Code: [cv\_speckel.m](https://github.com/weigcdsb/state-space-clustering/blob/main/LDS/constraint/speckel/cv_speckel.m)

Training dataset (3/4 for each neuron)



Test dataset (1/4 for each neuron)



True: 

|  |  |  |  |
| --- | --- | --- | --- |
|  | 3 clusters, p = 2 | 1 cluster, p = 2 | 1 cluster, p =6 |
| Llhd/spk in test dataset | (iteration 500 to 1000)  -1.0653  (iteration 2500 to 5000)  -1.0579 | (iteration 500 to 1000)  -1.0673 | (iteration 500 to 1000)  -1.0550, already greater than (3 clusters, p = 2, iteration 2500 to 5000) |
|  | (iteration 2500 to 5000) |  |  |

The convergence is achieved based on the trace plots (3 clusters & p = 2). I still think the “clustered” version will be very similar to the traditional PLDS, if p is chosen appropriately.

But I think this is meaningful. Just like Joshua’s paper, based held-out llhd, their method is close to previous methods. However, they can extract more structure (‘switching of state’ in their paper) on the same level of prediction power.

Follow the same rationale, I view “clustering” **majorly as an inference tool**, while keeping the prediction power. In other words, using “clustering” can give us more insights to the functional group, while not sacrificing the prediction ability.

After using the “clustering”, neuroscientists may guess the neurons within cluster may talk to each other (physically or chemically), and then do experiments to prove that. (My undergraduate classmates did a lot of these kinds of things, e.g. signal pathways)

Appendix:

When nClus = 1, p = 6, the average loading is (iteration 500 to 1000)

0.4079 0.2549 -2.1383 0.5827 -0.4771 -0.3496

2.2746 -0.5854 -2.0637 1.2762 -1.1696 -0.2830

3.3095 -1.2815 -2.2928 1.1612 -0.8106 0.2114

4.9536 -1.6957 -2.1908 1.3250 -1.3200 0.2438

6.0346 -1.9556 -1.9125 1.4851 -1.1831 0.3320

6.9765 -2.5887 -1.4726 1.6916 -1.4666 0.4313

7.8523 -3.5531 -1.8094 2.0951 -1.2040 0.9533

8.5900 -4.0420 -2.6808 2.3093 -0.9638 0.9116

10.6535 -4.9874 -2.5334 2.8074 -1.2398 1.2474

11.1037 -5.0568 -2.1528 2.8438 -1.2386 1.4641

3.4634 -0.7903 -3.5077 1.6517 -1.7356 -0.5731

4.4801 -1.3138 -3.3158 1.5712 -1.2721 -0.3260

4.7281 -1.5054 -3.4827 1.5071 -1.8854 -0.4663

3.8809 -1.6033 -3.3278 1.2671 -1.0282 0.3728

4.7842 -1.4567 -2.5352 1.3512 -1.5074 -0.1447

5.2149 -1.7661 -3.0964 2.0407 -1.3887 0.1622

4.9251 -2.0116 -3.1697 2.0131 -1.5071 0.5639

5.0399 -2.3511 -3.2848 2.0682 -1.1359 0.5290

4.1141 -2.2874 -3.7915 2.0295 -0.7592 0.8324

5.2442 -2.4148 -3.5454 2.4111 -0.9925 0.9126

4.8534 -3.7286 -0.9880 1.6230 0.6654 1.5425

3.4547 -3.3873 -0.2185 1.3959 1.3192 1.8796

1.7446 -3.4111 -0.8460 1.6530 1.5626 2.0768

0.7442 -2.6104 -0.2261 1.1884 1.7790 1.8694

1.0093 -2.8919 -0.0893 0.8656 2.0448 1.8613

-0.5368 -2.3560 -0.1200 0.7802 2.4478 2.0016

-1.8628 -2.1261 0.8551 0.8300 2.6078 2.1123

-3.1352 -1.6715 0.8455 0.3617 2.7454 2.1134

-3.6227 -1.5088 0.8564 0.6375 3.1536 2.3943

-4.8606 -1.4422 1.4271 0.2468 3.4706 2.2777