Principal components transform

February 27, 2022

[ ]:
```
import numpy as np
import matplotlib.pyplot as plt
import random
from sklearn.decomposition import PCA
```

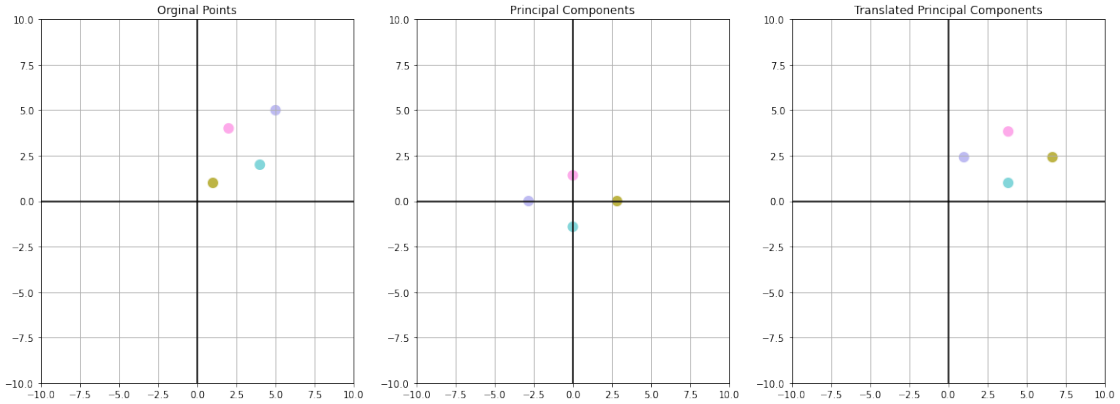
[ ]:
```
def random_colors(n):
    colors = []
    for i in range(n):
        colors.append("#%06x" % random.randint(0, 0xFFFFFF))
    return colors
```

[ ]:
```
def apply_PCA(Points):

    # set a unique color for each point
    colors = random_colors(len(Points))

    # PCA model
    pca = PCA(n_components = 2)
    principalComponents = pca.fit_transform(Points)

    # translation
    principalComponents_translated = np.zeros_like(principalComponents)
    min_x = principalComponents[:,0].min()
    principalComponents_translated[:,0] = principalComponents[:,0] - min_x + 1

    min_y = principalComponents[:,1].min()
    principalComponents_translated[:,1] = principalComponents[:,1] - min_y + 1


    # ploting

    fig, axs = plt.subplots(1, 3,figsize=(20,7))

    axs[0].scatter(Points[:,0],Points[:,1],marker='o',linewidths=5,color=colors)
    axs[0].grid()
    axs[0].axis(xmin=-10,xmax=10,ymin=-10, ymax=10)
    axs[0].hlines(y=0,xmin=-10,xmax=10,colors='k')
    axs[0].vlines(x=0,ymin=-10,ymax=10,colors='k')
    axs[0].set_title("Orginal Points")

    axs[1].scatter(principalComponents[:,0],
    principalComponents[:,1],marker='o',linewidths=5,color=colors)
    axs[1].grid()
```

```
    axs[1].axis(xmin=-10,xmax=10,ymin=-10, ymax=10)
    axs[1].hlines(y=0,xmin=-10,xmax=10,colors='k')
    axs[1].vlines(x=0,ymin=-10,ymax=10,colors='k')
    axs[1].set_title("Principal Components")

    axs[2].scatter(principalComponents_translated[:,0],
    principalComponents_translated[:,1],marker='o',linewidths=5,color=colors)
    axs[2].grid()
    axs[2].axis(xmin=-10,xmax=10,ymin=-10, ymax=10)
    axs[2].hlines(y=0,xmin=-10,xmax=10,colors='k')
    axs[2].vlines(x=0,ymin=-10,ymax=10,colors='k')
    axs[2].set_title("Translated Principal Components")
```

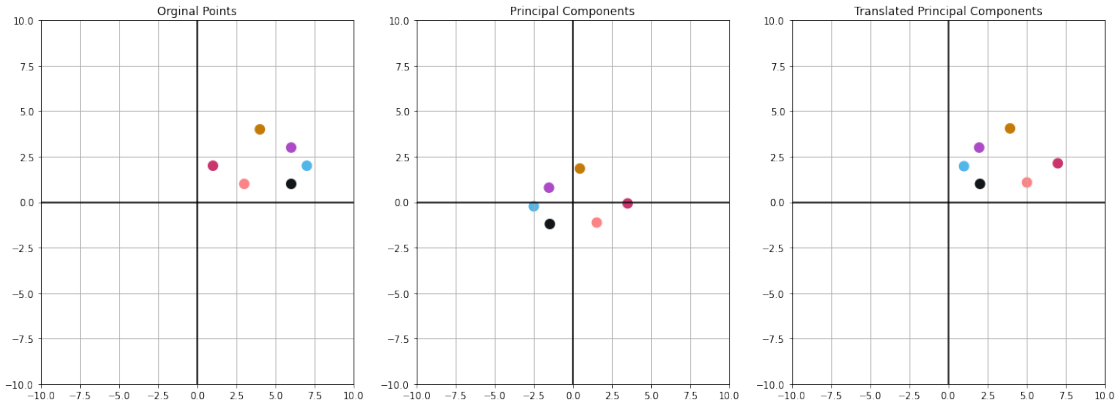
[ ]:
```
Points1 = np.array([[1,1],[2,4],[4,2],[5,5]])
```

[ ]:
```
apply_PCA(Points1)
```



[ ]:
```
Points2 = np.array([[1,2],[3,1],[4,4],[6,1],[6,3],[7,2]])
```

[ ]:
```
apply_PCA(Points2)
```



[ ]:
```
Points3 = np.random.randint(-4,4,(6,2))
```

[ ]:
```
apply_PCA(Points3)
```