# Why the likelihood becomes $-inf$ of the binary tensor logistic model

Jiaxin Hu

Email: jhu267@wisc.edu

Date: 2019.3.16

# 1 Why the likelihood becomes $-inf$

- Consider the model:

$$logit(\mathbb{E}Y) = G = U \times_1 A \times_2 B \times_3 C$$

- The formula I used to calculate the likelihood is:

$$l(U, A, B, C) = \langle Y, U \times_1 A \times_2 B \times_3 C \rangle - \sum_i \sum_j \sum_k (1 + exp(U \times_1 A \times_2 B \times_3 C))$$

- The direct reason of why the likelihood drops to $-inf$ is that the second term $\sum_i \sum_j \sum_k (1 + exp(U \times_1 A \times_2 B \times_3 C))$ becomes to $inf$, which is because the elements in estimated tensor $U \times_1 A \times_2 B \times_3 C$ have really large absolute value like $1e + 13, -1e - 13$. That's means there are something wrong with the coefficient when upgrade the factor matrices of core tensor.

# 2 Remedies

- Take upgrading $U$ as an example (the $-inf$ may happen at any step whatever upgrading core tensor or factor matrix). The $glm$ model is :

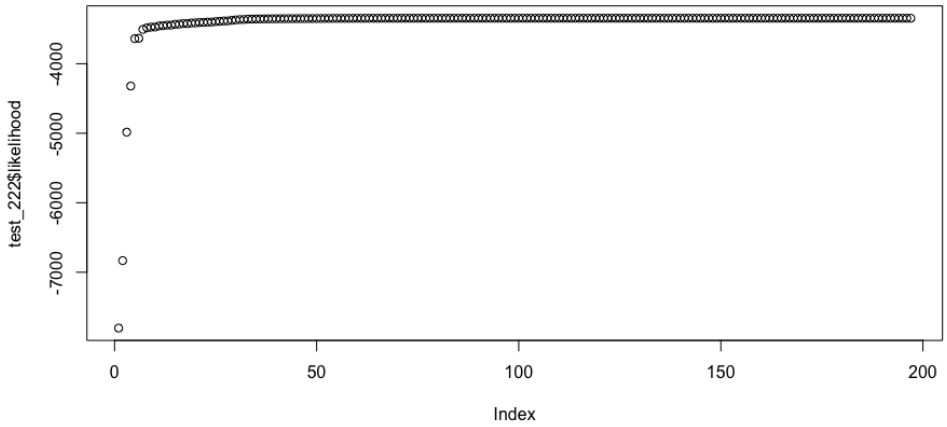
$$G^{(t)}[i', j', k'] = \sum_{ijk} U^{(t+1)}[i, j, k] A^{(t)}[i, i'] B^{(t)}[j, j'] C^{(t)}[k, k']$$

- I tried to initialize the start coefficient of $glm$ by $U^{(t)}$. I get a good result and likelihood is indeed increase than last step using this strategy. So I also implement this in upgrading factor matrix steps.

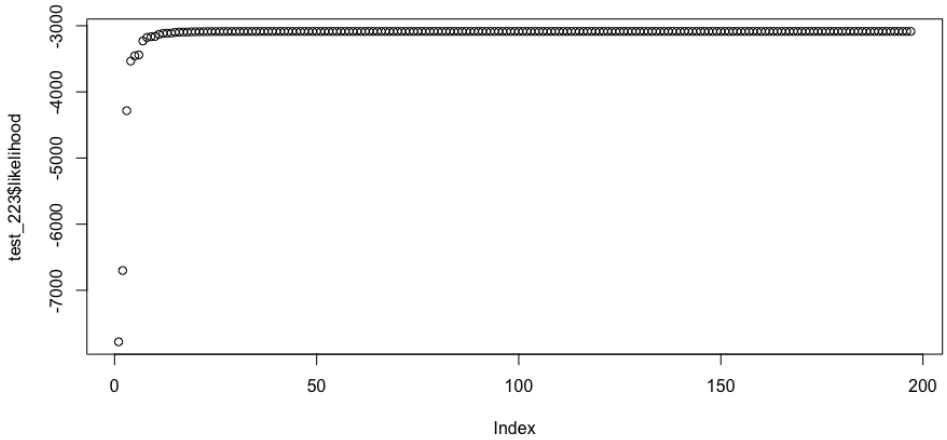
- However, if I set all the upgrading use the $glm$ with a start coefficient, there still may produce $-inf$ in some steps. So I let the function use this strategy only when the $glm$ without initialization gives a $-inf$.

- When I choose a low rank of $tucker$ like $k = c(2, 2, 2), k = c(2, 2, 3), k = (3, 3, 3)$ gives a good result (max iteration is 50).

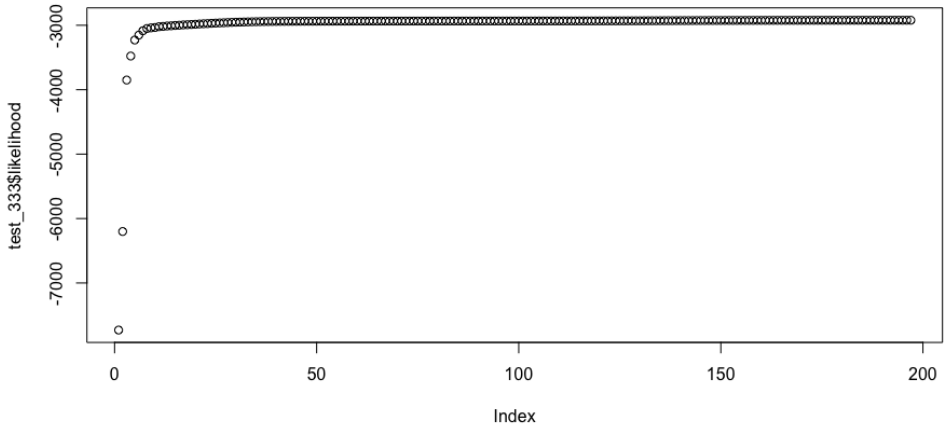
- $k = c(2, 2, 2)$**:** The likelihood stop to increase when it is $-3345.464$. And the plot is:




- $k = c(2, 2, 3)$**:** The likelihood stop to increase when it is $-3085.890$. And the plot is:




- $k = c(3, 3, 3)$**:** The likelihood stop to increase when it is $-2922.194$. And the plot is:




# 3 Still Problems...

- Although the strategy that initialize the start coefficient of *glm* works well when I choose low ranks of *tucker*, it can not fix the problem when I choose the rank higher.
- Take $k = c(3, 3, 4)$ for example. (In last report I give the result of $k = c(3, 3, 5)$, but I found that there was a typo in my code, thus the result of $k = c(3, 3, 5)$ in last report would not correct. After I corrected my code, I found when $k = c(3, 3, 5)$ there are still the same problem with $k = c(3, 3, 4)$ ).
- When $k = c(3, 3, 4)$, even though I initialize the start coefficient of *glm*, it still produces $-inf$ in just two iterations:

```
> test$likelihood
  [1] -7653.564 -5582.346 -3475.216 -3124.555 -3042.824
  [6] -2993.308      -Inf      -Inf      -Inf
```

  Then I check the step upgrading $C$ which produce the first $-inf$.

- The correlation matrix of predictor when use *glm* to upgrade $C$ is :

```
> cor(x3)
           [,1]         [,2]         [,3]        [,4]
[1,]  1.00000000 -0.07107255 -0.45134257 -0.8798396
```

```
[2,] -0.07107255  1.00000000 -0.09925495 -0.1404045
[3,] -0.45134257 -0.09925495  1.00000000  0.1723305
[4,] -0.87983960 -0.14040454  0.17233053  1.0000000
```

Noticed that the first column and the fourth column are negative correlated, which may be a reason leads to $-inf$.

- And we can find in this step we get a upgraded $C$. The correlation matrix of the new $C$ is:

```
> cor(problem$M3)
        [,1]       [,2]       [,3]       [,4]
[1,]  1.0000000 -0.9999933  0.9999945 -0.9999789
[2,] -0.9999933  1.0000000 -0.9999983  0.9999940
[3,]  0.9999945 -0.9999983  1.0000000 -0.9999923
[4,] -0.9999789  0.9999940 -0.9999923  1.0000000
```

I think the problem may happen here. And the problem may related to the choice of the core tensor's rank.