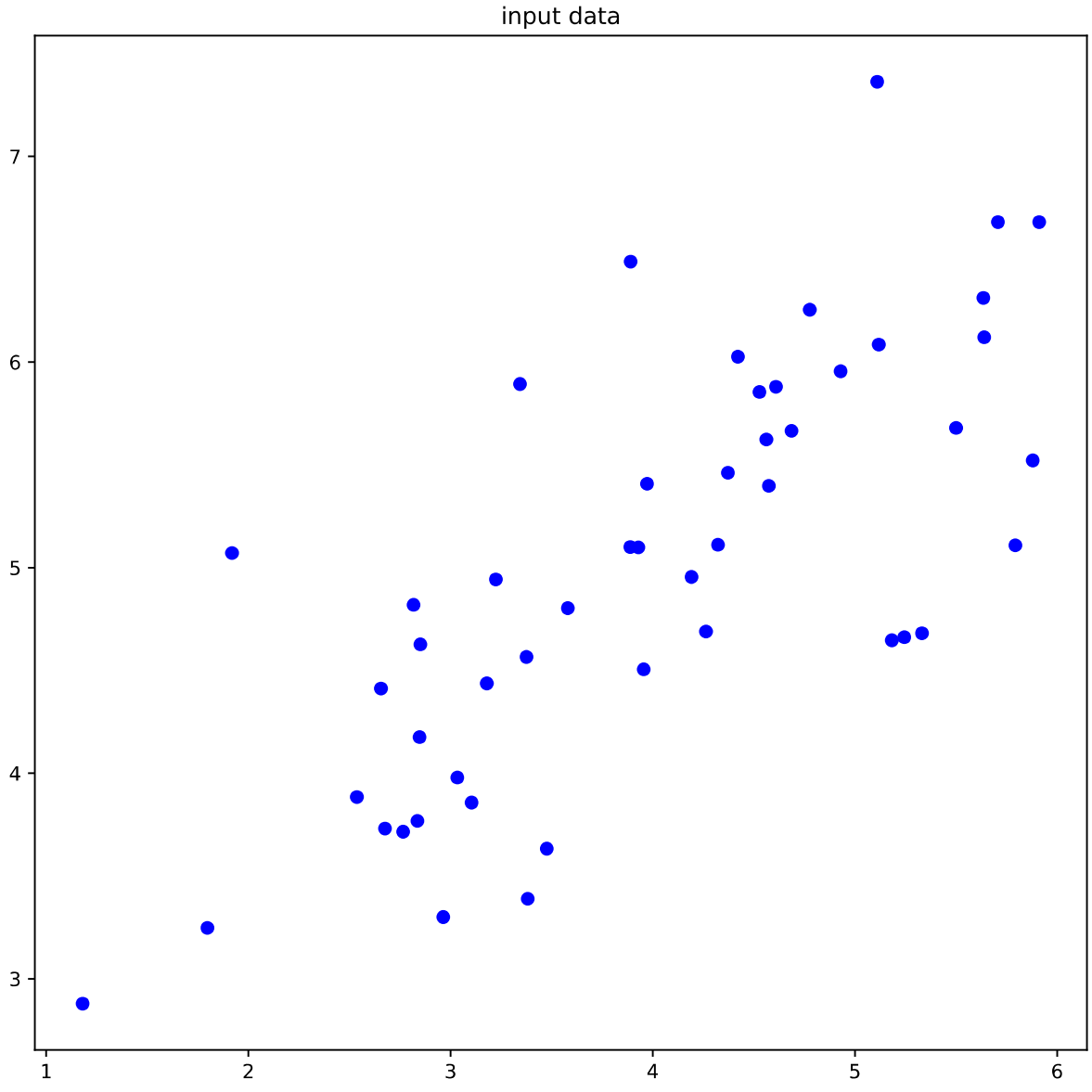
results

1. plot the input data after the normalization using Z-scoring in blue

In [ ]:

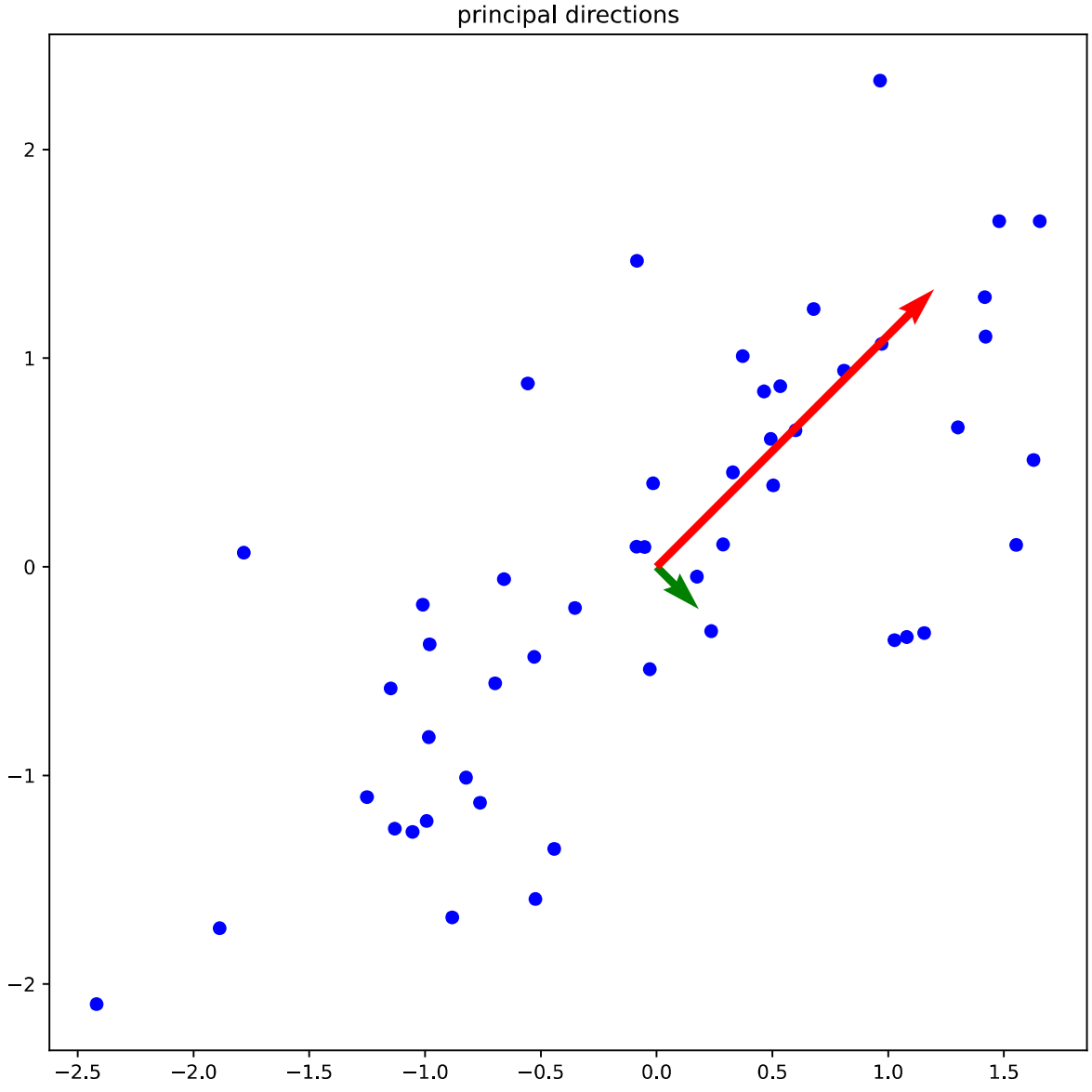
```
plt.figure(figsize=(8,8))
plt.title('input data')
# ================================================
# fill up the blank
#
plt.scatter(x,y,color='blue')
#
# ================================================
plt.tight_layout()
plt.show()
```



2. plot the first principal component in red and the second

principal components in green on the normalized data in blue

In [ ]:
```
plt.figure(figsize=(8,8))
plt.title('principal directions')
# ================================================
# fill up the blank
#
plt.scatter(zx, zy, color='blue')
S, U = compute_eigen(compute_covariance(feature))
origin = np.array([[0, 0],[0, 0]])
plt.quiver(*origin, S[:,0], S[:,1], color=['green','red'], scale = U/5)
#
# ================================================
plt.tight_layout()
plt.show()
```

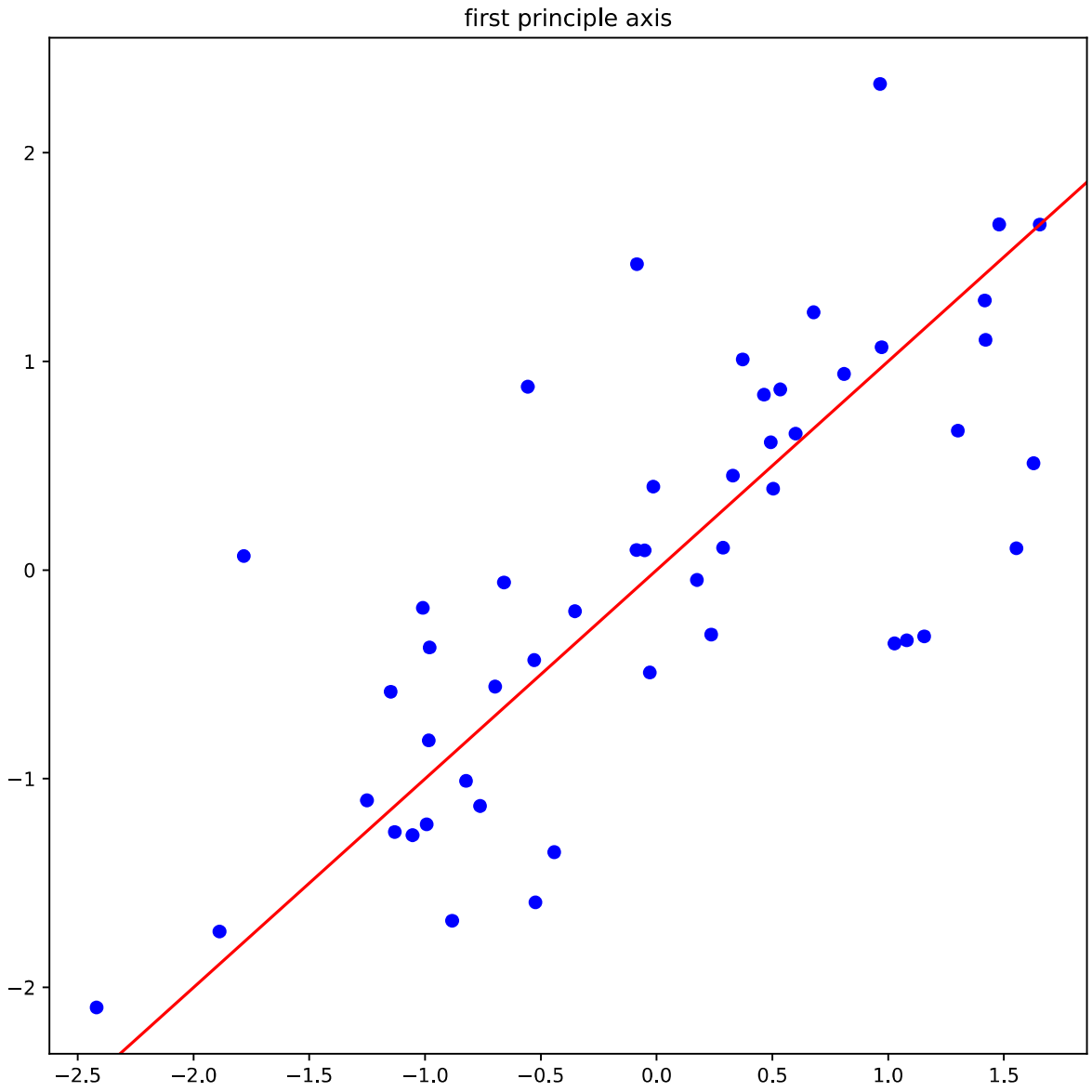


---

3. plot the first principal axis in red on the normalized data in blue

In [ ]:
```
plt.figure(figsize=(8,8))
plt.title('first principle axis')
# ================================================
```

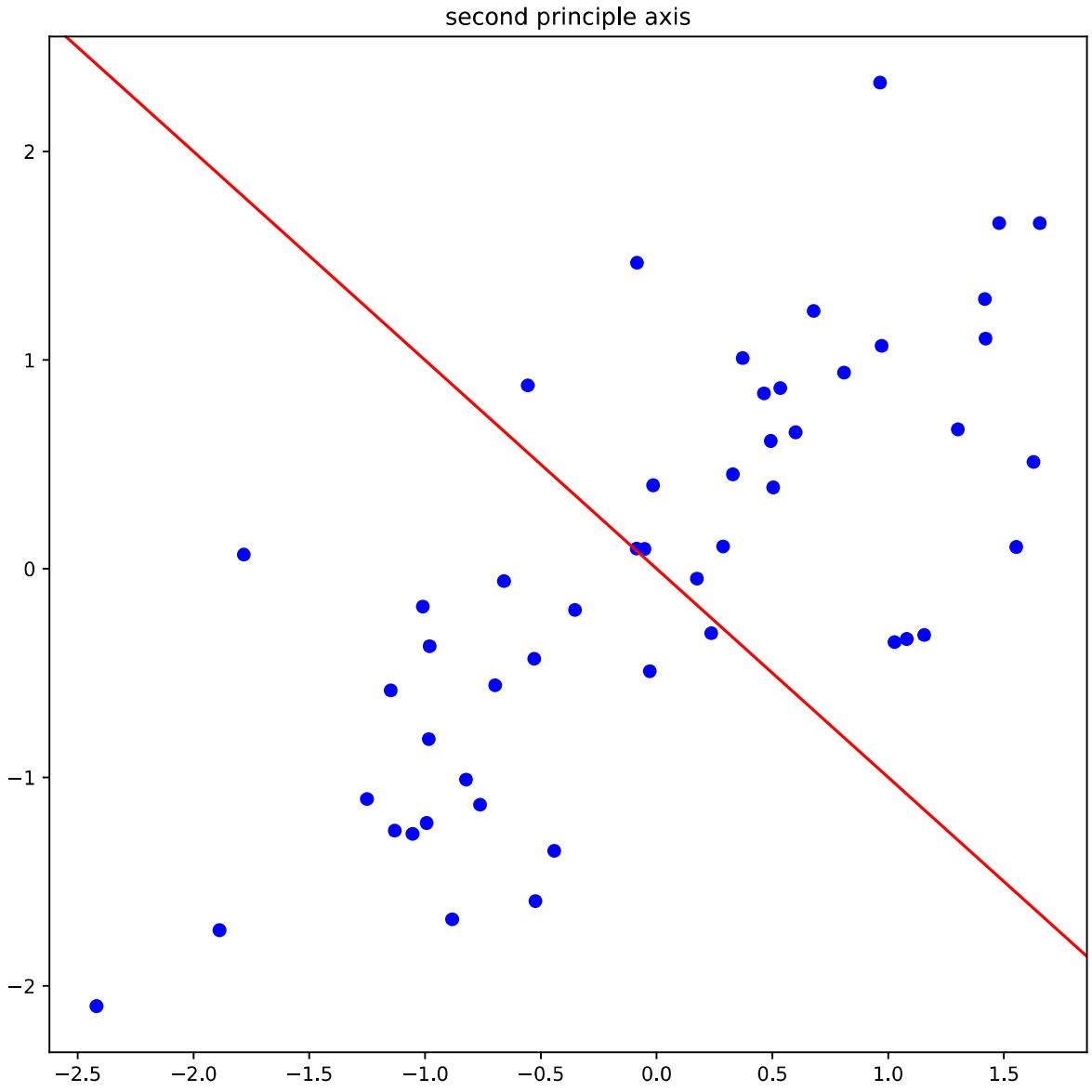
```
# fill up the blank
#
plt.scatter(zx, zy, color='blue')
plt.axline(np.array([0,0]),S[:,0], color='red')
# ================================================
plt.tight_layout()
plt.show()
```



4. plot the second principal axis in red on the normalized data in blue

In [ ]:

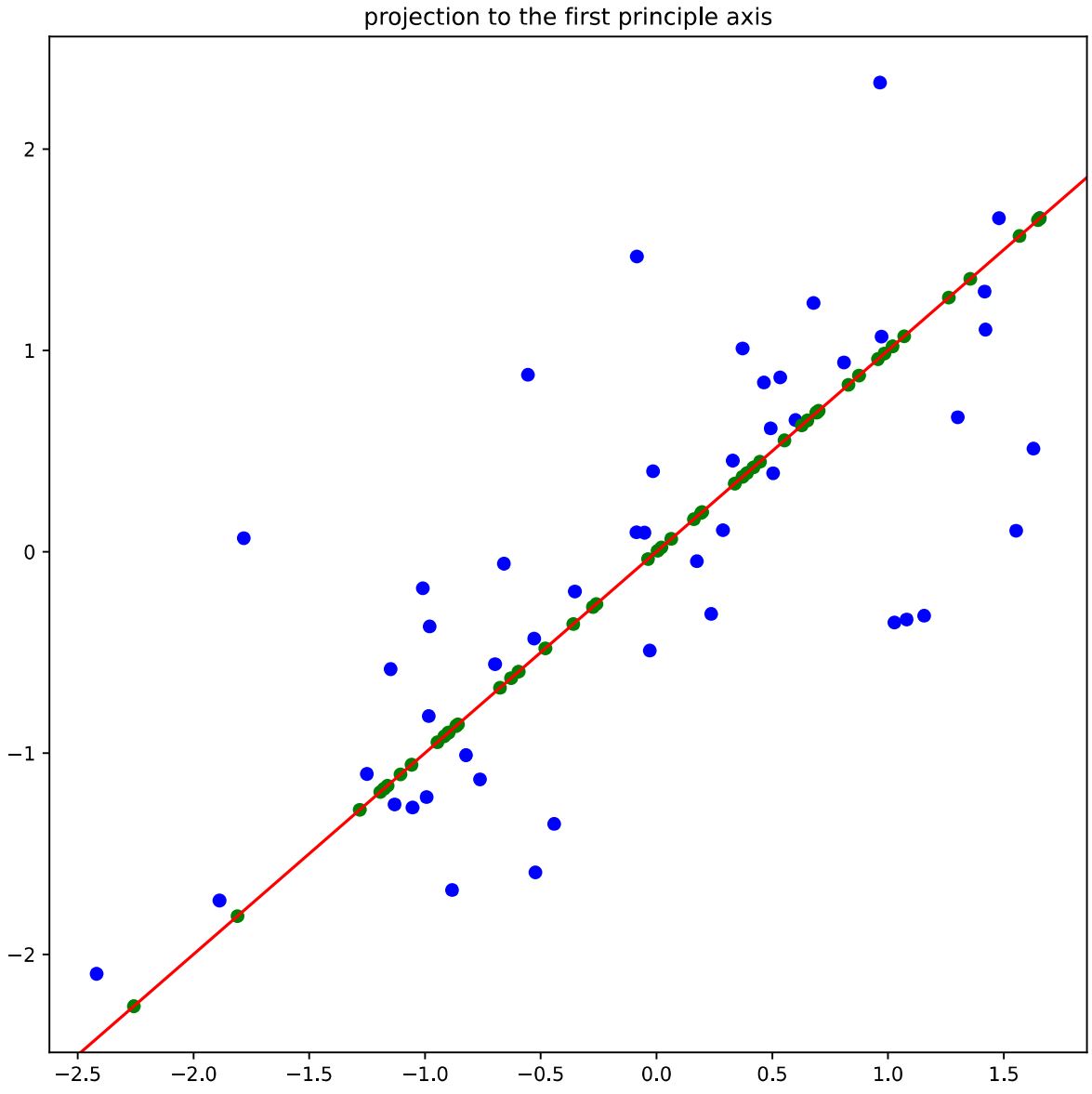
```
plt.figure(figsize=(8,8))
plt.title('second principle axis')
# ================================================
# fill up the blank
plt.scatter(zx, zy, color='blue')
plt.axline(np.array([0,0]),S[:,1], color='red')
# ================================================
plt.tight_layout()
plt.show()
```



5. plot the projection of data in green onto the first principal axis in red

In [ ]:

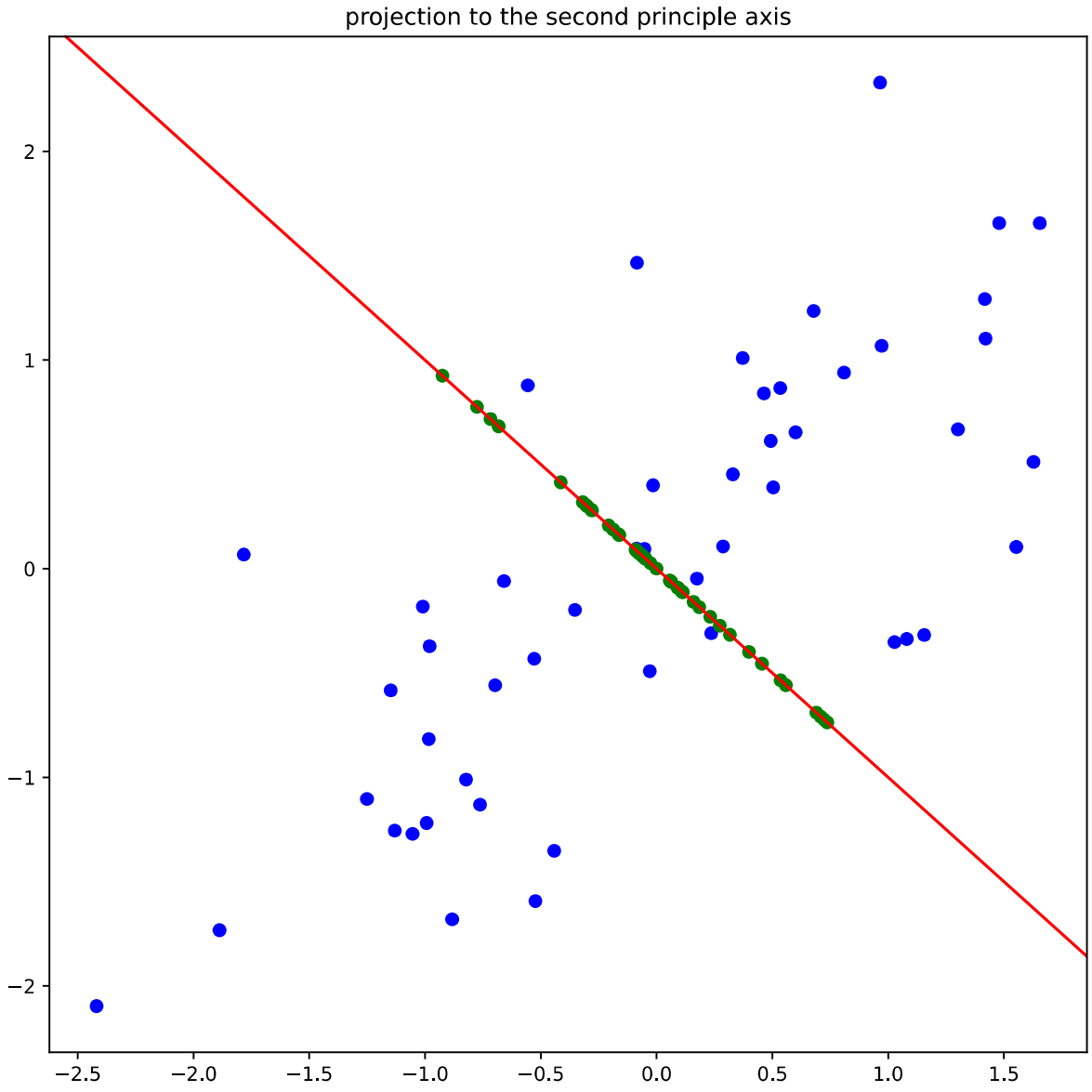
```
plt.figure(figsize=(8,8))
plt.title('projection to the first principle axis')
# ================================================
# fill up the blank
plt.scatter(zx, zy, color='blue')
plt.axline(np.array([0,0]),S[:,0], color='red')
DP = compute_projection_onto_line(feature,S[:,0])
plt.scatter([DP[:,0]],[DP[:,1]],color='green')
#
#
# ================================================
plt.tight_layout()
plt.show()
```



6. plot the projection of data in green onto the second principal axis in red

In [ ]:

```
plt.figure(figsize=(8,8))
plt.title('projection to the second principle axis')
# ================================================
# fill up the blank
plt.scatter(zx, zy, color='blue')
plt.axline(np.array([0,0]),S[:,1], color='red')
DP = compute_projection_onto_line(feature,S[:,1])
plt.scatter([DP[:,0]],[DP[:,1]],color='green')
#
# ================================================
plt.tight_layout()
plt.show()
```

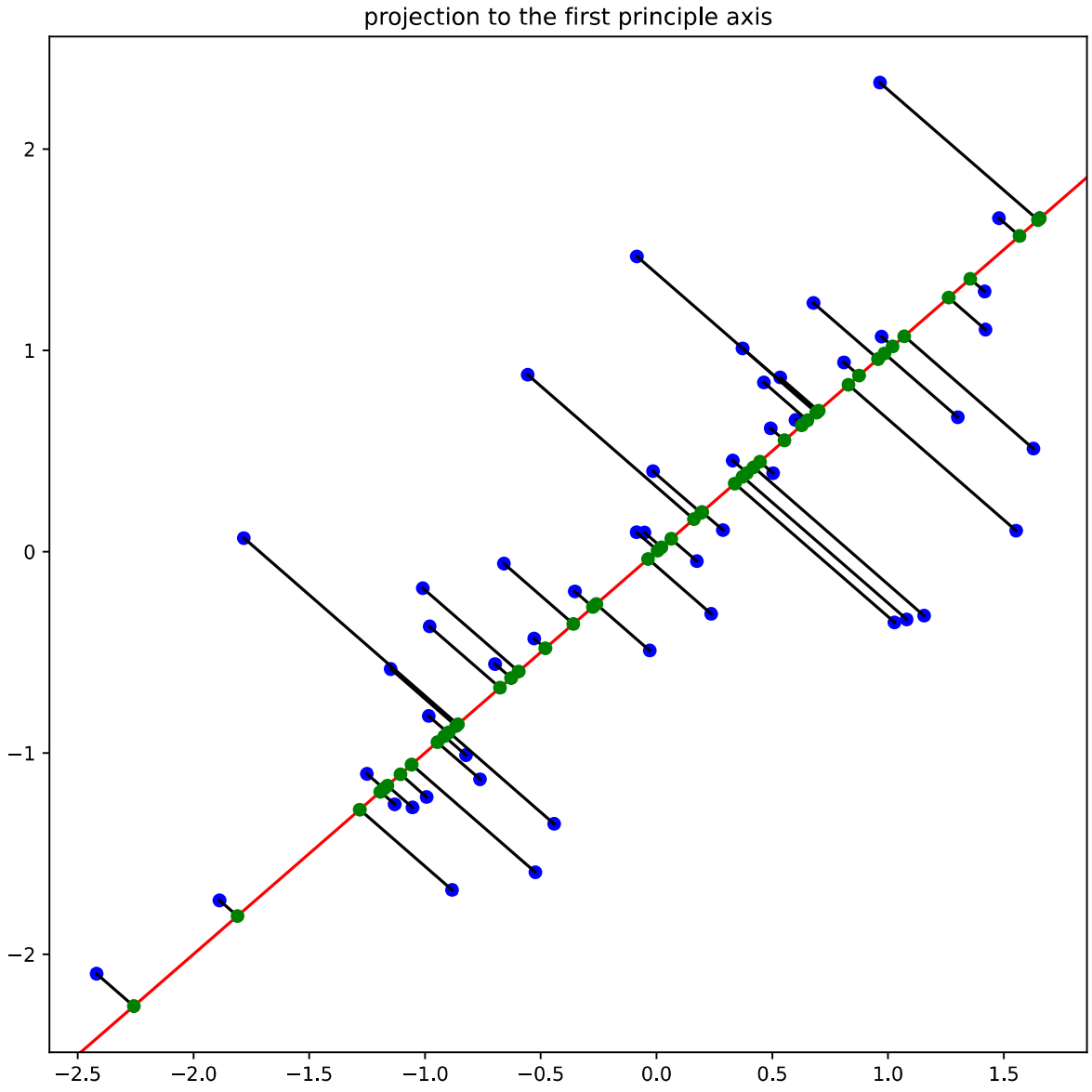


7. plot the projection line in grey onto the first principal axis

In [ ]:

```
plt.figure(figsize=(8,8))
plt.title('projection to the first principle axis')
# ================================================
# fill up the blank
plt.scatter(zx, zy, color='blue')
plt.axline(np.array([0,0]),S[:,0], color='red',zorder = 1)
DP = compute_projection_onto_line(feature,S[:,0])

#plt.plot(zx[0],zy[0],DP[0,0],DP[0,1],'-', color='black'
#print(DP)
#plt.plot([zx,zy],np.transpose(DP),color='black',linestyle='--')
for i in range(number_data):
    plt.plot([zx[i],DP[i,0]],[zy[i],DP[i,1]], color='black',zorder = 2)
    #plt.contour(DP[i,:],Df[i,:])
plt.scatter([DP[:,0]],[DP[:,1]],color='green',zorder = 3)
#
# ================================================
plt.tight_layout()
plt.show()
```



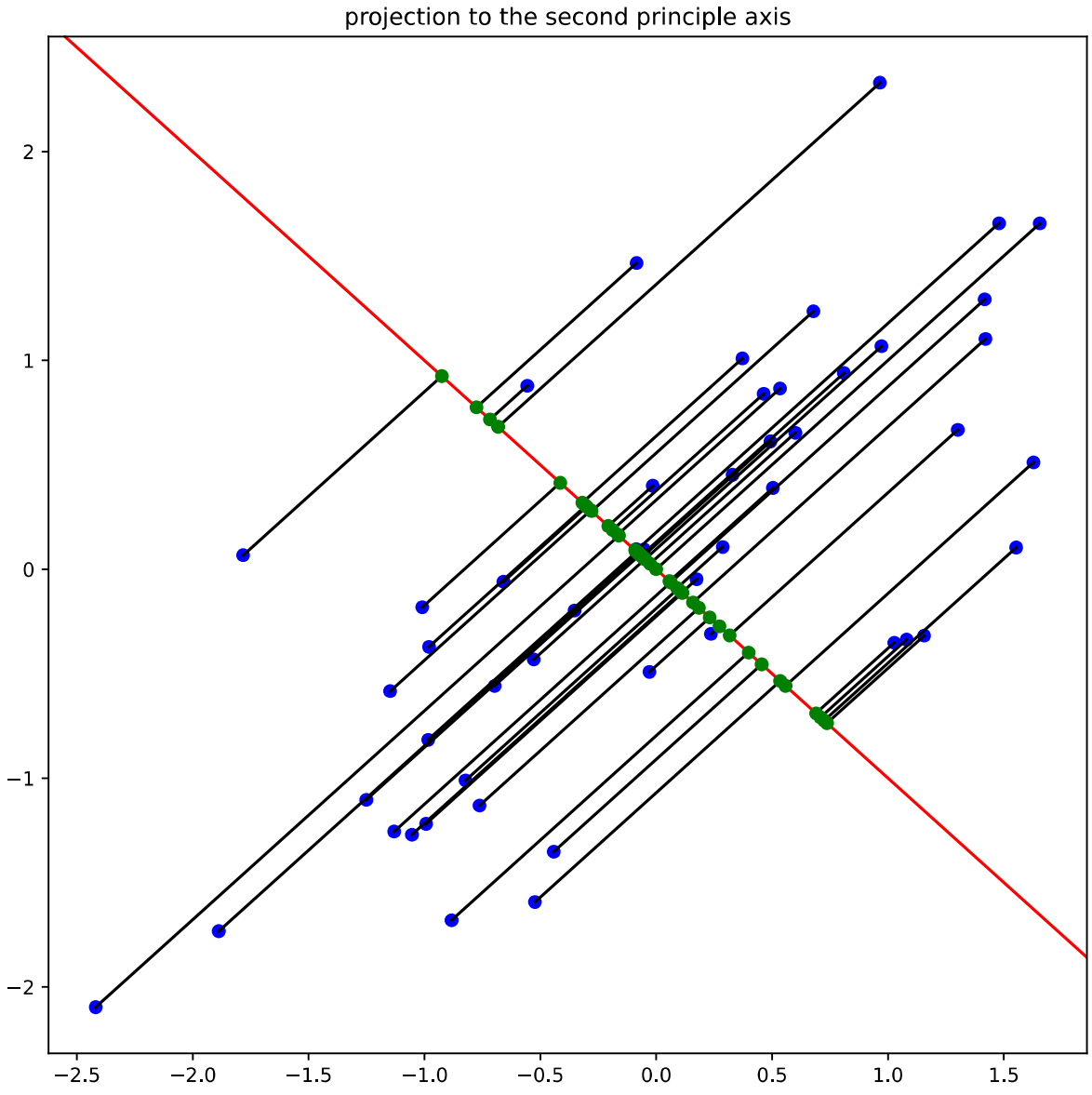
---

8. plot the projection line in grey onto the second principal axis

In [ ]:

```
plt.figure(figsize=(8,8))
plt.title('projection to the second principle axis')
# ================================================
plt.scatter(zx, zy, color='blue')
plt.axline(np.array([0,0]),S[:,1], color='red',zorder = 1)
DP = compute_projection_onto_line(feature,S[:,1])

#plt.plot(zx[0],zy[0],DP[0,0],DP[0,1],'-', color='black'
#print(DP)
#plt.plot([zx,zy],np.transpose(DP),color='black',linestyle='--')
for i in range(number_data):
    plt.plot([zx[i],DP[i,0]],[zy[i],DP[i,1]], color='black',zorder = 2)
    #plt.contour(DP[i,:],Df[i,:])
plt.scatter([DP[:,0]],[DP[:,1]],color='green',zorder = 3)
# ================================================
plt.tight_layout()
plt.show()
```


projection to the second principle axis
2
1
0
−1
−2
−2.5
−2.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5