בנימין מנשה 204938203
Signal Processing 2022

Assignment 7

1) **Information Theory Uses**

A. The research question is whether the neuronal activity during seizure-like bursts is more synchronized or less synchronized than baseline activity. Essentially, the researchers used the measures of information theory to see whether the signal-to-noise ratio increased or decreased during SLA. Spoiler: it unexpectdaly decreased, but because of more noise, not less signal.

B. H(t)

bins = 36 distribution = [15, 15, 5, 1] / 36

H(t) = - [2*15/36*log2(15/36) + 5/26*log2(5/36) + 1/36*log2(36)] = **1.456**

mean(H(n))

n1: distribution = [2, 2] / 4 H(n1) = - [2*0.5*log2(0.5)] = 1
n2: distribution = [3, 1] / 4 H(n2) = - [0.75*log2(0.75) + 0.25*log2(0.25] = 0.811
n3: distribution = [1,2,1] / 4 H(n3) = - [2*0.25*log2(0.25) + 0.5*log2(0.5)] = 1.5
n4: distribution = [3,1] / 4 H(n4) = - [0.75*log2(0.75) + 0.25*log2(0.25] = 0.811
n5: distribution = [3,1] / 4 H(n5) = - [0.75*log2(0.75) + 0.25*log2(0.25] = 0.811
n6: distribution = [3,1] / 4 H(n6) = - [0.75*log2(0.75) + 0.25*log2(0.25] = 0.811
n7: distribution = [4] / 4 H(n7) = - [1*log2(1)] = 0
n8: distribution = [4] / 4 H(n8) = - [1*log2(1)] = 0
n9: distribution = [2, 2] / 4 H(n9) = - [2*0.5*log2(0.5)] = 1

Mean(H(n)) = (2*1 + 4*0.811 + 1.5) / 8 = **0.843**

C. Wrote code in python, Attached in appendix A. For each bin size, we get a spike count, then calculate the overall entropy (using the distribution of spike counts), then subtract the mean entropy for each bin across trials.

D. Generated a poisson-like neuron which changed its firing rate every 50 ms, for 2 seconds. This is the table of the firing rates and the best bin the function found (I ran the function 100 times and got the approximate mean and std for bin size, as well as the approximate chance of this method getting a bin size of 50):

Firing rate 1 (Hz)	Firing rate 2 (Hz)	P(bin size = 50)	Best bin size mean (std)
100	0	0.07	64.21 (16.62)
70	20	0.04	87.08 (10.37)
50	20	0	91.47 (6.15)

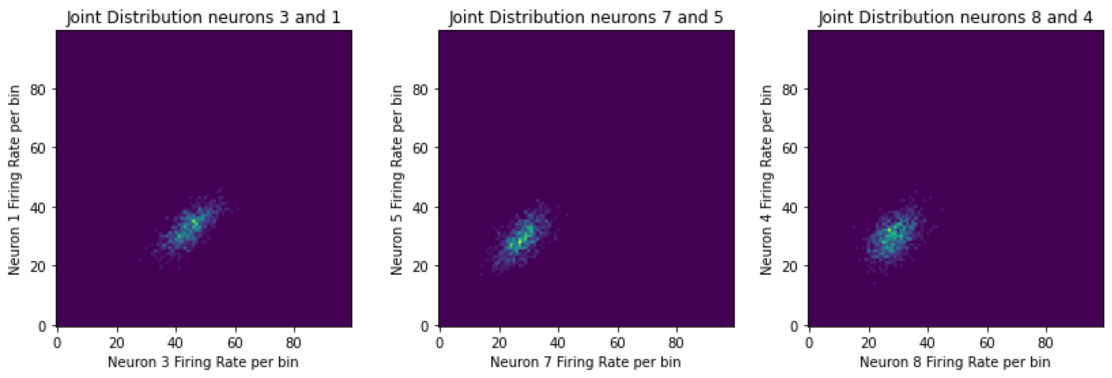
Overall, this method doesn't look so good, especially with short time samples such as the one we used. I presume there is too much variation in the Poisson process, and when the firing rate changes are not that large, the estimation of information is just not that accurate.

E. Spk_mat:

a. Best bin size = 99.

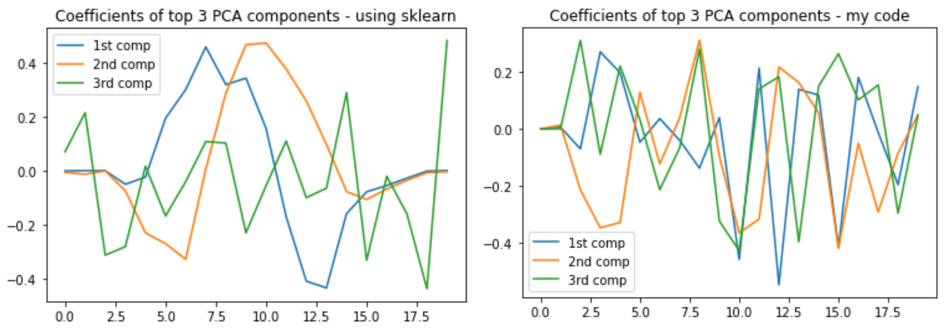
b. Entropy for each neuron (different bin size for each, but all close to 99): [1.22, 1.15, 1.13, 1.11, 1.08, 1.09, 1.11, 1.15]

c. Using bins of size 100, we got that:
neuron 3: neuron 1, neuron 7: neuron 5, neuron 8: neuron 4



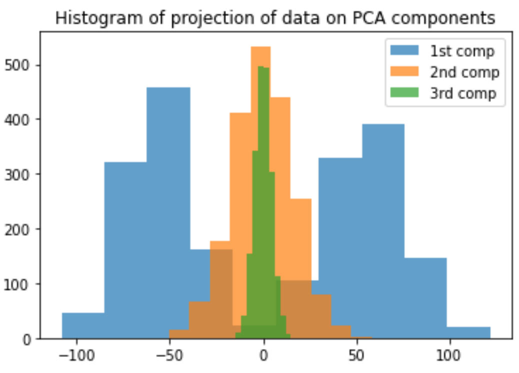
2) PCA and clustering

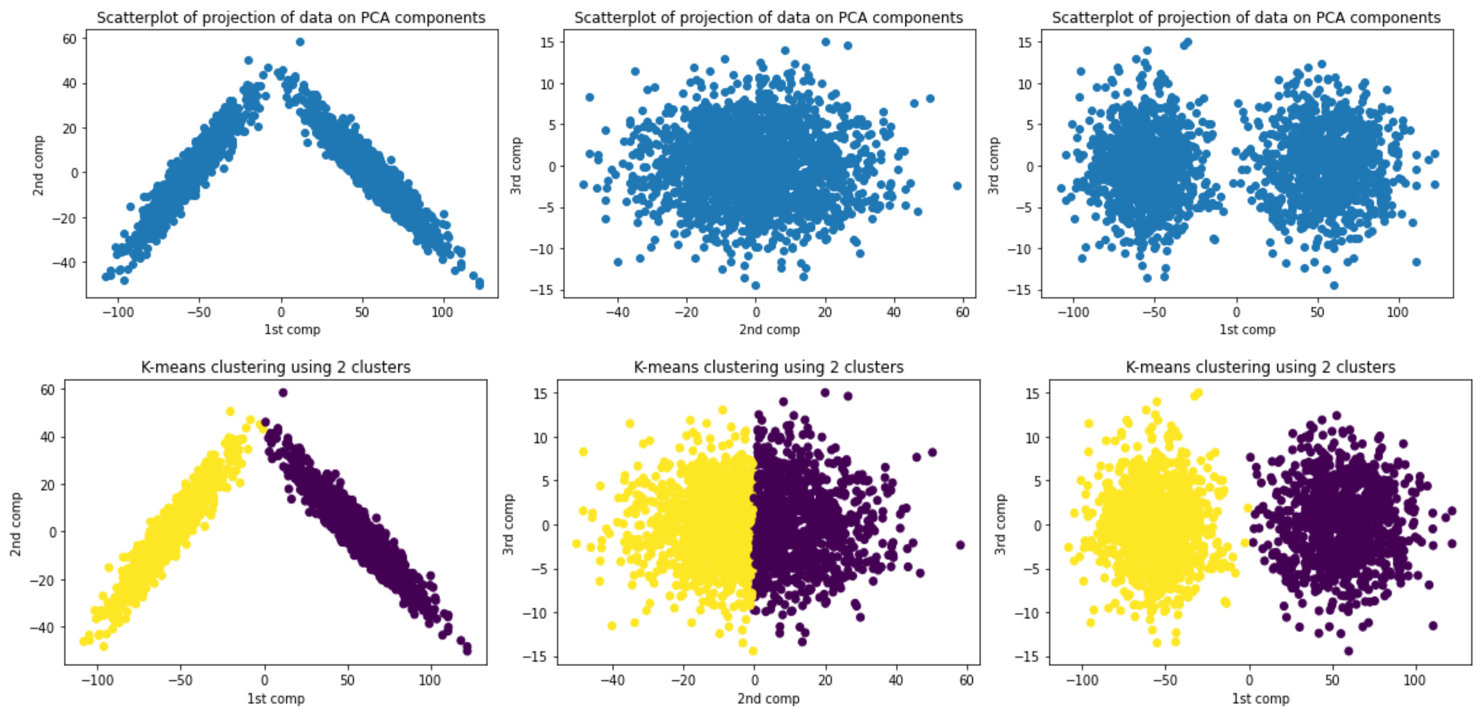
A. Figures:



B. Wrote code in python, attached in appendix B. The results were different, as can be seen above. However, when projecting the data unto the PCA components, I got the same scatterplots as in section C, except with components 2 and 3 are reversed (see appendix). I assume what happens is that my projection and sklearn's projections are equivalent (project to the same axes), but one of flips the direction of the axis.

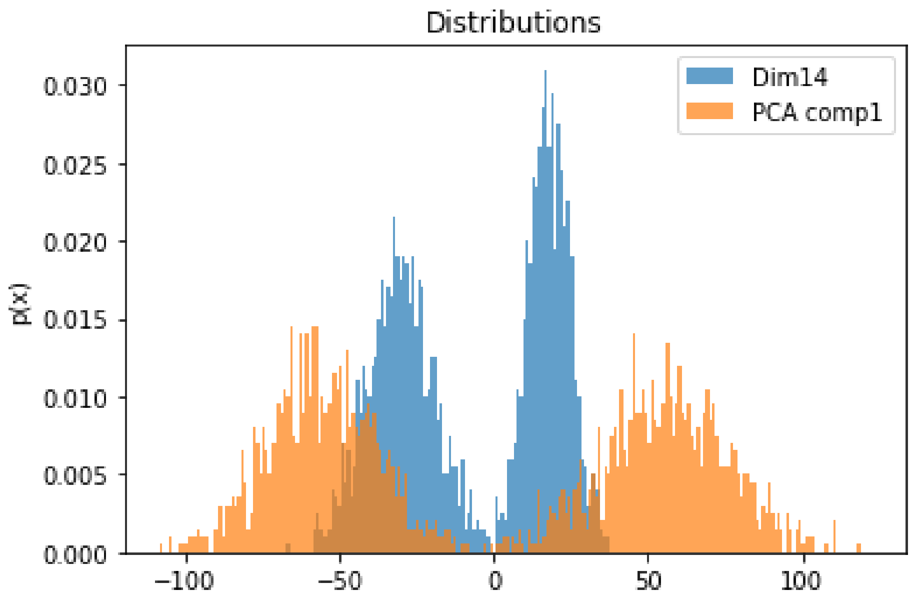
C. Figures attached below. As can be seen by eye, PCA component 1 splits the data rather nicely into 2 groups (can be seen both on the distribution and the scatter plots), while components 2 and 3 do not split the data any further. Furthermore, it is clear to me that the data is clustered into 2 groups in the scatterplots which use the 1st PCA components, therefore I used a K-means algorithm with 2 clusters on the PCA projections. We can clearly see the algorithm worked (of course, for scatterplot of PCA components 2 and 3, there are no 2 clusters, so it just arbitrarily split the data in half). I did not see any point trying other cluster numbers or methods (especially since k-mean worked so well, usually we would try EM when k-mean has a problem with elliptic data, but here we have no such problem).



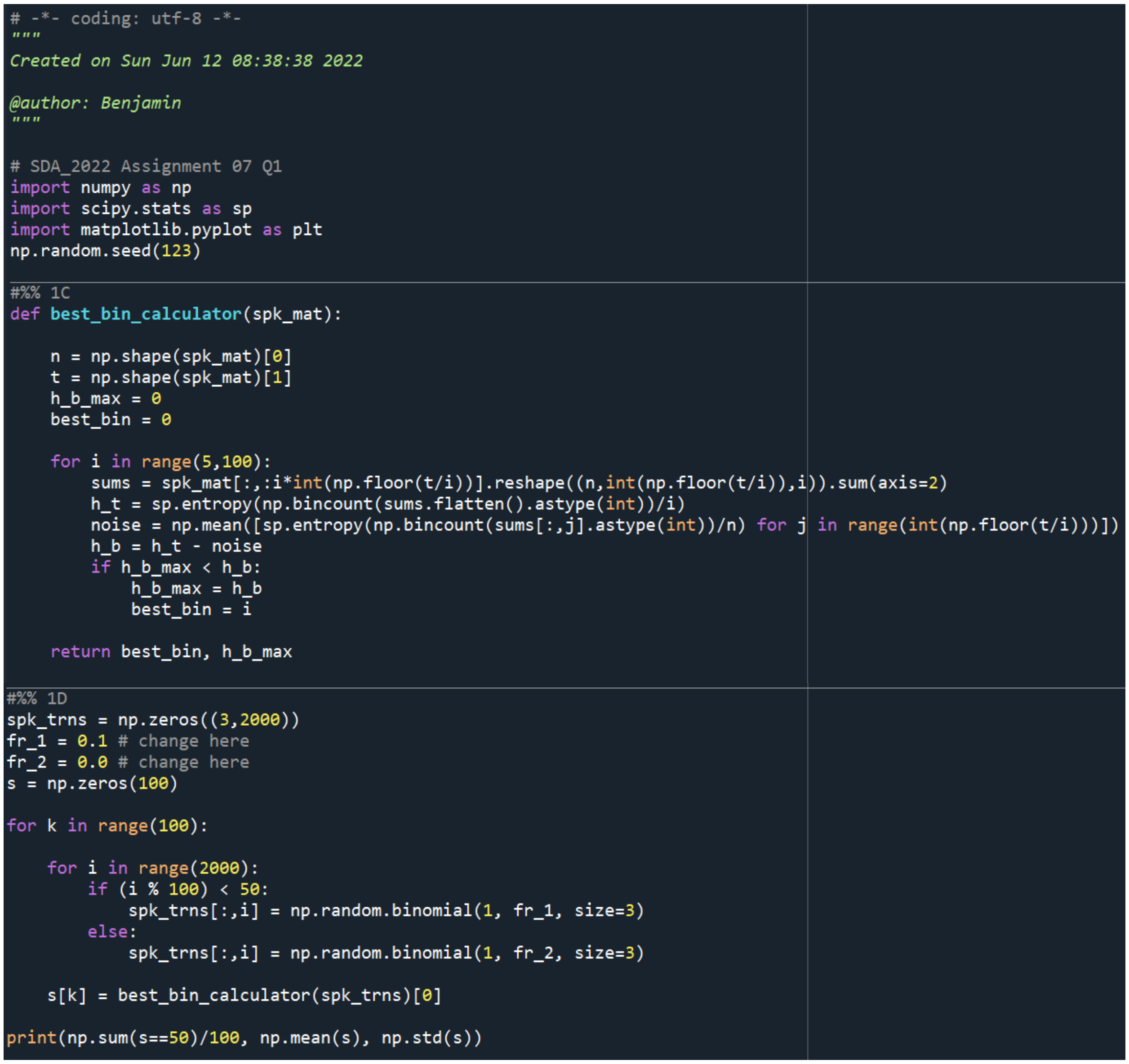


D. 1st component: 0.866
   2nd component: 0.063
   combined: 0.929
   remaining variance: 0.071

E. Column with highest variance = 14 (var=669.48).
   Used an approximation of the distribution using a bins of size 1
   entropy of column = 4.25
   entropy of first PCA component = 4.99

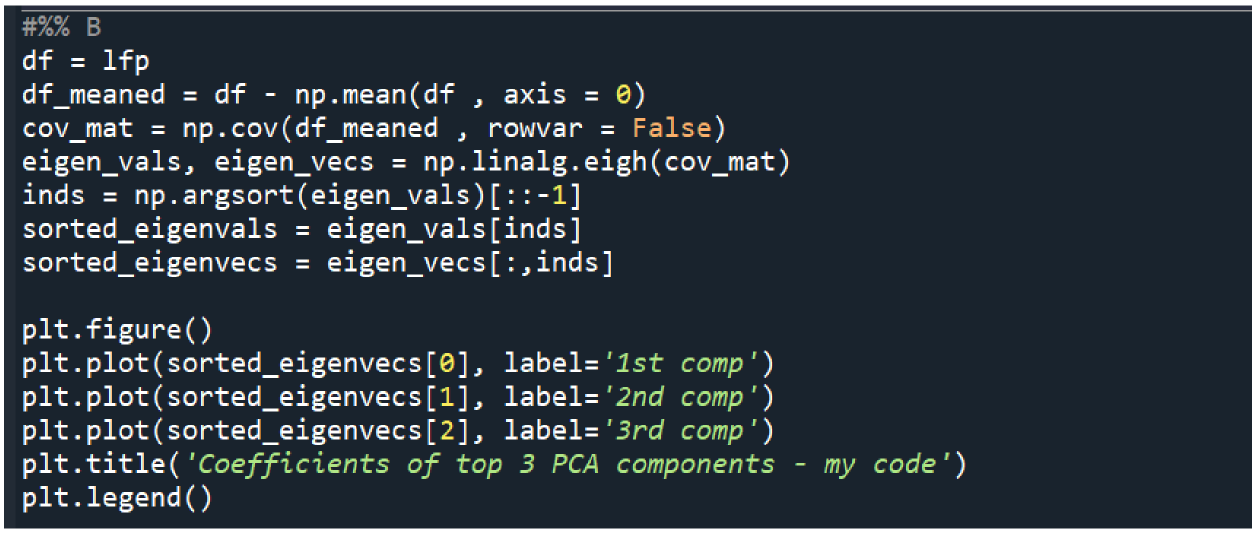


Appendix A – code for Question 1



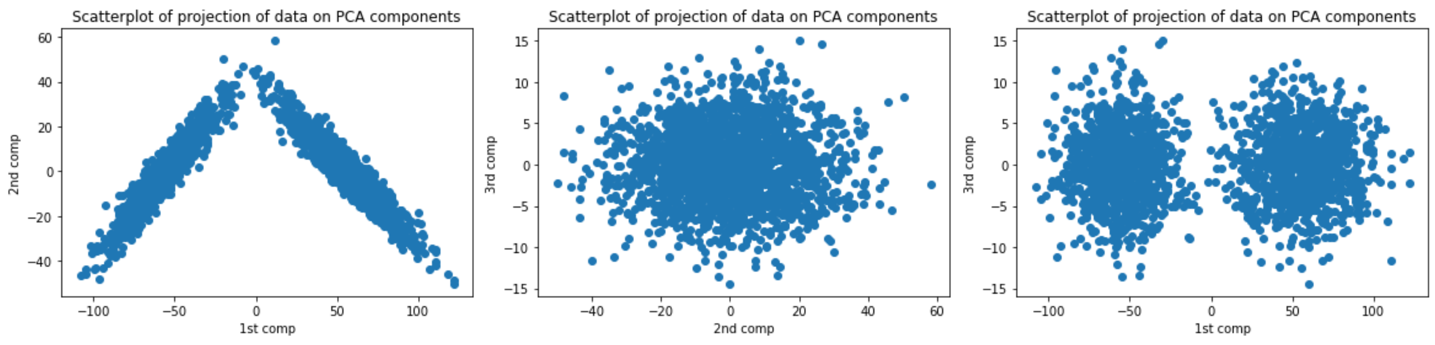
```
# -*- coding: utf-8 -*-
"""
Created on Sun Jun 12 08:38:38 2022

@author: Benjamin
"""

# SDA_2022 Assignment 07 Q1
import numpy as np
import scipy.stats as sp
import matplotlib.pyplot as plt
np.random.seed(123)

#%% 1C
def best_bin_calculator(spk_mat):

    n = np.shape(spk_mat)[0]
    t = np.shape(spk_mat)[1]
    h_b_max = 0
    best_bin = 0

    for i in range(5,100):
        sums = spk_mat[:,:i*int(np.floor(t/i))].reshape((n,int(np.floor(t/i)),i)).sum(axis=2)
        h_t = sp.entropy(np.bincount(sums.flatten().astype(int))/i)
        noise = np.mean([sp.entropy(np.bincount(sums[:,j].astype(int))/n) for j in range(int(np.floor(t/i)))])
        h_b = h_t - noise
        if h_b_max < h_b:
            h_b_max = h_b
            best_bin = i

    return best_bin, h_b_max

#%% 1D
spk_trns = np.zeros((3,2000))
fr_1 = 0.1 # change here
fr_2 = 0.0 # change here
s = np.zeros(100)

for k in range(100):

    for i in range(2000):
        if (i % 100) < 50:
            spk_trns[:,i] = np.random.binomial(1, fr_1, size=3)
        else:
            spk_trns[:,i] = np.random.binomial(1, fr_2, size=3)

    s[k] = best_bin_calculator(spk_trns)[0]

print(np.sum(s==50)/100, np.mean(s), np.std(s))
```

Appendix B – code for question 2

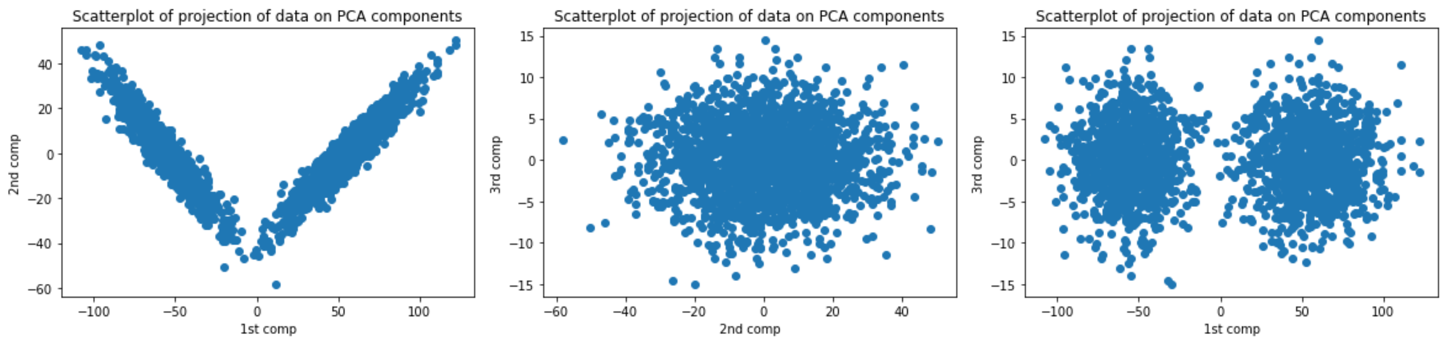


```python
#%% B
df = lfp
df_meaned = df - np.mean(df , axis = 0)
cov_mat = np.cov(df_meaned , rowvar = False)
eigen_vals, eigen_vecs = np.linalg.eigh(cov_mat)
inds = np.argsort(eigen_vals)[::-1]
sorted_eigenvals = eigen_vals[inds]
sorted_eigenvecs = eigen_vecs[:,inds]

plt.figure()
plt.plot(sorted_eigenvecs[0], label='1st comp')
plt.plot(sorted_eigenvecs[1], label='2nd comp')
plt.plot(sorted_eigenvecs[2], label='3rd comp')
plt.title('Coefficients of top 3 PCA components - my code')
plt.legend()
```

sklearn PCA projections:



my PCA projections:


Scatterplot of projection of data on PCA components
2nd comp
1st comp
Scatterplot of projection of data on PCA components
3rd comp
2nd comp
Scatterplot of projection of data on PCA components
3rd comp
1st comp