Python

Zobrazování dat

VŠCHT

2019

Matplotlib

Modul matplotlib - některé funkcionality

- umožňuje generovat grafy nejrůznějších typů, 2D i 3D grafy
- umožňuje vykreslovat více grafů do jednoho okna
- možnost volit popisky os, barvu čar, název grafu, legendu,...
- export grafů do běžných typů souborů (jpeg, eps, tiff, bmp,...)
- umožnujě zobrazit obrázky uložené v jpeg, eps, tiff, bmp,...
- podporuje TeX
- podporuje zobrazení grafů v nejrůznějších typech souřadnic
- lze anotovat jednotlivé grafy
- pomocí widgetů lze implementovat do pygtk, Tk, Qt aplikací
- možnost vytvářet animace
- podporuje event handling (odchytávání událostí)

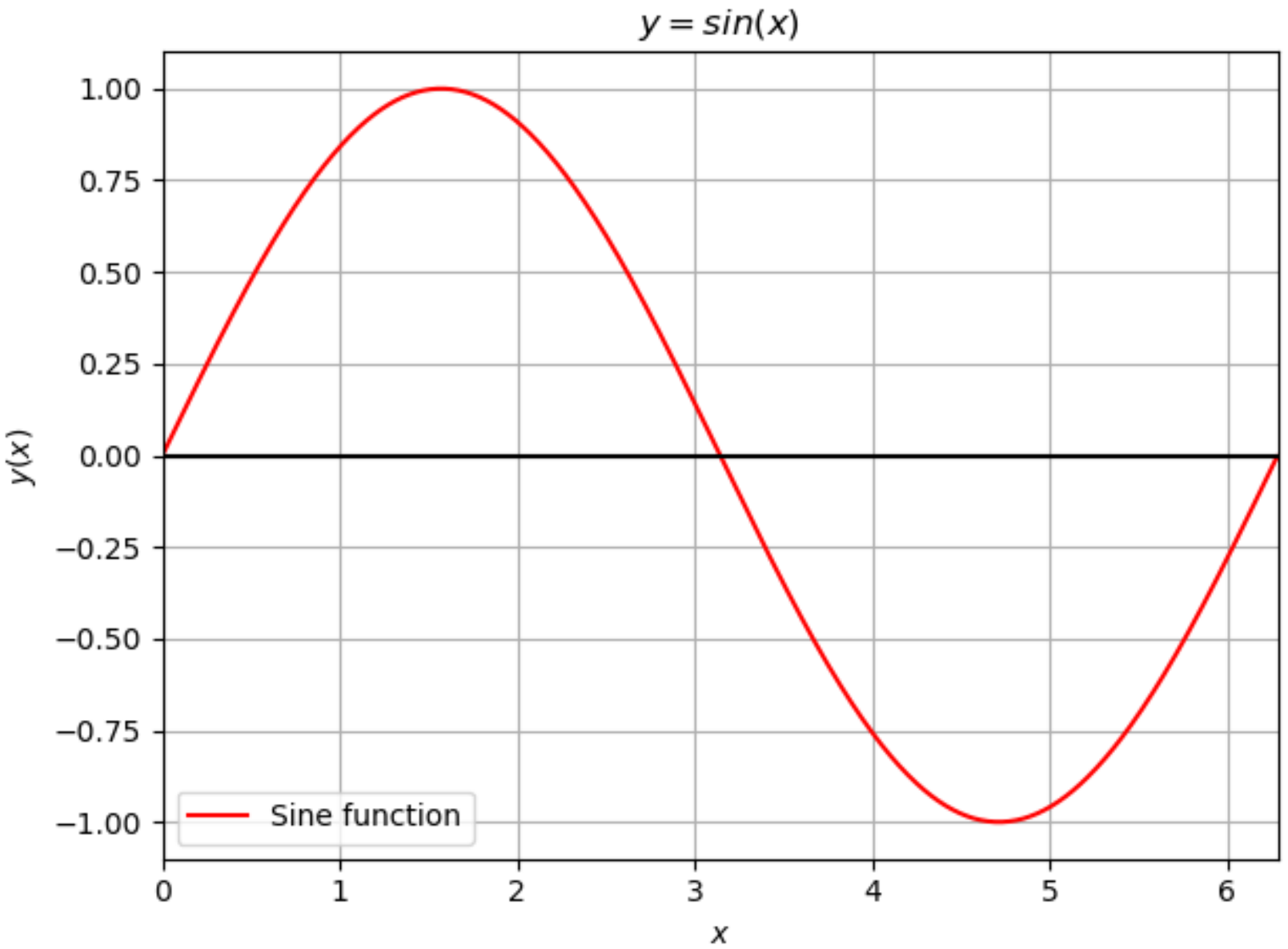
Graf funkce sinus

```
import matplotlib.pyplot as plt
import numpy as np

x = np.arange(0, 2 * np.pi, 0.01)
y = np.sin(x)

plt.plot(x, y, color="r") # plot values in y to against values in x
plt.xlim(x[0], x[-1]) # set x axis limits
plt.ylim(min(y) + 0.1 * min(y), max(y) + 0.1 * max(y)) # set y axis limits
plt.title("$y = sin(x)$") # graph title
plt.xlabel("$x$") # naming x axis with latex style font
plt.ylabel("$y(x)$") # naming y axis
plt.grid() # turn on grid
plt.tight_layout() # removes extra space around graph
plt.axhline(y=0, color='k') # make bold x axis at y=0
plt.legend(["Sine function"], loc=3) # add legend with desired text on desired position
plt.savefig("sine.png") # save figure to file
plt.show() # show window with graph
```

Graf funkce sinus - výsledek

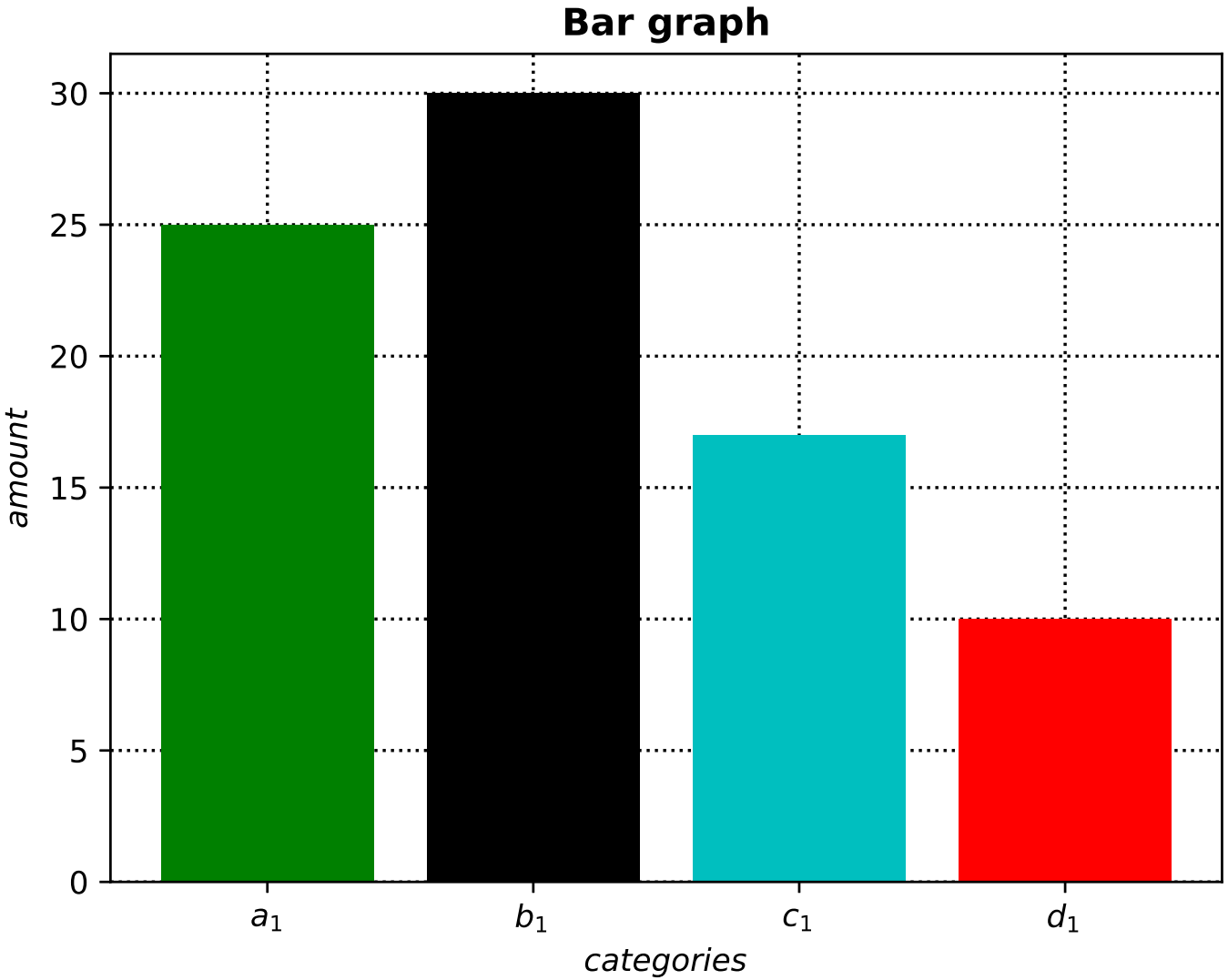


Sloupcový graf

```
import matplotlib.pyplot as plt

data = {"a": 25, "b": 30, "c": 17, "d": 10}  # set data to plot
plt.bar(list(data.keys()), data.values(), color=("g","k","c","r"),
        zorder=2, tick_label=list(data.keys()))
# plot bars with desired colors and bring them up (zorder)
plt.grid(zorder=1, linestyle=":", linewidth=1, color="k")
# brings dotted grid behid bars and set width and color of grid
plt.xticks(list(data.keys()), ["$a_1$", "$b_1$", "$c_1$", "$d_1$"])  # set x labels
plt.xlabel("$categories$")  # set description of x axis
plt.ylabel("$amount$")  # set description of y axis
plt.title("Bar graph", fontweight="bold")  # make bold title
plt.savefig('bar_graph.eps', format='eps', dpi=1000)  # save figure as svg with 1000 dpi
plt.show()  # show figure with graph
```

Sloupcový graf - výsledek

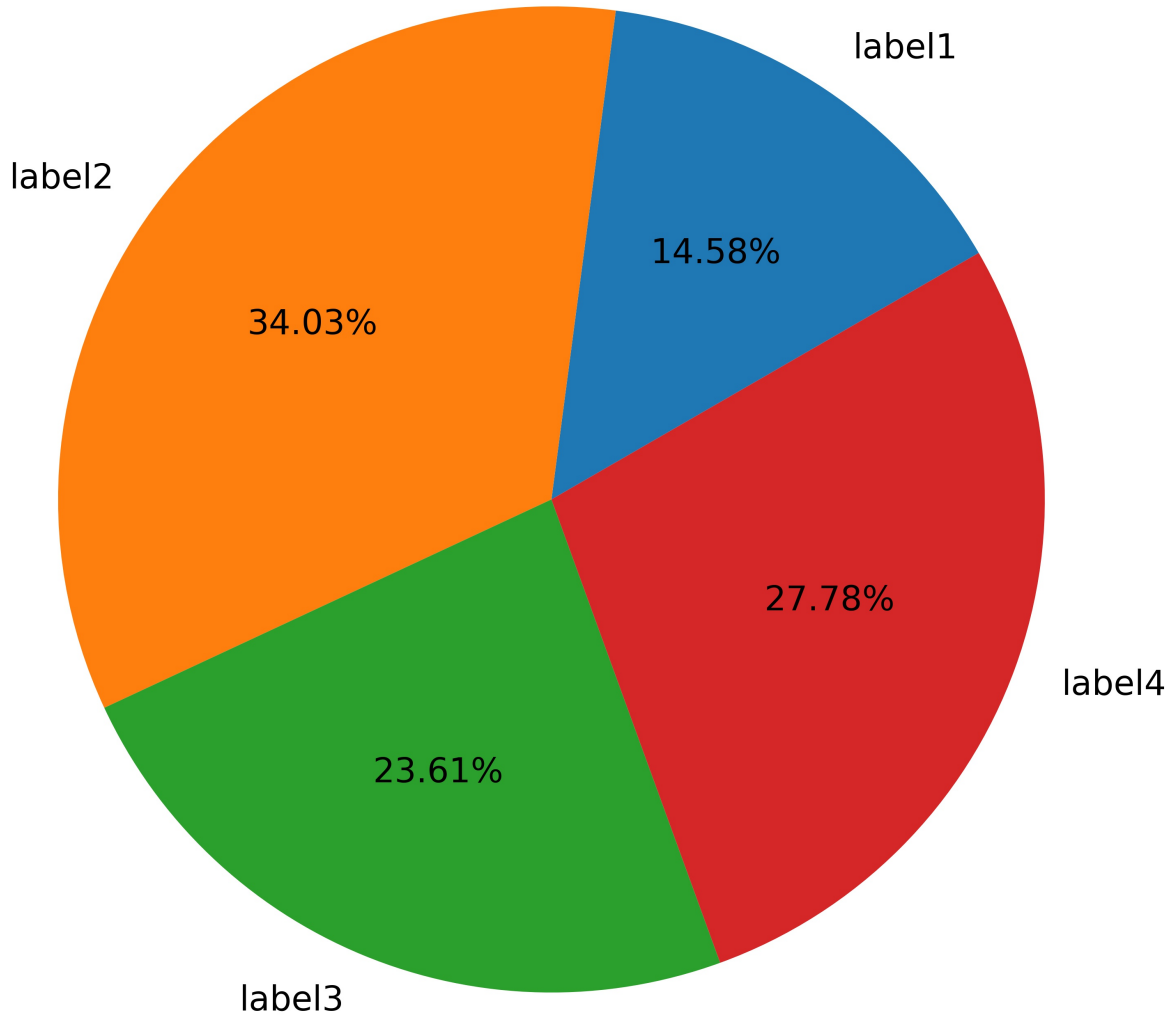


Koláčový graf

```
import matplotlib.pyplot as plt

labels = 'label1', 'label2', 'label3', 'label4'
sizes = [10.5, 24.5, 17, 20]
# prepare data
plt.gcf().canvas.set_window_title('Labels pie graph')
# set the window title
plt.pie(sizes, labels=labels, autopct='%1.2f%%',
        shadow=False, startangle=30)
# plot pie chart with percentage values (2 decimals shown), rotated for 30 degrees
plt.axis('equal')  # to make pie circular
plt.tight_layout()  # remove extra spaces
plt.savefig('pie_chart.jpg', dpi=600)  # safe graph as jpg with 600 dpi
plt.show()  # show window with graph
```

Koláčový graf - výsledek

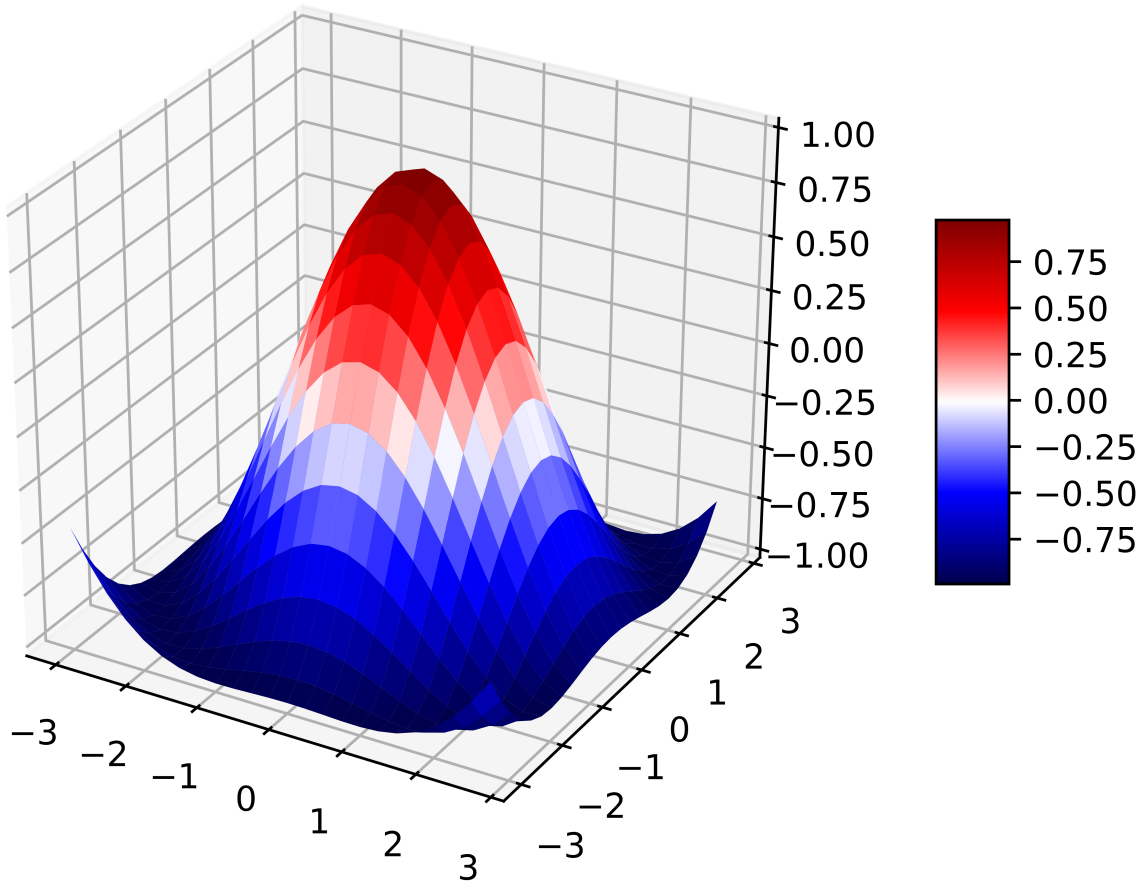


3D graf

```
from mpl_toolkits import mplot3d
import matplotlib.pyplot as plt
from matplotlib import cm
from matplotlib.ticker import LinearLocator, FormatStrFormatter
import numpy as np


fig = plt.figure()
ax = fig.gca(projection='3d') # set type of graph projection
X = np.arange(-3, 3, 0.25) # prepare x values
Y = np.arange(-3, 3, 0.25) # prepare y values
X, Y = np.meshgrid(X, Y) # creat grid made of x and y axis
Z = np.cos(np.sqrt(X**2 + Y**2)) # compute function values based on grid
surf = ax.plot_surface(X, Y, Z, cmap=cm.seismic, antialiased=True)
# plot surface, with specific color map
ax.set_zlim(-1.0, 1.0) # set z axis limits
fig.colorbar(surf, shrink=0.4, aspect=5)
# add colorbar right to graph with 0.4 size
fig.savefig("3d_graph.png", dpi=1000) # save graph as png with 1000 dpi
plt.show() # show window with graph
```

3D graf - výsledek



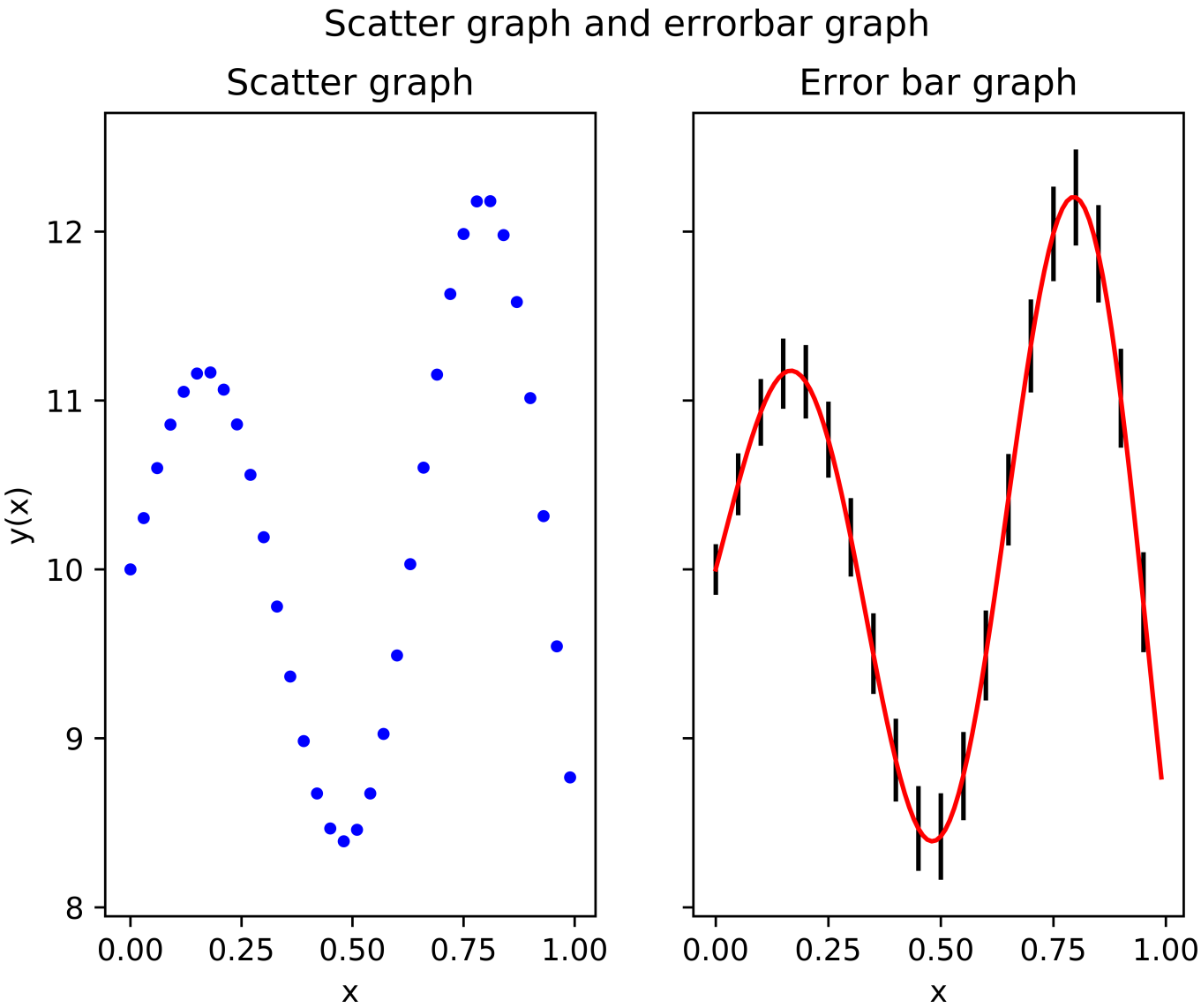
2 grafy v jednom obrázku - scatter plot, error graph

```
import numpy as np
import matplotlib.pyplot as plt


x = np.arange(0, 1.0, 0.01)  # prepare x
y = 10 + np.exp(x) * (np.sin(10 * x))  # prepare data to plot
y_error = 0.15 + 0.15 * np.sqrt(x)  # prepare errors

# create figure for 2 plots in row sharing y axis
fig, axs = plt.subplots(nrows=1, ncols=2, sharey=True)
ax = axs[0]  # select first subplot
ax.scatter(x[::3], y[::3], marker=".", color="b")
ax.set_xlabel("x")
ax.set_ylabel("y(x)")
# plot scatter graph with blue dots using every 3rd point of x and y
ax.set_title("Scatter graph")  # set subplot title
ax = axs[1]  # select second subplot
# plot errorbar graph with error bars only for every 5th sample
# graph has red color, errorbars have black color
ax.errorbar(x, y, yerr=y_error, errorevery=5, ecolor="k", color='r')
ax.set_title("Error bar graph")  # set tile of second subplot
ax.set_xlabel("x")
# set title of subplots
fig.suptitle("Scatter graph and errorbar graph")
plt.savefig("2graphs.png", dpi=1000)  # save figure as png with 1000 dpi
plt.show()  # show window with graphs
```

2 grafy v jednom obrázku - scatter plot, error graph



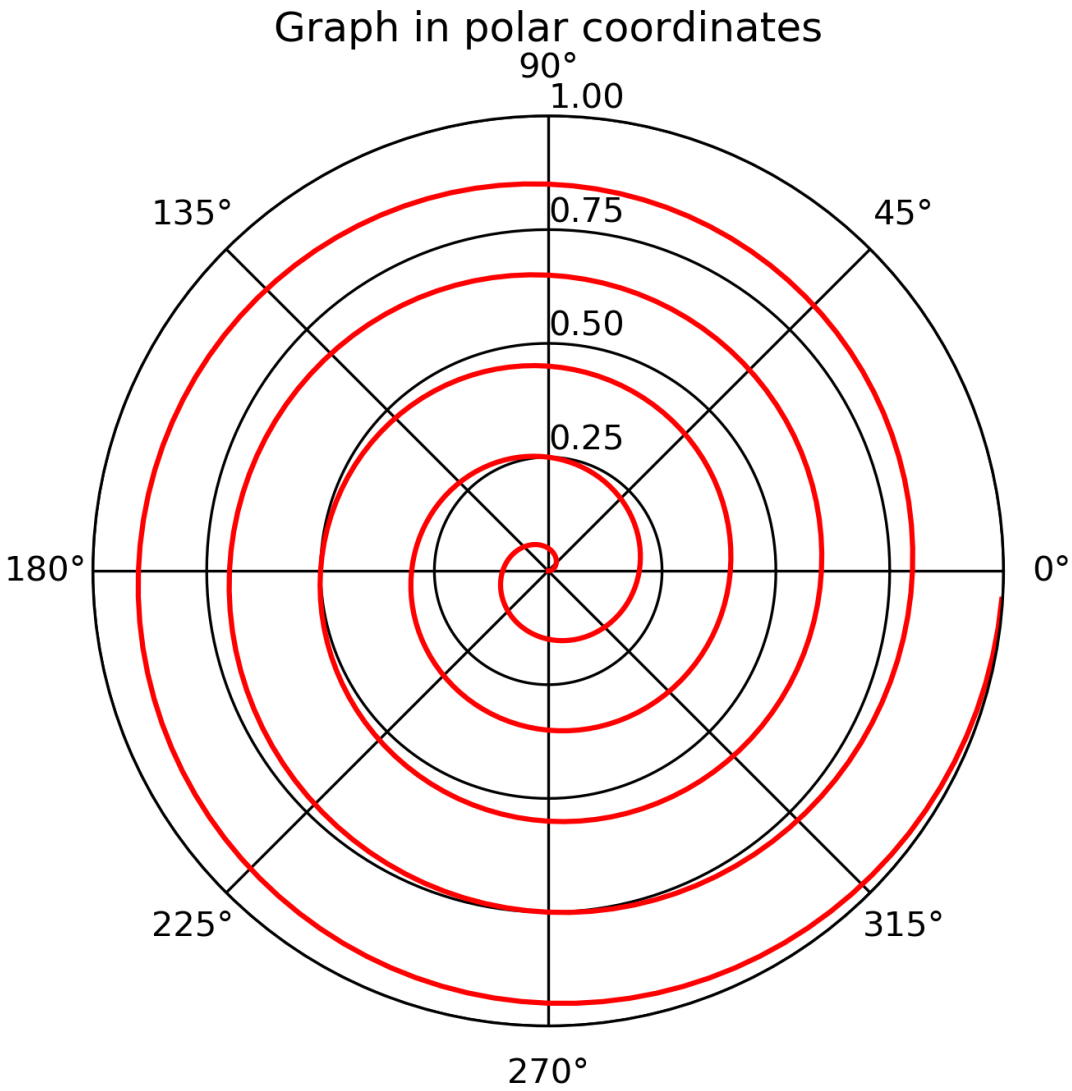
Graf v polárních souřadnicích

```
import numpy as np
import matplotlib.pyplot as plt


r = np.arange(0, 1, 0.002)  # prepare radius
theta = 10 * np.pi * r  # prepare angle

ax = plt.subplot(111, projection='polar')  #set 1 subplot with polar projection
ax.plot(theta, r, color="r")  # plot red graph
ax.set_rmax(1)  # set maximum of r axis
ax.set_rticks([0.25, 0.5, 0.75, 1])  # set r ticks
ax.set_rlabel_position(90)  # rotate l labels to +90 degrees from default location
ax.grid(True, color="k")  # make grid with black color
ax.set_title("Graph in polar coordinates", va='bottom')  # set graph title
plt.savefig("polar.png", dpi=300)  # save figure as png with 300dpi
plt.show()  # show window with graph
```

Graf v polárních souřadnicích - výsledek



Histogram

```
import matplotlib.pyplot as plt
import numpy as np


np.random.seed(111)  # set random seed
points = 2000000  # set number of points
bins = 50  # set number of bins
x = np.random.randn(points)  # prepare data sampled from normal distribution
plt.hist(x, bins=bins, density=False)  # plot histogram, y has count scale
plt.title("Histogram")  # set title
plt.savefig("histogram.png", dpi=300)
plt.show()  # show window with histogram
```

Histogram - výsledek