Single Figure

The steps of create a figure in matplotlib

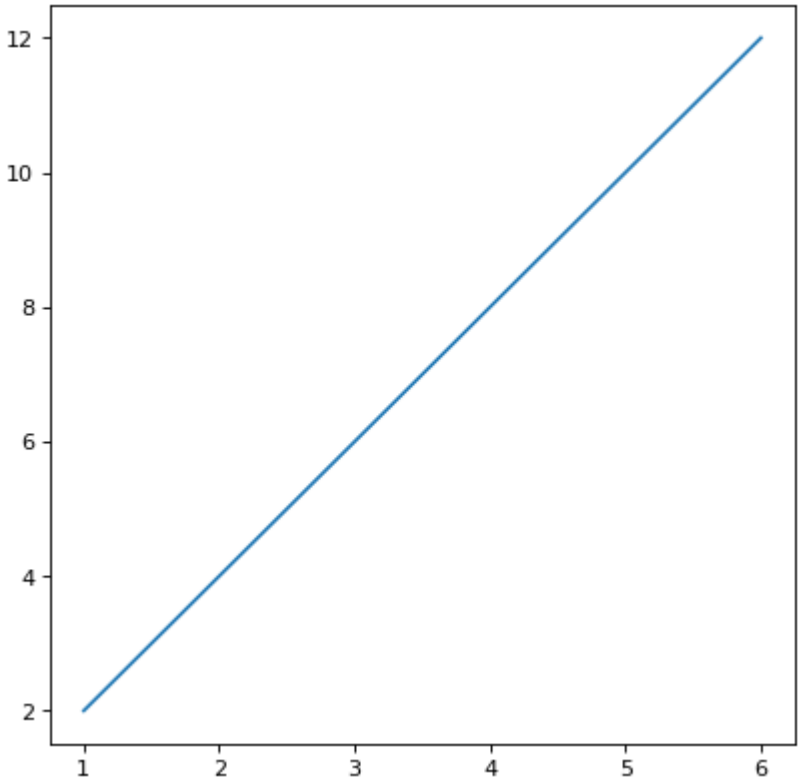
1. Create canvas: plt.figure()
2. Draw the graph: plt.plot(x,y)
3. Show the graph: plt. show()

In [1]:
```
import matplotlib
import matplotlib.pyplot as plt
import numpy as np
```

In [2]:
```
print('matplotlib: {}'.format(matplotlib.__version__))
```

```
matplotlib: 3.6.2
```

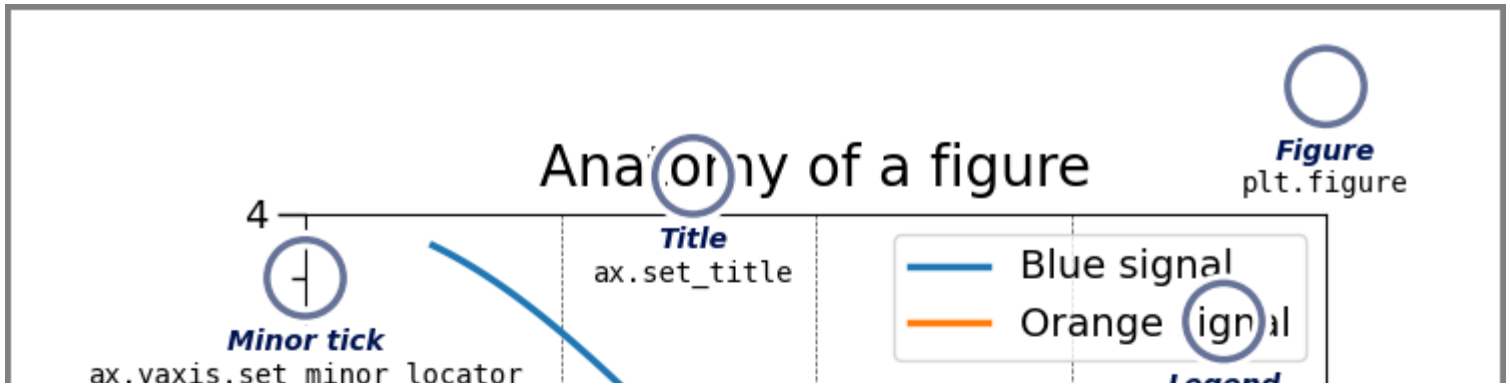
In [3]:
```
plt.figure(figsize = (6, 6), dpi = 80)  #  create an inch-by-inch image, which would be 80-by-80 pixels.
plt.plot([1,2,3,4,5,6],[2,4,6,8,10,12])
plt.show()
```

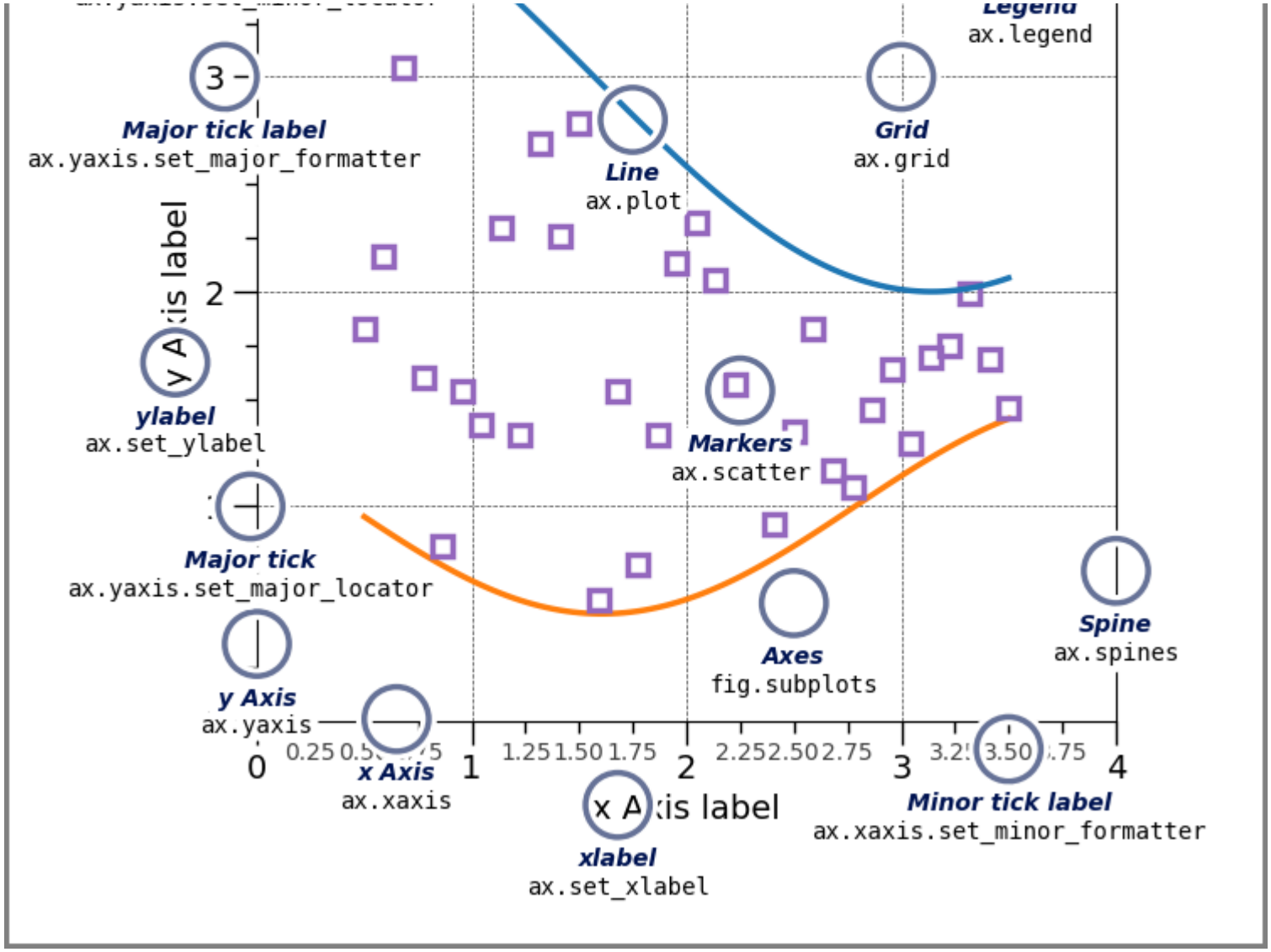


The Anatomy of a figure

This figure shows the name of several matplotlib elements composing a figure

https://matplotlib.org/stable/gallery/showcase/anatomy.html





Add some other elements in the graph

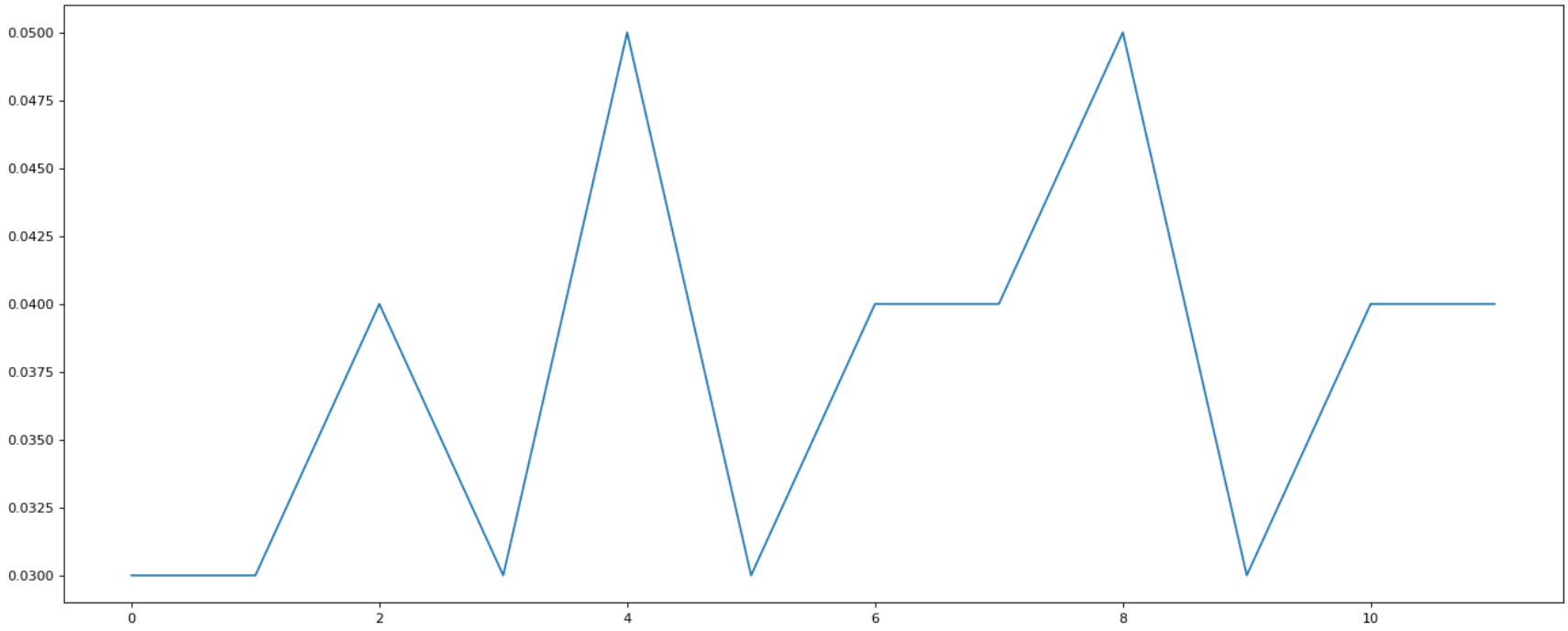
For example, graph the delinquent rates across the time.

In [4]:
```
import random
```

In [5]:
```
x = range(12)
random.seed(123)
y_30days_delinquent = [round(random.uniform(0.03, 0.05),2) for i in x]
```

In [6]:
```
plt.figure(figsize = (20, 8 ), dpi = 80)
plt.plot(x, y_30days_delinquent)
plt.show
```

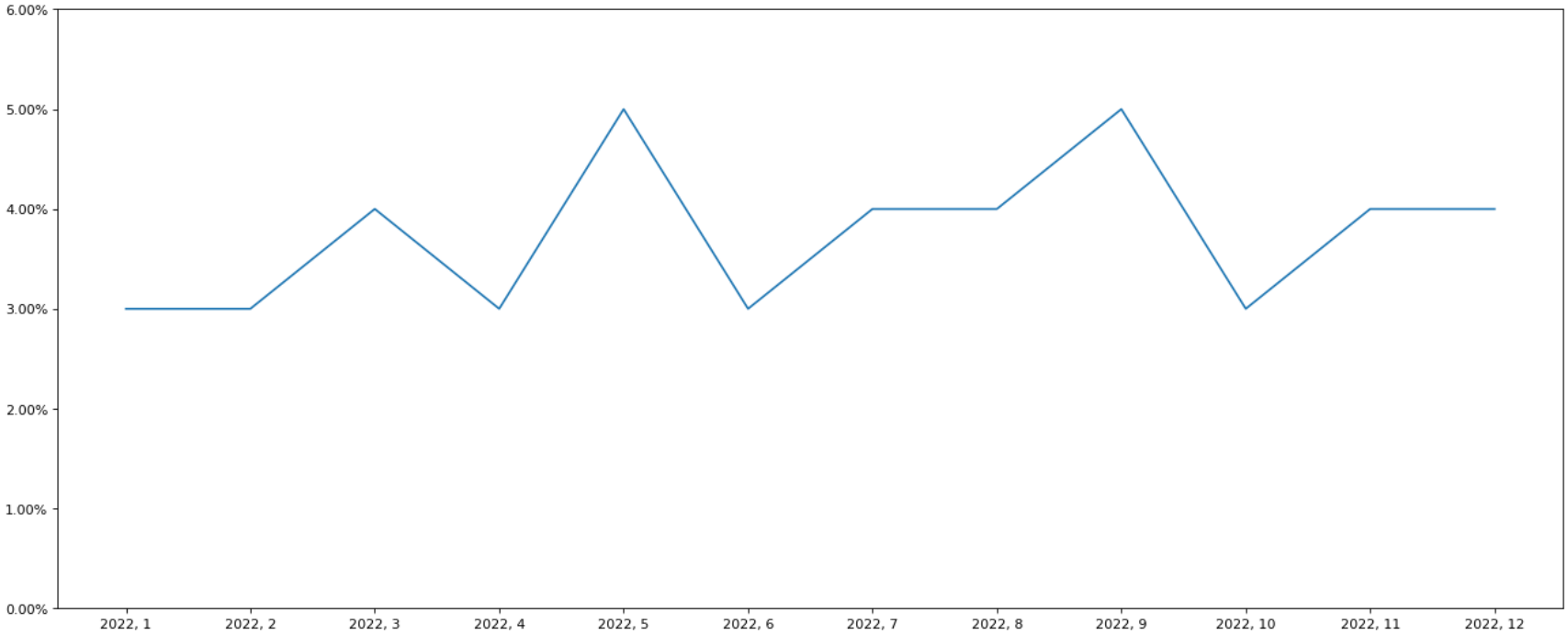
Out[6]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```



Change the x_ticks and y_ticks

In [7]:
```
x = range(12)
random.seed(123)
y_30days_delinquent = [round(random.uniform(0.03, 0.05),2) for i in x]

# Step 1
plt.figure(figsize = (20, 8 ), dpi = 80)

# set up the x_ticks_label, y_ticks_label
```

```
x_ticks_label = ["2022, {}".format(i + 1) for i in x]
step_size = 0.01
y_ticks = [i * step_size for i in range(7)]
y_ticks_label = ['{:.2%}'.format(value) for value in y_ticks]

# Step 2
plt.plot(x, y_30days_delinquent)

# Apply the x_ticks_label and y_tick_lable to the plt object
plt.xticks(x[::1], x_ticks_label[::1])
plt.yticks(y_ticks[:], y_ticks_label[:])

# Step 3
plt.show()
```



In real data, we have more than one year, we can create a list for month label with whatever length we want, for example:

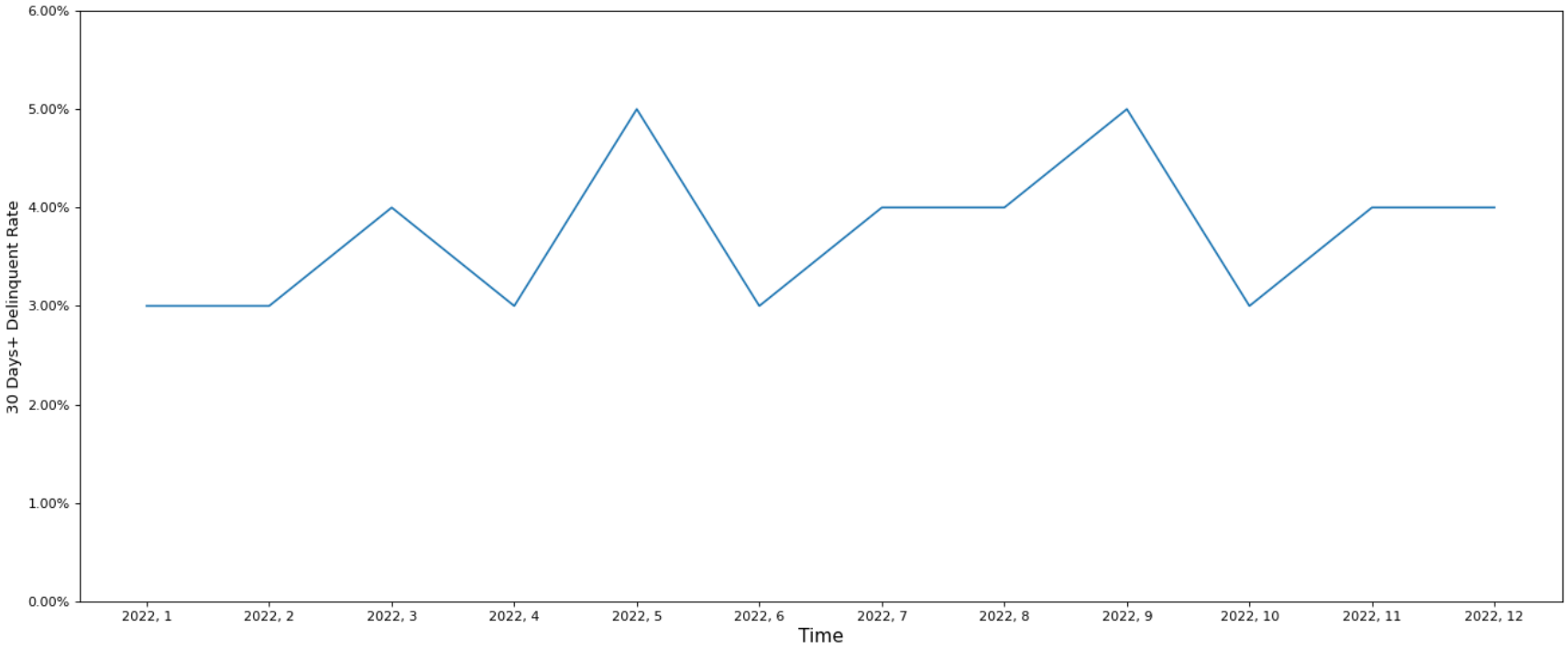
```
import datetime

start_date = datetime.date(2022, 1, 1)
end_date = datetime.date(2023, 12, 31)
```

```
months = []
current_date = start_date
while current_date <= end_date:
    month_str = current_date.strftime('%B %Y')
    months.append(month_str)
    current_date += datetime.timedelta(days = 32)
    current_date = current_date.replace(day = 1)


random.seed(1234)
y_30days_delinquent = [round(random.uniform(0.03, 0.05),2) for i in range(24)]
plt.figure(figsize = (20, 8), dpi = 80)
plt.plot(months, y_30days_delinquent)
plt.xticks(rotation = 45)
plt.show
```

Adding labels

In [8]:
```
x = range(12)
random.seed(123)
y_30days_delinquent = [round(random.uniform(0.03, 0.05),2) for i in x]

# Step 1
plt.figure(figsize = (20, 8 ), dpi = 80)

# set up the x_ticks_label, y_ticks_label
x_ticks_label = ["2022, {}".format(i + 1) for i in x]
step_size = 0.01
y_ticks = [i * step_size for i in range(7)]
y_ticks_label = ['{:.2%}'.format(value) for value in y_ticks]

# Step 2
plt.plot(x, y_30days_delinquent)

# Apply the x_ticks_label and y_tick_lable to the plt object
plt.xticks(x[::1], x_ticks_label[::1])
plt.yticks(y_ticks[:], y_ticks_label[:])
```

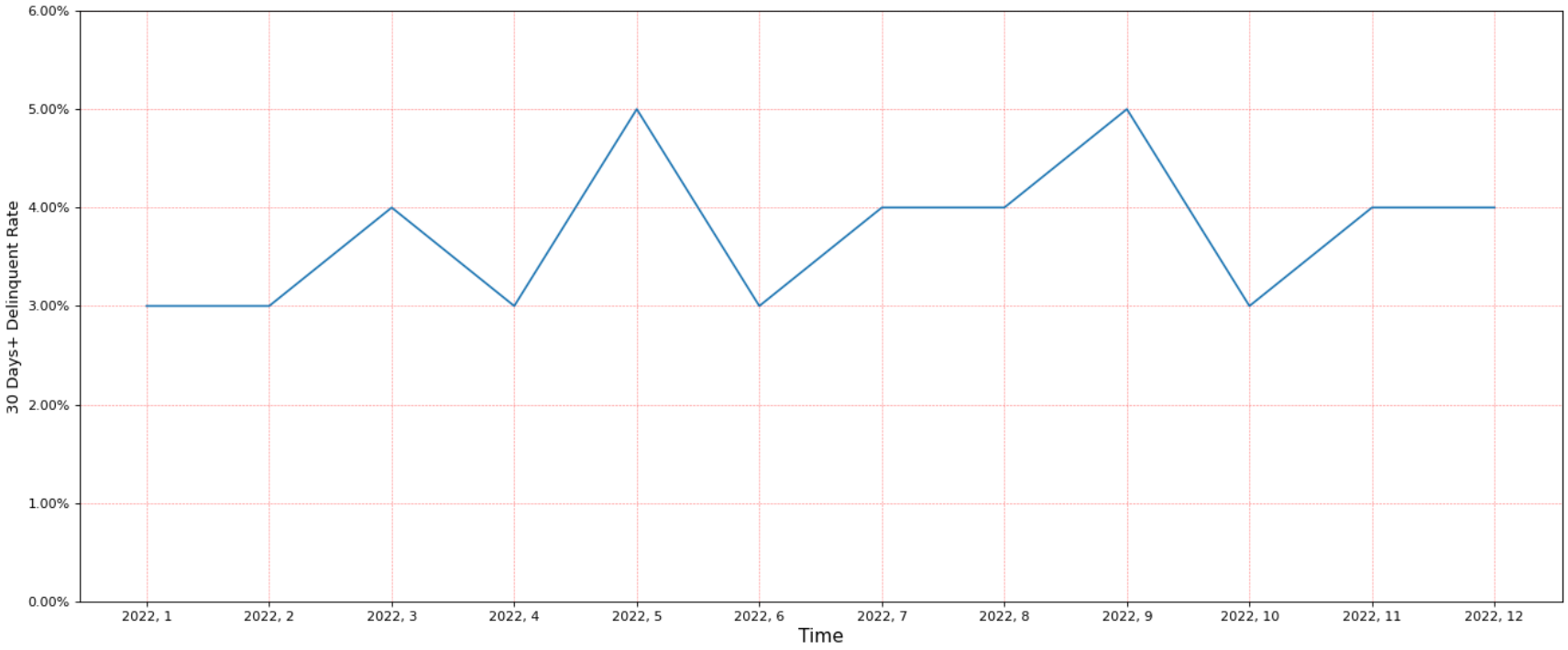
```
# Adding Labels
plt.xlabel("Time", fontsize = 14, loc = "center")   # "You can set it up using string like left, right and center"
plt.ylabel("30 Days+ Delinquent Rate", fontsize = 12, loc= "center")  # Bottom, top and center


# Step 3
plt.show()
```



Adding Grid

In [9]:
```
x = range(12)
random.seed(123)
y_30days_delinquent = [round(random.uniform(0.03, 0.05),2) for i in x]

# Step 1
plt.figure(figsize = (20, 8 ), dpi = 80)

# set up the x_ticks_label, y_ticks_label
x_ticks_label = ["2022, {}".format(i + 1) for i in x]
step_size = 0.01
```

```
y_ticks = [i * step_size for i in range(7)]
y_ticks_label = ['{:.2%}'.format(value) for value in y_ticks]

# Step 2
plt.plot(x, y_30days_delinquent)

# Apply the x_ticks_label and y_tick_lable to the plt object
plt.xticks(x[::1], x_ticks_label[::1])
plt.yticks(y_ticks[:], y_ticks_label[:])


# Adding labels
plt.xlabel("Time", fontsize = 14, loc = "center")   # "You can set it up using string like left, right and center"
plt.ylabel("30 Days+ Delinquent Rate", fontsize = 12, loc= "center")  # Bottom, top and center

# Adding Grids
plt.grid(axis = 'both', color = 'r', linestyle = '--', linewidth = 0.5, alpha = 0.5)


# Saving figure
plt. savefig("test.png")

# Step 3
plt.show()
```

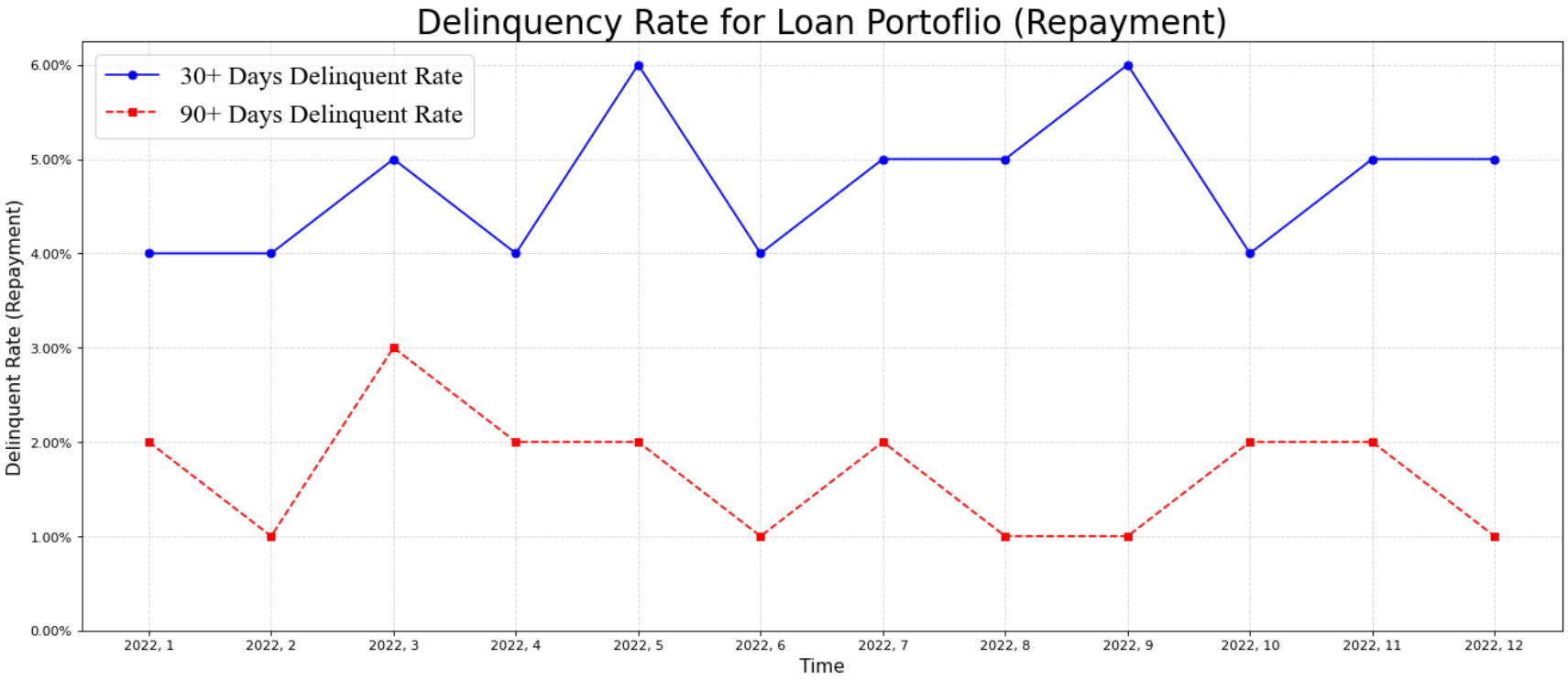


Adding another time series on the figure

In [10]:

```
x = range(12)
random.seed(123)
y_30days_delinquent = [round(random.uniform(0.04, 0.06),2) for i in x]
random.seed(12345)
y_90days_delinquent = [round(random.uniform(0.01, 0.03),2) for i in x]

# Step 1
plt.figure(figsize = (20, 8), dpi = 80)

# set up the x_ticks_label, y_ticks_label
x_ticks_label = ["2022, {}".format(i + 1) for i in x]
step_size = 0.01
y_ticks = [i * step_size for i in range(7)]
y_ticks_label = ['{:.2%}'.format(value) for value in y_ticks]

# Step 2
plt.plot(x, y_30days_delinquent, color = 'b', marker = 'o', label = "30+ Days Delinquent Rate")
```

```
plt.plot(x, y_90days_delinquent, linestyle = "dashed", color = 'r', marker = 's', label = "90+ Days Delinquent Rate")

# Apply the x_ticks_label and y_tick_lable to the plt object
plt.xticks(x[::1], x_ticks_label[::1])
plt.yticks(y_ticks[:], y_ticks_label[:])


# Adding labels
plt.xlabel("Time", fontsize = 14, loc = "center")   # "You can set it up using string like left, right and center"
plt.ylabel("Delinquent Rate (Repayment)", fontsize = 14, loc= "center")  # Bottom, top and center
plt. title("Delinquency Rate for Loan Portoflio (Repayment)", fontsize = 25)
# Adding Grid
plt.grid(True, linestyle = "--", alpha = 0.5)

# Adding Lengend
plt.legend(loc = "best", prop = {'family':'Times New Roman', 'size': 20})

plt. savefig("./test.png")
# Step 3
plt.show()
```



Color Symbol	Style Symbol
r Red	- solid
g Green	- - dashed
b Blue	-. dotted
w White	: dashdot
c Cyan	' ' Space
m Magenta	
y Yellow	
k Black	

For linestyle: https://matplotlib.org/stable/gallery/lines_bars_and_markers/linestyles.html

For Color: https://matplotlib.org/stable/tutorials/colors/colors.html

For Marker: https://matplotlib.org/stable/gallery/lines_bars_and_markers/marker_reference.html

Subplots

There are three type of subplots:

1. subplot

```
plt.subplot(nrows, ncols, index)
```

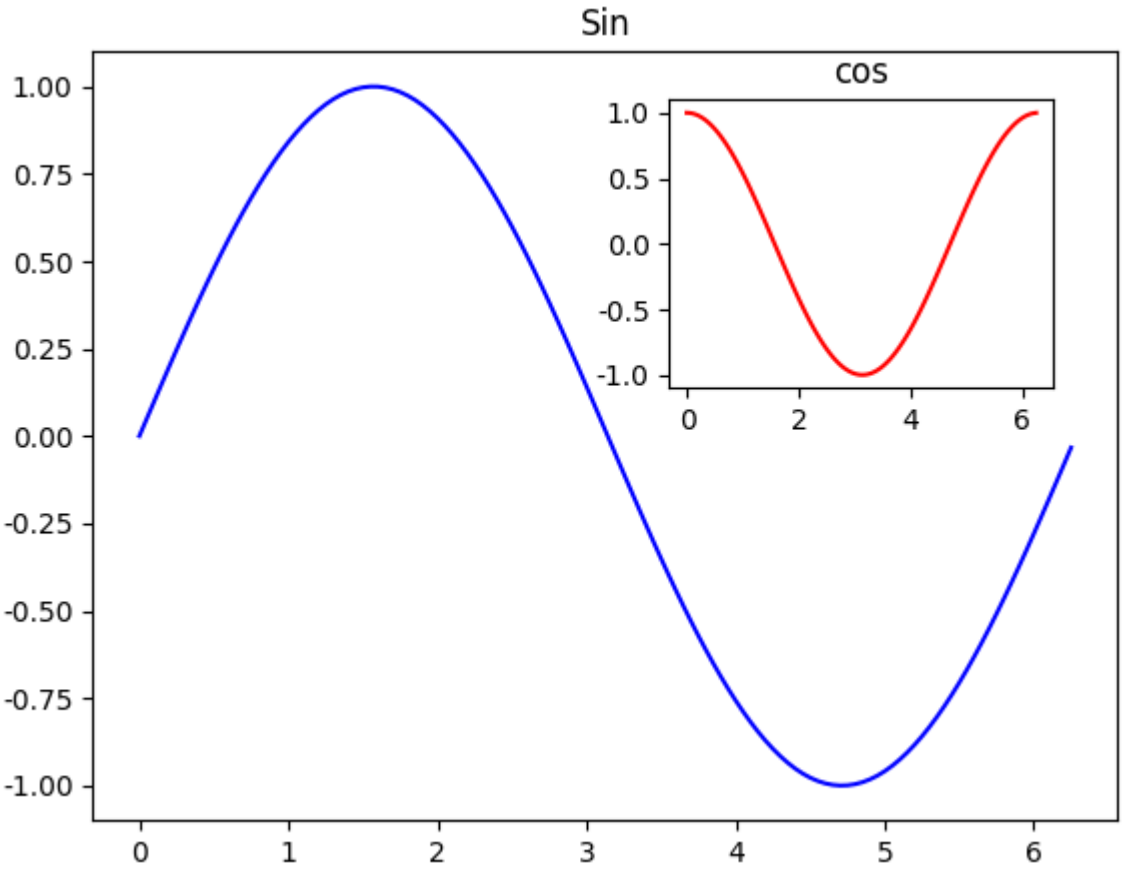
2. subplots

```
matplotlib.pyplot.subplots(nrows = 1, ncols = 1, **fig_kw)
```

Example of subplot (Add a subplot inside the main canvas)

In [11]:
```
import math
plt.rcParams['axes.unicode_minus']=False
x = np.arange(0, math.pi * 2, 0.05)
```

```
y = np.sin(x)
fig= plt.figure()

axes1 = fig. add_axes([0.1, 0.1, 0.8, 0.8]) # Main axis
axes2 = fig. add_axes([0.55, 0.55, 0.3, 0.3]) # Internal Ne
axes1. plot(x, y, 'b')
axes2. plot(x, np.cos(x),'r')
axes1.set_title('Sin')
axes2.set_title("cos")
plt.show()
```



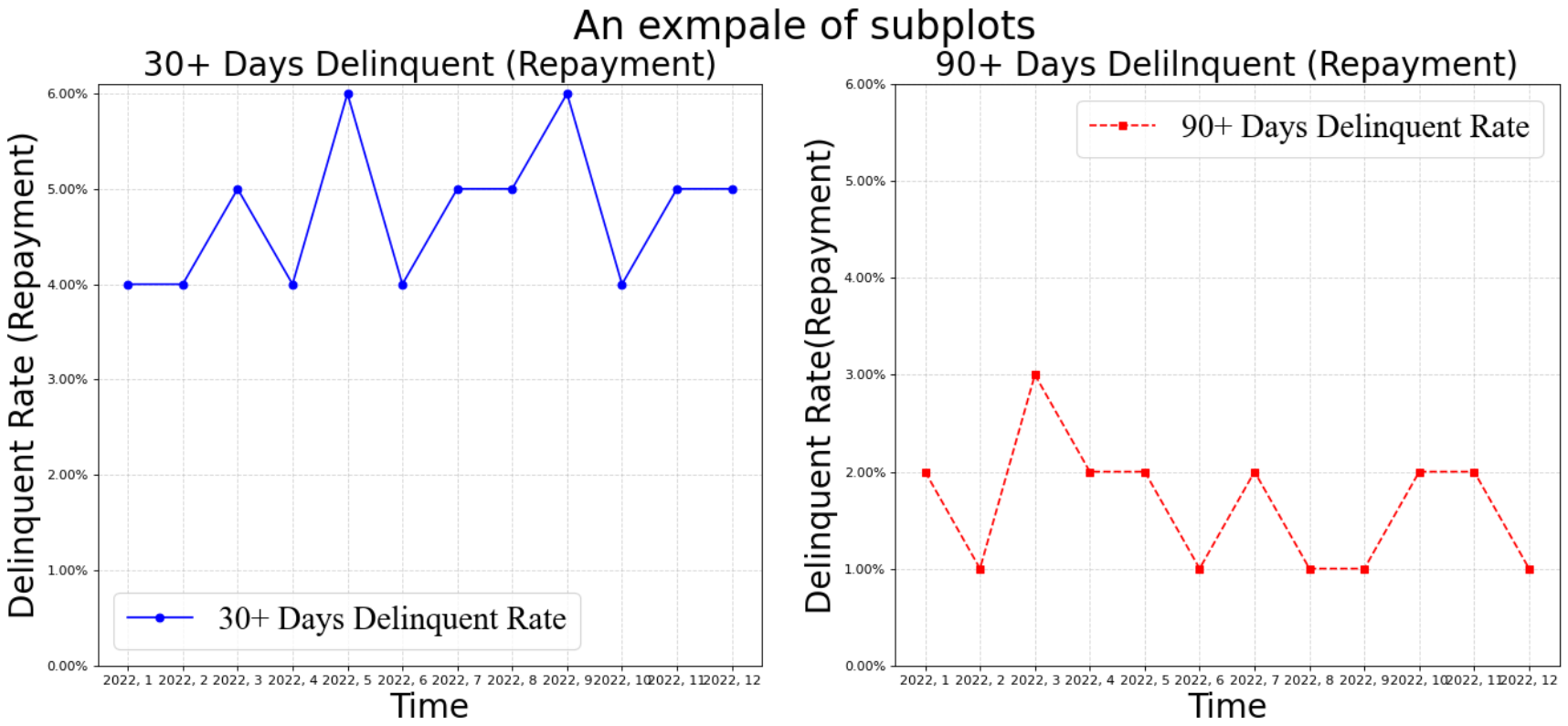
Subplots

To show 30 days+ and 90 days+ delinquent in subplots

In [12]:

```
# 0. Prepare the data we want
x = range(12)
random.seed(123)
y_30days_delinquent = [round(random.uniform(0.04, 0.06),2) for i in x]
random.seed(12345)
y_90days_delinquent = [round(random.uniform(0.01, 0.03),2) for i in x]

# 1. We create the canvas
# plt.figure(figsize = (20, 8), dpi = 80)
fig, axes = plt.subplots(nrows = 1, ncols = 2, figsize = (20, 8), dpi = 80)

#2. Graph the figure
axes[0].plot(x, y_30days_delinquent, color = 'b', marker = 'o', label = "30+ Days Delinquent Rate")
axes[1].plot(x, y_90days_delinquent, linestyle = "dashed", color = 'r', marker = 's', label = "90+ Days Delinquent Rate'

x_ticks_label = ["2022, {}".format(i + 1) for i in x]
step_size = 0.01
y_ticks = [i * step_size for i in range(7)]
y_ticks_label = ['{:.2%}'.format(value) for value in y_ticks]

axes[0].set_xticks(x[::1])
axes[0].set_yticks(y_ticks[::])
axes[0].set_xticklabels(x_ticks_label[::1],fontsize = 10)
axes[0].set_yticklabels(y_ticks_label[::])
axes[1].set_xticks(x[::1])
axes[1].set_yticks(y_ticks[::])
axes[1].set_xticklabels(x_ticks_label[::1],fontsize = 10)
axes[1].set_yticklabels(y_ticks_label[::])

axes[0].grid(linestyle= "--", alpha= 0.5)
axes[1].grid(linestyle= "--", alpha= 0.5)

axes[0].set_xlabel("Time",fontsize= 25)
axes[0].set_ylabel("Delinquent Rate (Repayment)",fontsize = 25)
axes[0].set_title("30+ Days Delinquent (Repayment)", fontsize = 25)
axes[1].set_xlabel("Time",fontsize= 25)
axes[1].set_ylabel("Delinquent Rate(Repayment)",fontsize = 25)
axes[1].set_title("90+ Days Delilnquent (Repayment)", fontsize = 25)


axes[0].legend(loc = 0,prop= {'family':'Times New Roman','size':25})
axes[1].legend(loc = 0,prop= {'family':'Times New Roman','size':25})
```

```
fig.suptitle("An exmpale of subplots", fontsize = 30)
# plt. savefig("./test.png")
# Step 3
plt.show()
```



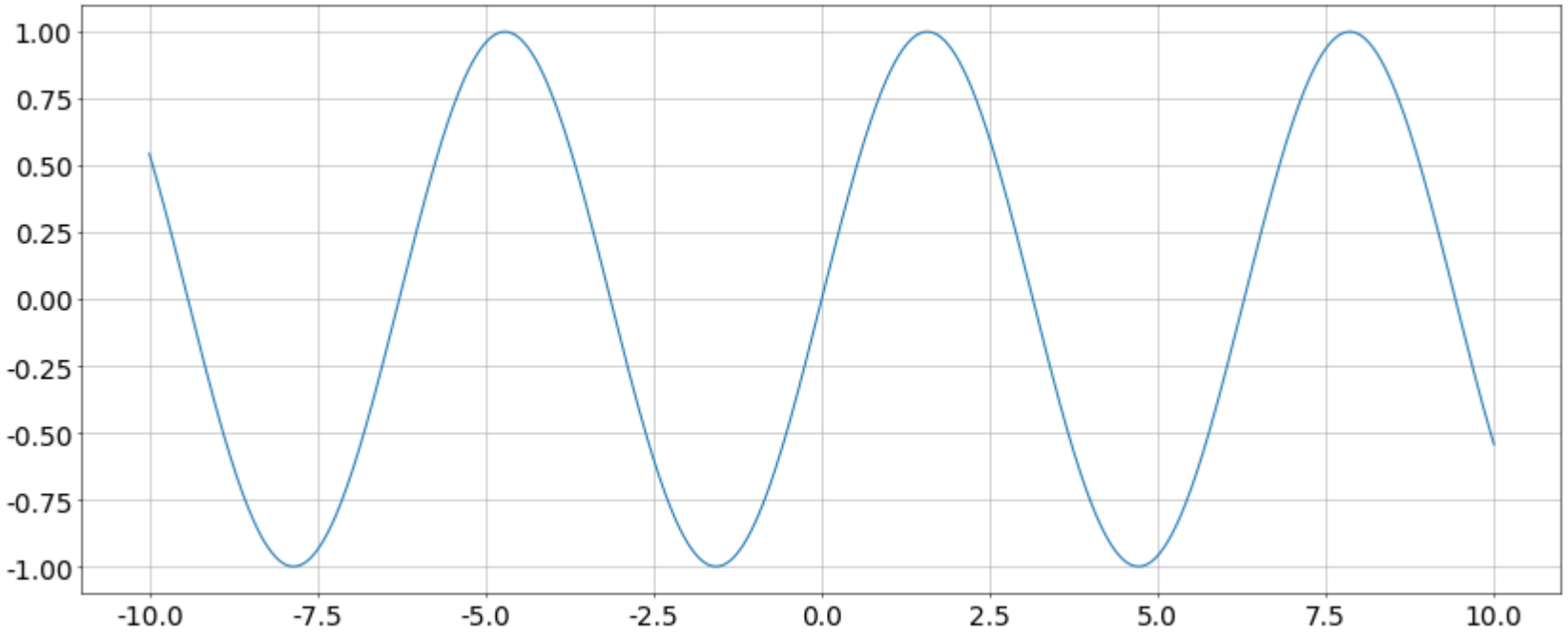
For more examples regarding 2 by 2 or N by N subplots, see the official documents also including sharing the indexes and axes.
https://matplotlib.org/stable/gallery/subplots_axes_and_figures/subplots_demo.html

Other basic graph

Line Graph

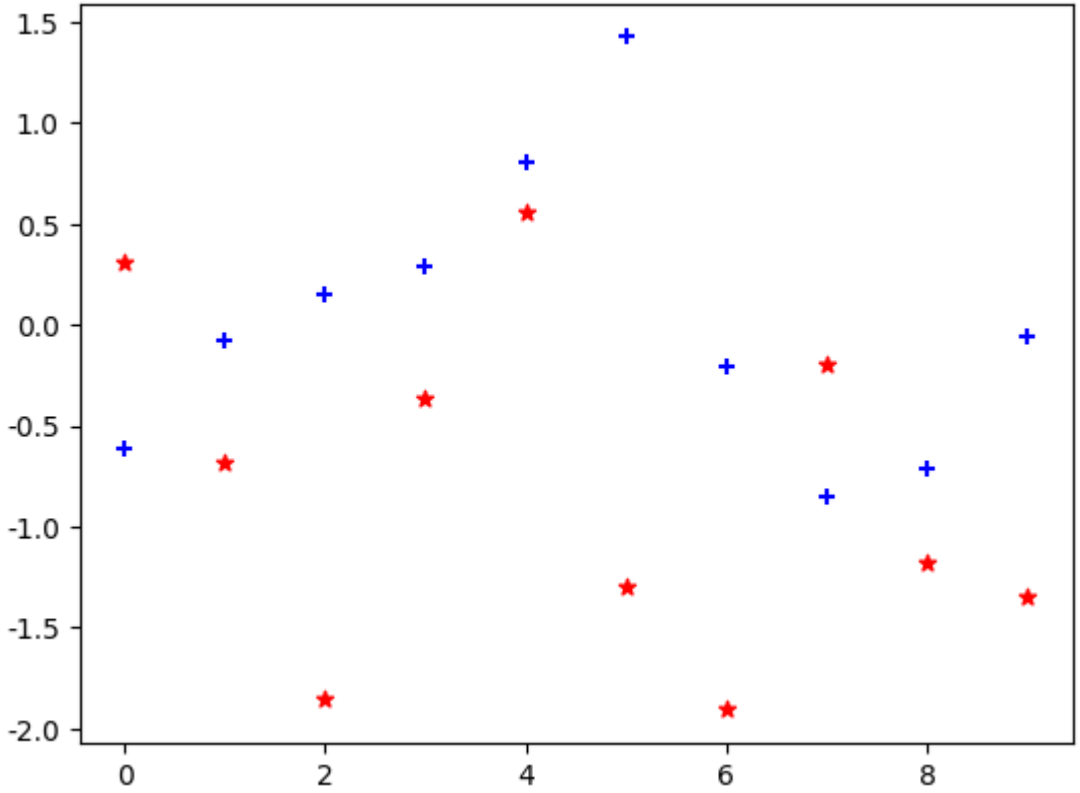
In [13]:
```
# 0.Prepare the date
x = np.linspace(- 10, 10, 1000)
y = np. sin(x)
```

```
# 1.create the canvas
plt.figure(figsize = (20, 8), dpi = 50)
plt.xticks(fontsize = 20)
plt.yticks(fontsize = 20)
# 2. Graph the line
plt.plot(x, y)
# 2.1 Add the grid line
plt. grid()
# 3.Show the graph
plt. show()
```



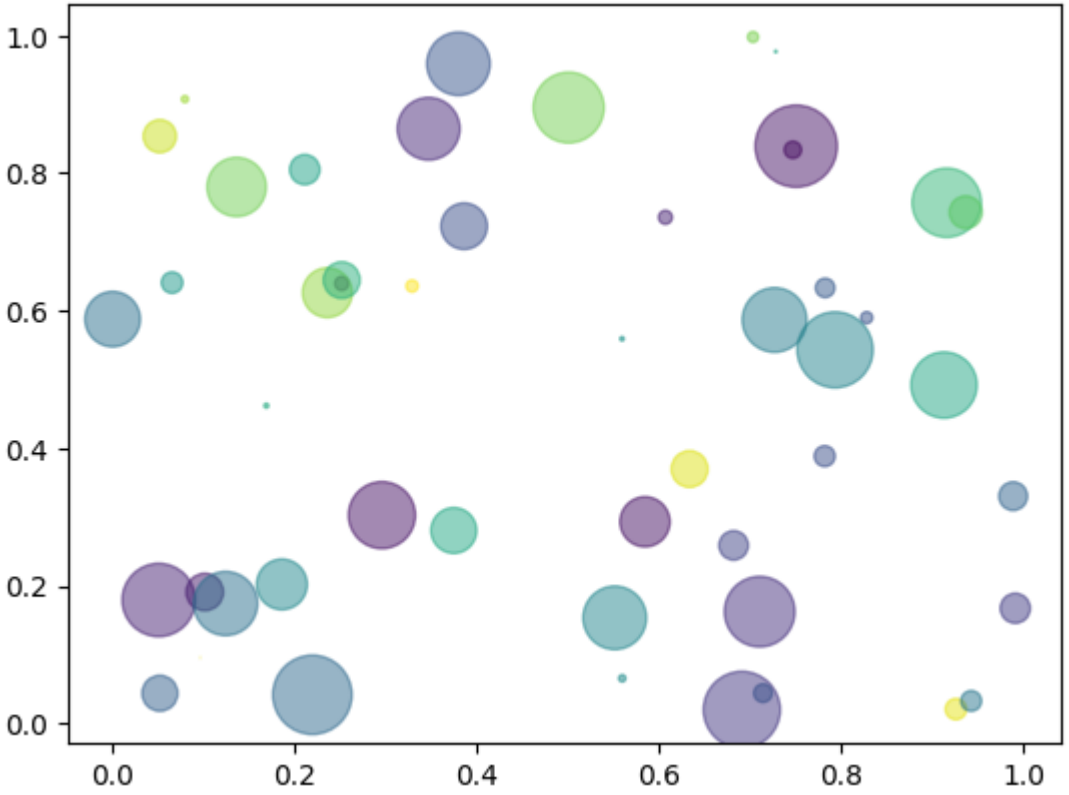
Scatter plot

In [14]:
```
x = np.arange(10)
y = np.random.randn(10)
z = np.random.randn(10)
plt.scatter(x, y, color = 'blue', marker = '+')
plt.scatter(x, z, color = 'red', marker = '*')
plt.show()
```



Bubble Plot

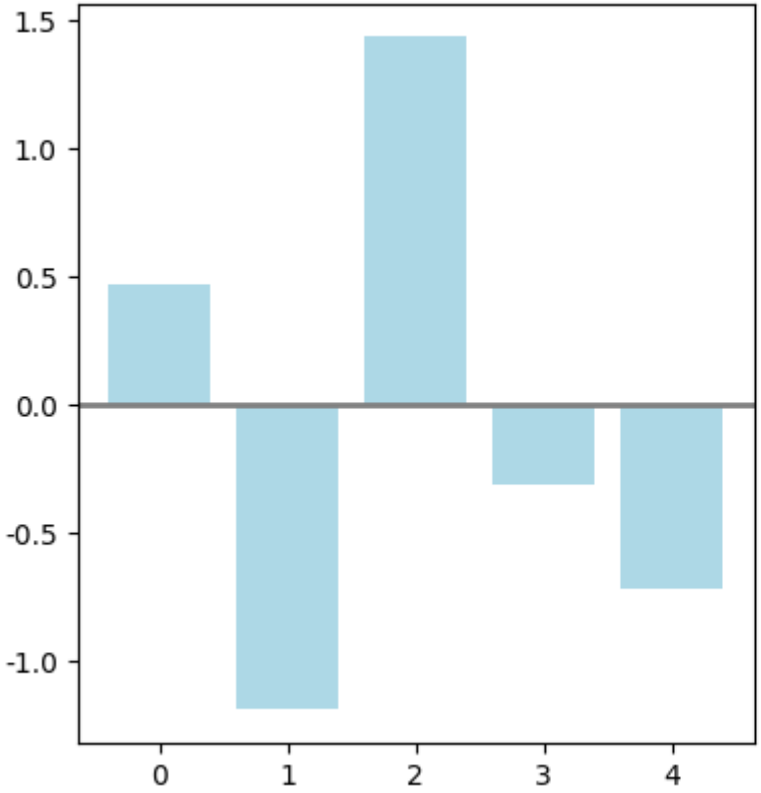
In [15]:
```
random.seed(1234)
N = 50
x = np.random.rand(N)
y = np.random.rand(N)
colors = np.random.rand(N)
area = (30 * np.random.rand(N))** 2
plt. scatter(x, y, s = area, c = colors, alpha = 0.5)
plt. show()
```

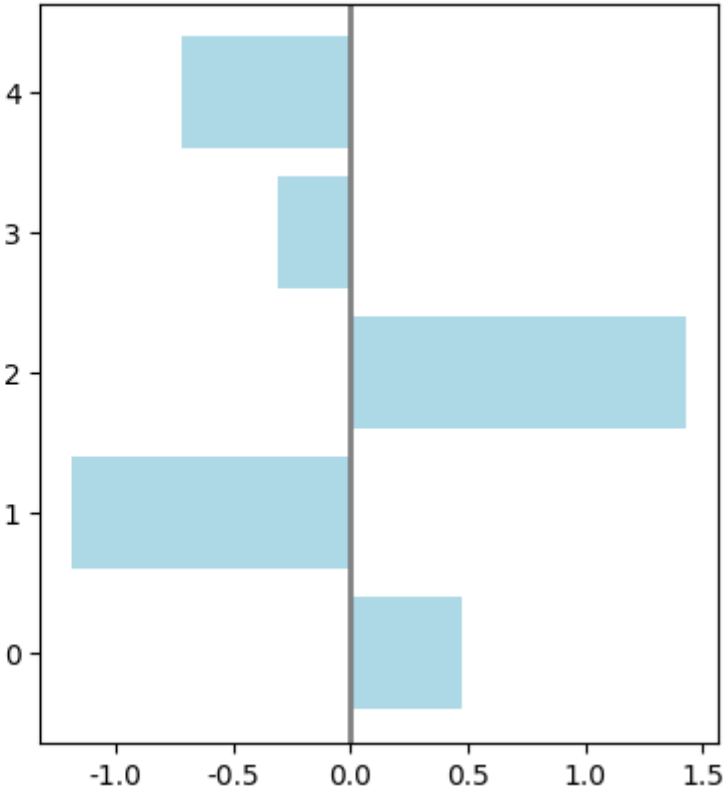


Bar Plot

In [16]:

```
np.random.seed(1234)
x = np.arange(5)
y = np.random.randn(5)
fig, axes = plt. subplots(ncols=2, figsize= plt.figaspect(1./ 2))
vert_bars = axes[0].bar(x, y, color = 'lightblue', align = 'center')
horiz_bars = axes[1].barh(x, y, color = 'lightblue', align = 'center')

axes[0]. axhline(0, color = 'gray', linewidth = 2)
axes[1]. axvline(0, color = 'gray', linewidth = 2)
plt. show()
```



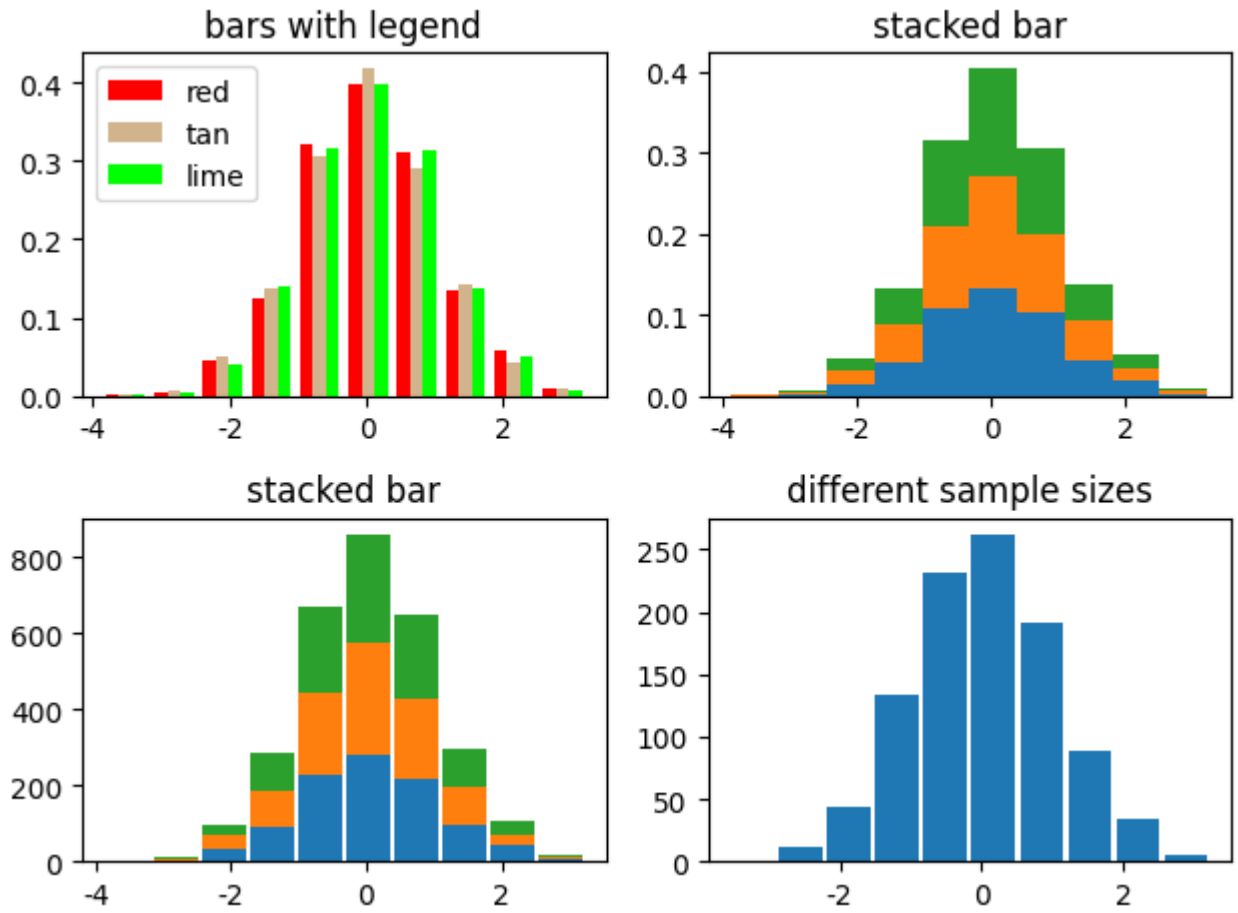


In [17]:
```
x
```

Out[17]:
```
array([0, 1, 2, 3, 4])
```

In [18]:
```
np.random.seed(1234)
n_bins = 10
x = np.random.randn(1000, 3)
fig, axes = plt.subplots(nrows = 2, ncols = 2)
ax0, ax1, ax2, ax3 = axes.flatten()  # flatten method to return a copy of the array collapsed into one dimension
colors = ['red', 'tan', 'lime']
ax0.hist(x, n_bins, density= True , histtype = 'bar', color = colors, label = colors)
ax0.legend(prop = {'size': 10})
ax0.set_title('bars with legend')

ax1.hist(x, n_bins, density = True , histtype = 'barstacked')
ax1.set_title('stacked bar')
```

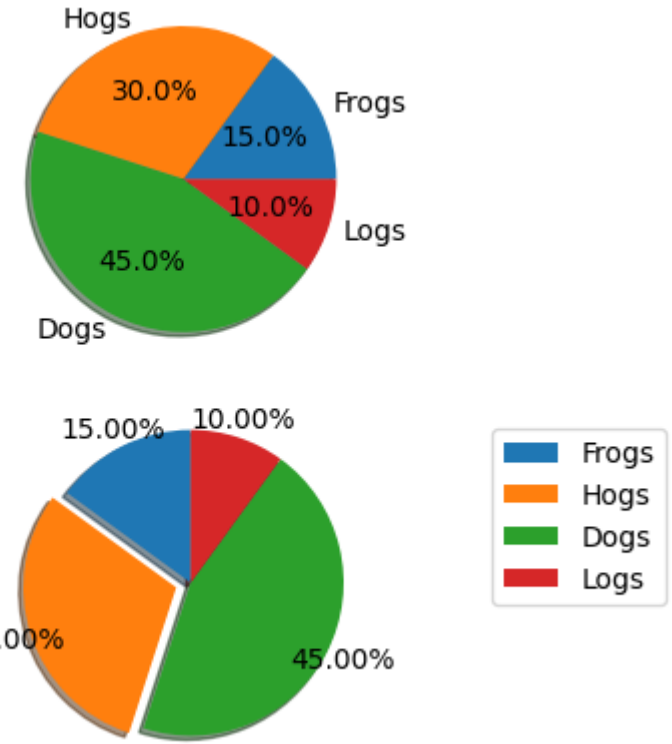
```
ax2.hist(x, histtype = 'barstacked', rwidth = 0.9)
ax2.set_title('stacked bar')

ax3.hist(x[:, 0], rwidth = 0.9)
ax3.set_title('different sample sizes')

fig.tight_layout()
plt.show()
```



Pie Chart

In [19]:
```
labels = 'Frogs', 'Hogs', 'Dogs', 'Logs'
sizes = [15, 30, 45, 10]
```

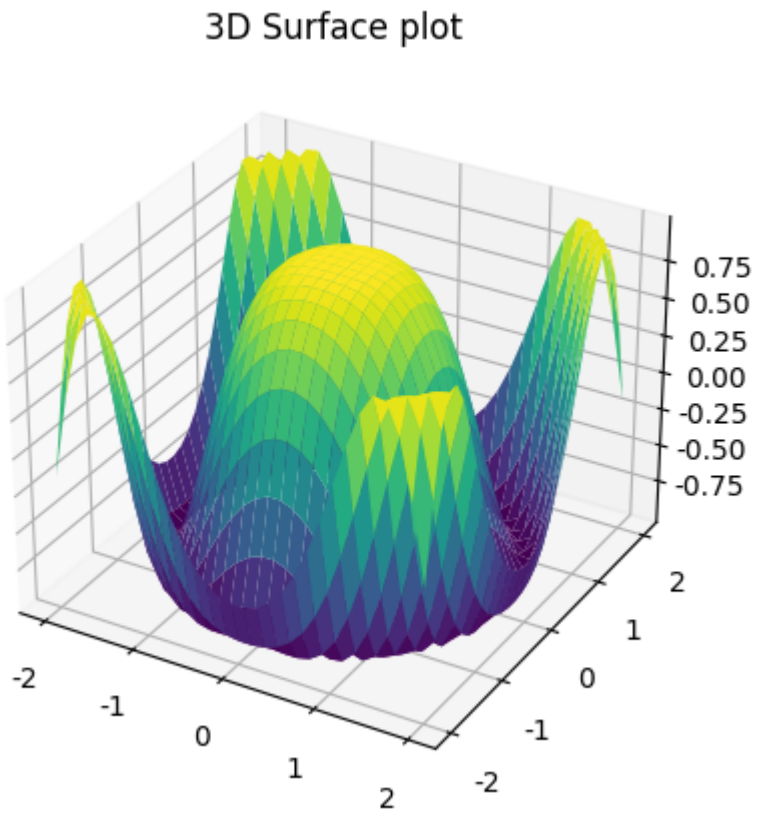
```
explode = (0, 0.1,0, 0) # only "explode" the 2nd slice (i.e. 'Hogs')
fig1, (ax1, ax2) = plt.subplots(2)
ax1.pie(sizes, labels = labels, autopct= '%1.1f%%', shadow = True)
ax1.axis('equal')
ax2.pie(sizes, autopct = '%1.2f%%', shadow = True , startangle = 90, explode = explode, pctdistance = 1.12)
ax2.axis('equal')
ax2.legend(labels = labels, loc = 'upper right')
plt.show()
```



In [20]:

```
from mpl_toolkits import mplot3d
import numpy as np
import matplotlib.pyplot as plt


# Calculate the inner product of two vectors
x = np.outer(np.linspace(- 2, 2, 30), np.ones(30))
# Calculate the transpose
y = x.copy().T
#Get the data for z using np.cosine function
```

```
z = np.cos(x ** 2 + y ** 2)

# Graph the 3D surface
fig = plt.figure()
ax = plt.axes(projection= '3d')
ax.plot_surface(x, y, z, cmap= 'viridis', edgecolor= 'none')
ax.set_title('3D Surface plot')
plt.show()
```



For more example, see https://matplotlib.org/stable/tutorials/toolkits/mplot3d.html

The best resource is always official website: https://matplotlib.org/stable/index.html

In [ ]: