Gray-scale image data analysis2

Chanwoo Lee, March 11, 2021

1 Algorithm improvement

There were three main problems in our algorithm `signT.R` that makes our output sub-optimal.

1. `signT.R` cannot take matrix input:

   Even though we can still analyze matrix data making array data, this issue makes our algorithm slower. Therefore, I have constructed a new algorithm that can take matrix value `signM.R`. I will incorporate `signT.R` and `signM.R` later.

2. `signT.R` gradient is not exactly right:

   While our algorithm had cost function based on mean of observed cost, $\frac{1}{|\Omega|}\sum_{\omega\in\Omega}|\mathcal{Y}(\omega)-\pi|_F\left((\text{sign}(\mathcal{Y}(\omega)-\pi)B(\omega)\right)$, the gradient function was based on sum of observed cost, $\sum_{\omega\in\Omega}|\mathcal{Y}(\omega)-\pi|_F\left((\text{sign}(\mathcal{Y}(\omega)-\pi)B(\omega)\right)$. This is okay when matrix or tensor dimension is small because it gives us right direction. However, when the dimension is high, the gradient value is too big to go next step, which causes algorithm to stop after one gradient evaluation. Therefore, I change the gradient function scaling by the number of observed values.

3. `signT.R` cannot iterate well with warm start option:

   Our algorithm uses option `start = linear` which uses output of `fit_continuous`. Previous condition for stopping iteration includes `binary_obj[iter]>0.01)`. However, this sometimes makes our algorithm give us output from `fit_continuous` not from `Alt`. This problem is critical because the algorithm is not actually optimizing our weighted loss function but optimizing Frobenius loss function. Therefore, I deleted `binary_obj[iter]>0.01)` condition in the while loop.

Handling those issues, `signM.R` gives us much better estimation than previous result as in Figure 1.

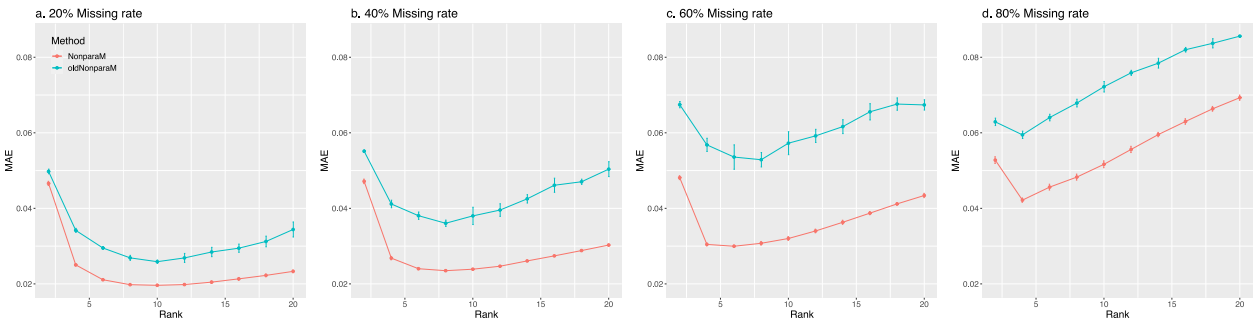


Figure 1: Estimation error versus rank under different missing rates. Panels (a)-(d) correspond to missing rate $20\%, 40\%, 60\%$, and 80%, respectively. Error bar represents the standard error over 5 repeated image recovery under the same conditions.

2 Hot air balloons image comparison

Figure 1 shows the mean absolute error versus rank under the different missing rates $20\%, 40\%, 60\%$, and 80%. I randomly sampled the entries of image matrix according to the missing rate and performed matrix completion based on hard, soft impute methods and NonparaM method with rank $r = 2, , 4, \ldots, 20$ and $H = 20$. I repeated this analysis 5 times to check the stability of analysis. Soft impute method needs the penalty parameter λ as an input. I choose the the penalty parameter λ that minimizes MAE in the test dataset in each missing rate. For example, λ is choosen to be 0.26,0.21,0.21, and 0.16 respectively for missing rates $20\%, 40\%, 60\%$, and 80% in order. Figure 3 shows that out method performs the best across all cases.

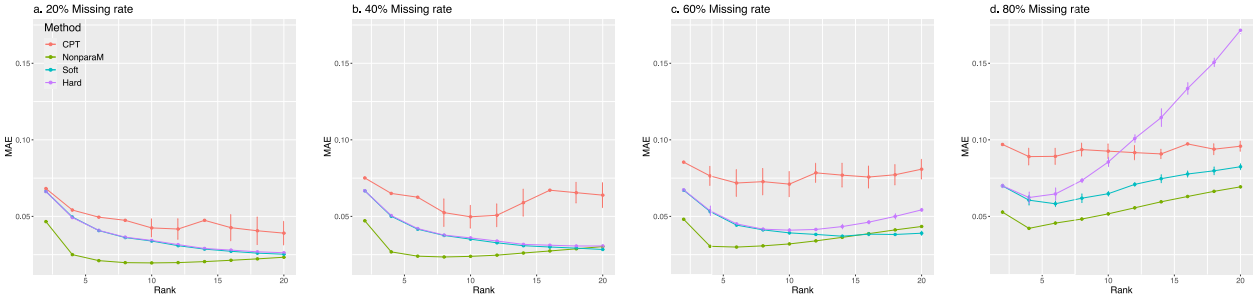


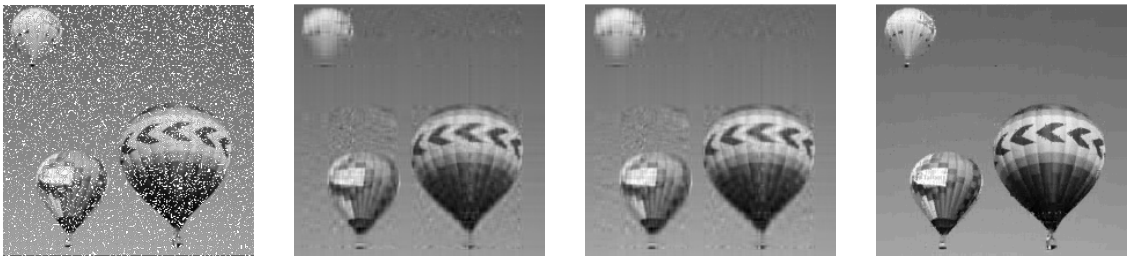
Figure 2: Estimation error versus rank under different missing rates. Panels (a)-(d) correspond to missing rate $20\%, 40\%, 60\%$, and 80%, respectively. Error bar represents the standard error over 5 repeated image recovery under the same conditions. Soft impute uses the best penalty parameter λ that minimizes MAE in the test dataset.

Next, I visualize the output of image matrix completion based on three methods (hard, soft, nonparaM). I set the best rank r based on the result in Figure 1. The penalty parameter in soft impute method is chosen based on performance in the test dataset. Table 1 summarizes hyper parameters used for matrix completion. We can see that our method does not require high rank for having the best performance in estimation. Small rank is enough even if true rank of image matrix might not be small. This is because our analysis is based on sign function and uses low-sign rank. This shows the our method's benefit of using sign function in analysis.

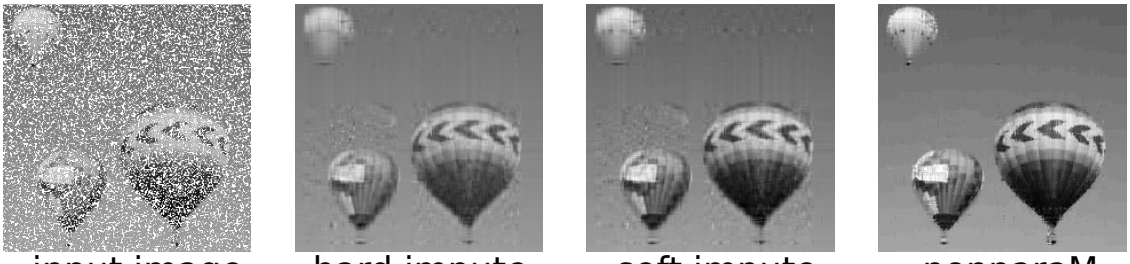
Rank	20% missing	40% missing	60% missing	80% missing
NonparaM	10	8	6	4
Soft impute	20 ($\lambda = 0.26$)	20 ($\lambda = 0.21$)	14 ($\lambda = 0.21$)	6 ($\lambda = 0.16$)
Hard impute	20	18	10	4

Table 1

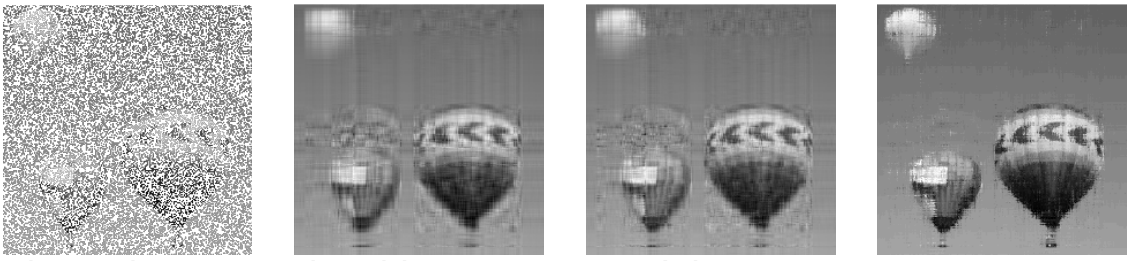
Figure 3 shows the image with missing entries, and recovered images from three different methods. We see that our method has much clearer recovered image and closer to the original image in all missing rates over two alternative methods.



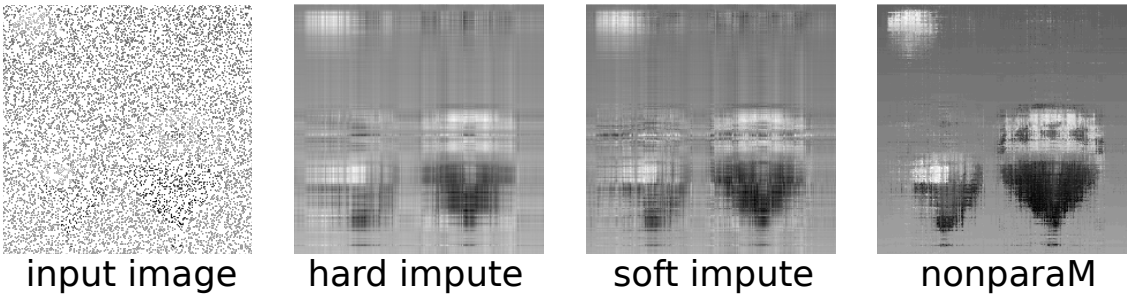
(a) When missing rate is 20%.



(b) When missing rate is 40%.



(c) When missing rate is 60%.



(d) When missing rate is 80%.

Figure 3: Performance of image recovery of our nonparametric regression method (NonparaM), soft impute, and hard impute methods depending on different missing rates.