Some Evidence Theory and Checking Algorithm

Zhuoyan Xu

4/18/2019

1 Evidence theory

Consider the semi-supervised binary tensor factorization, under the scenario where treat covariate as predictors. The general model is:

$$U = logit(\mathbb{E}Y) = B \times_1 X$$
$$B = C \times_1 N_1 \times_2 N_2 \times_3 N_3$$

where Y is the given binary tensor, X is the given covariate. N_1, N_2, N_3 are the factor matrices, C is the core tensor. B is the intermediate tensor.

1.1 Conclusion 1 : Each update has unique solution, and each update increases log-likelihood

To write in a general form, we have:

$$U = logit(\mathbb{E}Y) = C \times_1 XN_1 \times_2 N_2 \times_3 N_3$$

1.1.1 Multi-process of GLM

First, we prove that the update of factor matrices or core tensor is just the multi-process of regular GLM.

For updating N_3 when fix C, N_1, N_2(update N_2 is the same) Since we have:

$$U_{(3)} = N_3 C_{(3)} [N_2 \otimes (XN_1)]^T$$

Where $U_{(3)}, C_{(3)}$ are the unfold matrices of tensor U,C through mode-3.
Recalling the matrix form GLM, which is identical to this scenario:
Consider

$$logit[E(Y))_{n*p}] = U_{n*p} = X_{n*R} \times \beta_{R*p}$$

Where $U_{n*p} = (u_1, \ldots, u_p)$, $\beta_{R*p} = (\beta_1, \ldots, \beta_p)$

We have:

$$\begin{aligned} u_1^{N*1} &= X^{n*R} \times \beta_1^{R*1} \\ u_2^{N*1} &= X^{n*R} \times \beta_2^{R*1} \\ &\vdots \\ u_p^{N*1} &= X^{n*R} \times \beta_P^{R*1} \end{aligned}$$

Then we implement regular GLM on each equation(i.e. each column of matrix β). These are multiprocessing and mutually independent GLM processes.

For updating N_1 when fix C, N_2, N_3

Since we have:

$$U_{(1)} = XN_1C_{(1)}[N_3 \otimes N_2]^T$$

Where $U_{(1)}, C_{(1)}$ are the unfold matrices of tensor U,C through mode-1.

Recalling the form on note **Semi-Supervised Binary Tensor Factorization on dnations data**:

$$U^{d_1 \times d_2 d_3} = Y^{d_1 p} W^{pr_1} G^{r_1 \times d_2 d_3}$$

we have already proved that through some vectorization of matrix, it can be written as regular GLM form :

$$Y = X\beta$$

where β is the vectorization of N_1 in this scenario.

Thus it's still GLM.

For updating C when fix N_1, N_2, N_3

Recalling the note **Unsupervised Binary Tensor Factorization**, we have already proved that through some vectorization of matrix, it can be written as regular GLM form :

$$Y = X\beta$$

where β is the vectorization of C in this scenario. Thus it's still GLM.

1.1.2 Conclusion 1 on GLM

Then we just need to show the conclusion is satisfied on the GLM, in this case logistic regression.

Consider the form:

$$logit(\mathbb{E}\tilde{Y}) = U = X\beta$$

Recalling the negative log-likehood(i.e. cross-entropy), we have:

$$\begin{aligned} Q &= -\sum_{i=1}^{n} \{\tilde{y}_i \log(\pi_i) + (1 - \tilde{y}_i)\log(1 - \pi_i)\} \\ \pi_i &= \frac{\exp(\boldsymbol{x}_i' \boldsymbol{\beta})}{1 + \exp(\boldsymbol{x}_i' \boldsymbol{\beta})} \end{aligned}$$

Where x_i are one observation. If our dimension is p, then $x_i, \beta \in \mathbb{R}^p$

Consider $\tilde{y_i} \in \{0, 1\}$, let

$$y_i = 2\tilde{y_i} - 1 \in \{-1, 1\}$$

Then:

$$\begin{aligned} Q &= -\sum_{i=1}^{n} log\left\{\frac{1}{1 + \exp(-y_i x_i' \beta)}\right\} \\ &= \sum_{i=1}^{n} log\left\{1 + \exp(-y_i x_i' \beta)\right\} \end{aligned}$$

Then we consider the optimization problem:

$$\begin{aligned} \beta^* &= \arg\min_{\beta} Q && (1) \\ &= \arg\min_{\beta} \sum_{i=1}^{n} log\left\{1 + \exp(-y_i x_i' \beta)\right\} && (2) \end{aligned}$$

Since we have:

$$\begin{aligned} \frac{\partial Q}{\partial \beta} &= -\sum_{i=1}^{n} \frac{y_i}{1 + \exp(y_i x_i' \beta)} x_i \\ \frac{\partial^2 Q}{\partial \beta^2} &= \sum_{i=1}^{n} \left[\frac{y_i}{1 + \exp(y_i x_i' \beta)}\right]^2 x_i x_i^T \end{aligned}$$

Since for $\forall\ z \in \mathbb{R}^p$, we have:

$$z^T \frac{\partial^2 Q}{\partial \beta^2} z = \sum_{i=1}^{n} \left[\frac{y_i}{1 + \exp(y_i x_i' \beta)}\right]^2 (z^T x_i)^2 \geq 0$$

The equal sign holds if and only if z=0. Then we conclude that the Hessian matrix is positive definite.

According to:

$$\nabla^2 Q(\beta) > 0 \text{ for all } \beta \in \text{dom}(Q) = \mathbb{R}^p$$

We can conclude Q is strictly convex with respect to β. Then the optimization problem (2) has unique solution.

Consider the update at t step:

$$\begin{aligned} \beta^{(t)} &= \arg\min_{\beta} Q(\beta^{(t-1)}) \\ &= \arg\min_{\beta} \sum_{i=1}^{n} log\left\{1 + \exp(-y_i x_i' \beta^{(t-1)})\right\} \end{aligned}$$

According to what we derived, each update has unique solution. And each update won't decrease the log-likelihood. If $\beta^{(t-1)}$ reach the global minimum of opbejctive function, the update will remain the same:$\beta^{(t)} = \beta^{(t-1)}$.

1.2 Conclusion 2 : The log-likelihood is invariant to orthogonalization

Suppose we have:

$$U = logit(\mathbb{E}Y) = B \times_1 X$$
$$B = C \times_1 N_1 \times_2 N_2 \times_3 N_3$$

where Y is the given binary tensor, X is the given covariate. N_1, N_2, N_3 are the factor matrices, C is the core tensor. B is the intermediate tensor.

When we conduct the orthogonalization and normalization on N_1 (N_2, N_3 are the same), such as SVD:

$$N_1 = U\Sigma V^T$$

Take $\tilde{N_1} = U$ as orthonormal version of N_1, let $P = \Sigma V^T$, then:

$$N_1 = \tilde{N_1}P$$

Let $C = \tilde{C} \times_1 P^{-1}$ Then we have:

$$\begin{aligned} B &= C \times_1 N_1 \times_2 N_2 \times_3 N_3 \\ &= \tilde{C} \times_1 P^{-1} \times_1 \tilde{N_1}P \times_2 N_2 \times_3 N_3 \\ &= \tilde{C} \times_1 \tilde{N_1} \times_2 N_2 \times_3 N_3 \end{aligned}$$

Then the log-likelihood is the same.

2 Check on unsupervised

Recalling unsupervised model:

$$logit\left\{\mathbb{E}\left[\mathcal{X}^{d_1d_2d_3}\right]\right\} = \mathcal{G}^{r_1r_2r_3} \times_1 A^{d_1r_1} \times_2 B^{d_2r_2} \times_3 C^{d_3r_3}$$

Where d_1, d_2, d_3 is dimension of tensor. The r_1, r_2, r_3 is the dimension of low rank tensor.

Let U denote $logit\left\{\mathbb{E}\left[\mathcal{X}^{d_1d_2d_3}\right]\right\}$.

Take $d_1 = d_2 = d_3 = 20$, $r_1 = r_2 = r_3 = 4$.

2.1 The MSE of ten simulations

In ten simulation(each simulation contains 25 updates). The MSE is shown below:

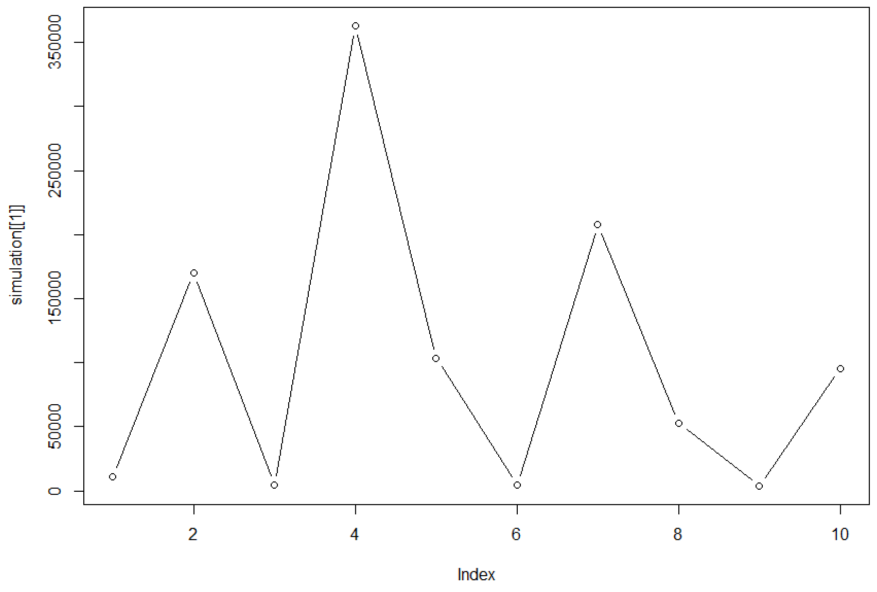


Figure 1: The MSE error of U

It still shows large difference between estimated and true U. Then I checked all the logLik, it all shows increasing pattern. Hold 4-th simulation for example:

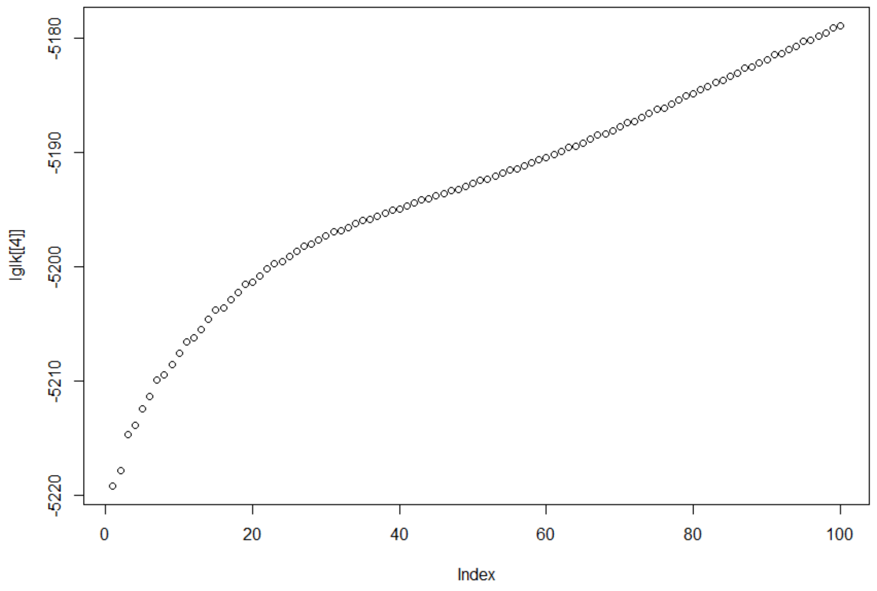
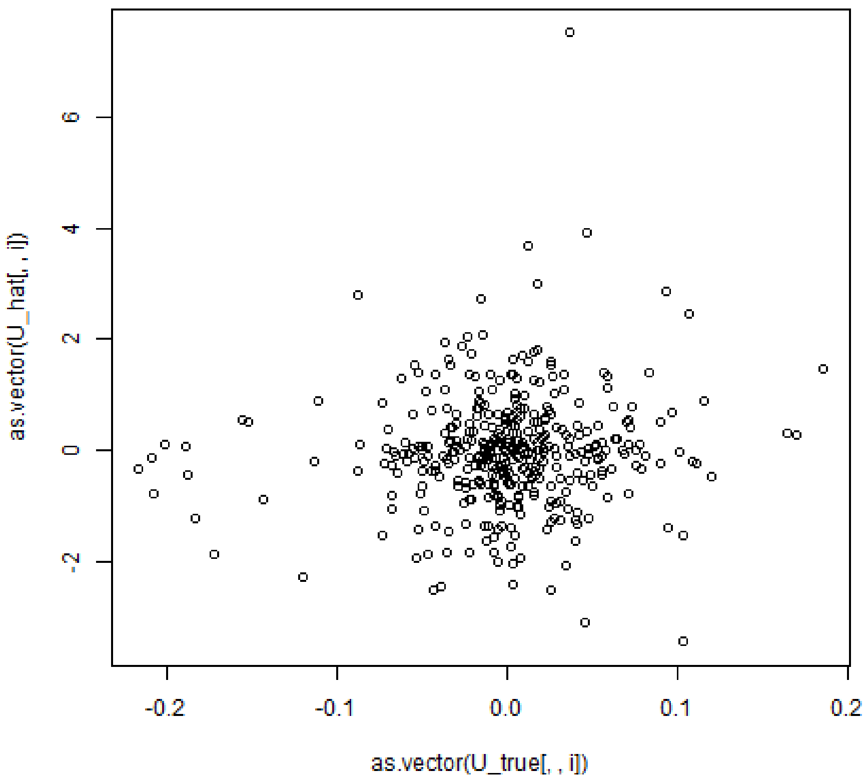


Figure 2: 4-th simulation logLik

Then, we can conclude different simulations cause different convergence rate, but it'll all converge, since it's all increasing and bounded. We can set update times to be larger to deal with this situation This is what we'll do in next subsection.

2.1.1 Check on one simulation

I set.seed(37) then check the algorithm, the logLik converge(each update is smaller than 0.005). The MSE of U is 4226.743. Then I plot one of the slices of the U and U-hat(they basically shows all the same pattern). It shows a erratic pattern. Like this:



8 .png

Figure 3: U slice

The U and U-hat tensor is shown like:

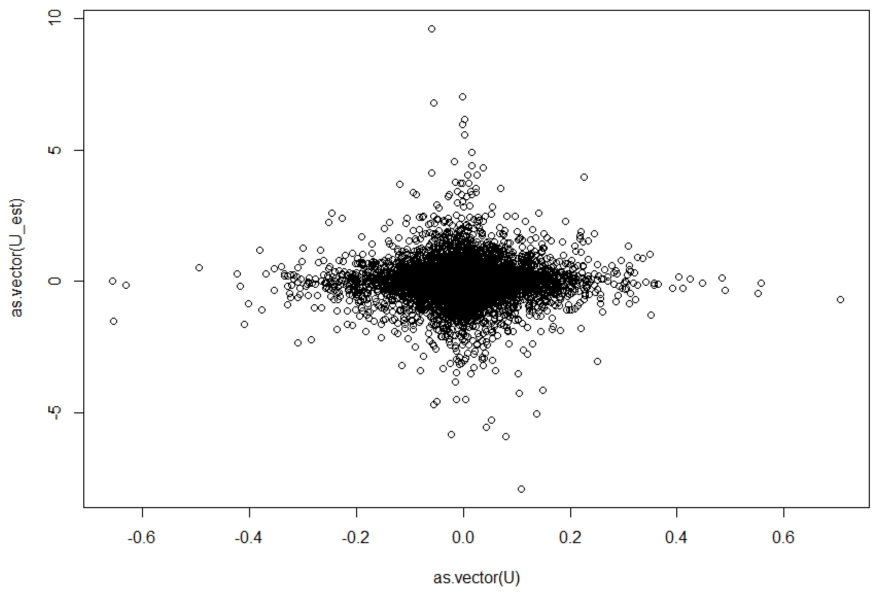


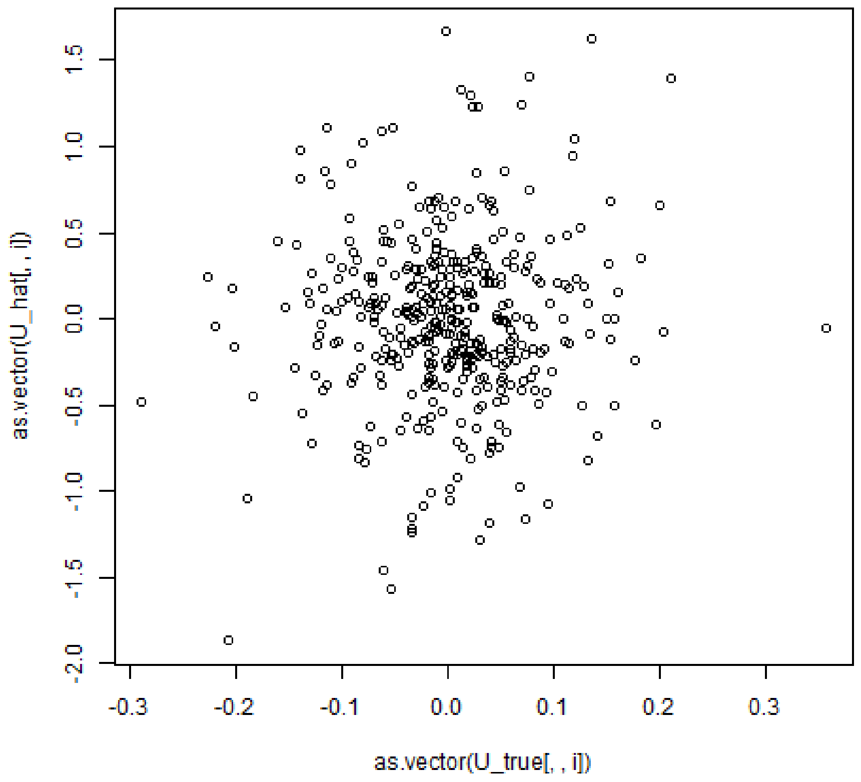
Figure 4: U tensor in real scale

It might because sometimes I think it converge, but actually it didn't.(Since the improvement for logLik for each update may not show a monotone deceasing trend when updating times increasing). Then I set updating times up to 200.

The improvement of every step of update is decreasing.It turns out when updating 60 times, the improvement of logLik converge to 1e-5. When finish updating 200 times, the improvement of logLik converge to 1e-9.

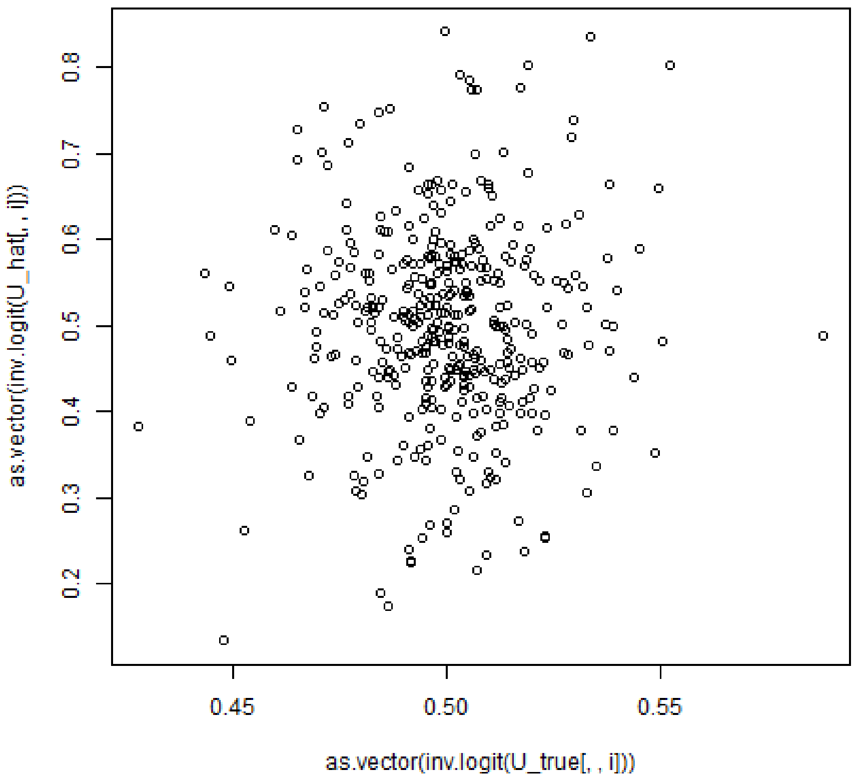
Under this scenario, the MSE is 4351.015.

Then I plot one of the slices of the U and U-hat(they basically shows all the same pattern). It still shows a erratic pattern. Like this:



8 .png

Figure 5: U mode 3 slice 8 200updates in real scale



8 .png

Figure 6: U mode 3 slice 8 200updates in logic scale

The U and U-hat tensor is shown like:

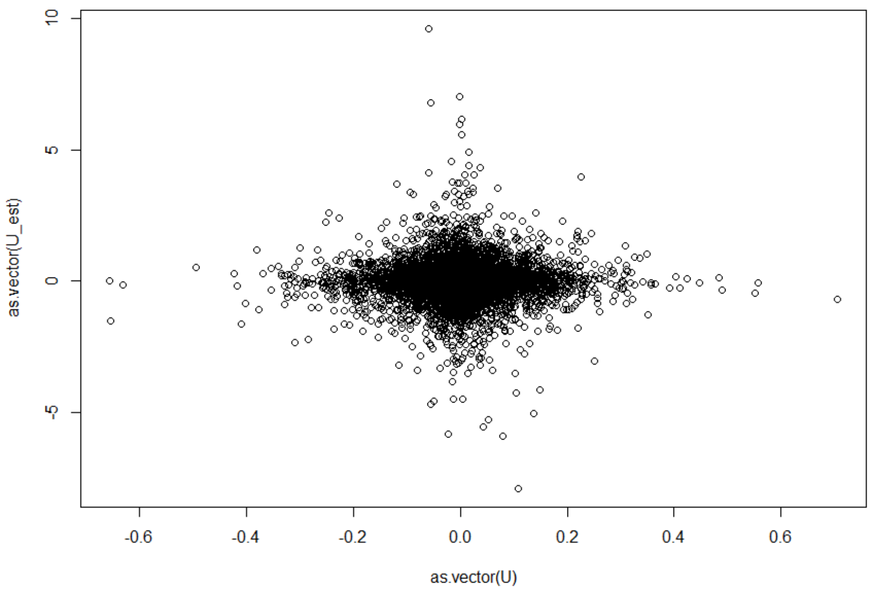


Figure 7: U tensor 200 update in real scale

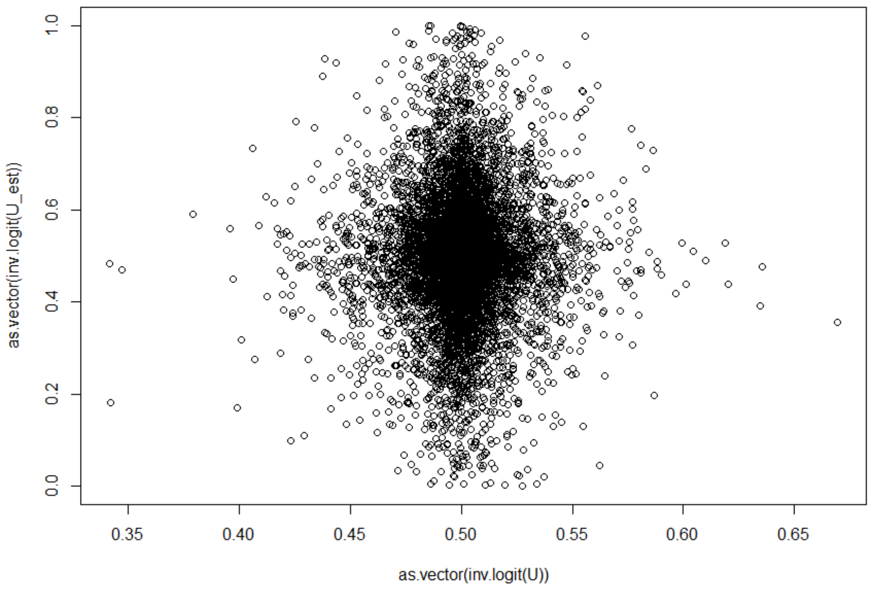


Figure 8: U tensor 200 update in logic scale

2.2 Specify initialization

When I specify the initialization, still the set.seed(37) case. Then I update 100 times. The updating is a little different. The improvement of every step of update is not monotone. It turns out when updating core tensor, the improvement of logLik is tiny, like 1e-10. when updating fantor matrices, the improvement of logLik is relatively larger, like 1e-1. The logLik is shown like:

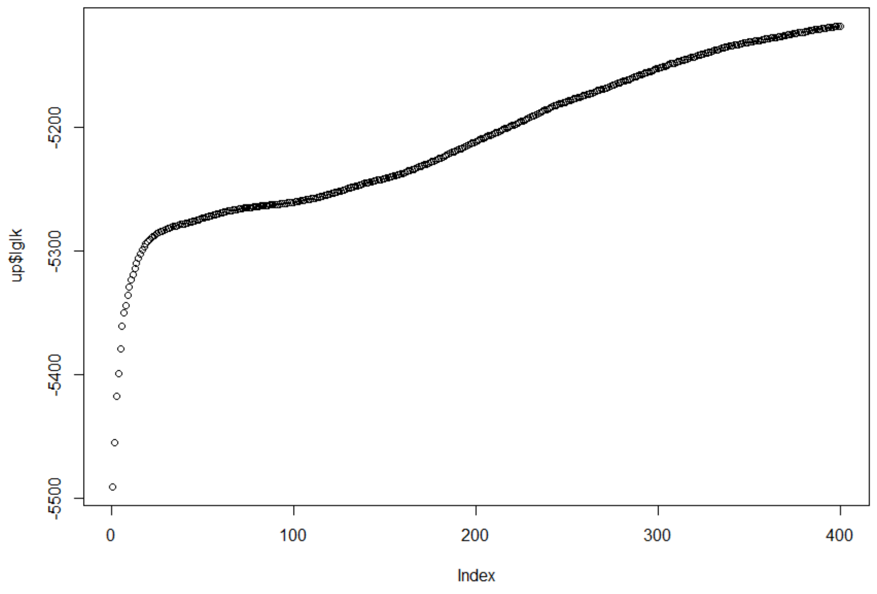
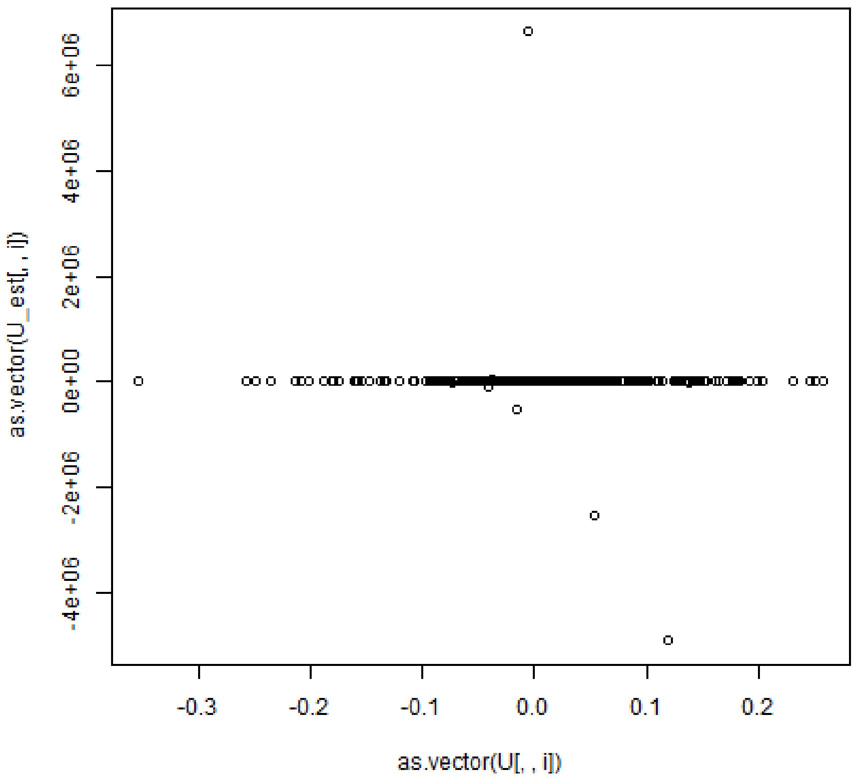


Figure 9: logLik with initialization

Under this scenario, the MSE is 1.478751e+18.

Then I plot one of the slices of the U and U-hat in real scale(they basically shows all the same pattern). Depite some very large elements, all the other are very closed to 0. Like this:



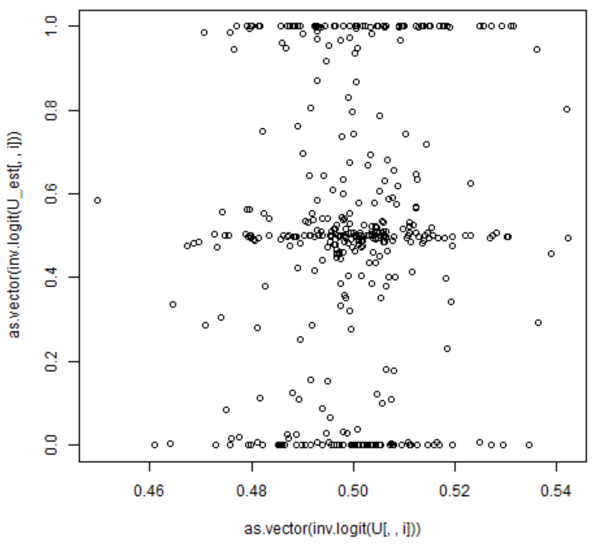
9 .png

Figure 10: U slice 9 with initialization

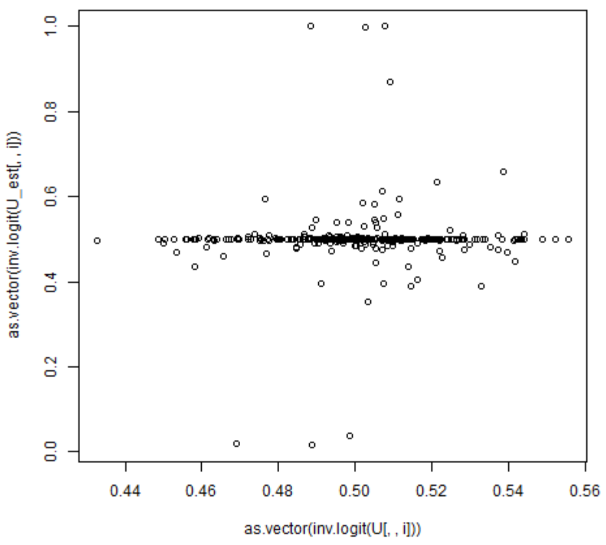
The slices of the U and U-hat in logic scale(they basically shows all the same pattern) is like these two patterns:

7 .png

13 .png



(a) U mode3 slice with ini sigmoid slice 7



(b) U mode3 slice with ini sigmoid slice 13

Figure 11: Three simple graphs

The U and U-hat tensor in the real scale is shown like:

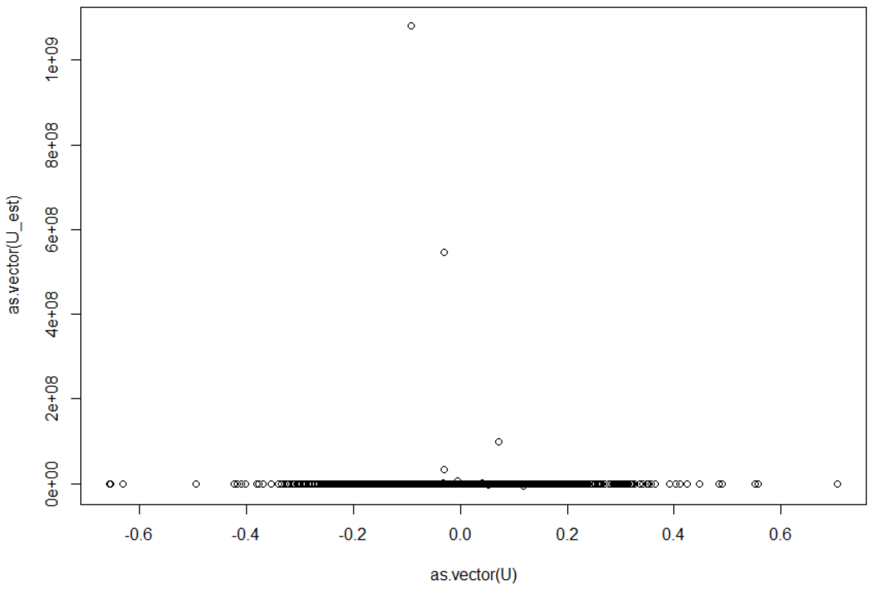


Figure 12: U tensor with initialization

The U and U-hat tensor in the logic scale is shown like:

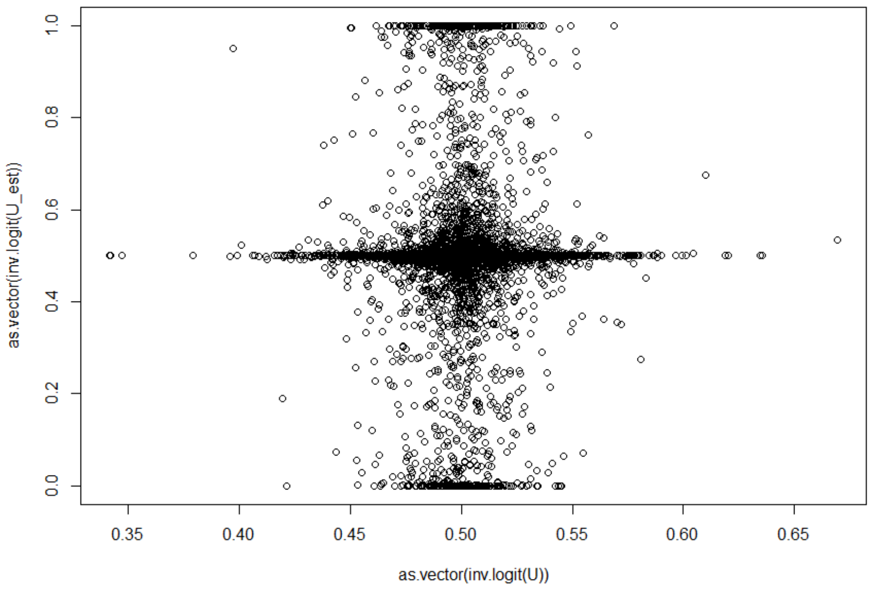
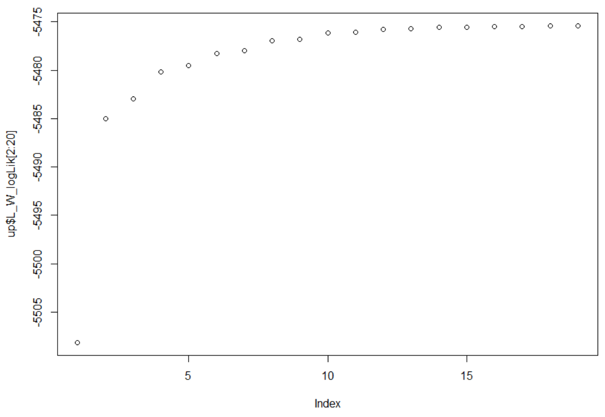


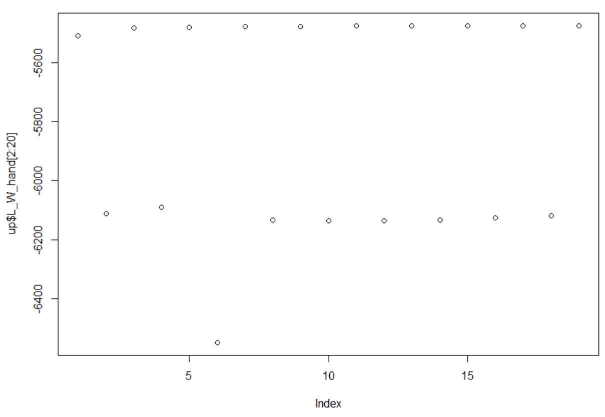
Figure 13: U tensor with initialization in logic scale

2.3 Check on semisupervised

I apply a 10 times simulation, and the logLik got by GLM function is increasing, but the logLik calculated by hand is fluctuating on updating of N_1. For easy to compare, I just compare the logLik before or after updating N_1. Like shown below:



(a) logLik got by GLM function



(b) logLik calculated by hand3

Figure 14: Three simple graphs

The number is shown like:

```
    up$L_W_logLik[2:20]  got by function
 [1] -5508.127 -5484.996 -5482.932 -5480.212 -5479.504 -5478.270 -5477.969 -5476.977
 [9] -5476.787 -5476.165 -5476.056 -5475.770 -5475.721 -5475.585 -5475.564 -5475.487
[17] -5475.477 -5475.429 -5475.423
   up$L_W_hand[2:20]  got by direct calculating
 [1] -5508.127 -6111.753 -5482.932 -6091.340 -5479.504 -6548.824 -5477.969 -6134.677
 [9] -5476.787 -6137.256 -5476.056 -6137.266 -5475.721 -6133.482 -5475.564 -6127.016
[17] -5475.477 -6119.058 -5475.423
```