Principal Component Analysis

import library

In [ ]:

```
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.colors as colors
from matplotlib import cm
import time
import util
```

load data

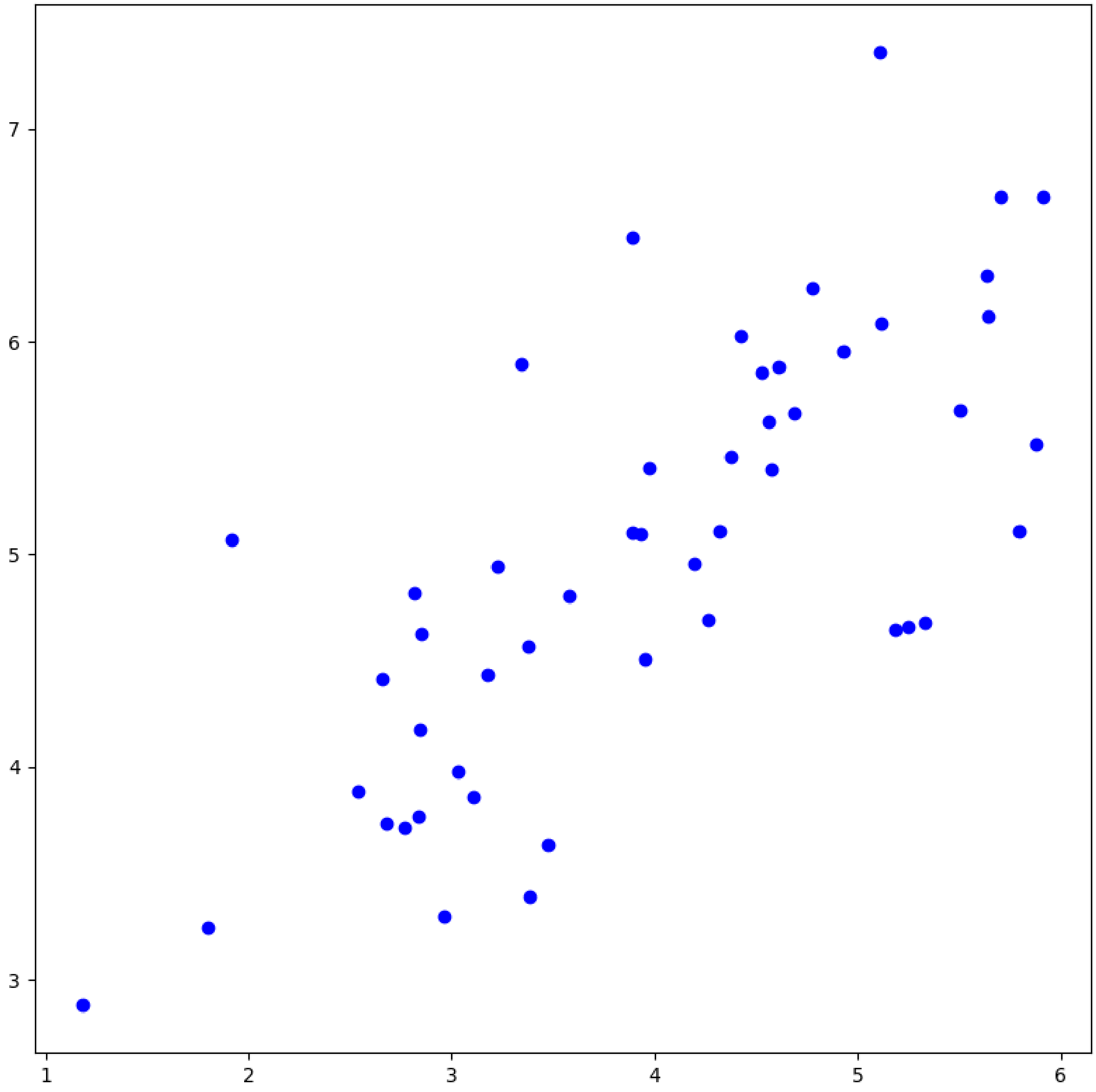
In [ ]:

```
fname_data  = '12_data.csv'
data        = np.genfromtxt(fname_data, delimiter=',')
num_data    = np.size(data, 0)
num_feature = np.size(data, 1)
data        = np.reshape(data, (num_data, num_feature))
```

plot the data

In [ ]:

```
plt.figure(figsize=(8,8))
plt.scatter(data[:,0], data[:,1], color='blue')
plt.tight_layout()
plt.show()
```



Normalization (Z-scoring)

In [ ]:

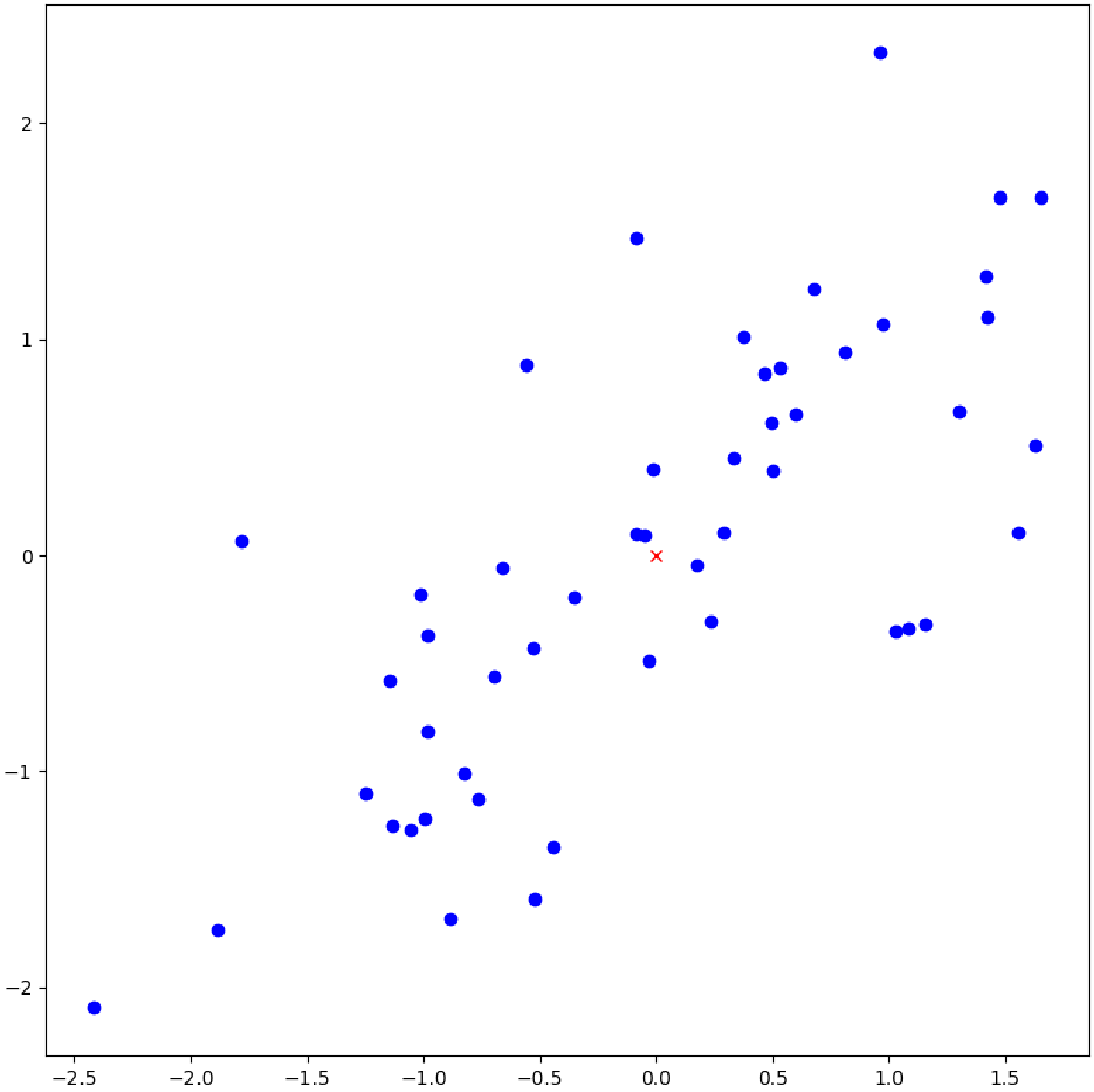
```
data_norm = util.normalize(data)
```

In [ ]:

```
def plot_01():
    plt.figure(figsize=(8,8))
    plt.scatter(data_norm[:,0], data_norm[:,1], color='blue')
    plt.plot(0, 0, 'x', color='red')
    plt.tight_layout()
    plt.show()
```

In [ ]:

```
plot_01()
```



In [ ]:
```
def plot_02():
    print('mean = (%.2f, %.2f)' % (data_norm.mean(axis=0)[0], data_norm.mean(axis=0)
    print('std = (%.2f, %.2f)' % (data_norm.std(axis=0)[0], data_norm.std(axis=0)[1]
```

In [ ]:
```
plot_02()
```

```
mean = (0.00, 0.00)
std = (1.00, 1.00)
```

compute principal components

`np.matmul` , `np.transpose` , `np.linalg.eig` , `np.argsort` , `plt.quiver` , `plt.scatter`

In [ ]:
```
pc1 = util.get_principal_component_first(data_norm)
```

In [ ]:
```
pc2 = util.get_principal_component_second(data_norm)
```

In [ ]:
```
def plot_03():
    print('first principal component = (%.2f, %.2f)' % (pc1[0], pc1[1]))
    print('second principal component = (%.2f, %.2f)' % (pc2[0], pc2[1]))
```

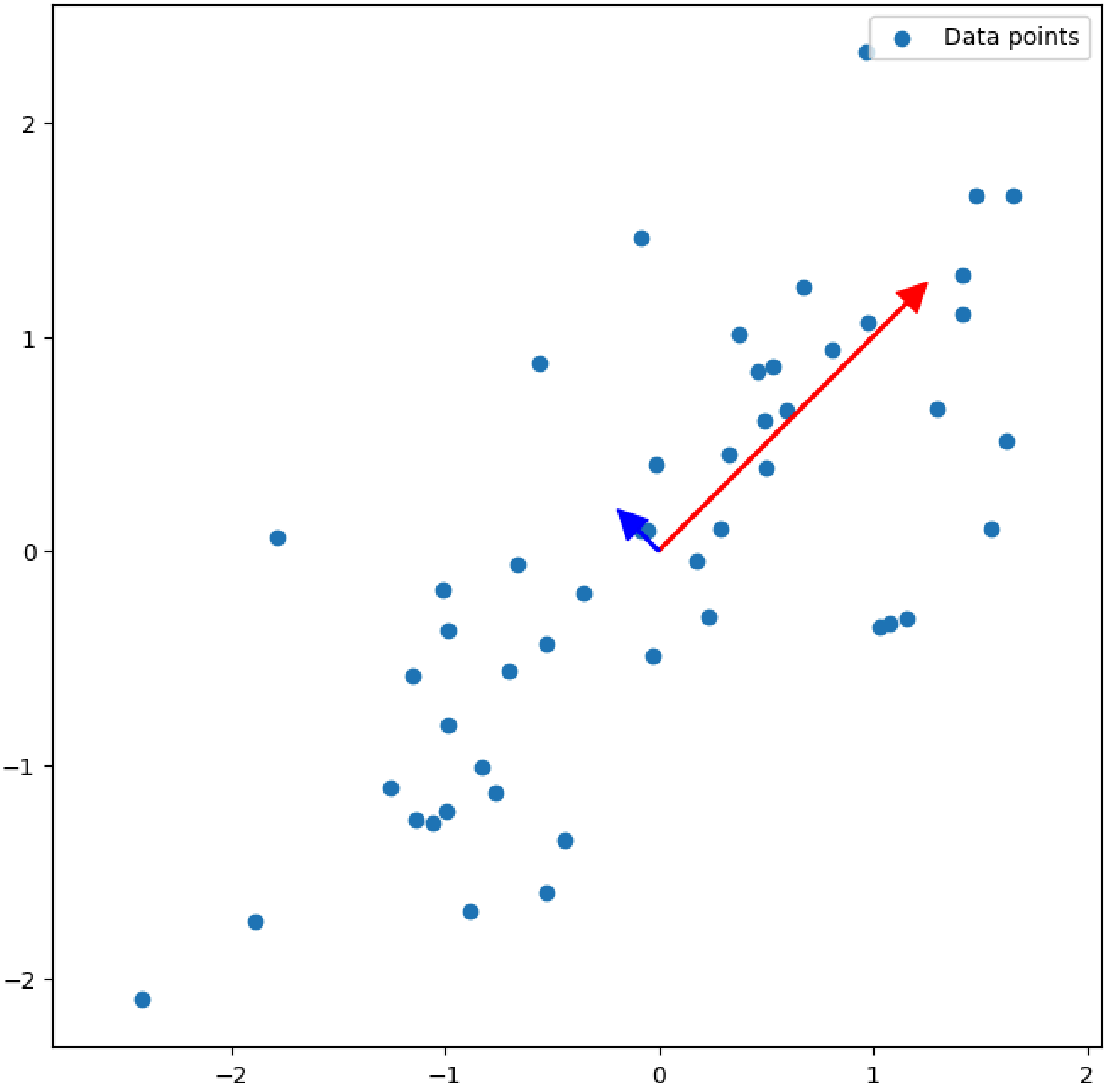
plot the principal components

```
In [ ]: plot_03()
```

```
first principal component = (0.71, 0.71)
second principal component = (-0.71, 0.71)
```

```
In [ ]: def plot_04():
            util.plot_principal_component(data_norm, pc1, pc2)
```

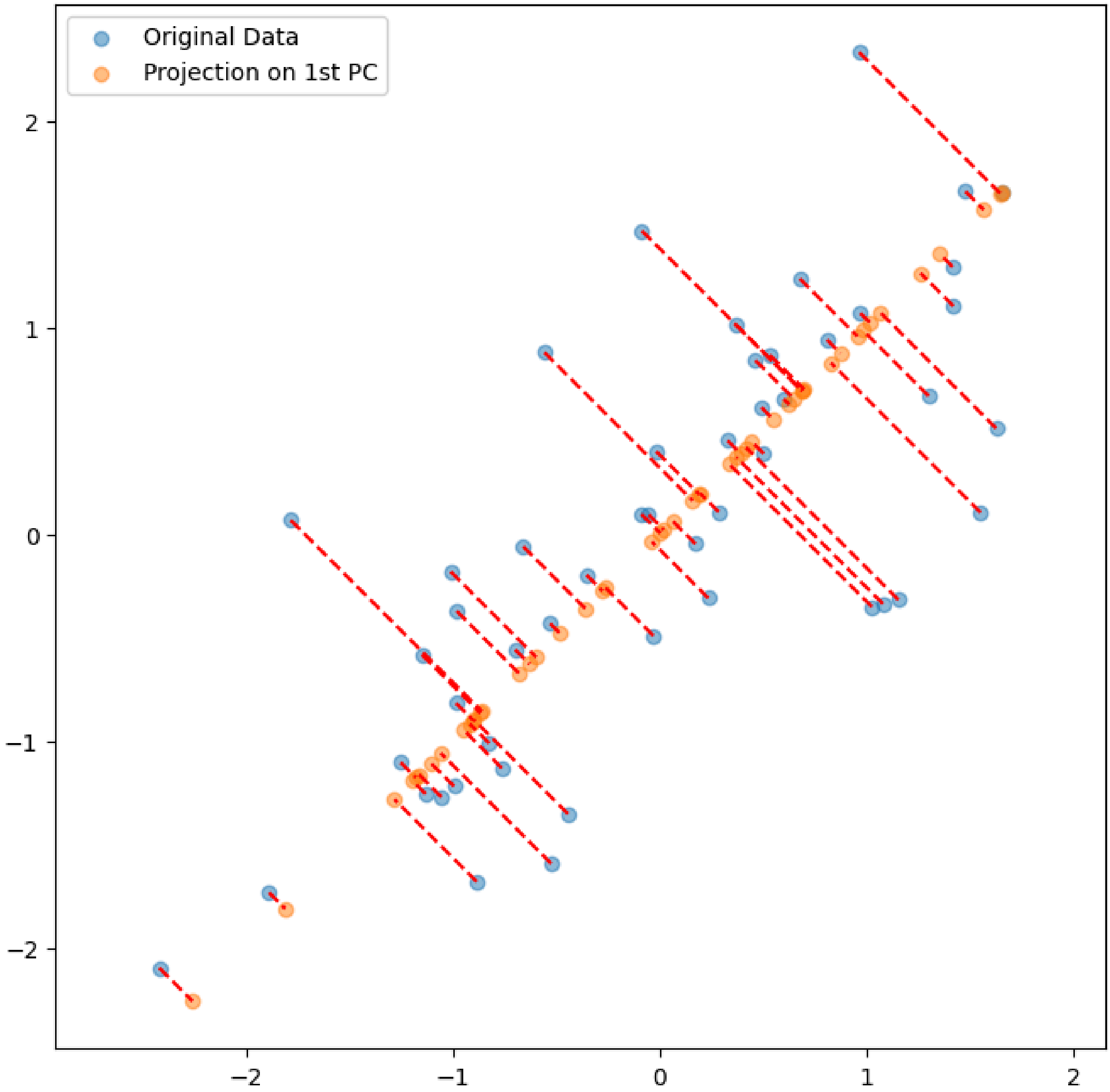
```
In [ ]: plot_04()
```



plot the projection of the data on the first principal components

```
In [ ]: def plot_05():
            util.plot_projection_principal_component(data_norm, pc1)
```

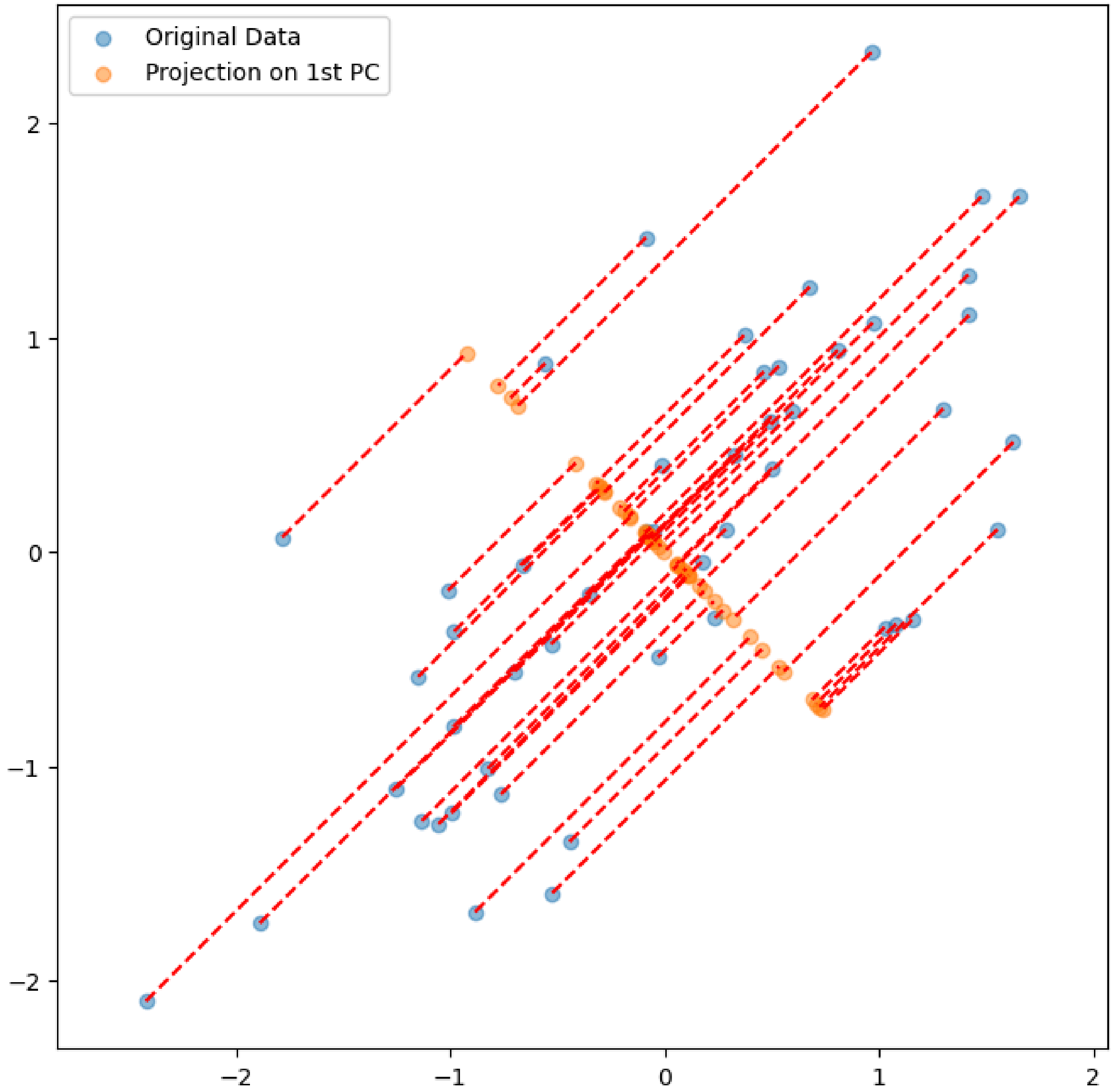
```
In [ ]: plot_05()
```



plot the projection of the data on the secpmd principal components

```
In [ ]: def plot_06():
            util.plot_projection_principal_component(data_norm, pc2)
```

```
In [ ]: plot_06()
```

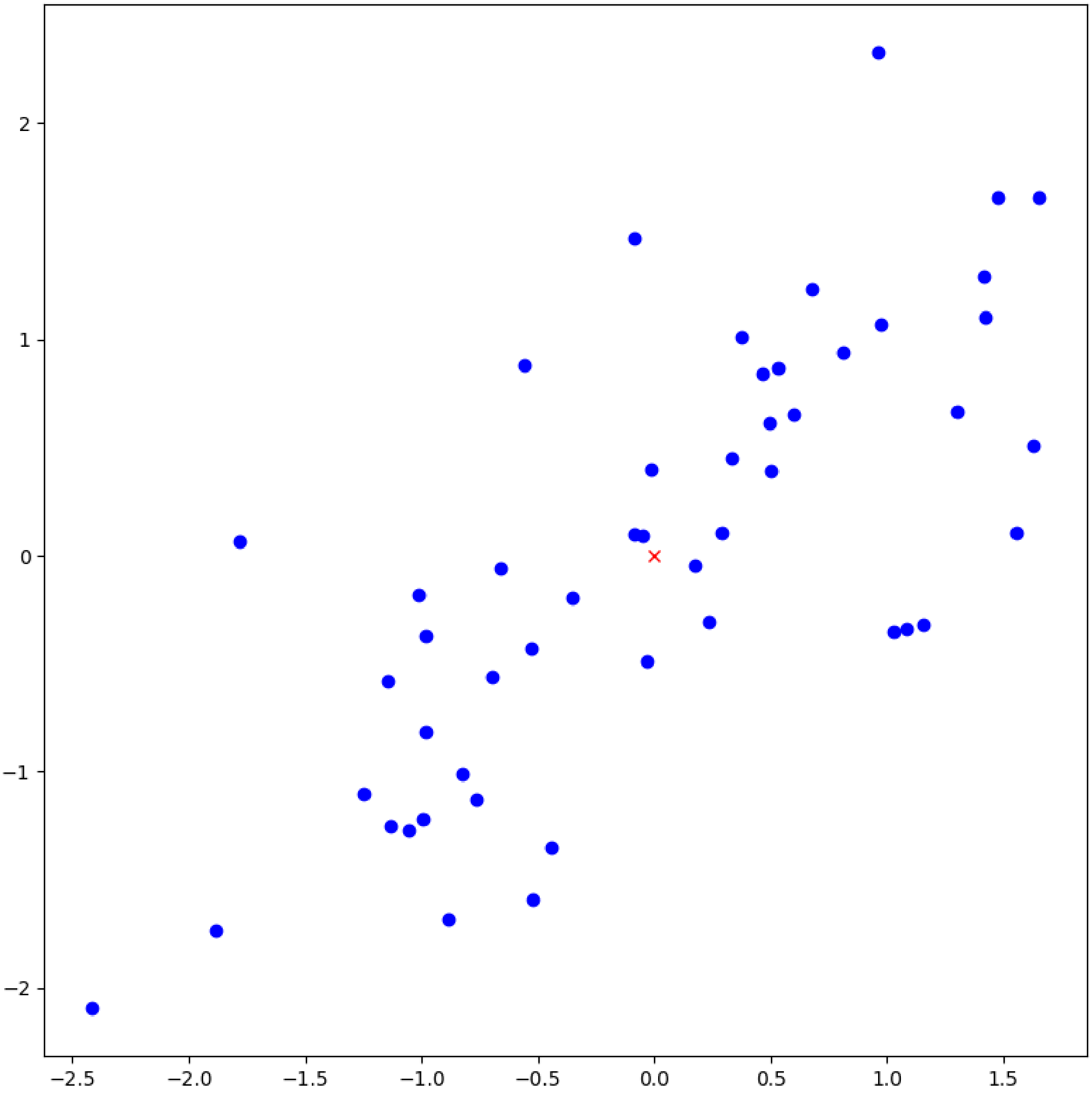


results

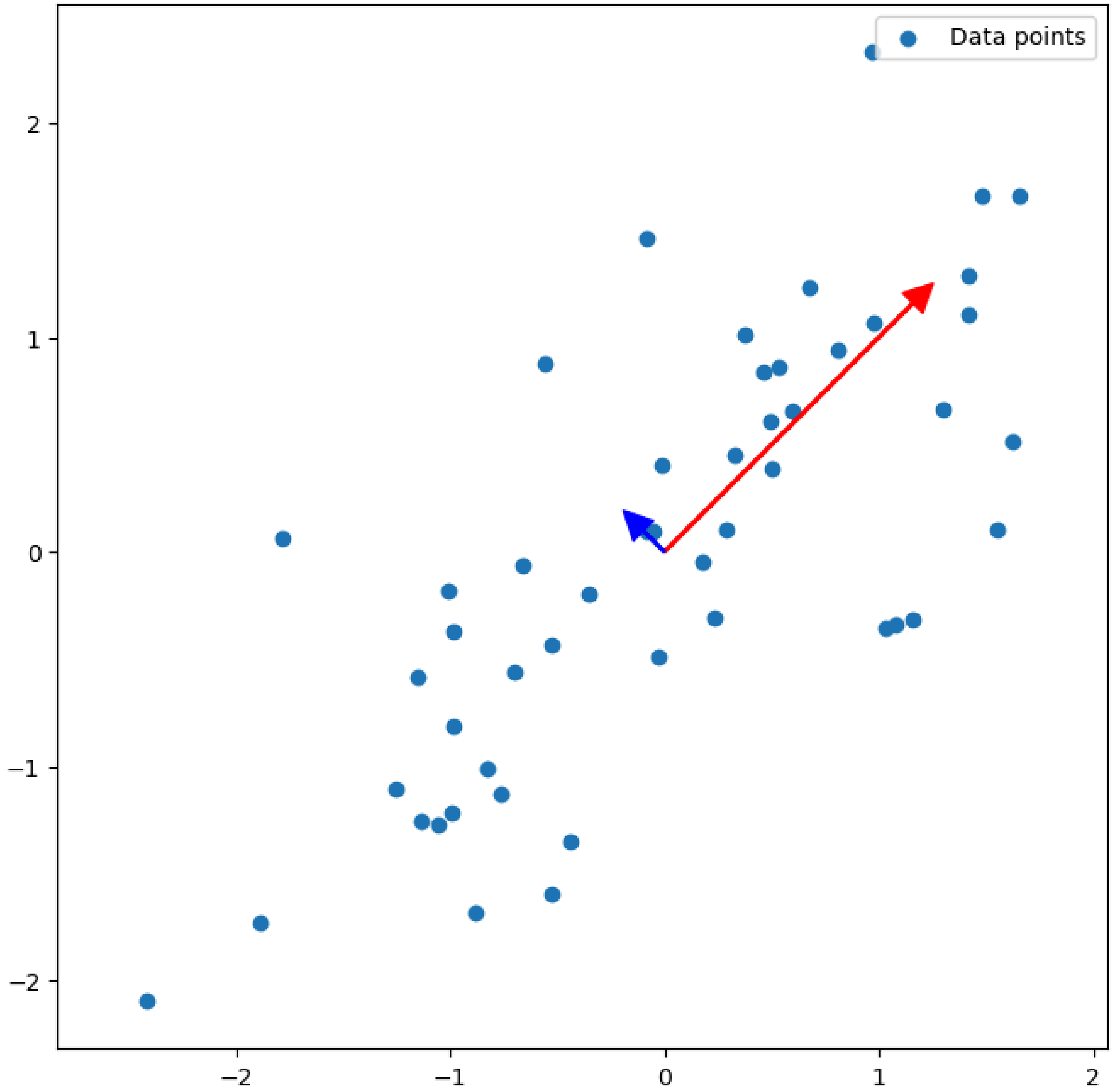
In [ ]:

```
number_result = 6

for i in range(number_result):
    title           = '# RESULT # {:02d}'.format(i+1)
    name_function   = 'plot_{:02d}()'.format(i+1)

    print('')
    print('##########################################################################
    print(title)
    print('##########################################################################
    print('')

    eval(name_function)
```

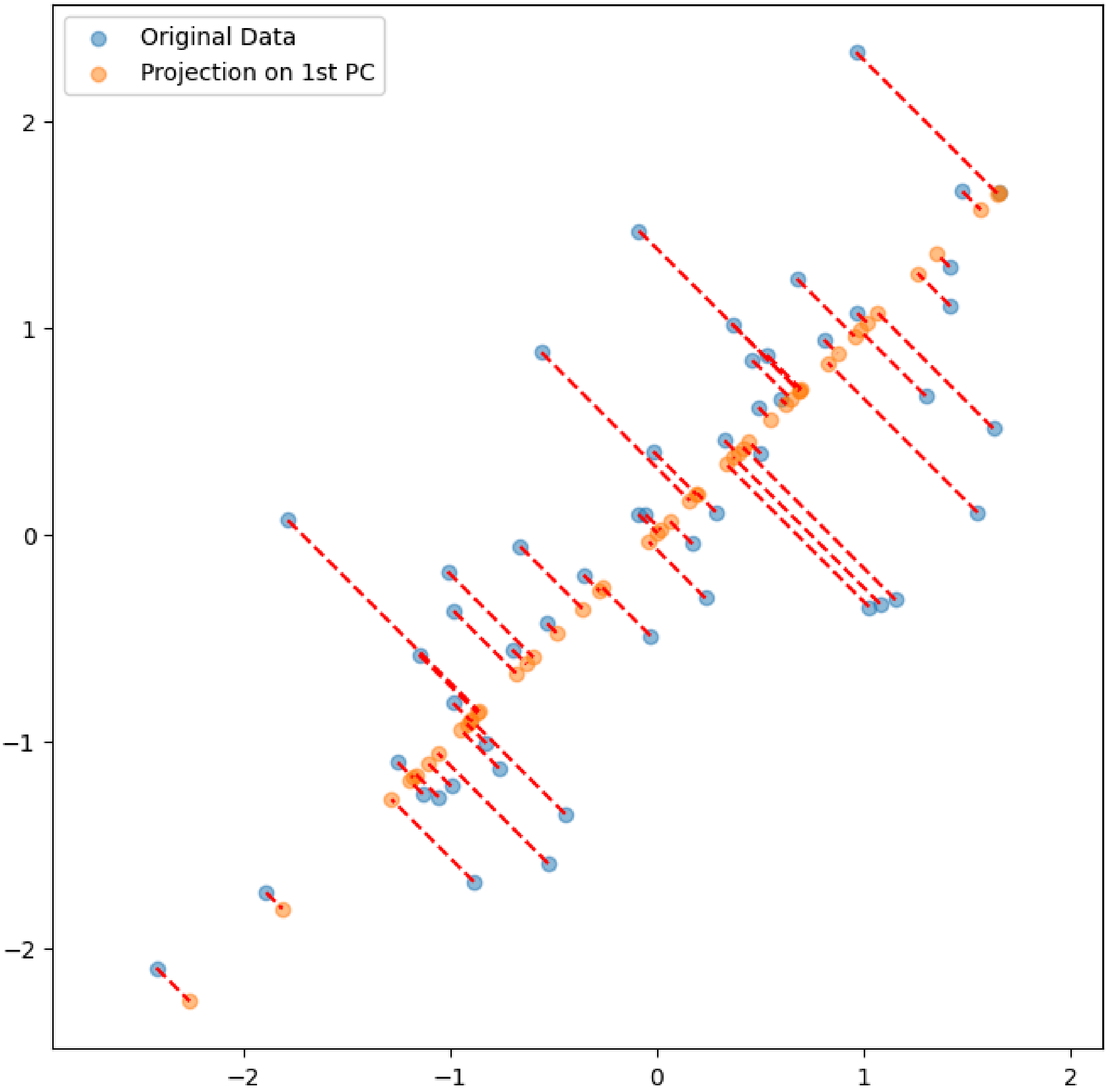
```
###############################################################################
# RESULT # 01
###############################################################################
```



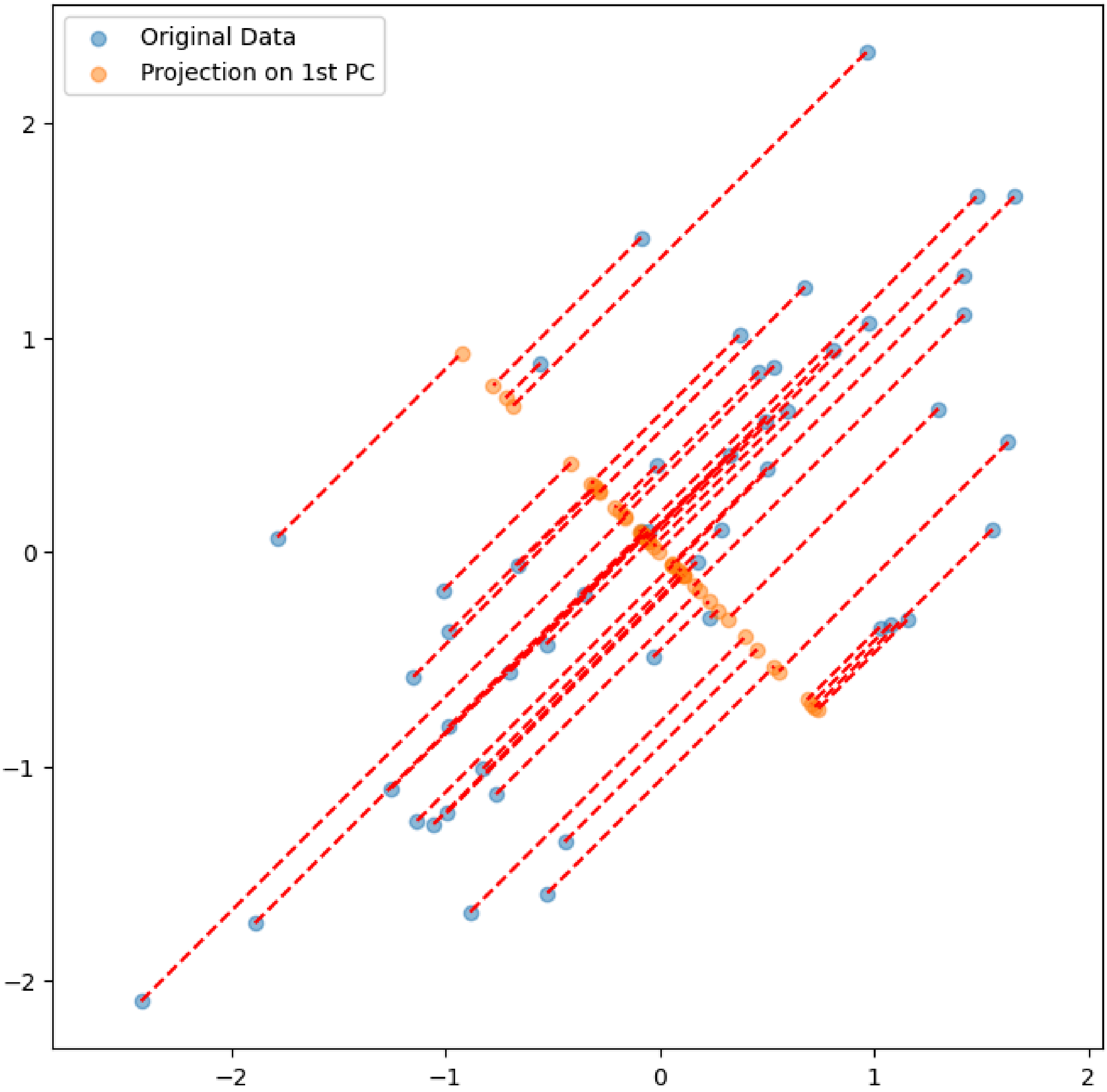
```
###############################################################################
# RESULT # 02
###############################################################################

mean = (0.00, 0.00)
std = (1.00, 1.00)

###############################################################################
# RESULT # 03
###############################################################################

first principal component = (0.71, 0.71)
second principal component = (-0.71, 0.71)

###############################################################################
# RESULT # 04
###############################################################################
```



```
################################################################################
# RESULT # 05
################################################################################
```



```
################################################################################
# RESULT # 06
################################################################################
```



```
In [ ]:
```