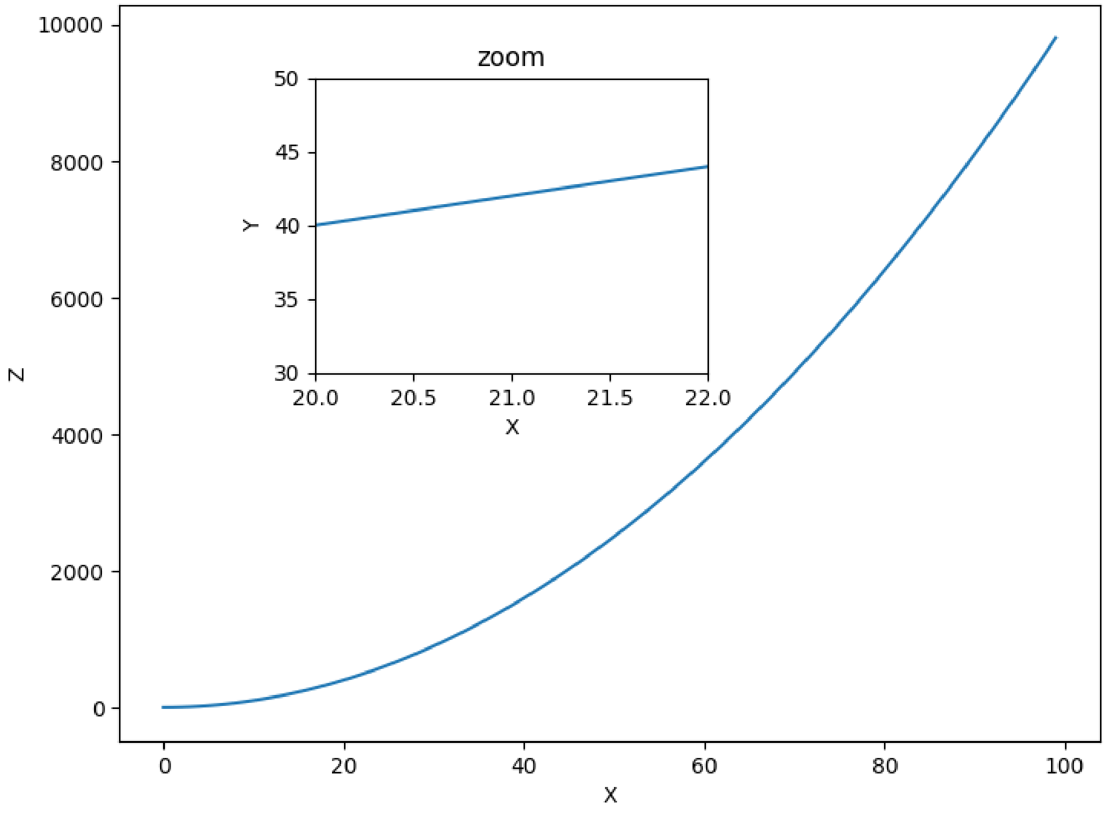
```
# Add second axis at [0.2,0.5,.4,.4] - larger inset
ax2 = fig.add_axes([0.2,0.5,.4,.4])
```



** Now use x,y, and z arrays to recreate the plot below. Notice the xlimits and y limits on the inserted plot:**

[7]:
```
# Create a figure object
fig = plt.figure()

# Add first axis at [0,0,1,1] - full figure
ax1 = fig.add_axes([0,0,1,1])

# Add second axis at [0.2,0.5,.4,.4] - larger inset (zoom window)
ax2 = fig.add_axes([0.2,0.5,.4,.4])

# Plot on main axis (x vs z - quadratic curve)
ax1.plot(x, z)
ax1.set_xlabel('X')
ax1.set_ylabel('Z')
```

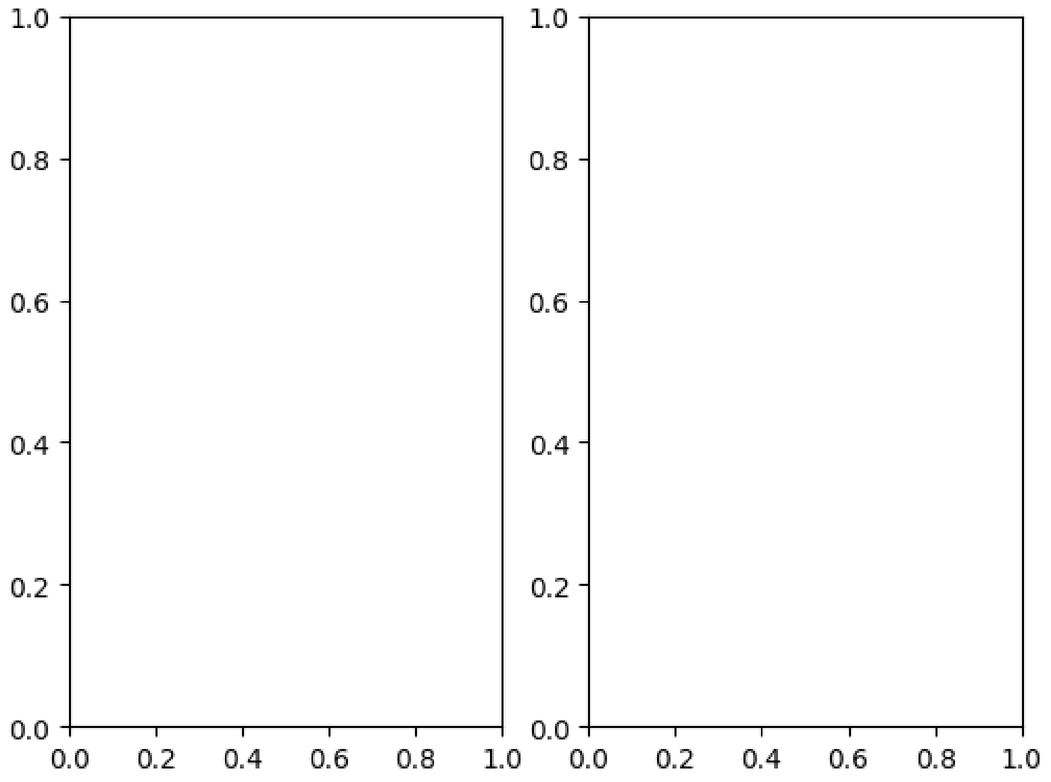
```
# Plot on inset axis with limits (zoom into x vs y - linear relationship)
ax2.plot(x, y)
ax2.set_xlabel('X')
ax2.set_ylabel('Y')
ax2.set_xlim(20, 22)
ax2.set_ylim(30, 50)
ax2.set_title('zoom')

plt.show()
```



3.5 Exercise 4

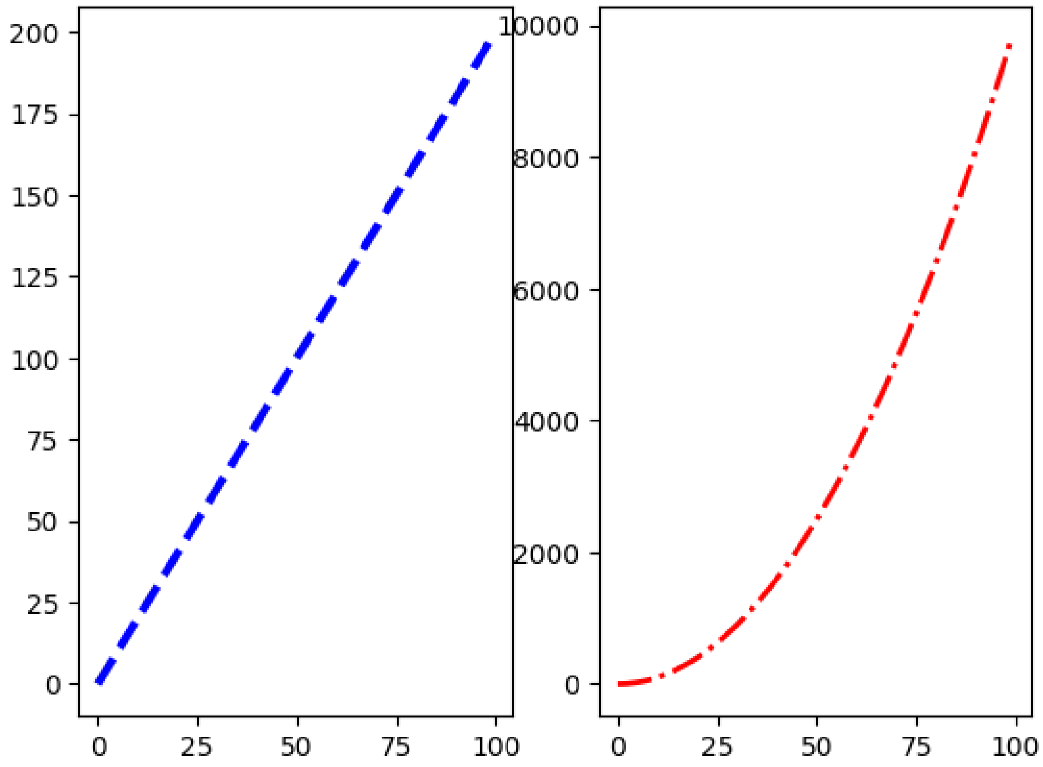
** Use plt.subplots(nrows=1, ncols=2) to create the plot below.**

[8]:
```
# Create subplots with 1 row and 2 columns
fig, axes = plt.subplots(nrows=1, ncols=2)

plt.show()
```



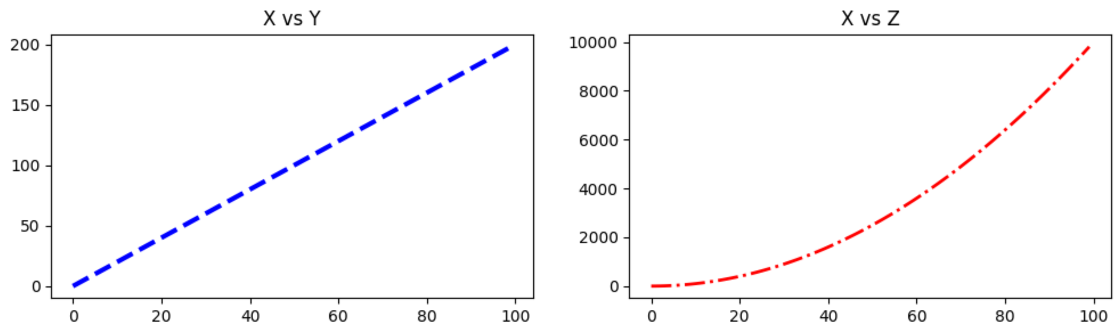
** Now plot (x,y) and (x,z) on the axes. Play around with the linewidth and style**

[9]:
```python
# Create subplots with 1 row and 2 columns
fig, axes = plt.subplots(nrows=1, ncols=2)

# Plot (x,y) on first subplot with linewidth and style
axes[0].plot(x, y, color='blue', linewidth=3, linestyle='--')

# Plot (x,z) on second subplot with different linewidth and style
axes[1].plot(x, z, color='red', linewidth=2, linestyle='-.')

plt.show()
```



** See if you can resize the plot by adding the figsize() argument in plt.subplots() are copying and pasting your previous code.**

[10]:
```python
# Create subplots with 1 row and 2 columns and custom figure size
fig, axes = plt.subplots(nrows=1, ncols=2, figsize=(12, 3))

# Plot (x,y) on first subplot with linewidth and style
axes[0].plot(x, y, color='blue', linewidth=3, linestyle='--')
axes[0].set_title('X vs Y')

# Plot (x,z) on second subplot with different linewidth and style
axes[1].plot(x, z, color='red', linewidth=2, linestyle='-.')
axes[1].set_title('X vs Z')

plt.show()
```


X vs Y
200
150
100
50
0
0
20
40
60
80
100
X vs Z
10000
8000
6000
4000
2000
0
0
20
40
60
80
100

4 Great Job!