Evaluating LASSO Performance in High-Dimensional Data: A Comparison Between Pixel and Frequency Domains

Siyang Ren, Nichole E. Carlson, William L. Lippitt, Yue Wang

Notes

Everything surrounded by [] are my thoughts/questions/to-dos.

- William comments in blue underline
- Repo organization: code, reports, and results directories should subdirectories of ESFGSP_paper, not of ESFGSP_paper/Simulations. Please reorganize the repo. This rearrangement, while arbitrary when considering only one repo, is consistent with the standard structure of most CIDA repos and aids other team members navigate your repos.
- I've made two changes to the header you might want to look at. The header is the section of the tex file that comes before `\begin{document}`.
  - I changed your figure path to a relative path that should work on everyone's computer. While it *is not* appropriate to use relative paths in code files (e.g. R, python, Rmarkdown, etc) because you never know what someone's working directory might be, it *is* appropriate in tex files! For tex files, relative paths will always be executed as relative to the directory in which the tex file is contained. So long as the tex file and the figures are all in a repo, this allows relative paths to function on all computers with a local copy of the repo.
  - I added some custom commands/functions, one of which allowed me to make comments in blue underline. While that particular custom command may not be of interest to you, the capacity to make a custom command in Latex might be. The line
    `\newcommand{\ed}[1]{{\color{blue}{\uline {#1}}}}`
    defines a new command `\ed{}` which takes 1 input and outputs it in blue underline. In mathematical reports, I might make the command `\E{}{}`
    `\newcommand{\E}[2]{\text{E}_{#1}\left[#2\right]}`
    which allows me to render expectations nicely with automatically resizing brackets:

$$\text{E}\left[\sum_{i=1}^{n} X_i^2\right]; \qquad \text{E}_M\left[X_1\right]$$

- I've started a bibliography file for you. If you look in the reports directory, you'll see a references.bib file. If you look at the end of this document, you'll see that this bibliography has been read in. When you look up a reference in google.scholar.com, there is a citation link underneath. At the bottom of the example citations, there is a link to a bibtex citation. This can be pasted into the references.bib file, and then citations to that reference can be made in document. If you look to the place in the tex file where you made reference to using Murakami's notation, I have added a citation to the corresponding document. Compiling the references in the tex file is a slightly different process than the usual compiling of the text document into a pdf when you make edits. It can be specific to the tex editor you use (I use texmaker), but the process often looks as follow: compile as you usually do (the tex file will find what references it needs here). Then, do a BibTex compile by, in the file bar, clicking on Tools -¿ BibTex (this makes the bibliography information available). The compile as you usually do twice (this renders both the references and the citations). If needed, google latex bibliographies or feel free to ask me for help in person.

Introduction

This study evaluates the performance of LASSO models in high-dimensional feature selection by comparing results from two different data representations: the original pixel space and a frequency space derived from eigen decomposition using Moran's eigenvectors. The pixel space corresponds to the raw data where each feature is a pixel, while the frequency space transforms these features into frequency components.[1]

We investigate whether fitting LASSO in the frequency space improves feature selection and prediction accuracy compared to the pixel space. Two simulation scenarios are used: one assuming sparsity in the pixel space and the other assuming sparsity in the frequency space. By comparing the results, we aim to determine which representation provides better performance for LASSO in high-dimensional datasets.

Methods

[2]

[3]

[1] Phrasing is a bit imprecise here. Frequency space doesn't do a transformation, it is the result of a transformation. "...while the frequency space is obtained from a transformation of pixel space into spatially independent frequencies."

[2] Some fussy people consider it bad form to have two sections headers one after the other without any text between them, as you do here. I personally don't care, certainly not for an internal report, but it's something to keep in mind as we move towards a manuscript. Journal editors are fussy.

[3] This methods section is a bit jumbled. When writing a report like this, it's good to imagine that a new student is joining the team and is going to learn about the project by reading your report. What information might they need to know? What order should information be presented in? It takes a lot of time and experience to learn how to structure reports well in this sense, and this particular report is harder because you need to talk about theory, simulations, and analyses instead of just analyses. Here's my recommendation to get you started. In the methods section, have 4 subsections: Frequency space,

Data Simulation

Let $\mathbf{x}$ be a column vector representing the pixel values of a single observation, where the total number of pixels is $n = 256$. The covariance matrix $\mathbf{C}$ follows an exponential correlation structure:

$$\mathbf{C}_{ij} = -\exp(\mathrm{dist}(i,j)),$$

where $\mathrm{dist}(i,j)$ is the distance between pixels i and j on a 16×16 grid.[4] To center the data, we use the centering matrix $\mathbf{M} = \mathbf{I} - \mathbf{1}\mathbf{1}'/n$.[5] The decomposition[6] of $\mathbf{MCM}$ is given by:

$$\mathbf{MCM} = \mathbf{E}_{\text{full}} \boldsymbol{\Lambda}_{\text{full}} \mathbf{E}'_{\text{full}},$$

where $\mathbf{E}_{\text{full}}$ represents the eigenvectors (is the matrix of eigenvectors), and $\boldsymbol{\Lambda}_{\text{full}}$ the diagonal matrix of eigenvalues [The notations above come from reference: [1]].[7] The transformation of the pixel values $\mathbf{x}$ into the frequency space is then:[8]

$$\mathbf{x}_{\text{freq}} = \mathbf{E}^T \mathbf{x}.$$

Here, the covariance matrix of $\mathbf{x}_{\text{freq}}$, $\mathbf{E}^T\mathbf{CE}$, is expected to be diagonal [this requires verification of $\mathbf{E}$'s orthogonality].[9]

For each simulation, $\mathbf{X}$ represents the matrix of observations, with each row corresponding to one of the 1000 total observations.[10] The transformation of $\mathbf{X}$ into the frequency domain is:

$$\mathbf{X}_{\text{freq}} = \mathbf{XE}.$$

We define the coefficient vectors in both spaces as follows:[11] $\boldsymbol{\beta}$ for the pixel space and $\mathbf{b}$

Models, Simulations, and Analyses. In Frequency space, go over the Moran eigendecomposition and how you transform an image from pixel space to frequency space and back. In Models, briefly introduce the LASSO model and how applying it to pixel or frequency data will change whether you assume sparse signal in pixels or sparse signal in frequencies. Make sure you give enough info that in the Simulations section, you don't need to introduce any new notation. In Simulations, specify how you will simulate a dataset using the notation from Models. In Analyses, tell me exactly what you're going to do and what you're going to report. Make sure that for everything you're going to report in Results, you've already told me somewhere in Methods that you're going to report it.

[4] This definition needs to be modified to have $C_{ii} = 0$ along the diagonal.

[5] 'Centering data' is a very broad concept which typically refers to centering all observations of a variable to have mean 0, repeated for each variable. This centering operation is different in that it centers one observation of many variables, for each observations, that is it centers each image to have mean hue of 0. Because this centering is non-standard in that sense, you may want to say M centers images rather than data.

[6] eigendecomposition, not just decomposition. There are many other decompositions that might have been used here.

[7] You introduce E_{full}, Λ_{full}, and E'_{full} notation for transpose here, but then for the rest of the report use E, Λ, and E^T notation for transpose. Either is fine, but pick just one.

[8] You've just used x for the first time, but you haven't introduced the notation. x is vectorization of an image, meaning a length 256 vector where each element corresponds with the hue of a pixel.

[9] Nope! E is definitely orthogonal! That's a property of eigendecomposition when applied to a symmetric matrix like MCM.

[10] "...represents a 1000×256 matrix of 1000 observed images, with each row corresponding to one image."

[11] Coefficient vectors in what model? Why? You and I know the answer, but these concepts need to be introduced. This brings me back to the idea of a new student who has never heard of the project learning about it through your report. It may be hard to believe, but that is the level of detail you will need to understand your own reports when you look back at them in two or three years to write your dissertation.

for the frequency space. The relationship between the two is:

$$\boldsymbol{\beta} = \mathbf{E}\mathbf{b},$$

ensuring that $\mathbf{X}\boldsymbol{\beta} = \mathbf{X}_{\text{freq}}\mathbf{b}$.

Two simulation scenarios are considered:

1. **Pixel space sparsity:** Here, $\boldsymbol{\beta}$ is sparse, with non-zero values limited to a central 8×8 region of the pixel grid. The covariate matrix $\mathbf{X}$ is generated from a multivariate normal distribution with mean 0 and covariance matrix $\mathbf{C}$.[12] The response variable $\mathbf{y}$ is binomial, with success probability determined by $\boldsymbol{\eta} = \mathbf{X}\boldsymbol{\beta}$, where the non-zero entries in $\boldsymbol{\beta}$ are constant and chosen such that the success probability[13] $\mathbf{p} = \frac{1}{1+\exp(-\boldsymbol{\eta})}$ is roughly uniformly distributed in $[0, 1]$.

2. **Frequency space sparsity:** In this scenario, the coefficient vector $\mathbf{b}$ is sparse, with 10% of its 256 entries randomly set to non-zero values. The covariate matrix $\mathbf{X}_{\text{freq}}$ is generated from a multivariate normal distribution with zero mean and a diagonal covariance matrix, where the eigenvalues decrease along the diagonal [I am considering different options for the step size in the diagonal covariance matrix. Currently, I am using a constant step size, but further investigation may be needed to determine the most appropriate choice.].[14] The non-zero entries in $\mathbf{b}$ are constant and chosen such that outcome $\mathbf{y}$[15] is drawn from a binomial distribution with a success probability $\mathbf{p}$ roughlyuniformly distributed in $[0, 1]$.

After simulating data from either space, the values in the alternate space are calculated using the described transformations. When simulating data from the frequency space, even though the diagonal covariance matrix is chosen randomly and not calculated via $\mathbf{E}^T\mathbf{C}\mathbf{E}$, $\mathbf{E}$ is still used to transform the data back to the pixel space.[16]

[12] This matrix you've called C is not the same matrix as in MCM. We should discuss.

[13] Success probability of what? You and I know, but you have to introduce that there will be two groups of images and that this probability is a probability of membership.

[14] I've seen a couple of your thoughts on variances throughout this report, especially as they relate to Figures 5, 6, 8, and 10. You have good ideas I'd like to see you pursue (try that proof!!). To throw in my two cents, ... (I'm realizing that's a traditional English phrase/pun, relying on cents, as in pennies as a currency, being pronounced identically to sense, as in intelligence or common sense. The phrase thus means casually offering my thoughts without confidence in their value.) I would try using constant variance as well as variances as described in Sarah's spVBM work. The group should sit down and work out what the second of these options really means since Sarah thought it was a good idea. More generally, this may just be related to a signal to noise ratio. When variability due to noise is small relative to the magnitude of the coefficient (the signal), we have a higher signal to noise ratio and more powerful tests. We may have variable signal to noise, resulting in variable power in Figure 10.

[15] This is the first sentence in which you refer to an outcome y!! You need to introduce group labels and groups much, much earlier than where you currently do in the results.

[16] We should sit down together and talk through the covariance matrices and assumptions together. The distinction between the covariance matrix C used to define the frequency space and the covariance matrix of the images is both very subtle and very important, and I can see your struggle in your notation. (which is good! I can see what you're thinking!)

Model Evaluation

To compare the performance of LASSO in both the pixel space and frequency space, we fit two models for each simulation context: one using covariates in the pixel space and another using covariates in the frequency space. The optimal regularization parameter λ is selected via cross-validation, using the binomial deviance as the performance metric. The dataset is split into training (80%) and test (20%) sets, and the cross-validation is performed using 10 folds.[17]

Two values of λ are considered:

- `lambda.min`, which minimizes the cross-validated error.
- `lambda.1se`, the largest λ within one standard error of the minimum.

After selecting the optimal λ, we evaluate model performance on the test set. The performance metrics include:

- Accuracy
- Area Under the Curve (AUC)
- P-values for each covariate

P-values are computed using the **hdi** package [need to provide further details on how the package computes p-values]. The entire simulation is repeated 500 times. We report the mean and standard deviation of accuracy and AUC. For p-values, we report the percentage of cases where $p < 0.05$ at each covariate location [consider whether p-value adjustment is needed here].[18]

Results

Effect Size Determination

In Simulation 1, we evaluated the distribution of the success probability $\mathbf{p}$ at different non-zero values of $\boldsymbol{\beta}$ (0.01, 0.05, 0.1, 0.2, and 1) (this is methods). As shown in Figure 1, a value of 0.1 produced the most uniform distribution of $\mathbf{p}$, making it the optimal choice for model fitting in this scenario (this is results).

Similarly, in Simulation 2, we assessed the distribution of $\mathbf{p}$ at various non-zero values for $\mathbf{b}$ (0.1, 0.15, 0.2, 0.25, and 0.3) (this is methods). As shown in Figure 2, the value of 0.2 resulted in the most uniform distribution of $\mathbf{p}$, making it the best option for this simulation (this is results).

[17] I discover in the Results section that you simulate 500 datasets from each simulation. I should discover that here, as it describes what you did.

[18] You computed and reported more than you describe here. For example, you computed the mean group differences and reported them in Figures 3 and 5, but don't tell me you did that until the Results section. Every comparison and computation needs to be described in Methods so that someone can replicate what you did, and then reported in Results so that someone can see if they get the same thing.

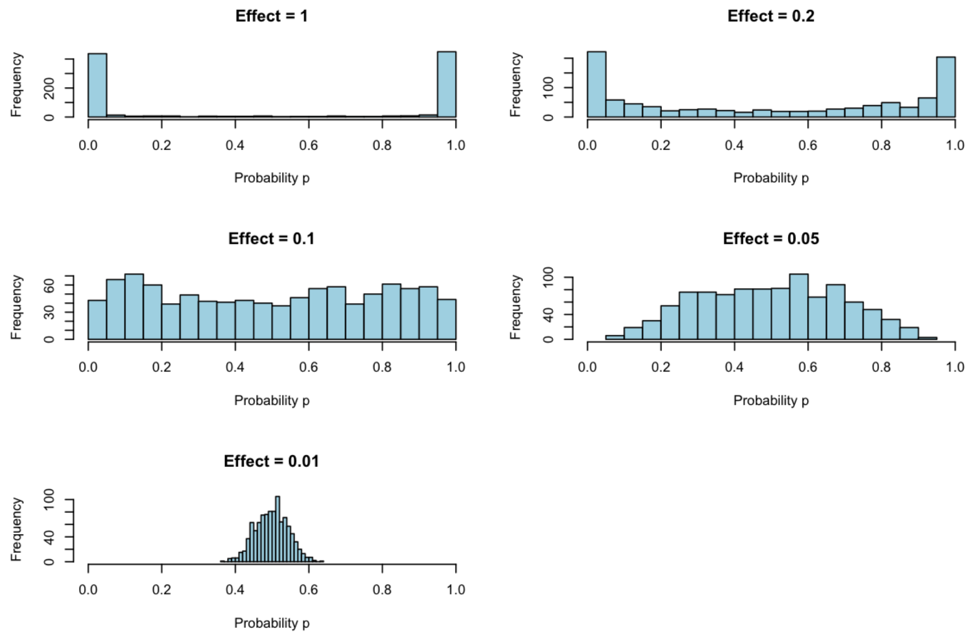

Figure 1: Distribution of success probability **p** at different non-zero values in Simulation 1.

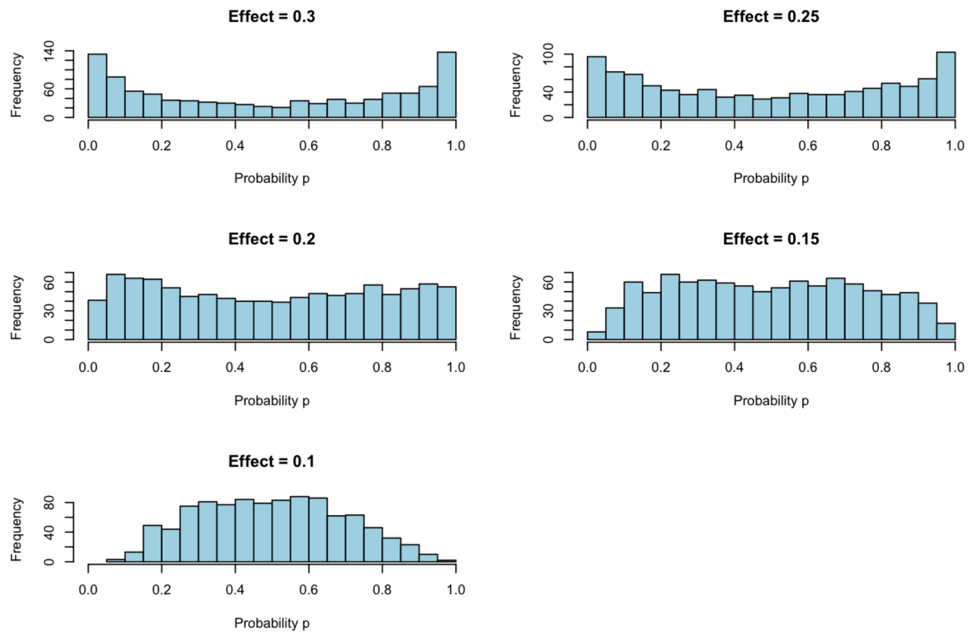

Figure 2: Distribution of success probability **p** at different non-zero values in Simulation 2.

Group Mean Difference

In this subsection, we examine the group mean differences in covariate values between instances where $y = 1$ and $y = 0$ for both Simulation 1 and Simulation 2.

Figure 3[19] [20] [21] presents the group mean differences for Simulation 1, with the heatmap on the left showing that regions corresponding to non-zero coefficients in $\boldsymbol{\beta}$ exhibit larger mean differences between $y = 1$ and $y = 0$, as larger covariate values in these locations have higher probabilities of being assigned to $y = 1$. The scatterplot on the right displays the group mean differences in the frequency domain, where each point represents a frequency component; frequencies associated with larger eigenvalues tend to have larger mean differences. Figure 4 shows the actual coefficients used in Simulation 1, where non-zero coefficients in $\boldsymbol{\beta}$ are localized to specific pixels, corresponding to the areas with larger mean differences in the group comparison.

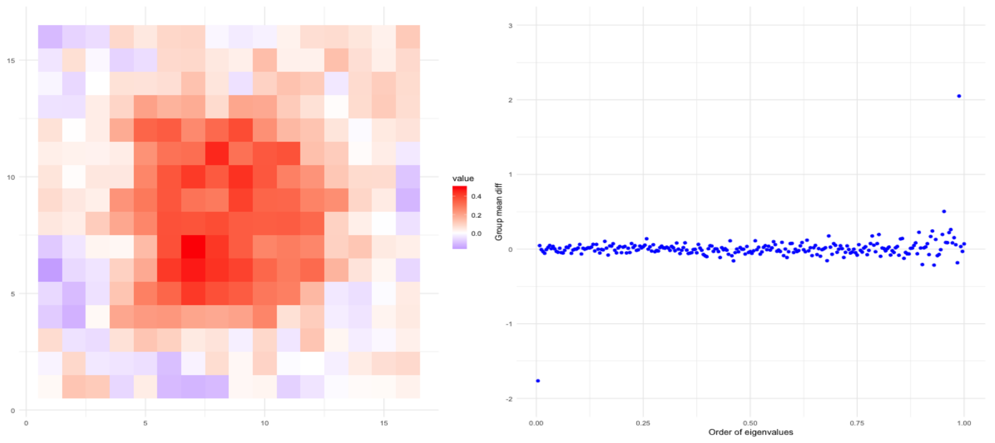

Figure 3: Group mean difference in covariate values between instances where $y = 1$ and $y = 0$ in Simulation 1, shown for both the pixel space (left) and frequency space (right).

Figure 5 shows the group mean differences for Simulation 2, while Figure 6 displays the actual coefficients.[22] The non-zero coefficients in $\mathbf{b}$ are uniformly set to 0.2. However, the scatterplot in the frequency space does not clearly highlight the non-zero components, with increasing variance observed for larger eigenvalues. This variance pattern is consistent with the diagonal covariance matrix used in the simulation. The difficulty in identifying the non-zero components suggests that the effect size may be too small relative to the variance, making detection challenging. [Further adjustments to either the effect size or the covariance matrix could improve the detectability of these non-zero coefficients in future analyses.]

[19] For the pixel plots in Figures 3 and 4, I would consider using tile color to indicate true signal. For reference, in ggplot, color refers to the outline of tiles while fill refers to the inner color, so you'd have fill indicating true signal and color indicating group difference.

[20] The axes and color scales for the corresponding plots in Figures 3 and 4 should be legible and the same scale. Let me know if you need help making them use the same scale. To get the scales to be large enough to read: if you're using the f_savemyplot function I sent you, I recommend resaving the plot at half the size but in the same proportion. This will render the text larger relative to the plots.

[21] For all plots against eigenvalue order, I like to include a vertical line at the location of the 0 eigenvalue to distinguish associations with positive spatial correlations from those with negative spatial correlations. If you do this, be sure to update figure captions accordingly.

[22] For the order of eigenvalue plots in Figures 5 and 6, I would consider using color and/or shape to indicate true signal.

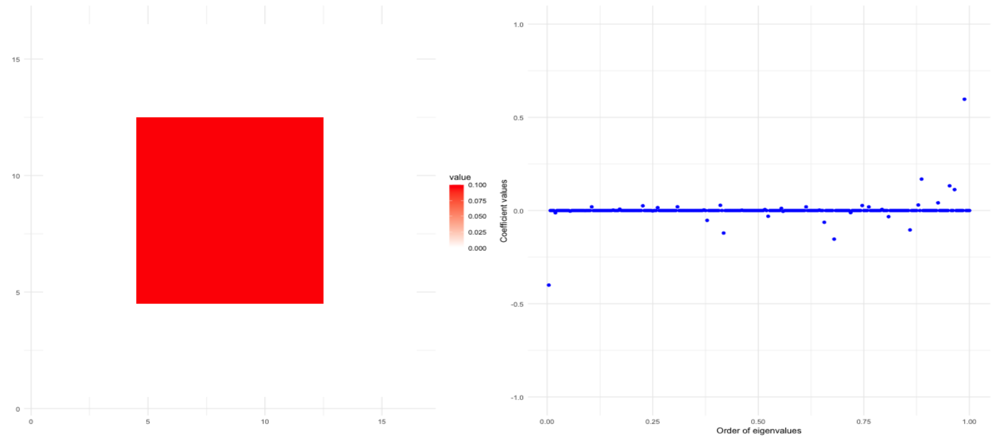

Figure 4: Actual coefficients in Simulation 1 for the pixel space (left) and frequency space (right).

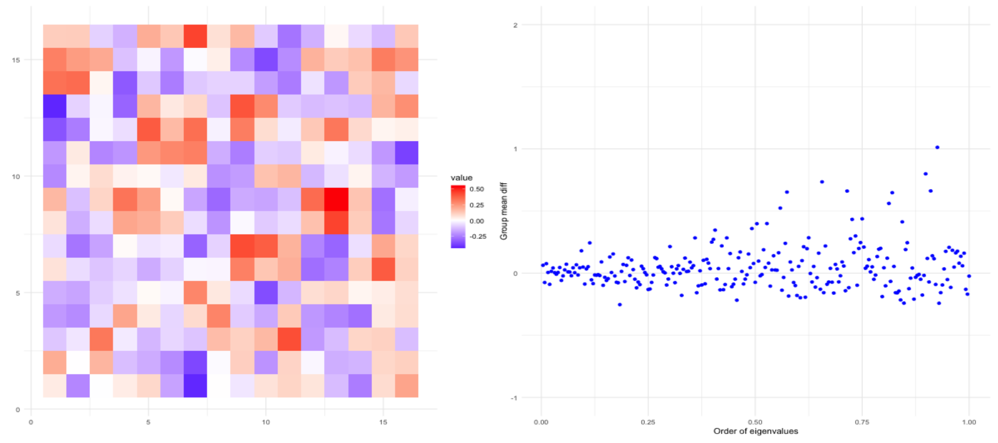

Figure 5: Group mean difference in covariate values between instances where $y = 1$ and $y = 0$ in Simulation 2.

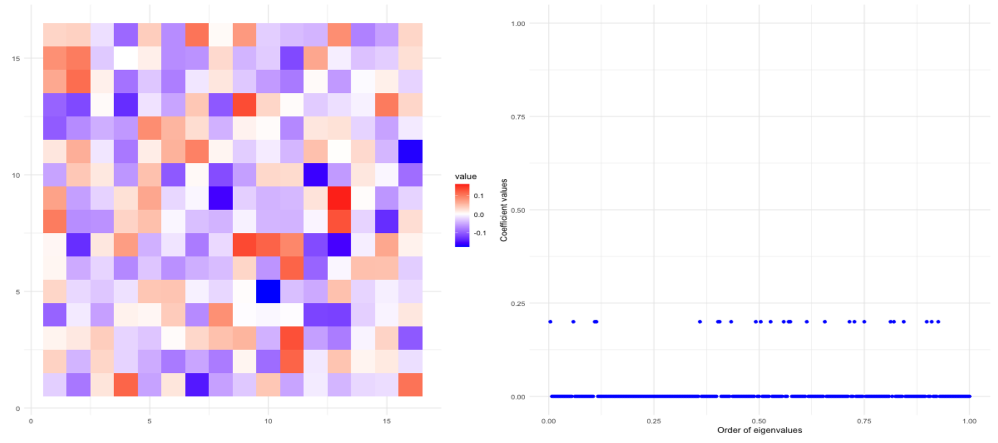

Figure 6: Actual coefficients in Simulation 2 for the pixel space (left) and frequency space (right).

AUC and Accuracy

Table 1 summarizes the average AUCs and accuracies over 500 iterations. In both Simulation 1 (pixel space sparsity) and Simulation 2 (frequency space sparsity), models fitted in the frequency space consistently outperformed those fitted in the pixel space. For example, in Simulation 1, using `lambda.min` as the regularization parameter, models trained with pixel space covariates achieved an AUC of 0.803 (SE = 0.031) and an accuracy of 72.6% (SE = 0.032). In contrast, models trained with frequency space covariates produced a slightly higher AUC of 0.826 (SE = 0.028) and a higher accuracy of 74.5% (SE = 0.030). A similar trend was observed in Simulation 2, with frequency space models showing superior performance regardless of the regularization parameter used.

Table 1: Comparison of AUC and accuracy between models fitted in the pixel space and frequency space across 500 iterations for Simulation 1 and Simulation 2.

Simulation	**Model in Pixel Space**		**Model in Frequency Space**	
	AUC (SE)	**Accuracy (SE)**	**AUC (SE)**	**Accuracy (SE)**
Simulation 1				
`lambda.min`	0.803 (0.031)	0.726 (0.032)	0.826 (0.028)	0.745 (0.030)
`lambda.1se`	0.800 (0.032)	0.722 (0.032)	0.826 (0.029)	0.745 (0.031)
Simulation 2				
`lambda.min`	0.755 (0.036)	0.684 (0.034)	0.812 (0.030)	0.732 (0.032)
`lambda.1se`	0.735 (0.039)	0.669 (0.038)	0.812 (0.031)	0.732 (0.032)

Coefficients Estimation

Figure 8[23] presents the mean estimated b values plotted against the order of eigenvalues. The order of eigenvalues are calculated the same way as above. For Simulation 1, `lambda.1se` shrinks the estimated coefficients more than `lambda.min`, as it provides a larger penalty on it. For Simulation 2, even though it is not obvious, I feel the estimated values has an increase trend as the eigenvalues increase. [Still, I am wondering whether this is related to the covariance matrix, the decreased diagonal values, consider math proof?].

The mean estimated coefficients across iterations were calculated, and Figure 7 displays the mean estimated $\boldsymbol{\beta}$ values. Two key observations can be made: (1) There is no significant difference in the estimated coefficients when using `lambda.min` versus `lambda.1se`, and (2) the estimated values align well with the actual values, indicating that each model is accurately identifying the relevant features.

[23] In Figures 7 and 8, you're reporting estimates without showing the truth, and you're only showing estimates where the model sparsity matches the simulation sparsity (pixel model reported for pixel sparsity simulation, frequency model reported for frequency sparsity simulation). I would like to see 12 more subplots in these two figures, corresponding with the true coefficients for each simulation in each space and the estimated coefficients according to each tuning parameter in each simulation. Remember that just as $\beta = Eb$ for true coefficients, we can transform estimates as well $\hat{\beta} = E\hat{b}$ to present coefficients $\hat{b}$ from a frequency model in pixel space and vice versa.

Figure 8 shows the mean estimated **b** values plotted against the order of eigenvalues. The eigenvalue ordering is consistent with earlier calculations. In Simulation 1, `lambda.1se` applies a stronger regularization penalty, shrinking the estimated coefficients more than `lambda.min`. For Simulation 2, although the trend is less clear, there seems to be an upward trend in the estimated values as the eigenvalues increase. This trend may be related to the structure of the covariance matrix, specifically its decreasing diagonal values [consider math proof?].

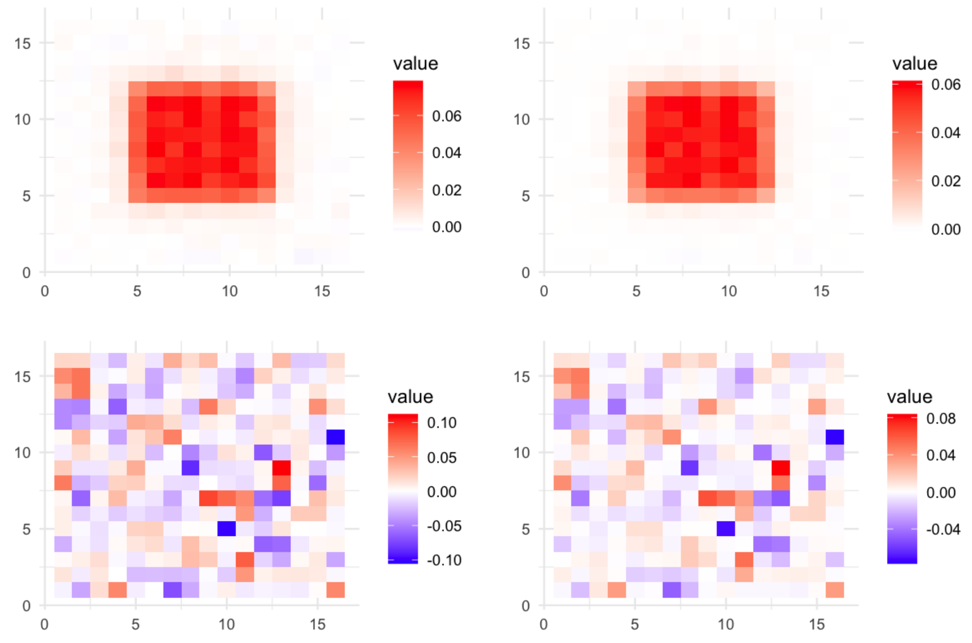

Figure 7: Mean estimated β values across simulations, with models fitted using `lambda.min` (left) and `lambda.1se` (right). The top row shows results for Simulation 1, while the bottom row shows results for Simulation 2.

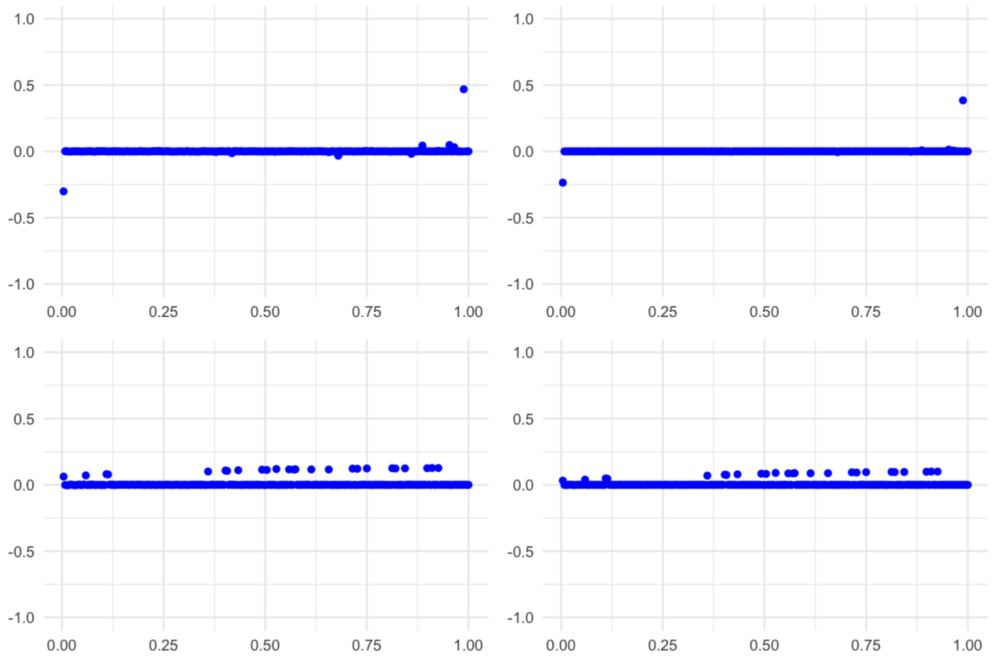

Figure 8: Mean estimated b values across simulations, plotted against ordered eigenvalues. Models fitted using `lambda.min` are on the left and models fitted with `lambda.1se` on the right. The top row shows results for Simulation 1, while the bottom row shows results for Simulation 2.

Significant P-values

It is interesting to observe that, although the heatmap for significance of $\boldsymbol{\beta}$ in Simulation 1 follows the pattern of the actual non-zero values, the percentage of significance is relatively low (Figure 9 left). In contrast, the non-zero values of $\mathbf{b}$ (Figure 10 left) show a much higher percentage of significance, reaching as high as 100% across iterations.

Another observation is that, although the non-zero effect size for $\mathbf{b}$ is constant in Simulation 2, the percentage of significant p-values increases as the eigenvalues grow (Figure 10 right). [I am considering creating a plot to visualize the actual β values in Simulation 2 and the actual b values in Simulation 1, where the non-zero values vary, and compare them with the corresponding percentage of significant p-values. The goal is to examine whether the size of the actual non-zero values correlates with the percentage of significance. I suspect that a larger absolute effect size should result in a higher percentage of significance, but this doesn't seem to be the case for b in Simulation 2, so I want to investigate other factors as well.]

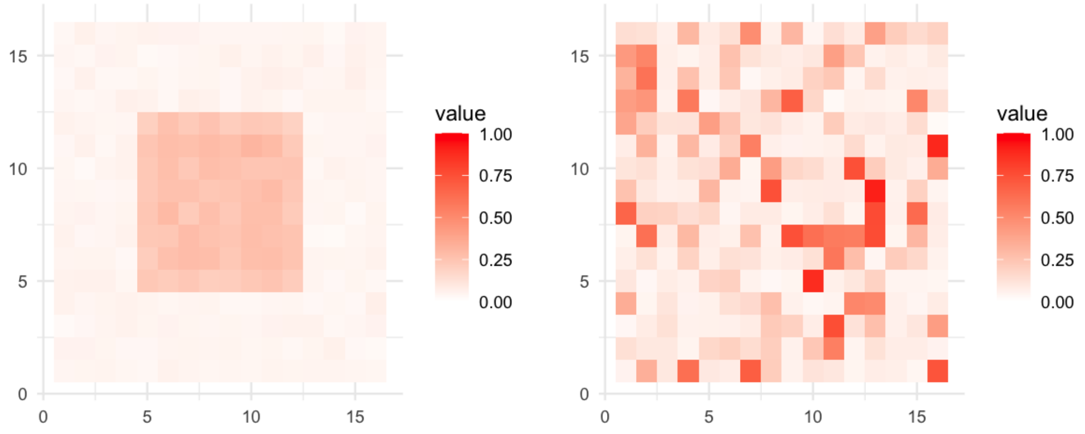

Figure 9: Percentage of significant p-values for elements of β when fitting models in the pixel space in Simulation 1 (left) and Simulation 2 (right).

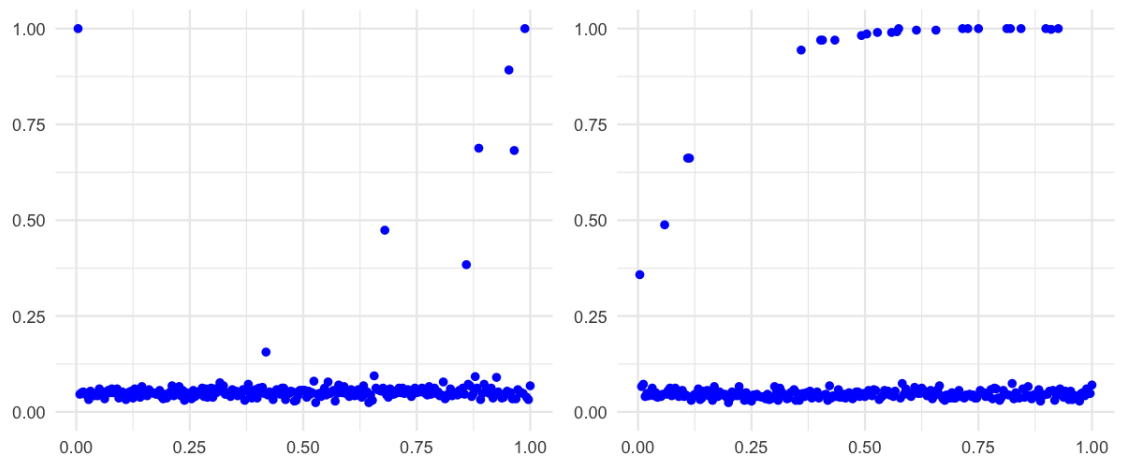

Figure 10: Percentage of significant p-values for elements of b across ordered eigenvalues in both simulations.

Future Work

- Adding details about how `hdi` package calculated p-values and why my permutation test didn't work.
- Increase b effect size (how to keep p evenly distributed in the same time?) see whether the pattern of coefficient estimates disappear or relieve.
- What is the next step in higher level?[24]

[24] We're going to move into more simulations and more complex contexts towards writing a full on paper!! This is fantastic work. Once we really work through this report and work out all the kinks, the next simulations should go faster. We also want to add spVBM back in as a methodology. For now, though, you need to stay focused on this report. Writing up what you've done really nicely takes a lot longer than you'd expect, but it's nearly always worth it.

References

[1] D. Murakami and D. A. Griffith. Eigenvector Spatial Filtering for Large Data Sets: Fixed and Random Effects Approaches. Geographical Analysis, 51(1):23–49, 2019.