March 21 tensor clustering

Yuchen Zeng

March 2019

1 Verification of the method of choosing k,r,l

Here I set n=20, p=20, q=20, the simulation results are shown as table 1. The accuracy rate is $\frac{times\ of\ choosing\ true\ k,r,l}{total\ simulation\ times}$.

2 Correction in last summary

In fact we cannot use such a definition to find the error rate in the last summary. The reason is after labeling all the modes, the labels may be different though the clustering results are the same. In this case, the mus tensor would be different from the original one even if the clustering result is correct. So we change our method of estimating the model into comparing the zeros we got in our model and the zeros in true tensor.

3 The simulation result of chooseLambda()

3.1 matrix situation

Here is the result of chooseLambda()(the pink dots) and sparseBC.BIC()(the blue triangles) (accoring to figure 1). They are almost consistent with each other. The first figure we use the original calculateBIC() method and the second figure we use the new calculateBIC() method. I think the model in the original paper is not good but the code is reasonable. And here we choose the $\lambda = 200$ which was selected by sparseBC.BIC() and chooseLambda(). And the results of label2() is as figure 2.

4 tensor situation

Figure 3 is the result (both method = "new" and method = "original" are the same).

k	r	l	accuracy rate	mean k	mean r	mean l	noise	sim.times
2	2	4	1	2(0)	2(0)	4(0)	1	5
4	3	4	1	4(0)	3(0)	4(0)	1	5
2	4	3	1	2(0)	4(0)	3(0)	1	5
4	3	4	1	4(0)	3(0)	4(0)	2	5
2	4	3	0.8	2(0)	3.8(0.2)	3(0)	2	5
2	3	4	1	2(0)	3(0)	4(0)	1	10
2	4	4	0.9	2(0)	4(0)	3.9(0.1)	1	10
3	3	3	1	3(0)	3(0)	3(0)	1	10
3	2	3	1	3(0)	2(0)	3(0)	1	10

Table 1: n=20,p=20,q=20, simulation of choosing k,r,l

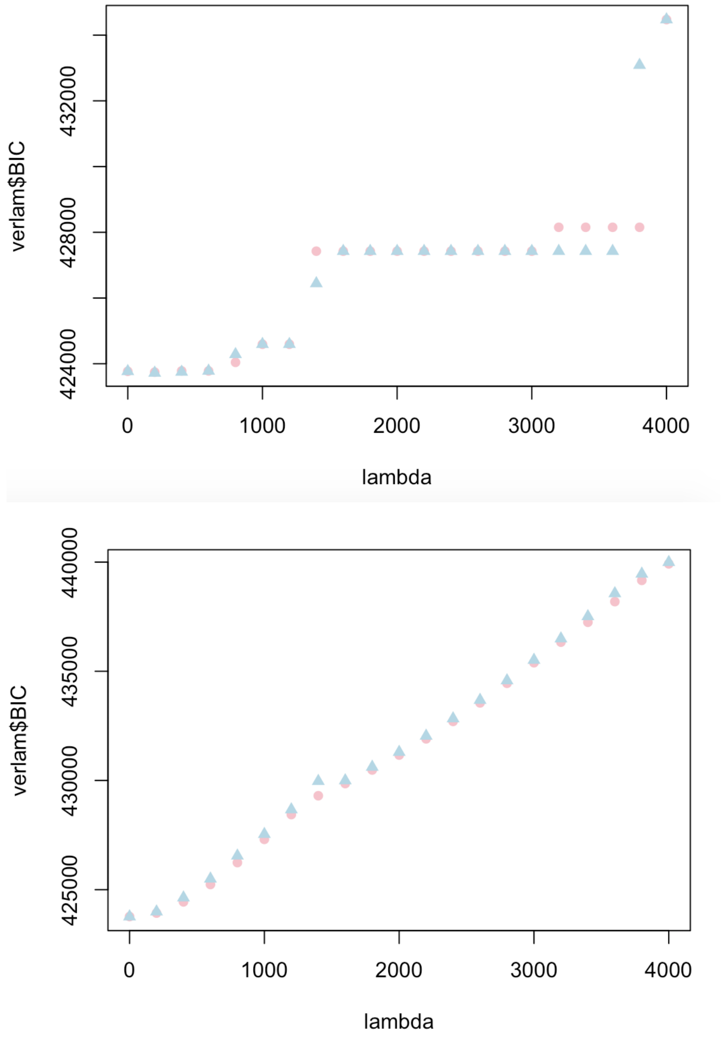

Figure 1:

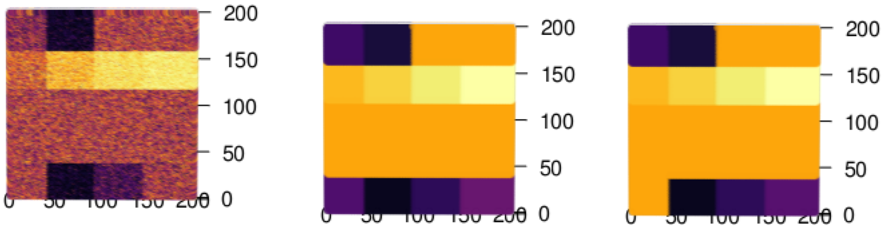

Figure 2:

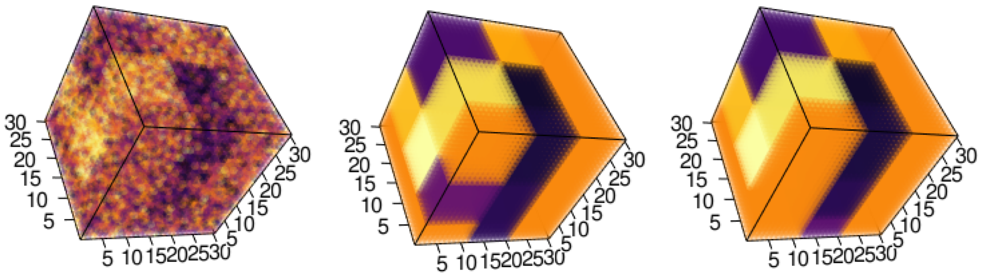

Figure 3:

A Rprofile of sparse.choosekrl()

This time I use multicore to run this function. Under the Mac/linux/Unix operation system, the speed of the code would be much faster than before. Here is the new Rprofile of this function:

```
$by.self
                  self.time self.pct total.time total.pct
#54                    0.56       56       0.56        56
<no location>          0.22       22       0.22        22
tensorsparse.R#85      0.08        8       0.08         8
#22                    0.04        4       0.04         4
#32                    0.04        4       0.04         4
#65                    0.04        4       0.04         4
#53                    0.02        2       0.02         2

$by.total
                  total.time total.pct self.time self.pct
#54                     0.56        56      0.56       56
<no location>           0.22        22      0.22       22
tensorsparse.R#85       0.08         8      0.08        8
#27                     0.08         8      0.00        0
#22                     0.04         4      0.04        4
#32                     0.04         4      0.04        4
#65                     0.04         4      0.04        4
#53                     0.02         2      0.02        2

$by.line
                  self.time self.pct total.time total.pct
#22                    0.04        4       0.04         4
#27                    0.00        0       0.08         8
#32                    0.04        4       0.04         4
#53                    0.02        2       0.02         2
#54                    0.56       56       0.56        56
#65                    0.04        4       0.04         4
<no location>          0.22       22       0.22        22
tensorsparse.R#85      0.08        8       0.08         8

$sample.interval
[1] 0.02

$sampling.time
[1] 1
```

Moreover, to speed up the functions, I change many for loops into mcapply.