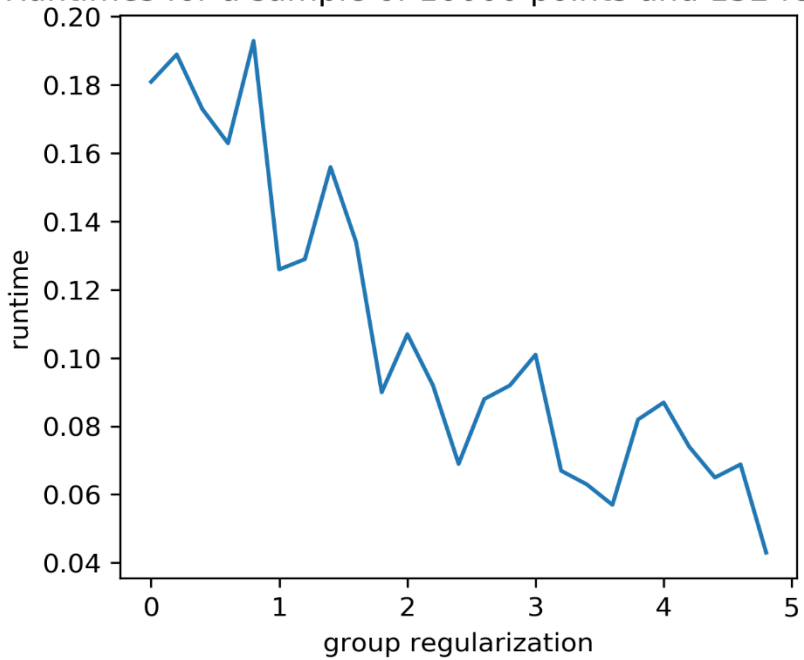
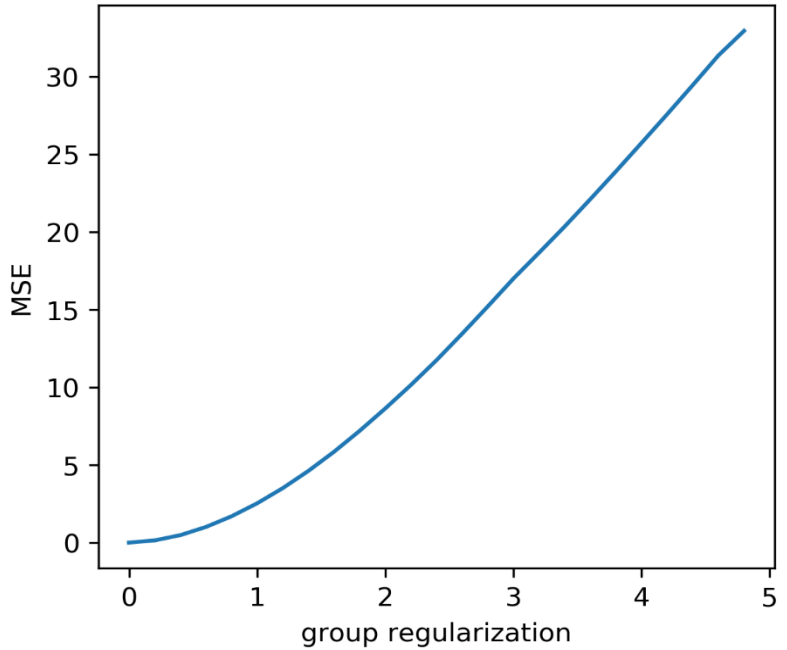
Group lasso: Wrong vs correct groups

Methods

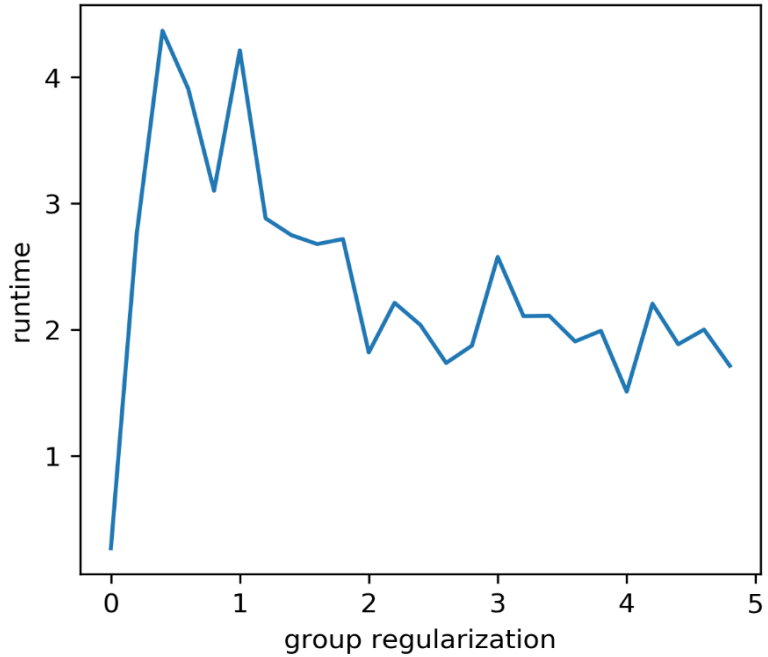
- The dataset was simulated according to the instructions in the assignment:
  - Standard deviation of coefficients (β) and the data matrix (X) was set to 1.
  - γ was set to 15 which is equivalent to the SNR
  - For correlated data a block covariance matrix was used to draw the samples from the multivariate normal distribution
- The package used was GroupLasso in python
  - The number of samples was set to 1000 for the analyses.
  - The number of groups and the group sizes were set to 12 and 11 respectively.
  - Group regularization was set to 1 unless otherwise stated
  - Number of iterations of the group lasso algorithm was set to 10000

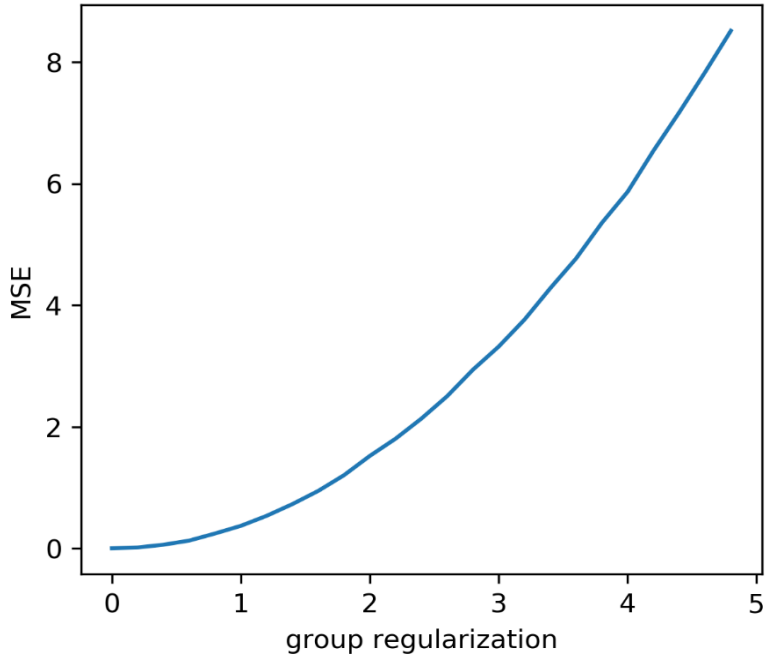
Runtimes for a sample of 10000 points and 132 features





Runtimes for a sample of 100 points and 2496 features



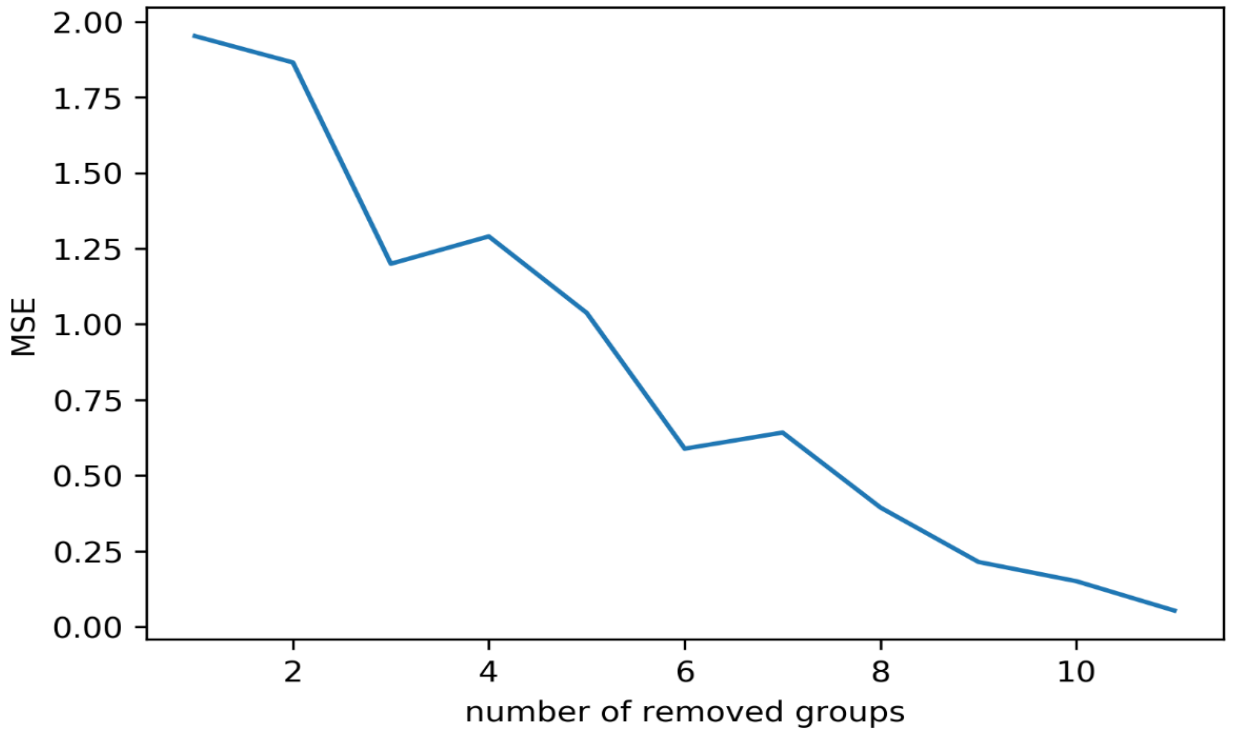


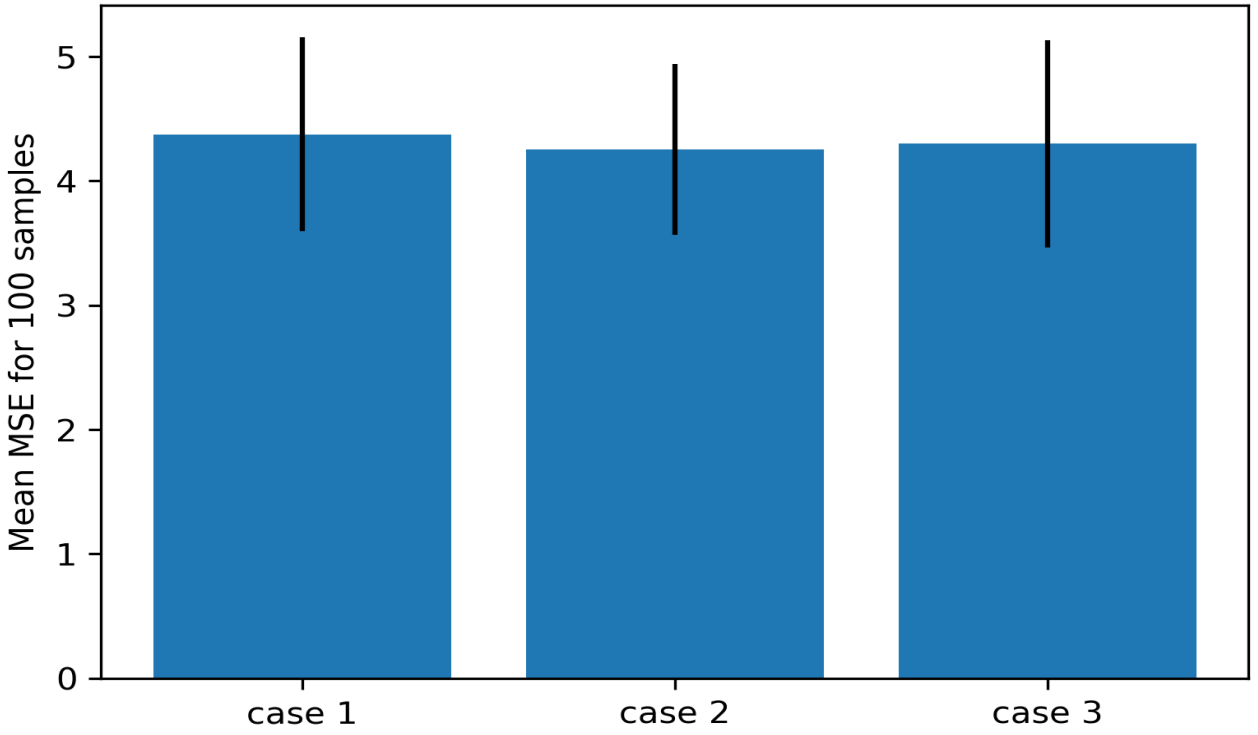
Comparison of runtime and MSE for two different scenarios: n>p and p>n. More group regularization leads to sparser groups and thus is more time-efficient especially in the n>p case however it also impacts the error to a greater extent compared to the p>n case. Based on these two graphs group regularization = 1 was chosen for all analyses in this summary.

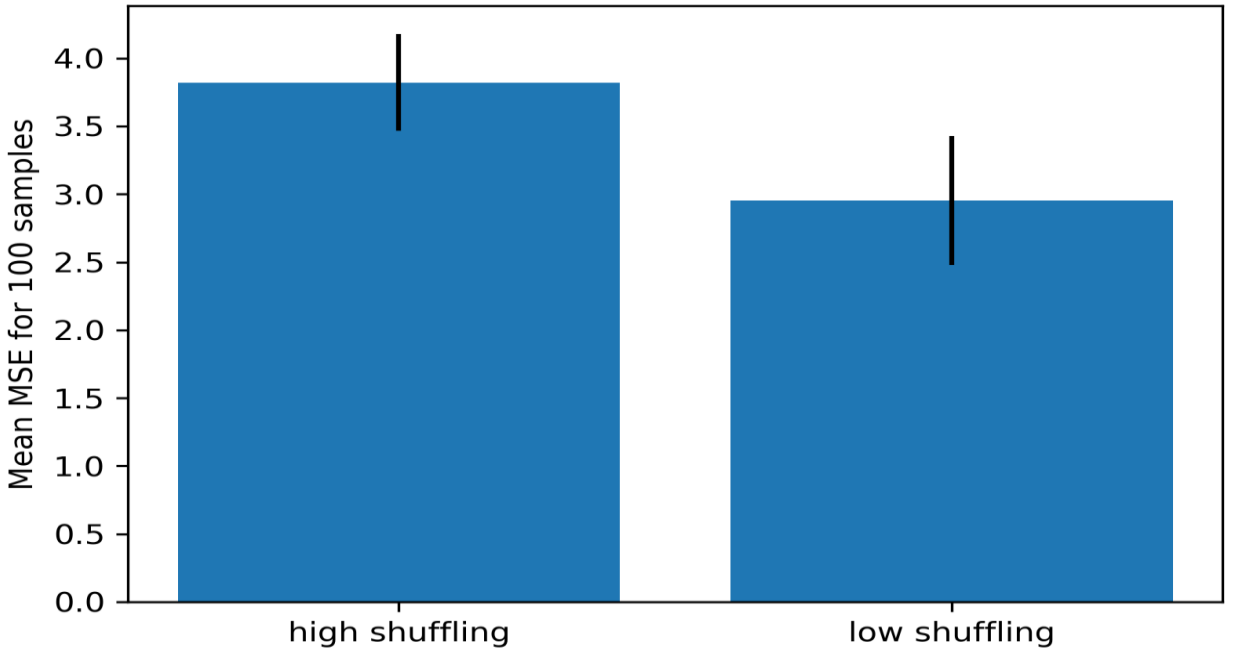
Effect of Different Groupings

I defined a function to somewhat automate different groupings:

- Using same groupings but with different order ([1 1 3 3 2 2] instead of [1 1 2 2 3 3] for example) did not significantly affect the MSE ($p=0.7$, student's t-test over 100 samples)
- However when the numbers of repeats are changed ([1 1 1 1 2 2 3 3 3] instead of [1 1 1 2 2 2 3 3 3]) or in some cases one or more groups are entirely removed in this scenario then the difference in MSE is highly significant and worse for the wrong grouping ($p<0.0001$, Welch' t-test). The order is preserved in this case.
- When different orders of repeat segments are used ([1 2 2 3 1 2 2 3 1] instead of [1 1 1 2 2 2 3 3 3]) the results are again highly statistically different ($p = 0$, I used some different tests all with the same results). The original order is preserved
- When the same scenario as above is used but this time the order is changed to some extent this scheme deteriorates performance similar to the above case with comparable measures.

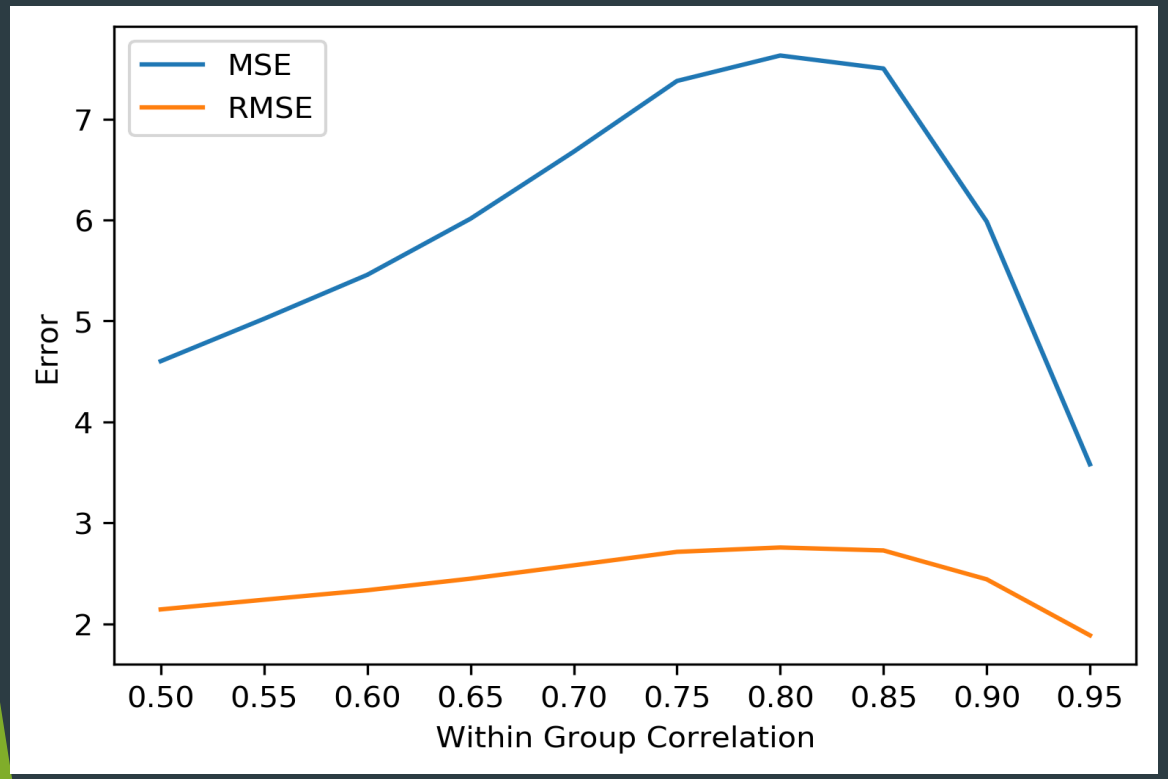


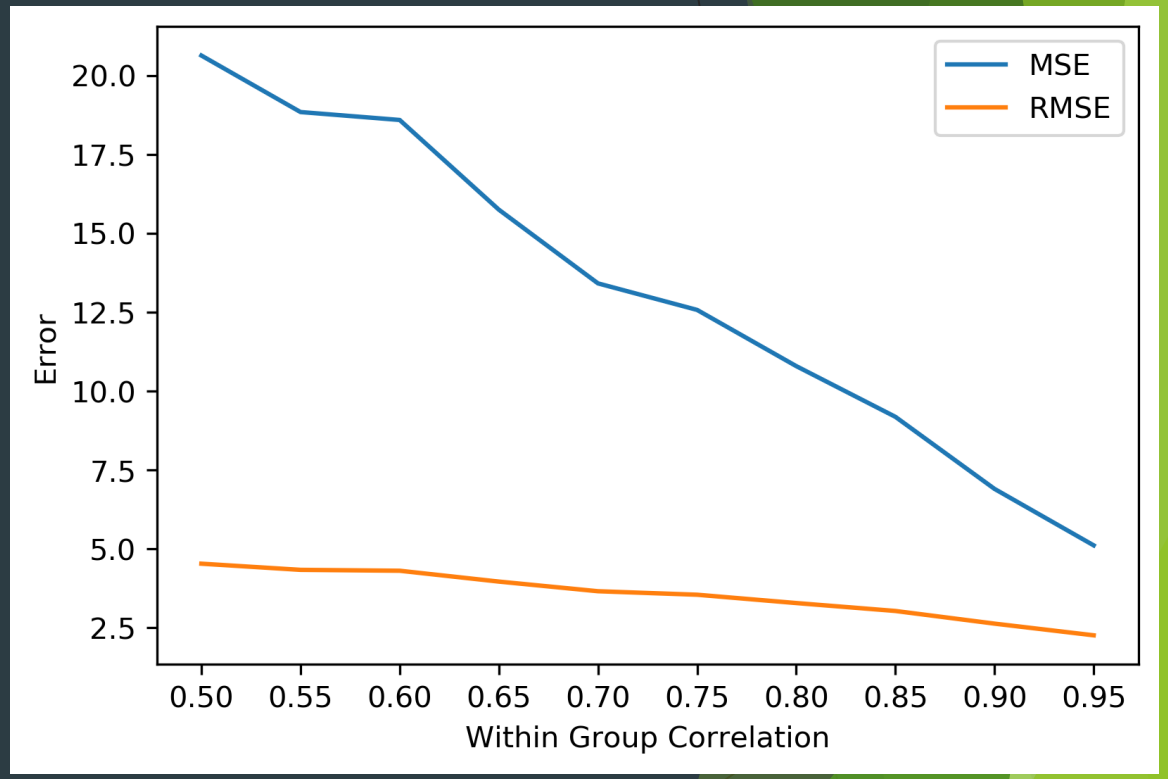




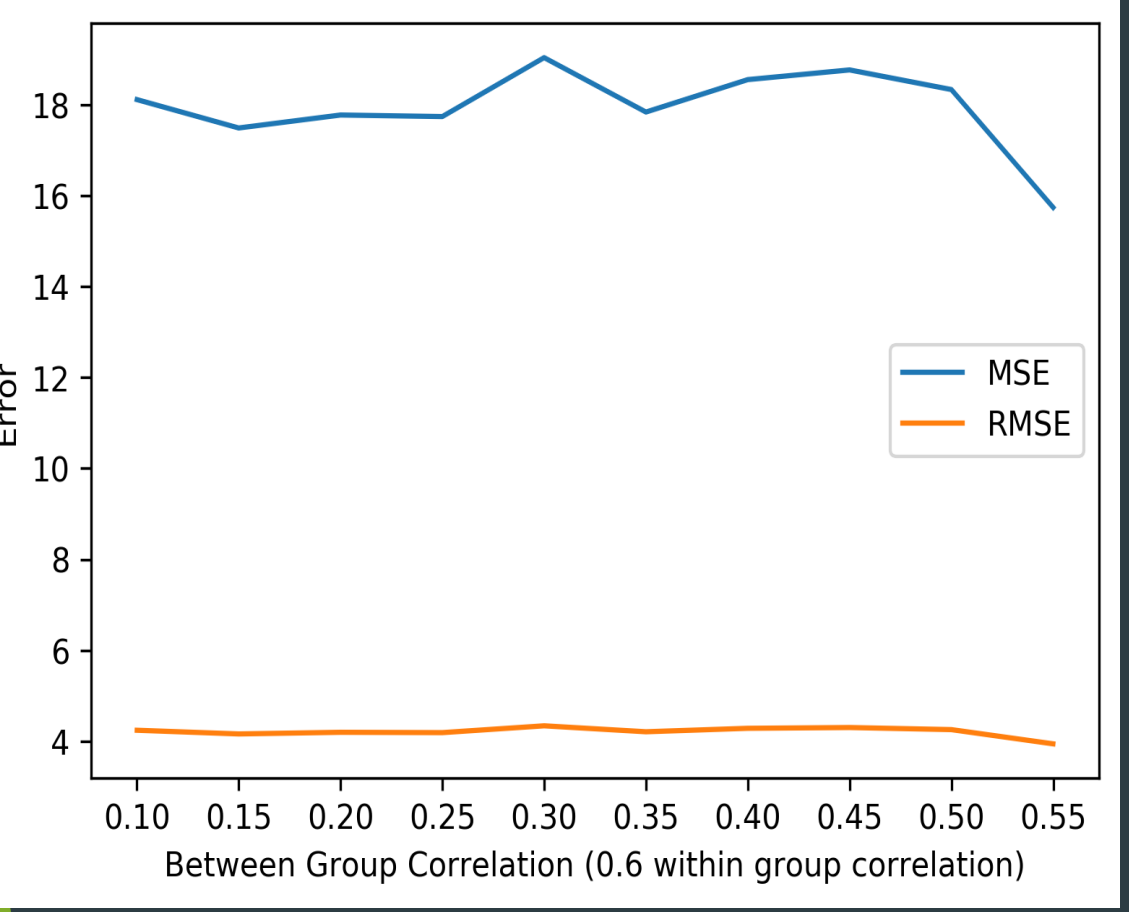
The figure to the upper left shows that as we remove the groups from list of groups the MSE becomes lower. The figure to the right corresponds to the last bullet point in the previous page the 3 cases are: [0,2,1,3,4,5,6,7,8,9,10,11],[0,1,5,3,4,2,7,6,8,9,10,11], [8,1,10,11,4,5,6,7,0,9,2,3] with 2,4, and 6 swaps in the order of groups respectively and then they are assigned different numbers of repeats (based on the scheme described in the 3rd bullet point in the previous page. The figure to the lower left corresponds to total group shuffling across the whole array of groups. High shuffling means that more members (more than half) were shuffled and low shuffling is the opposite case.

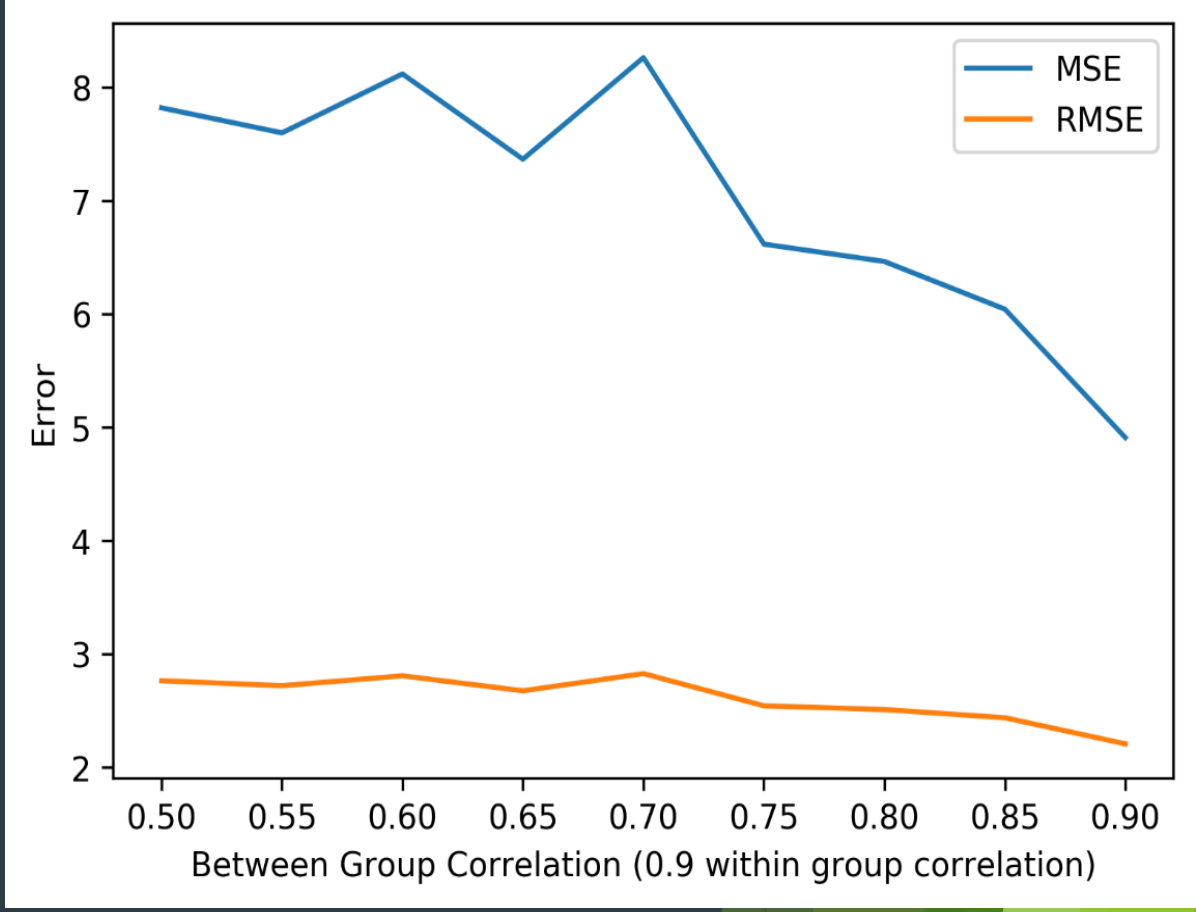
Correlated Samples





The left figure corresponds to group regularization 1 and the right figure to group regularization 5.


Error
18
16
14
12
10
8
6
4
MSE
RMSE
0.10
0.15
0.20
0.25
0.30
0.35
0.40
0.45
0.50
0.55
Between Group Correlation (0.6 within group correlation)


Error
8
7
6
5
4
3
2
MSE
RMSE
0.50
0.55
0.60
0.65
0.70
0.75
0.80
0.85
0.90
Between Group Correlation (0.9 within group correlation)

Discussion

Let us recap the definition of a sparse group lasso regularised machine learning algorithm. Consdier the unregularised loss function $L(\beta; \mathbf{X}, \mathbf{y})$, where β is the mode coefficients, $\mathbf{X}$ is the data matrix and $\mathbf{y}$ is the target vector (or matrix in the case of multiple regression/classification algorithms). Furthermore, we assume that $\beta = \left[\beta_1^T, \ldots, \beta_G^T\right]^T$ and that $\mathbf{X} = \left[\mathbf{X}_1^T, \ldots, \mathbf{X}_G^T\right]^T$, where β_g and $\mathbf{X}_g$ is the coefficients and data matrices corresponding to covariate group g. In this case, we define the group lasso regularised loss function as

$$L(\beta; \mathbf{X}, \mathbf{y}) + \lambda_1 ||\beta||_1 + \lambda_2 \sum_{g \in \mathcal{G}} \sqrt{d_g} ||\beta||_2$$

where λ_1 is the parameter-wise regularisation penalty, λ_2 is the group-wise regularisation penalty, $\beta_g \in \mathbf{d_g}$ and $\mathcal{G}$ is the set of all groups.

- We see that the group lasso is robust against wrong group assignments given that the order of the data is not changed. This is even true when the data are correlated. It seems that for group lasso the actual groups themselves do not matter as long as they preserve the original index span (if group 0 was the first 10 groups then if group 7 is replaced with 0, group 7 should also occupy the first 10 groups). From the definition of the loss function given above (obtained from GroupLasso documentation) we see that the actual grouping does not matter since they are summed and multiplied by the norm of the coefficients.
- However based on this loss function if some groups are more present than others then the loss function is more biased towards them (the $\sqrt{d}$ term) even with the regularization factor. On the other hand we are still using more features than half let's say and the probability that nonzero coefficients get multiplied by each other is higher in the last term if we increase the length of some groups with respect to others. So these two factors add together to bias the evaluation to those groups with more members and thus bias the learning.

Discussion (continued)

- For the case that we remove groups one by one we can see from the definition of the loss function that with fewer groups the last term should decrease as it is a sum over all groups. Also fewer groups means that a lower number of features are zero and thus we get lower MSE. So lowering the number of groups acts like making the group lasso denser. This result persists even with correlated data.
- Random shuffling of groups puts unrelated coefficients beside each other and the more the shuffling the worse the algorithm performs.
- Shuffling and different repeats disorganizes the groups even more than just shuffling and thus is the worst case. All of these cases would be worse for correlated data as correlated data are even more sensitive to coefficient groupings
- Thus group lasso is robust against wrong group assignments and removal of groups as long as the original repeating pattern of the groups is not changed.

Discussion on Correlations

- Group lasso supposedly handles group correlations better and one of the motivations behind its design was this improvement over lasso. We can see this from the two graphs is slide 6. However, we see an increase in MSE up to within group correlation of 0.8 when group regularization is equal to 1. This group regularization implies a denser group lasso and perhaps up to a certain correlation it becomes difficult for the dense lasso guess the interdependencies and after a certain threshold, which is around 0.85 correlation ,here the data become sufficiently correlated for the dense lasso to be able to reduce its dimensionality effectively (sort of like an 'aha'-moment).
- However, sparser groups (as shown in the right figure of slide 6) are less prone to becoming confused by the high dimensional dependencies present in these correlated data, although in general they perform worse they steadily show improvement with higher correlations and are thus suitable for high dimensional big data.
- The Group Lasso also seems to capture between group correlation as shown in slide 7 (at least for highly correlated data) although its scope seems to be more limited than the within_group_correlation. Perhaps the between group correlations become to complex even for the group lasso to handle them effectively.