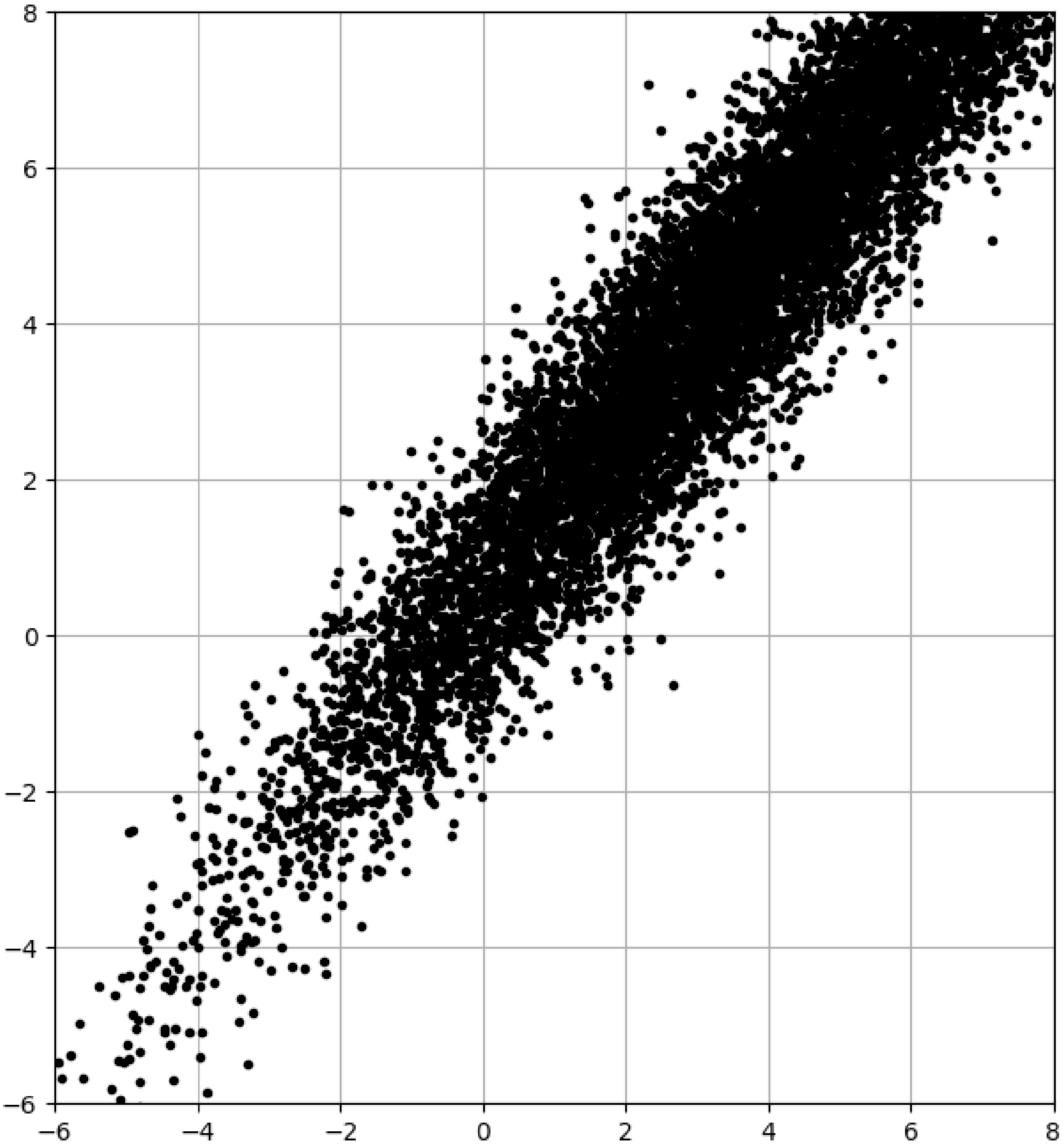
```python
# MK Lab4 - Rafał Klinowski
import matplotlib.pyplot as plt
import numpy as np
plt.rcParams['figure.figsize'] = [16, 8]

xC = np.array([2, 1])      # Center of data (mean)
sig = np.array([2, 0.5])   # Principal axes

theta = np.pi/3            # Rotate cloud by pi/3

R = np.array([[np.cos(theta), -np.sin(theta)],     # Rotation matrix
              [np.sin(theta), np.cos(theta)]])

# Załadowanie punktów z pliku
import pandas as pd
data = pd.read_csv('6.csv', sep=",", header=None)
data = data.to_numpy()

print(data)
```

```
[[ 4.88540487  1.08728543  1.97071503 ... -1.1085726   0.27366883
  -0.07498368]
 [-0.93198105  4.48977962  0.25308115 ...  0.81249114  0.13407353
   1.68978014]]
```

```python
# Rysowanie wczytanych punktów
nPoints = data.shape[1]
print("nPoints = ", nPoints)

X = R @ np.diag(sig) @ data + np.diag(xC) @ np.ones((2,nPoints))

fig = plt.figure()
ax1 = fig.add_subplot(121)
ax1.plot(X[0,:],X[1,:], '.', color='k')
ax1.grid()
plt.xlim((-6, 8))
plt.ylim((-6,8))

plt.show()
```

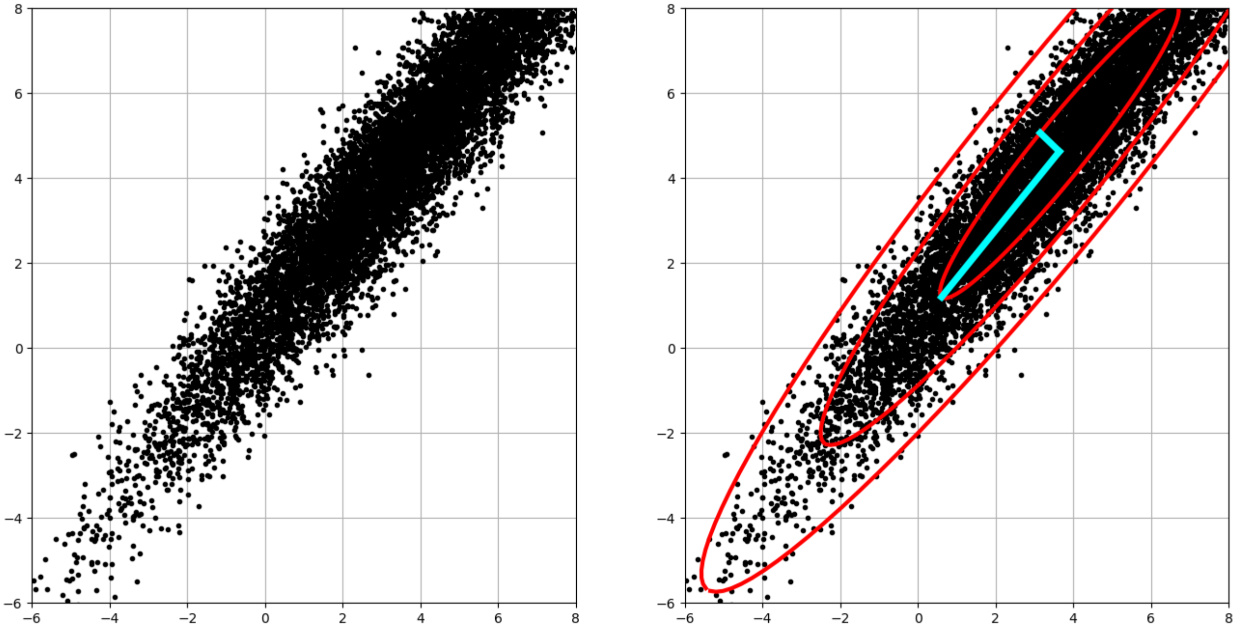
```
nPoints =  10000
```



```python
# Przygotowanie do SVD
Xavg = np.mean(X,axis=1)
B = X - np.tile(Xavg,(nPoints,1)).T

# Przeprowadzenie SVD
U, S, VT = np.linalg.svd(B/np.sqrt(nPoints),full_matrices=0)

print("---U---")
print(U)
print("---S---")
print(S)
```

```
# Wykresy
fig = plt.figure()
ax1 = fig.add_subplot(121)
ax1.plot(X[0,:],X[1,:], '.', color='k')
ax1.grid()
plt.xlim((-6, 8))
plt.ylim((-6,8))

ax2 = fig.add_subplot(122)
ax2.plot(X[0,:],X[1,:], '.', color='k')   # Plot data to overlay PCA
ax2.grid()
plt.xlim((-6, 8))
plt.ylim((-6,8))

# Przygotowanie PCA
theta = 2 * np.pi * np.arange(0,1,0.01)

# Pierwszy standardowy rozkład
Xstd = U @ np.diag(S) @ np.array([np.cos(theta),np.sin(theta)])

ax2.plot(Xavg[0] + Xstd[0,:], Xavg[1] +
Xstd[1,:],'-',color='r',linewidth=3)
ax2.plot(Xavg[0] + 2*Xstd[0,:], Xavg[1] +
2*Xstd[1,:],'-',color='r',linewidth=3)
ax2.plot(Xavg[0] + 3*Xstd[0,:], Xavg[1] +
3*Xstd[1,:],'-',color='r',linewidth=3)

# Plot principal components U[:,0]S[0] and U[:,1]S[1]
ax2.plot(np.array([Xavg[0], Xavg[0]+U[0,0]*S[0]]),
         np.array([Xavg[1],
Xavg[1]+U[1,0]*S[0]]),'-',color='cyan',linewidth=5)
ax2.plot(np.array([Xavg[0], Xavg[0]+U[0,1]*S[1]]),
         np.array([Xavg[1],
Xavg[1]+U[1,1]*S[1]]),'-',color='cyan',linewidth=5)

plt.show()
```

```
---U---
[[-0.66248388 -0.74907617]
 [-0.74907617  0.66248388]]
---S---
[4.57597841 0.65799426]
```


8
6
4
2
0
−2
−4
−6
−6 −4 −2 0 2 4 6 8
8
6
4
2
0
−2
−4
−6
−6 −4 −2 0 2 4 6 8