Brain data and Music data application 2.

Chanwoo Lee

Dec 25, 2019

1 Brain Data Application

1.1 Clustering method

My approach for clustering on each mode is based on the relationship between Principal Component Analysis (PCA) and Singular Value Decomposition (SVD). Let X be a data matrix. Based on SVD, we have the following.

$$X = U\Sigma V^T.$$

where U is an orthornormal matrix and Σ is the diagonal matrix. Then, columns of V are principal directions or axes. Also, columns of $U\Sigma$ are principal components. Therefore, we can measure distance between two different data points by calculating rows of $U\Sigma$. Likewise, Suppose a data tensor Θ has Tucker decomposition as,

$$\Theta = \mathcal{C} \times_1 A_1 \times_2 A_2 \times_3 A_3.$$

Consider the task of clustering tensor slides along the mode-1. Let $A_1 \in \mathbb{R}^{d_1 \times r_1}$ be mode-1 factor matrix and $\mathcal{C} \in \mathbb{R}^{r_1 \times r_2 \times r_3}$ be the core tensor. My suggestion is to unfold Θ as

$$\Theta_{(1)} = A_1 \mathcal{C}_{(1)} (A_3 \otimes A_2)^T.$$

We can interpret $(A_3 \otimes A_2)$ as principal axes and $A_1\mathcal{C}_{(1)}$ as principal components. Therefore, my suggestion is to apply k-means regarding each row of $A_1\mathcal{C}_{(1)}$ as each point.

One way to check whether this method does make sense is to compare two cluster groups, one is from when you cluster tensor slides along the mode-1 and the other is from when you cluster tensor slides along the mode-2. Since mode-1 and mode-2 have the exactly same meaning in the brain data, two cluster groups are supposed to be similar. I did simple test to check validity of this method when we set tucker rank is (24,24,7). The following figure is the result of the two clusterings according to mode-1 and mode-2.

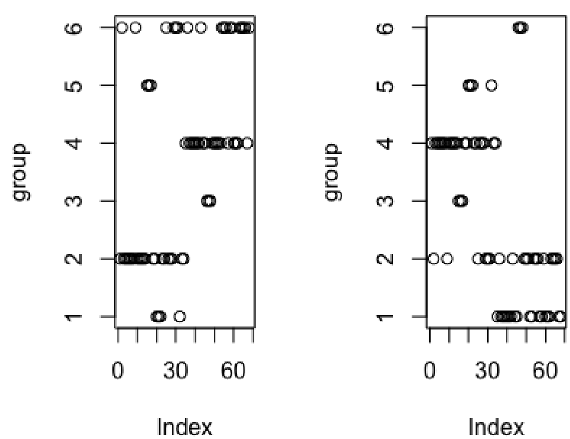


Figure 1: The left figure is from when you do clustering according to mode-1 and the right figure is from when you do clustering according to mode-2. We can easily check that 1 group in the left side matches with 5 group in the right side. Likewise, we can check the following matches: 2-4, 3-6, 4-1, 5-3, 6-2.

However, it turned out that there is not good cluster matches when you cluster the factor matrix A_1 and the factor matrix A_2 giving each column weight according to the norm of each row of the unfolded core tensor. The following figure is from when we cluster with weighting according to the norm of the unfolded core tensor.

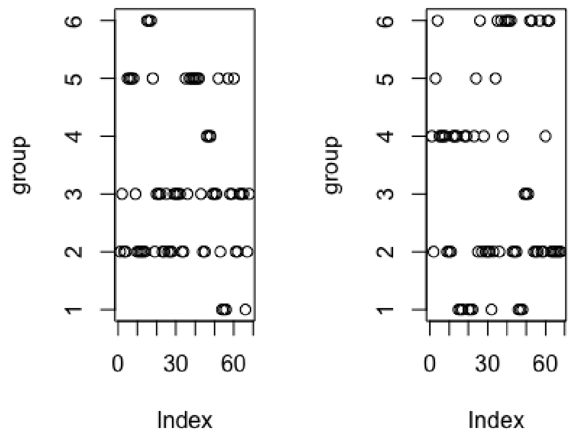


Figure 2: Unlike Figure 1, it is hard to match between the right and left side groups.

The following figure is the visualization of cluster groups of the first approach. Although there is a tendency that most of nodes is contained in the first and the second clusters, I think our algorithm make reasonable cluster according to brain region. The first cluster group is mostly the right side of the brain and the second cluster group is mostly the left side of the brain. The third cluster connects triangular nodes on the right side and the fourth cluster connects triangular nodes on the left side. The fifth cluster seems to represent bottom right part of the brain and sixth cluster seems to represent bottom left part of the brain. As you can see, our algorithm succeeded to give us reasonable explanation of the result.

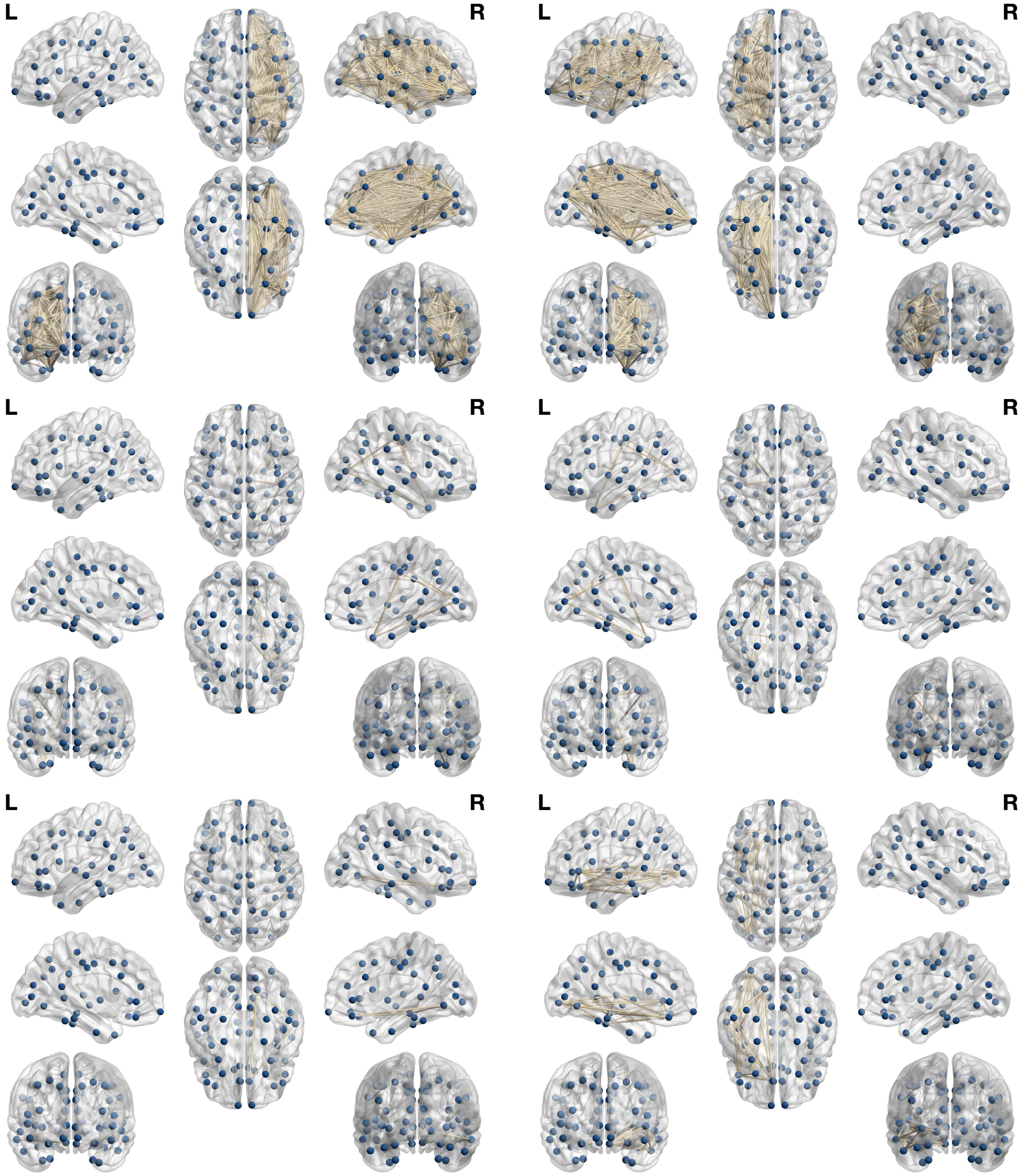


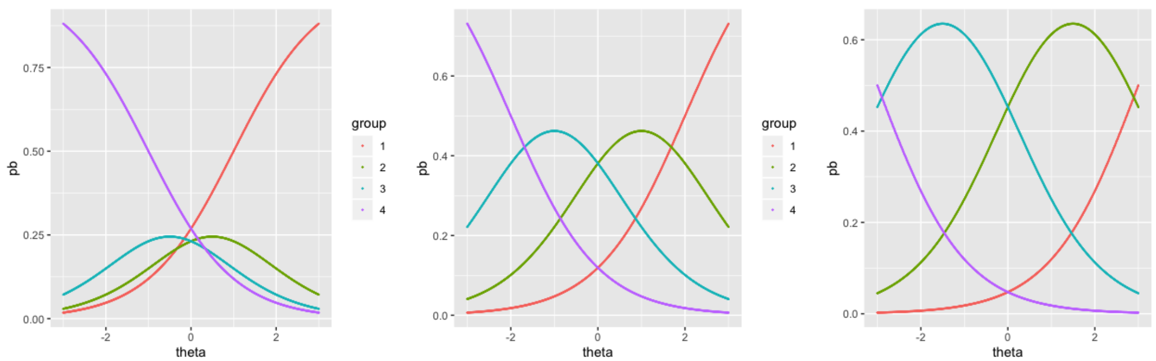
Figure 3: Brain image from the cluster groups.

2 Missing Data Application

2.1 Issues to be handled.

The music data is a $42 \times 139 \times 26$ dimensional tensor with 42 individuals, 139 songs and 26 contexts. The number of available data is only 2884 out of 151788 which implies that the missing rate is 98.09998%. I think this missing rate makes estimation of the parameters hard. There are some issues to be handled.

1. We can perform the expectation step well where we estimate $\hat{y} = \arg\max_{i \in [5]} P(y = i)$: ω values have major role in the expectation step because some values can not be obtained in some ω. Consider the following figures.



The figures show the probability $P(y = i)$ where $i \in [4]$ according to θ value where $\omega \in \{(-1, 0, 1), (-2, 0, 2), (-3, 0, 3)\}$. The leftmost side is when $\omega = (-1, 0, 1)$ and the rightmost side is when $\omega = (-3, 0, 3)$. We can check that the values 2,3 can not be obtained as an expectation step if we estimate $\hat{y} = \arg\max_{i \in [4]} P(y = i)$ when $\omega = (-1, 0, 1)$ because $P(y = 2), P(y = 3)$ are smaller than $\max(P(y = 1), P(y = 2))$ for any θ . If I update ω first from initial points A_1, A_2, A_3 which are random orthonormal matrices and $\mathcal{C}$ whose entries are generated from normal distribution, we get $\omega = (-0.4001899, 0.3022475, 1.0533067, 1.9447256)$. For this ω, the probability graph is like as follows.

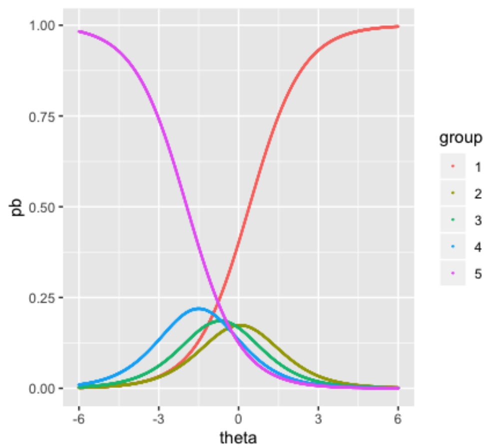


Figure 4

In this initial point, we can not get a good expectation step for y. I tried different initial points. One of them is the output of Tucker decomposition from the data where missing parts are filled with random sampling with the multinomial parameter of $(\frac{\text{\#of 1 in available data}}{\text{\#of available data}}, \cdots, \frac{\text{\#of 5 in available data}}{\text{\#of available data}})$. But still the probability plot looks like Figure 4.

However, I do not think that this ω is necessarily a wrong estimation. I did simulation with data generated by the following algorithm.

```
d=c(20,20,20)
r=c(3,3,3)
A_1 = matrix(runif(d[1]*r[1],min=-1,max=1),nrow = d[1])
A_2 = matrix(runif(d[2]*r[2],min=-1,max=1),nrow = d[2])
A_3 = matrix(runif(d[3]*r[3],min=-1,max=1),nrow = d[3])
C = as.tensor(array(runif(prod(r),min=-1,max=1),dim = r))
omega = c(-1,0,1)
theta = ttl(C,list(A_1,A_2,A_3),ms=1:3)@data
ttnsr <- realization(theta,omega)@data

```

Although ω in the simulation never assign the largest probability to the value 2 and 3, the number of 2 and 3 values in the data is 3420 out of 8000. Also our algorithm successfully estimate $\hat{\omega} = (1.05452469, -0.02471193, 0.99550457)$ and $\hat{\theta}$ as in the following figure.

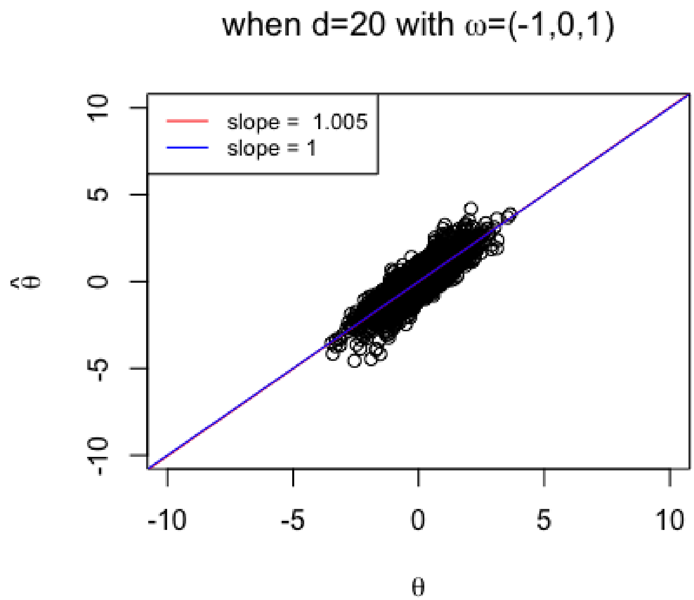


Figure 5: Y axis is the estimation $\hat{\theta}$ and X axis is the real θ

2. How to estimate y then?: My approach to this problem is to do random sampling with the multinomial parameter $(P(y=1), P(y=2), P(y=3), P(y=4), P(y=5))$ not estimating by $y = \arg\max_{i\in[5]} P(y=i)$. However, I am worried that this estimation does not have a good performance since I added randomness in the estimation.

3 To do list.

1. I am still waiting for the output of the brain data which I put on the server. I got outputs of the rank $(21,21,6), \cdots, (21,21,10), (22,22,6), \cdots, (25,25,6), \cdots, (25,25,9), (26,26,6)$ and the smallest BIC rank was $(24,24,10)$ from the output I got. I think that I need to do more rank experiments. However, as long as I can get the smallest BIC rank, then everything is going to be fine.

2. For music data, I need to construct a good way to estimate missing data.

3. I did literature review to find other methods analyzing ordinal tensor. However, I could not find good methods to compare with our method. I have plans to spend more time to find other methods.

4. I will do cross validation to compare our method with continuous Tucker decomposition method when I decide the best rank.