# 11_matplotlib_exercise_solution

October 18, 2020

## 1 Matplotlib Uebungen

Nehmt euch hierfuer nun Zeit, Matplotlib kann anfangs schwierig zu verstehen sein. Dies sind relativ einfache Diagramme, aber es trotzdem nicht einfach, wenn man das erste Mal mit matplotlib arbeitet.

Außerdem keine Sorge, falls ihr die Syntax von Matplotlib frustrierend findet. Wir werden als naechstes Seaborn kennenlernen. Seaborn ist weitaus handlicher und perfekt fuer Data Science Anwendungen. Seaborn baut allerdings auf matplotlib auf, weswegen es wichtig ist matplotlib kennengelernt zu haben.

** HINWEIS: ALLE BEFEHLE ZUM PLOTTEN EINER FIGUR SOLLTEN SICH ALLE IN DERSELBEN ZELLE BEFINDEN. DIE TRENNUNG IN MEHRERE ZELLEN KANN DAZU FUEHREN, DASS NICHTS ERSCHEINT. **

## 2 Übungen

Folge den Anweisungen, um die Diagramme mit diesen Daten neu zu erstellen:

### 2.1 Daten

[15]:
```
import numpy as np
x = np.arange(0,100)
y = x*2
z = x**2
```

** Importiere matplotlib.pyplot als plt und setze %matplotlib inline. Welchen Befehl verwendet man, wenn man einen anderen Editor verwendet? **

[16]:
```
import matplotlib.pyplot as plt
%matplotlib inline
# plt.show() wenn Jupyter Notebook nicht verwendet wird
```

### 2.2 Aufgabe 1

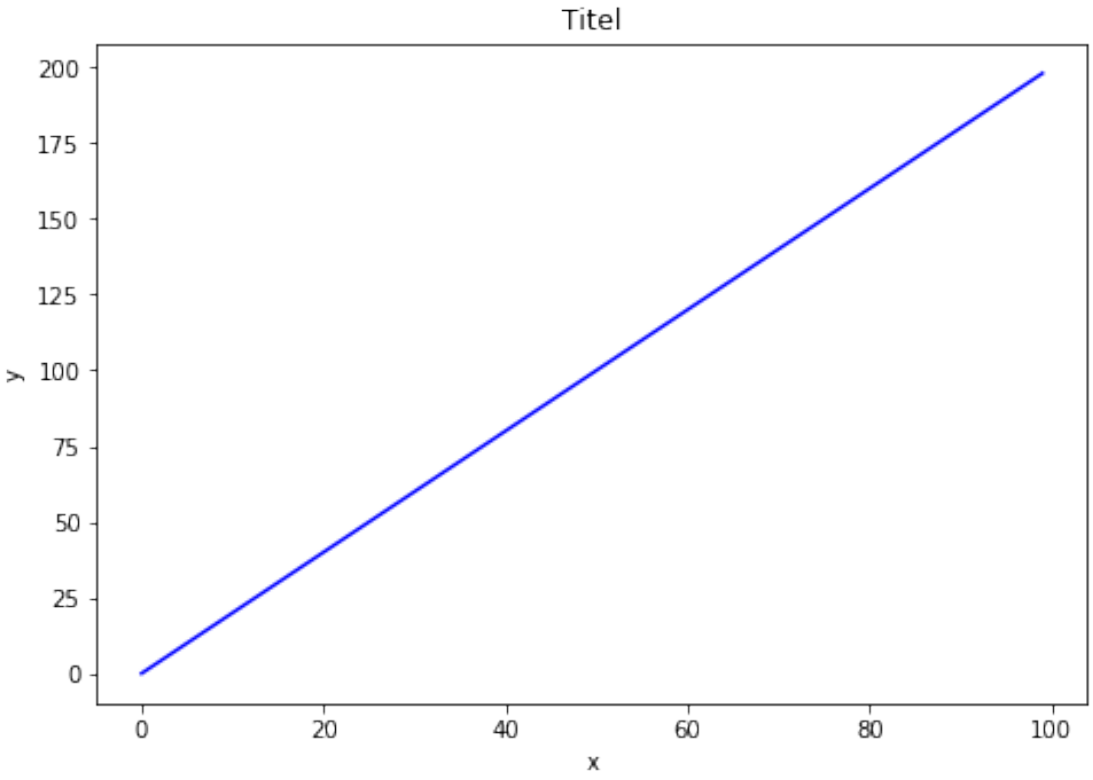
** Führe diese Schritte aus:**

- Erstelle ein Figure-Objekt namens fig mit plt.figure().
- Verwende add_axes, um eine Achse zur Figuren-Leinwand bei [0,0,1,1] hinzuzufuegen. Nenne diese ne
- Plotte (x,y) auf den Achsen und waehle Beschriftung und Titel so, dass sie mit dem untenstehenden

[17]:
```
fig = plt.figure()

ax = fig.add_axes([0,0,1,1])
ax.plot(x,y,'b')
ax.set_xlabel('x')
ax.set_ylabel('y')
ax.set_title('Titel')
```

[17]:
```
Text(0.5,1,'Titel')
```

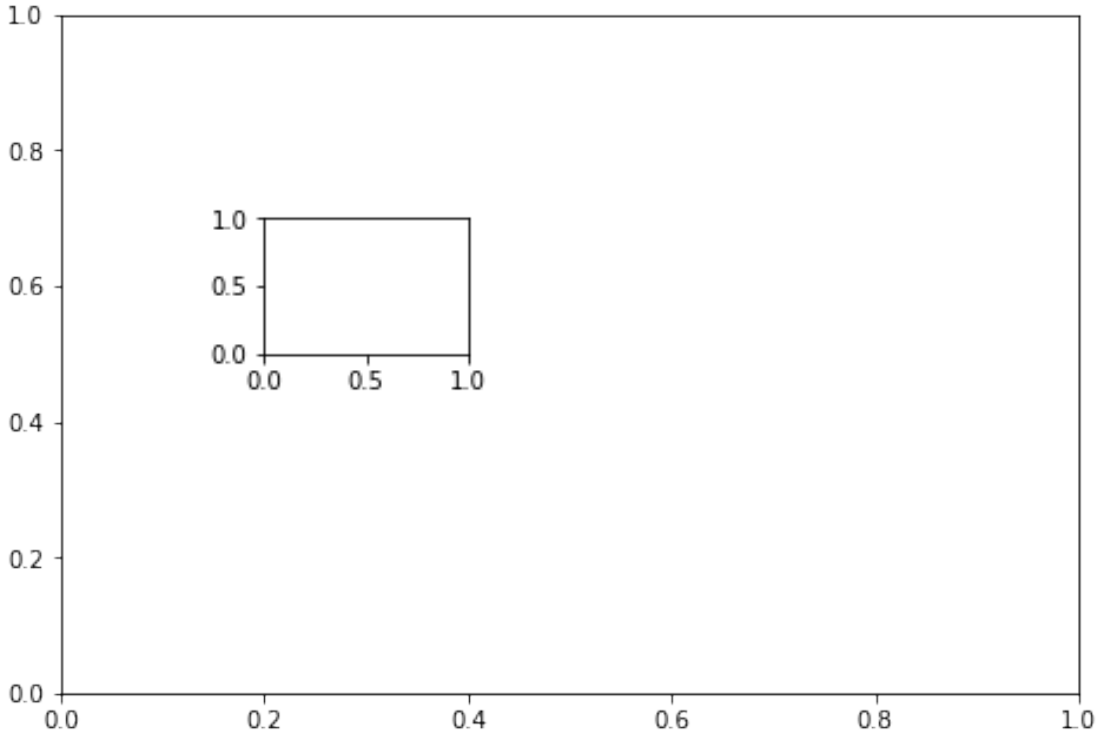



### 2.3 Aufgabe 2

** Erstelle ein Figurenobjekt und setze zwei Achsen darauf, ax1 und ax2. ax1 bei [0,0,1,1] und ax2 bei [0.2,0.5,.2,.2]. **

[18]:
```
fig = plt.figure()
```

```
ax1 = fig.add_axes([0,0,1,1])
ax2 = fig.add_axes([0.2,0.5,.2,.2])
```



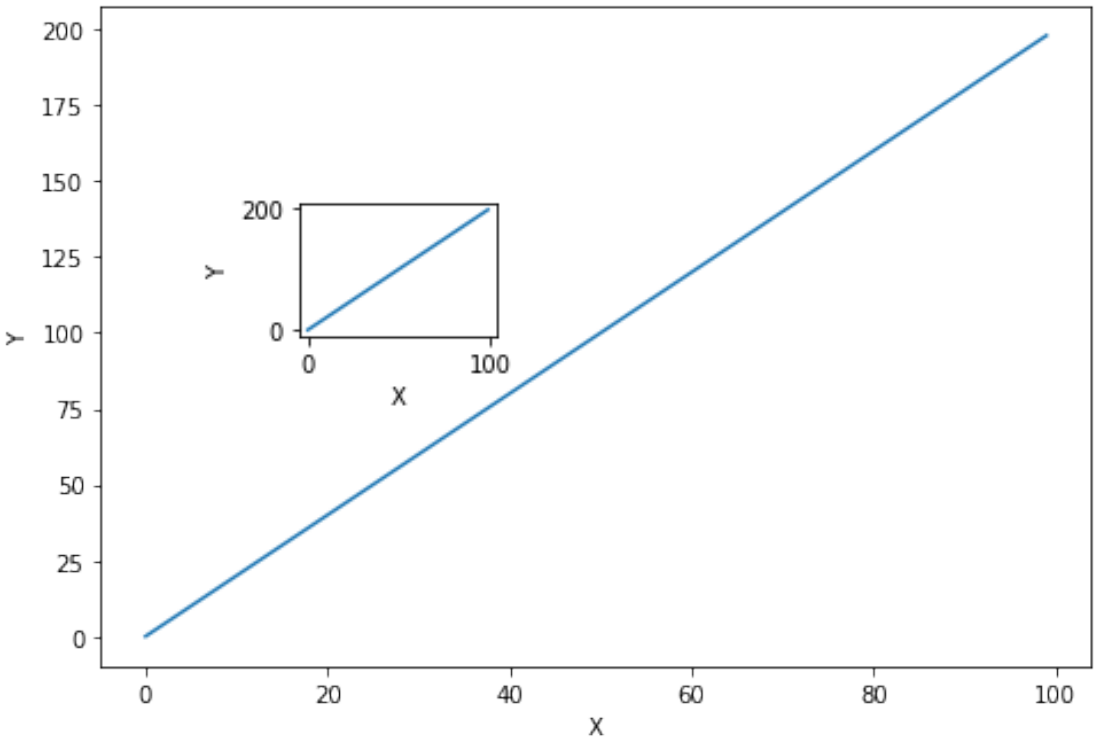

**Plotte nun (x,y) auf beiden Achsen. Und rufe dein Figurenobjekt auf, um es zu zeigen.**

[19]:
```
ax1.plot(x,y)
ax1.set_ylabel('Y')
ax1.set_xlabel('X')

ax2.plot(x,y)
ax2.set_ylabel('Y')
ax2.set_xlabel('X')

fig
```

[19]:

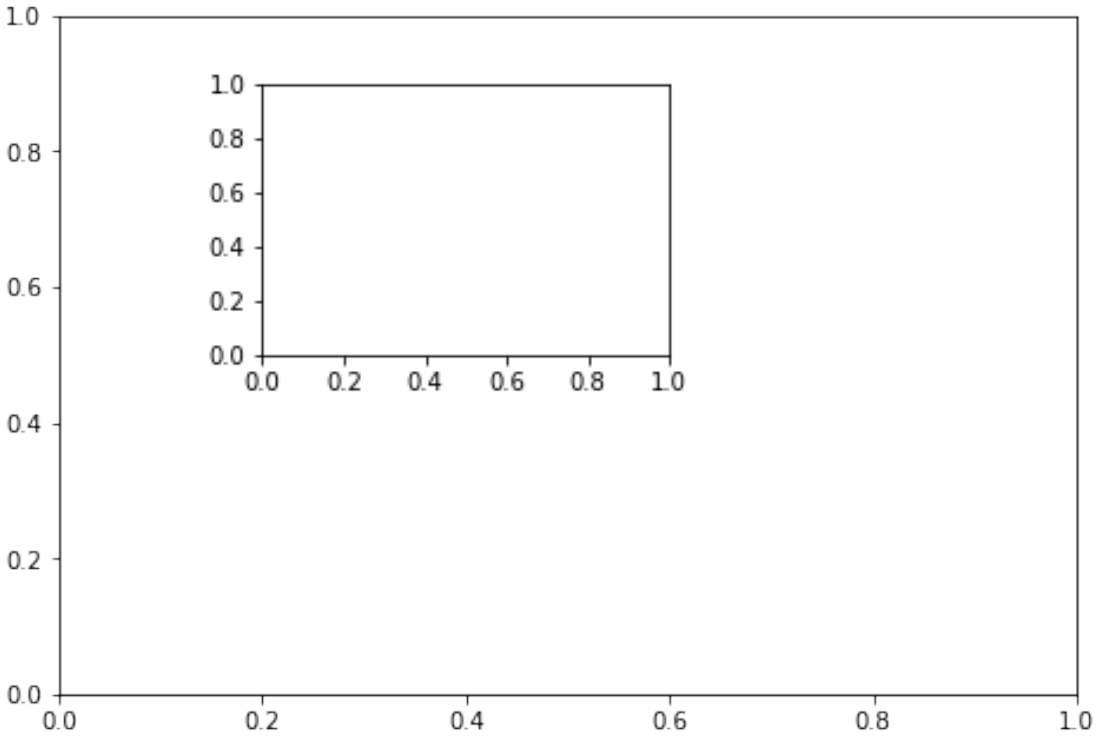



### 2.4 Aufgabe 3

**Erstelle die folgende Darstellung, indem du zwei Achsen zu einem Figurenobjekt bei[0,0,1,1] und[0.2,0.5,.4,.4] hinzufuegst.**

[20]:
```
fig = plt.figure()

ax1 = fig.add_axes([0,0,1,1])
ax2 = fig.add_axes([0.2,0.5,.4,.4])
```



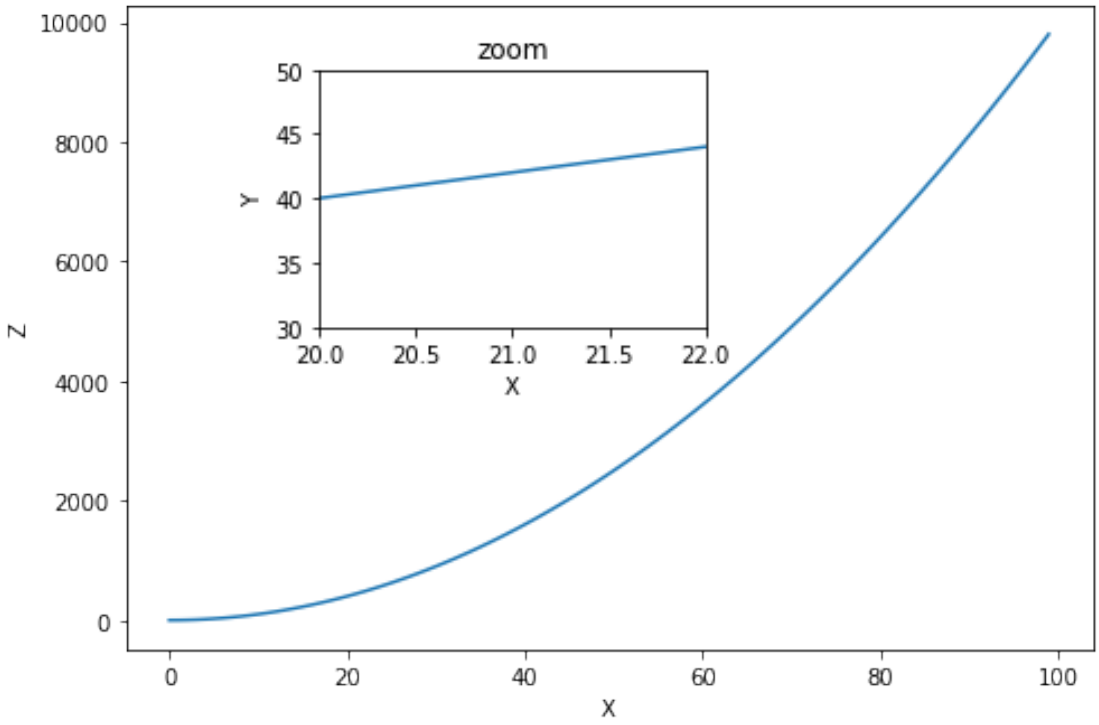

**Verwende nun x,y und z Arrays, um die folgende Darstellung wiederherzustellen. Beachte die x limits und ylimits auf dem eingefuegten Plot:**

[21]:
```
ax1.plot(x,z)
ax1.set_ylabel('Z')
ax1.set_xlabel('X')

ax2.plot(x,y)
ax2.set_ylabel('Y')
ax2.set_xlabel('X')
ax2.set_title('zoom')
ax2.set_xlim(20,22)
ax2.set_ylim(30,50)

fig
```

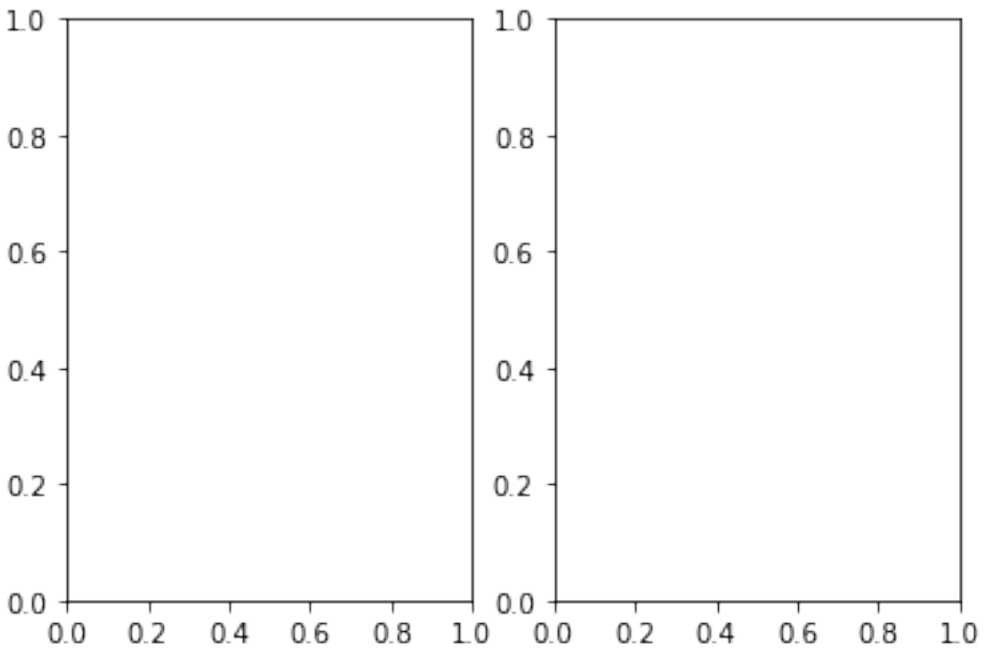
[21]:




### 2.5 Aufgabe 4

**Verwende plt.subplots(nrows=1, ncols=2), um die folgende Darstellung zu erstellen.**

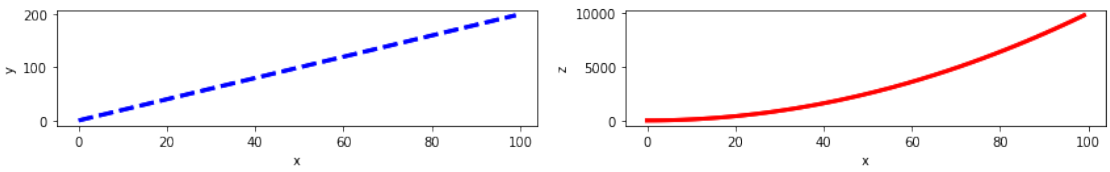
[22]:
```
fig,axes = plt.subplots(nrows=1, ncols=2)
```




** Plotte nun (x,y) und (x,z) auf den Achsen. Spiele mit der Linienbreite und dem Stil herum. Außerdem sorge dafuer, dass es keine Ueberlappung gibt.**

[25]:
```
axes[0].plot(x,y, color='blue', lw=3, ls='dashed')
axes[1].plot(x,z, color='red', lw=3)

fig.tight_layout()
fig
```

[25]:


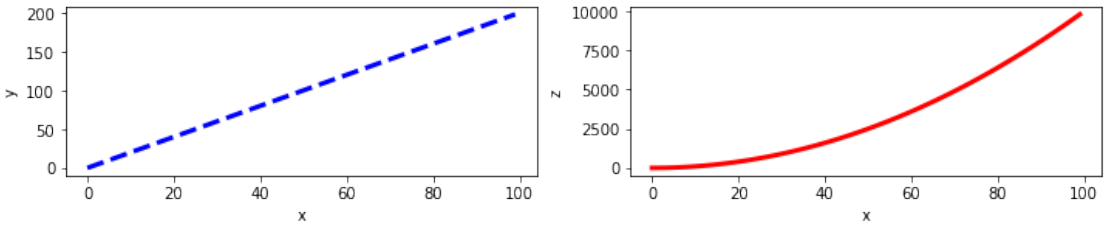

**Ueberpruefe, ob du die Groesse der Darstellung aendern kannst, indem du das Argument figsize() in plt.subplots() hinzufuegst. Kopiere deinen vorherigen Code**

[24]:
```
fig,axes = plt.subplots(nrows=1, ncols=2, figsize=(12,2))

axes[0].plot(x,y, color='blue', lw=3, ls='dashed') #ls='--' auch moeglich
axes[0].set_xlabel('x')
axes[0].set_ylabel('y')
```

```
axes[1].plot(x,z, color='red', lw=3)
axes[1].set_xlabel('x')
axes[1].set_ylabel('z')
```

[24]: `Text(0,0.5,'z')`




## 3 ENDE!