Optimal Hyperparameters and model averaging Analysis with Corrected ELBO

Contents

Overview

In this document, I present the results a grid search and model averaging experiment conducted across 100 trials after correcting the ELBO. I first analyze the distribution of the optimal hyperparameters selected by grid search. Next, I compare the results of model averaging to the grid search approach.

The first section shows the correction made to the ELBO. The following two main sections contain these analyses, while the fourth section describes the data and extraneous covariate generation.

ELBO Calculations

Let (y_i, x_i) be independent response-covariate pairs, with $y_i \in \mathbb{R}, x_i \in \mathbb{R}^p$. Consider the following model:

$$(\beta, \gamma) \sim \prod_{k=1}^{p} \delta_{\{0\}}(\beta_k)^{1-\gamma_k} \mathcal{N}\left(\beta_k; 0, \sigma^2\sigma_\beta^2\right)^{\gamma_k} \hat{\pi}^{\gamma_k}(1-\hat{\pi})^{1-\gamma_k} \tag{1}$$

$$y_i|\beta \sim \mathcal{N}\left(x_i^\top \beta, \frac{\sigma^2}{w_i}\right) \tag{2}$$

Approximate $[\beta, \gamma|y] \propto [y_i|\beta][\beta, \gamma]$ with q:

$$q(\beta, \gamma) = \prod_{k=1}^{p} \delta_{\{0\}}(\beta_k)^{1-\gamma_k} \mathcal{N}\left(\beta_k; \mu_k, s_k^2\right)^{\gamma_k} \alpha_k^{\gamma_k}(1-\alpha_k)^{1-\gamma_k} \tag{3}$$

Then, the ELBO is given by:

$$\mathbb{E}_q(\log[\beta, \gamma]) + \mathbb{E}_q(\log[Y|\beta]) - \mathbb{E}_q[\log q(\beta, \gamma)] \tag{4}$$

The corrected ELBO as currently in the code is:

$$\mathbb{E}_q(\log[\beta, \gamma]) = \sum_{k=1}^{p} \left(-\frac{\alpha_k}{2}\log(\sigma^2\sigma_\beta^2) - \frac{\alpha_k(s_k^2+\mu_k^2)}{2\sigma^2\sigma_\beta^2} + \alpha_k \log(\hat{\pi}) + (1-\alpha_k)\log(1-\hat{\pi})\right) \tag{5}$$

$$+ \tag{6}$$

$$\mathbb{E}_q(\log[Y|\beta]) = -\frac{1}{2\sigma^2}\sum_{i=1}^{n} w_i \left(\left\{y_i - \sum_{k=1}^{p} x_{ik}\mu_k\alpha_k\right\}^2 + \sum_{k=1}^{p} x_{ik}^2(\alpha_k(\mu_k^2+s_k^2) - \alpha_k^2\mu_k^2)\right) - \frac{n}{2}\log(2\pi\sigma^2) \tag{7}$$

$$- \tag{8}$$

$$\mathbb{E}_q[\log q(\beta, \gamma)] = \sum_{k=1}^{p} \left(-\frac{\alpha_k}{2}\log(s_k^2) - \frac{\alpha_k}{2} + \alpha_k\log(\alpha_k) + (1-\alpha_k)\log(1-\alpha_k)\right) \tag{9}$$

Prior to this correction, the last term in $\mathbb{E}_q(\log[Y|\beta])$ was $\frac{1}{2}\log(2\pi\sigma^2)$ instead of $\frac{n}{2}\log(2\pi\sigma^2)$.

Note that the expression for $\mathbb{E}_q(\log[Y|\beta])$ is missing a $\frac{1}{2}\sum_{i=1}^{n}\log(w_i)$. However, since the w_i are constant, the omission of this term does not affect the final model.

Optimal Pi

Hyperparameter specification

In this experiment, I applied the grid search algorithm in 100 trials. I generated the grid from the cartesian product of the following marginal grids:

```
# marginal grids
marg_grids
```

```
## $pip
##  [1] 0.00001 0.00010 0.00100 0.01000 0.02500 0.05000 0.10000 0.15000 0.20000
## [10] 0.25000 0.30000 0.35000 0.40000
##
## $ssq
##  [1] 1.0e-05 1.0e-04 1.0e-03 1.0e-02 2.5e-02 5.0e-02 1.0e-01 1.5e-01 2.0e-01
## [10] 2.5e-01 3.0e-01 3.5e-01 4.0e-01 4.5e-01 5.0e-01 5.5e-01 6.0e-01 6.5e-01
## [19] 7.0e-01 7.5e-01 8.0e-01 8.5e-01 9.0e-01 9.5e-01 1.0e+00 1.5e+00 2.0e+00
## [28] 2.5e+00 3.0e+00 5.0e+00 1.0e+01
##
## $sbsq
##  [1] 1.0e-05 1.0e-04 1.0e-03 1.0e-02 2.5e-02 5.0e-02 1.0e-01 1.5e-01 2.0e-01
## [10] 2.5e-01 3.0e-01 3.5e-01 4.0e-01 4.5e-01 5.0e-01 5.5e-01 6.0e-01 6.5e-01
## [19] 7.0e-01 7.5e-01 8.0e-01 8.5e-01 9.0e-01 9.5e-01 1.0e+00 1.5e+00 2.0e+00
## [28] 2.5e+00 3.0e+00 5.0e+00 1.0e+01
```

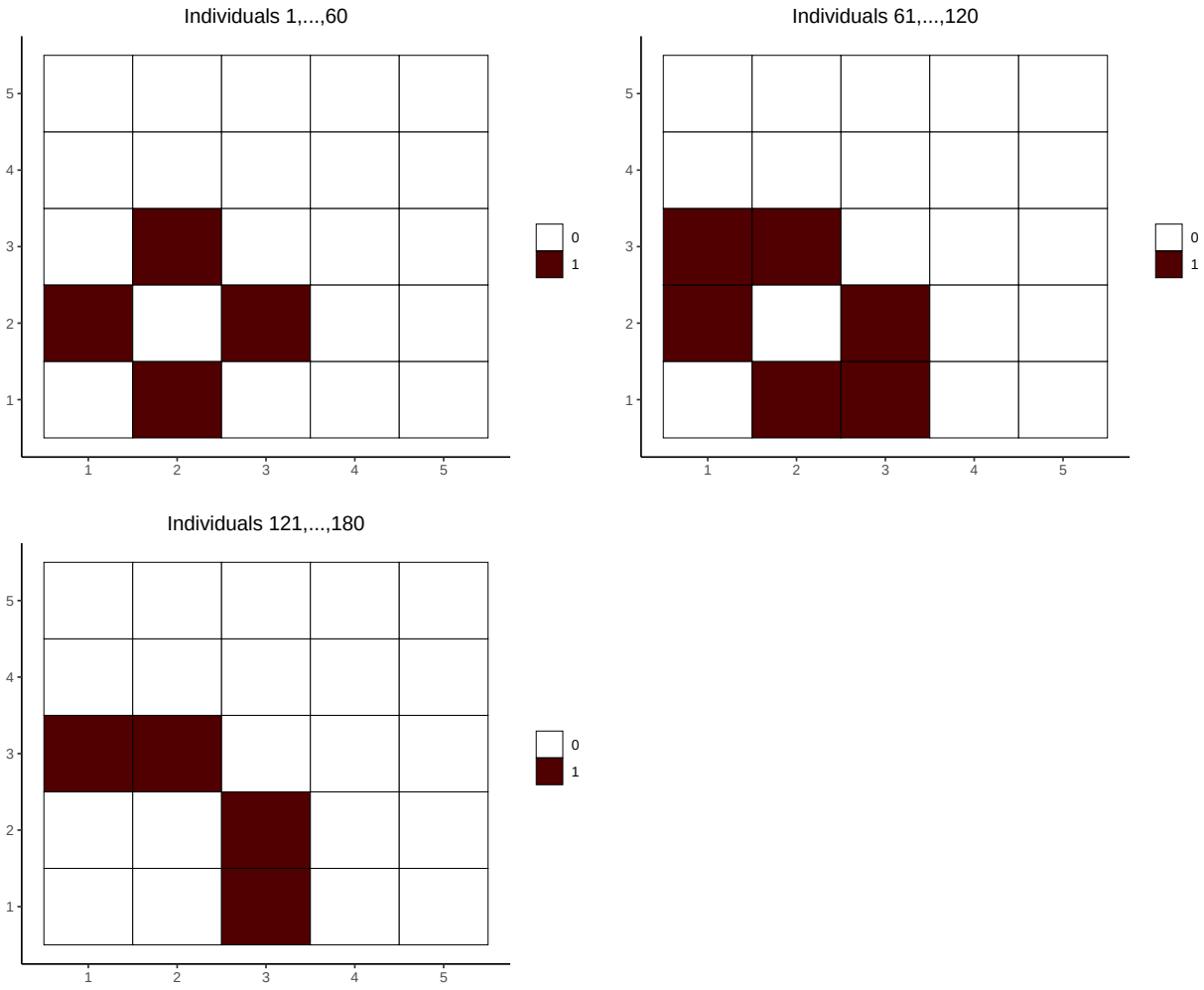
```
# number of grid points in the cartesian product
nrow(expand.grid(marg_grids))
```

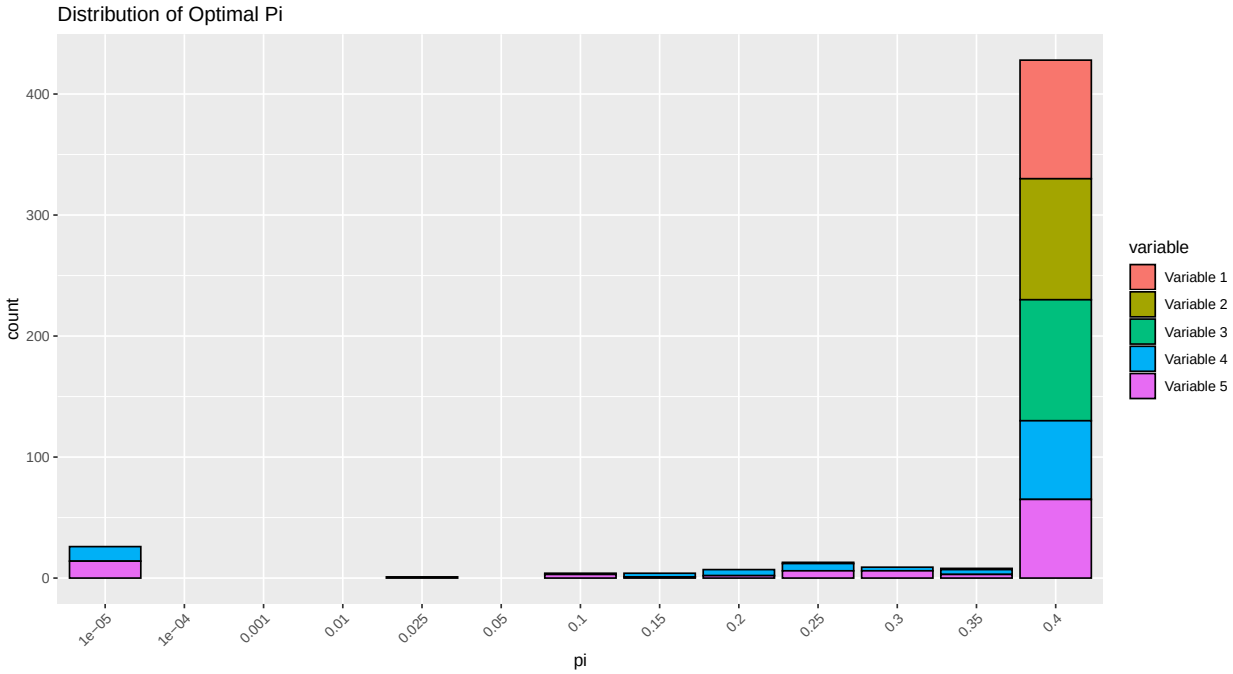
```
## [1] 12493
```

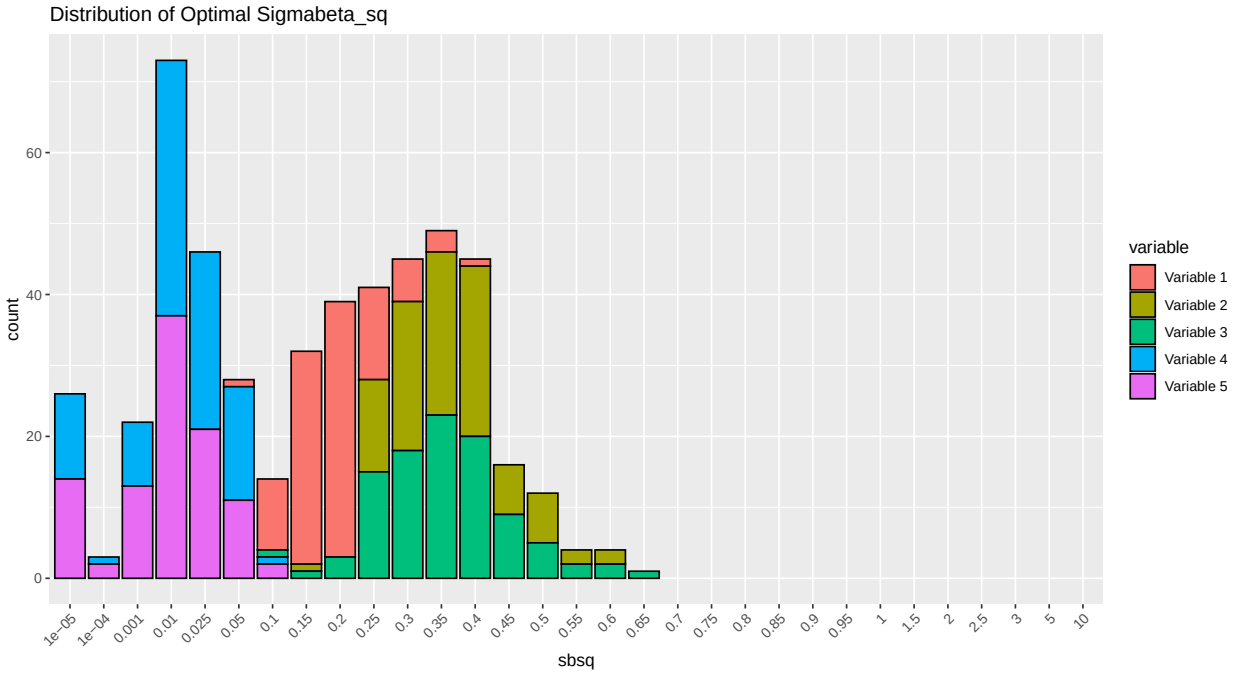
Optimal Hyperparameter Distribution

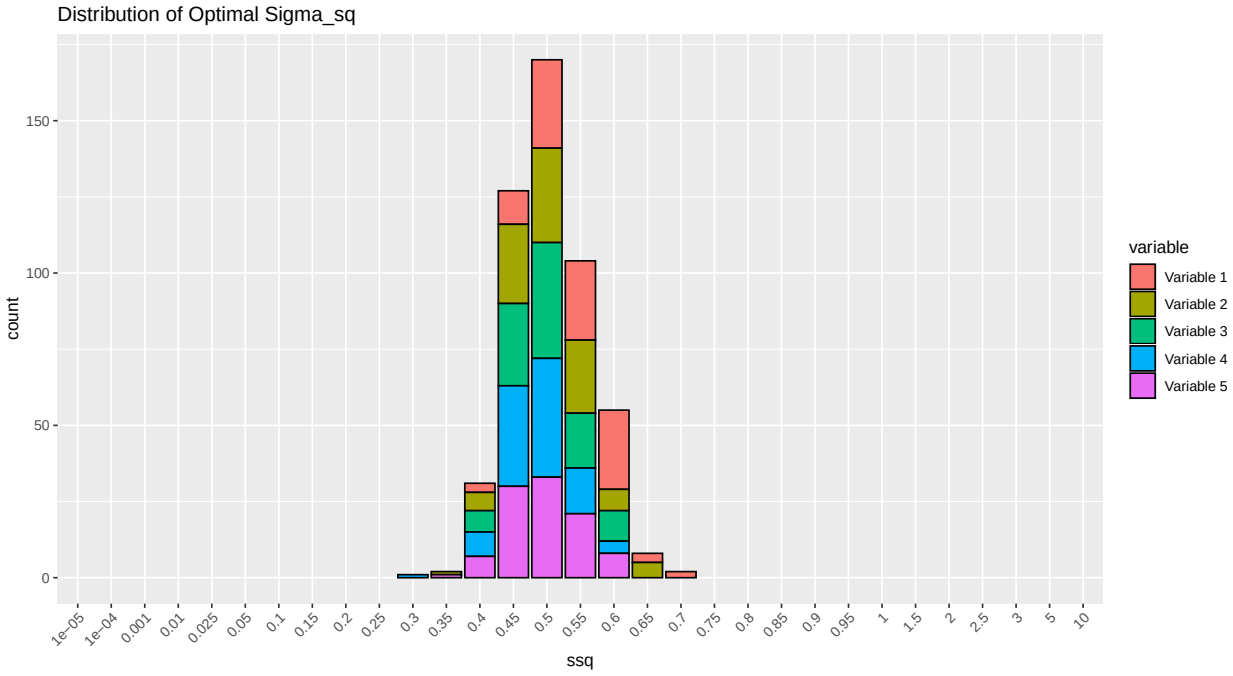
In this section, I display the distribution of optimal hyperparameters aggregated across all variables. Since a unique point in the hyperparameter grid was selected for each of the 5 variables, a total of 500 grid points are represented in each figure.

I begin by displaying the true underlying precision structures to provide context for the optimal pi.


Individuals 1,...,60
Individuals 61,...,120
Individuals 121,...,180
0
1
1
2
3
4
5


Distribution of Optimal Pi
count
pi
variable
Variable 1
Variable 2
Variable 3
Variable 4
Variable 5
0
100
200
300
400
1e−05
1e−04
0.001
0.01
0.025
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4



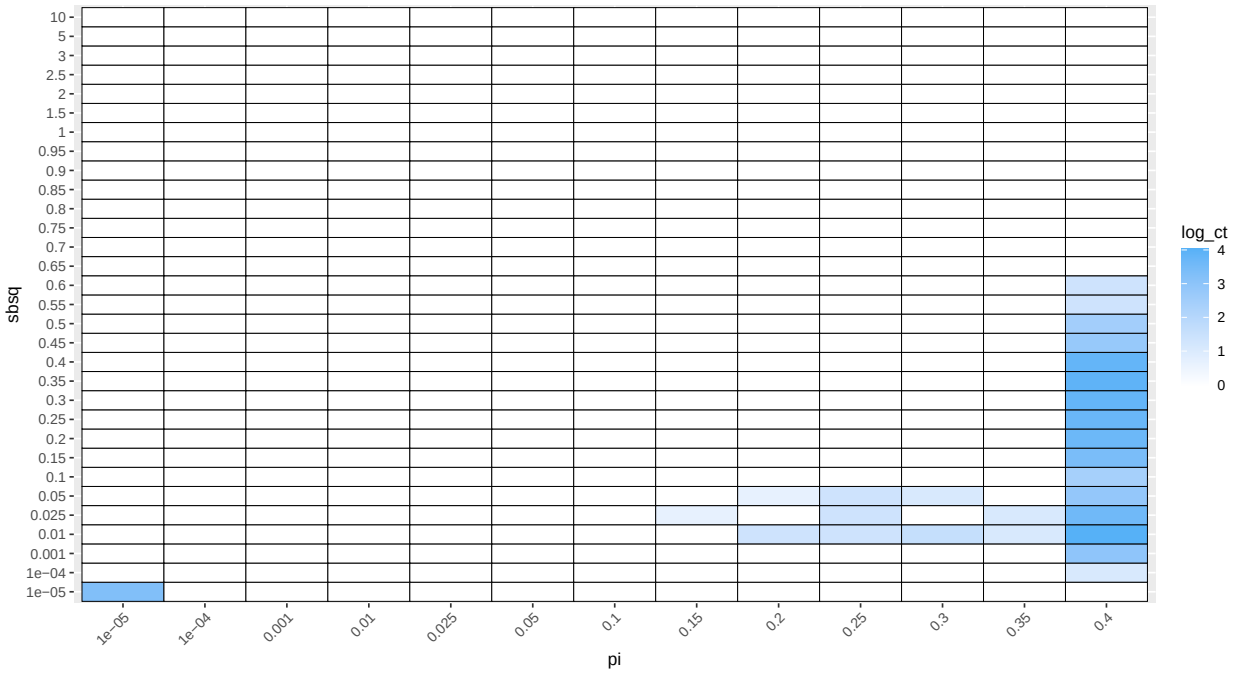


Optimal Sigmabeta_sq as a function of Pi

Here, I visualize the optimal sigmabeta_sq as a function of pi. Note that the count is on the log scale.

```
##
##         1e-05 1e-04 0.001 0.01 0.025 0.05 0.1 0.15 0.2 0.25 0.3 0.35 0.4 0.45
##   1e-05    26     0     0    0     0    0   0    0   0    0   0    0   0    0
##   1e-04     0     0     0    0     0    0   0    0   0    0   0    0   0    0
##   0.001     0     0     0    0     0    0   0    0   0    0   0    0   0    0
```

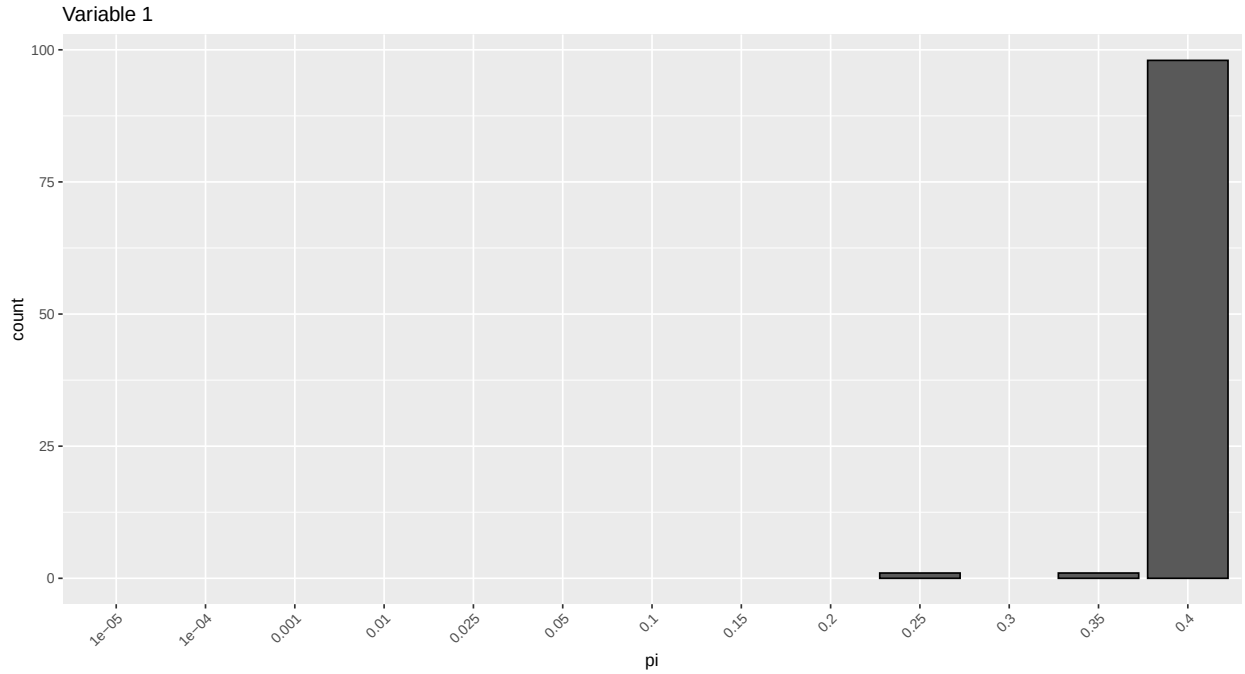
```
##    0.01      0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
##    0.025     0     0     0     1     0     0     0     0     0     0     0     0     0     0     0
##    0.05      0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
##    0.1       0     0     1     0     1     1     1     0     0     0     0     0     0     0     0
##    0.15      0     0     1     0     2     0     1     0     0     0     0     0     0     0     0
##    0.2       0     0     1     4     0     2     0     0     0     0     0     0     0     0     0
##    0.25      0     0     0     4     4     4     0     1     0     0     0     0     0     0     0
##    0.3       0     0     0     5     0     3     1     0     0     0     0     0     0     0     0
##    0.35      0     0     0     3     3     1     0     1     0     0     0     0     0     0     0
##    0.4       0     3    19    56    36    17    11    30    39    41    45    49    45    16
##
##          0.5 0.55 0.6 0.65 0.7 0.75 0.8 0.85 0.9 0.95   1 1.5   2 2.5   3   5 10
##    1e-05   0    0   0    0   0    0   0    0   0    0   0   0   0   0   0   0  0
##    1e-04   0    0   0    0   0    0   0    0   0    0   0   0   0   0   0   0  0
##    0.001   0    0   0    0   0    0   0    0   0    0   0   0   0   0   0   0  0
##    0.01    0    0   0    0   0    0   0    0   0    0   0   0   0   0   0   0  0
##    0.025   0    0   0    0   0    0   0    0   0    0   0   0   0   0   0   0  0
##    0.05    0    0   0    0   0    0   0    0   0    0   0   0   0   0   0   0  0
##    0.1     0    0   0    0   0    0   0    0   0    0   0   0   0   0   0   0  0
##    0.15    0    0   0    0   0    0   0    0   0    0   0   0   0   0   0   0  0
##    0.2     0    0   0    0   0    0   0    0   0    0   0   0   0   0   0   0  0
##    0.25    0    0   0    0   0    0   0    0   0    0   0   0   0   0   0   0  0
##    0.3     0    0   0    0   0    0   0    0   0    0   0   0   0   0   0   0  0
##    0.35    0    0   0    0   0    0   0    0   0    0   0   0   0   0   0   0  0
##    0.4    12    4   4    1   0    0   0    0   0    0   0   0   0   0   0   0  0
```



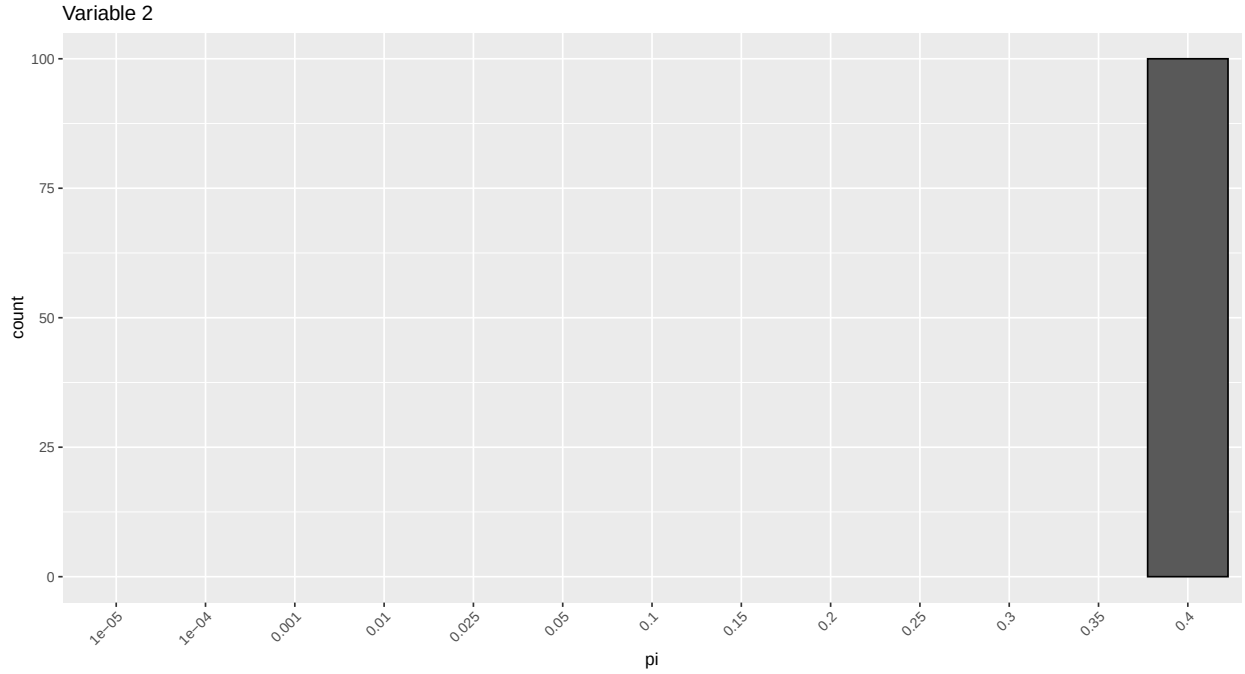
Optimal Pi by variable

In this section, I visualize the optimal pi by variable.

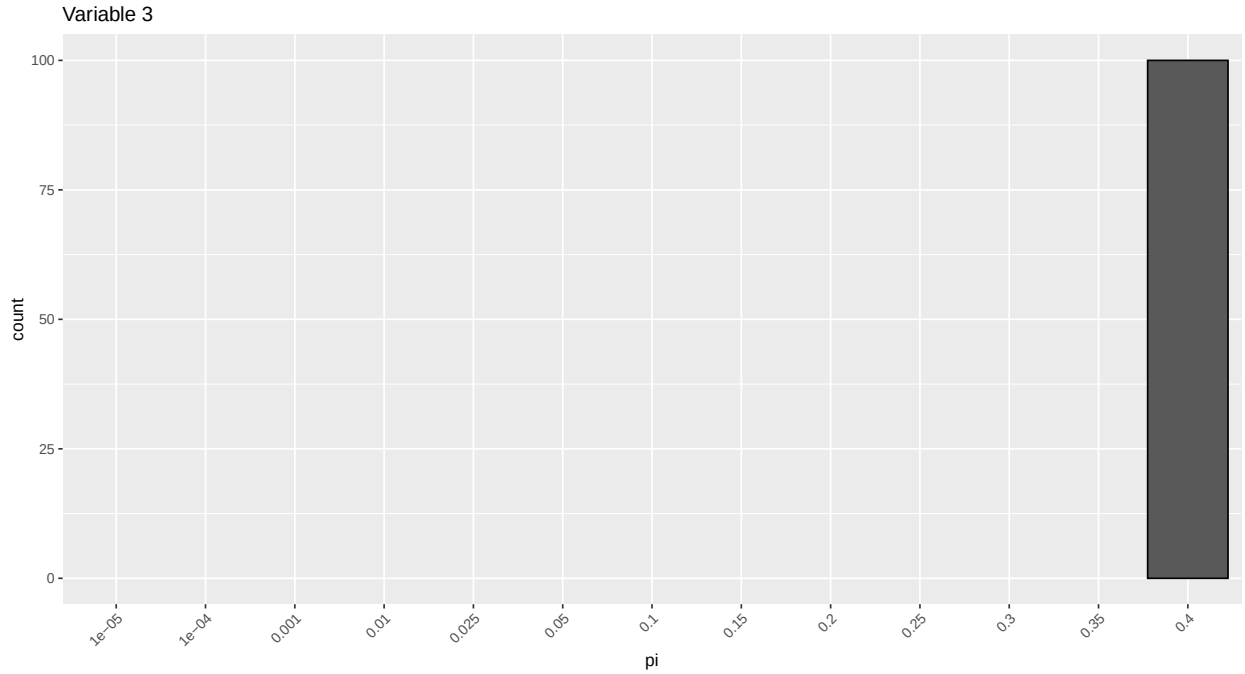
```
## [[1]]
```


Variable 1
100
75
50
25
0
count
1e−05
1e−04
0.001
0.01
0.025
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
pi

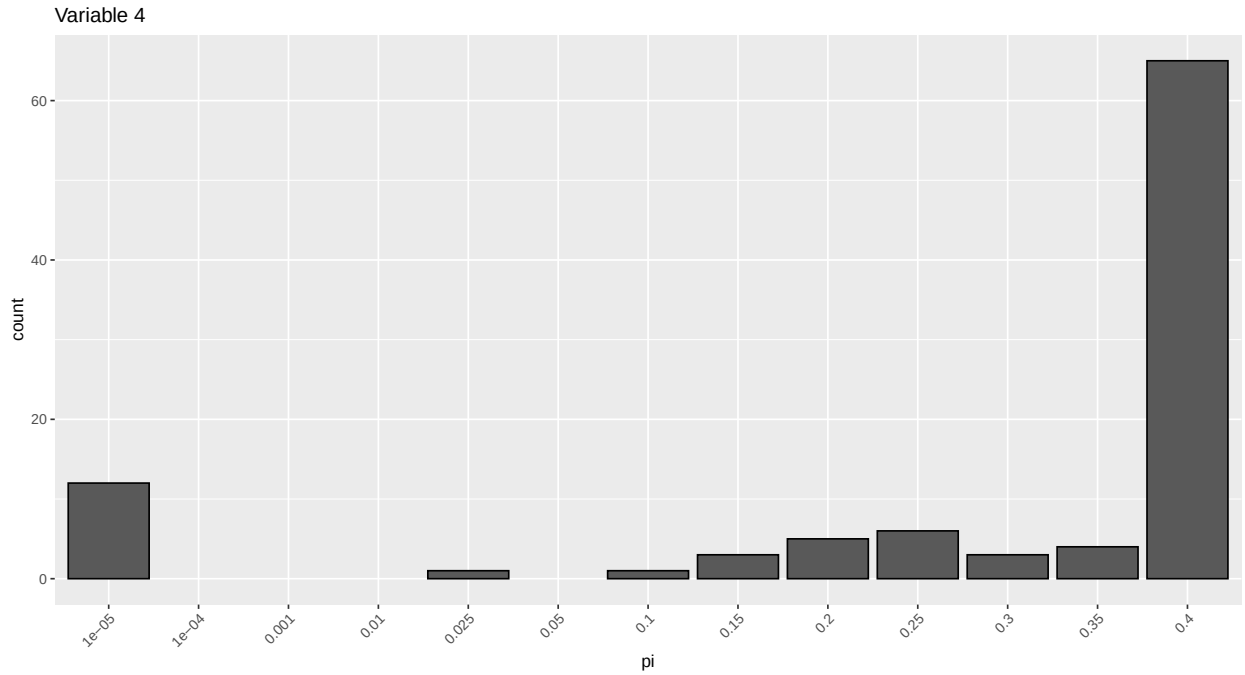
##
[[2]]


Variable 2
100
75
50
25
0
count
1e−05
1e−04
0.001
0.01
0.025
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
pi

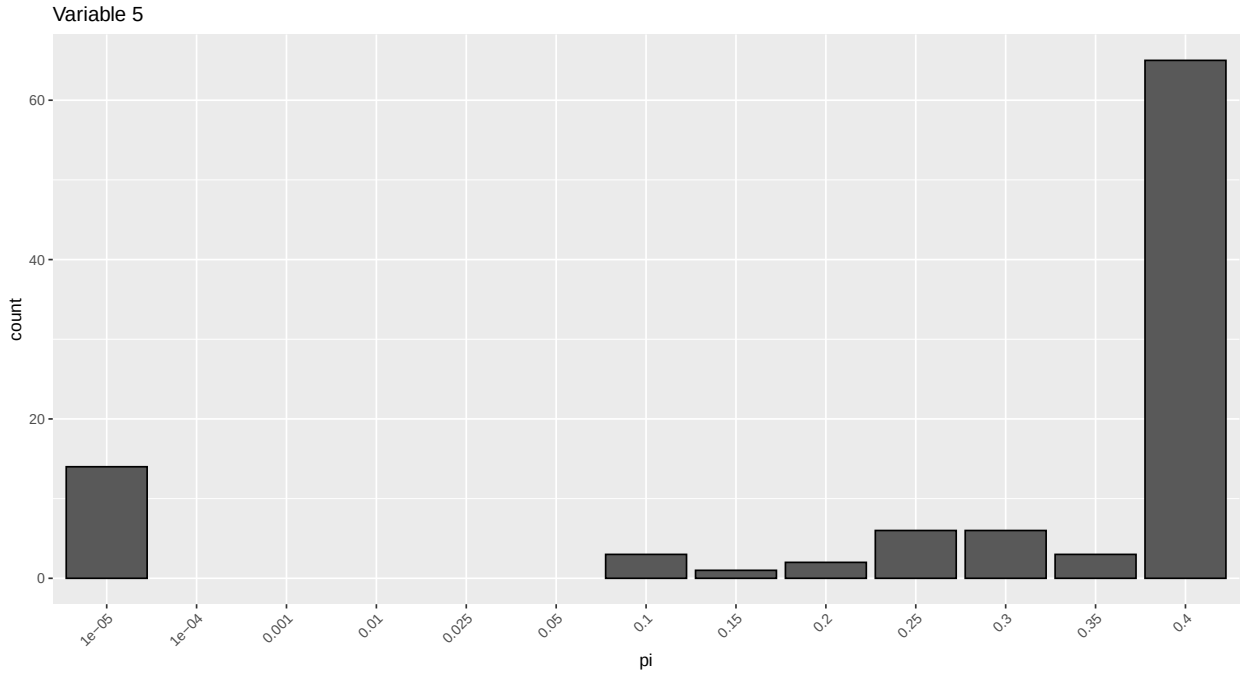
##
[[3]]


Variable 3
100
75
50
25
0
count
1e−05
1e−04
0.001
0.01
0.025
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
pi

##
[[4]]


Variable 4
60
40
20
0
count
1e−05
1e−04
0.001
0.01
0.025
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
pi

##
[[5]]



$p = 20$

In this section, I present the same analyses as above for a dataset with $p = 20$.

The marginal grids are:

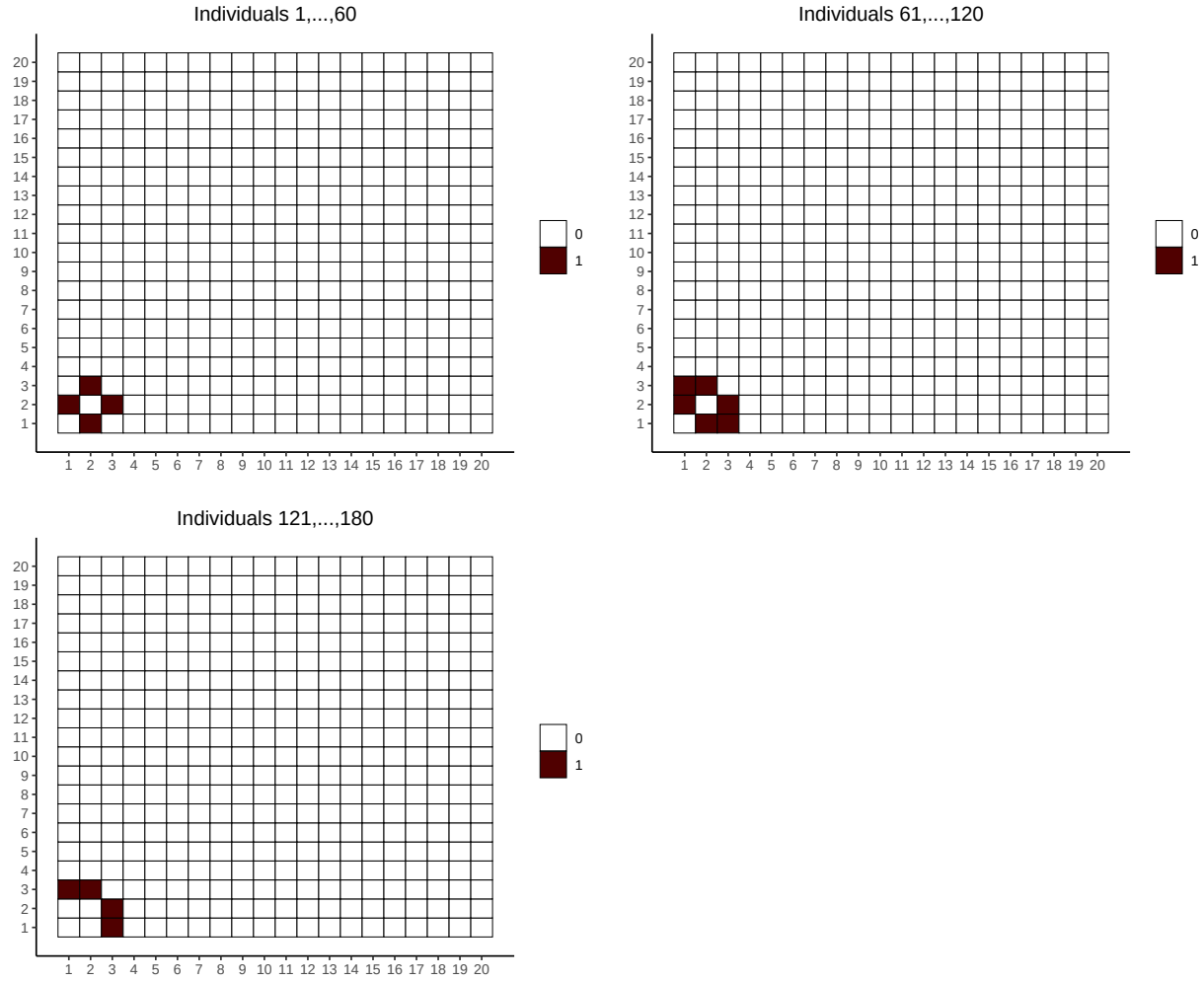
```
# marginal grids
marg_grids
```

```
##        pip     ssq    sbsq
## 1  0.00001 1.0e-05 1.0e-05
## 2  0.00010 1.0e-04 1.0e-04
## 3  0.00100 1.0e-03 1.0e-03
## 4  0.01000 1.0e-02 1.0e-02
## 5  0.02500 2.5e-02 2.5e-02
## 6  0.05000 5.0e-02 5.0e-02
## 7  0.15000 2.0e-01 2.0e-01
## 8  0.25000 3.5e-01 3.5e-01
## 9  0.35000 5.0e-01 5.0e-01
## 10 0.45000 6.5e-01 6.5e-01
## 11 0.55000 8.0e-01 8.0e-01
## 12 0.65000 9.5e-01 9.5e-01
## 13 0.75000 1.0e+00 1.0e+00
## 14 0.85000 1.5e+00 1.5e+00
## 15 0.95000 2.0e+00 2.0e+00
## 16 0.99000 2.5e+00 2.5e+00
## 17 0.99900 3.0e+00 3.0e+00
## 18 0.99990 5.0e+00 5.0e+00
## 19 0.99999 1.0e+01 1.0e+01
```

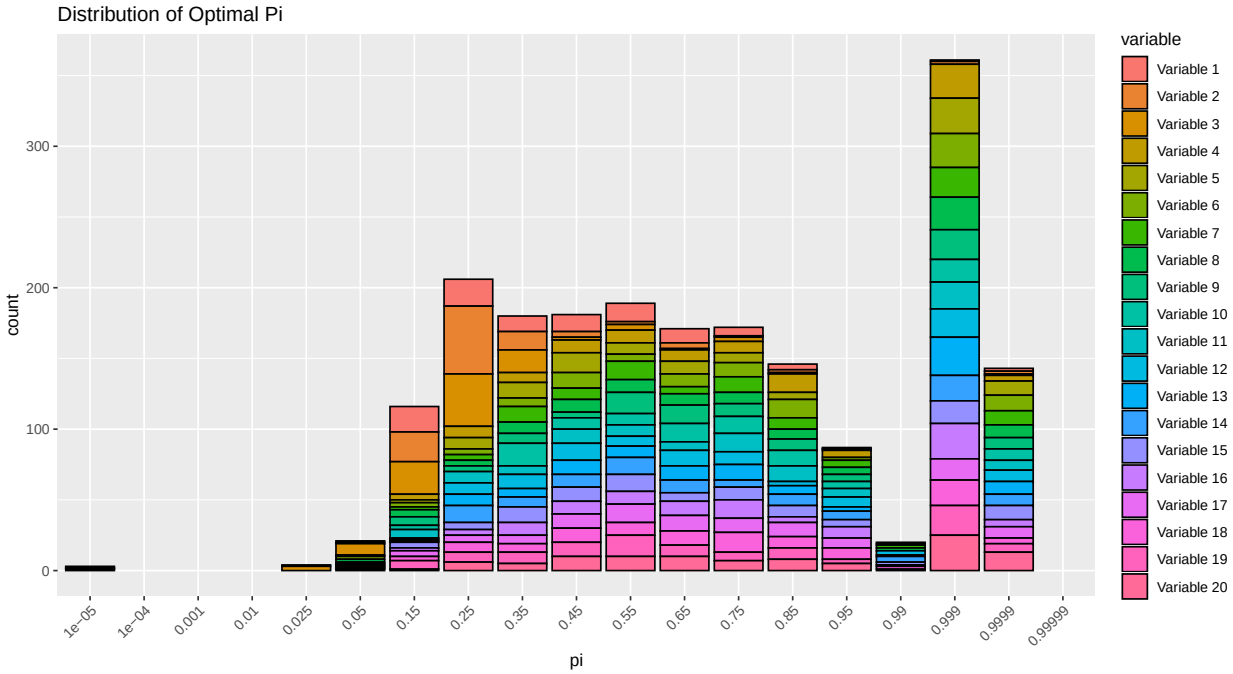
```
# number of grid points in the cartesian product
nrow(expand.grid(marg_grids))
```

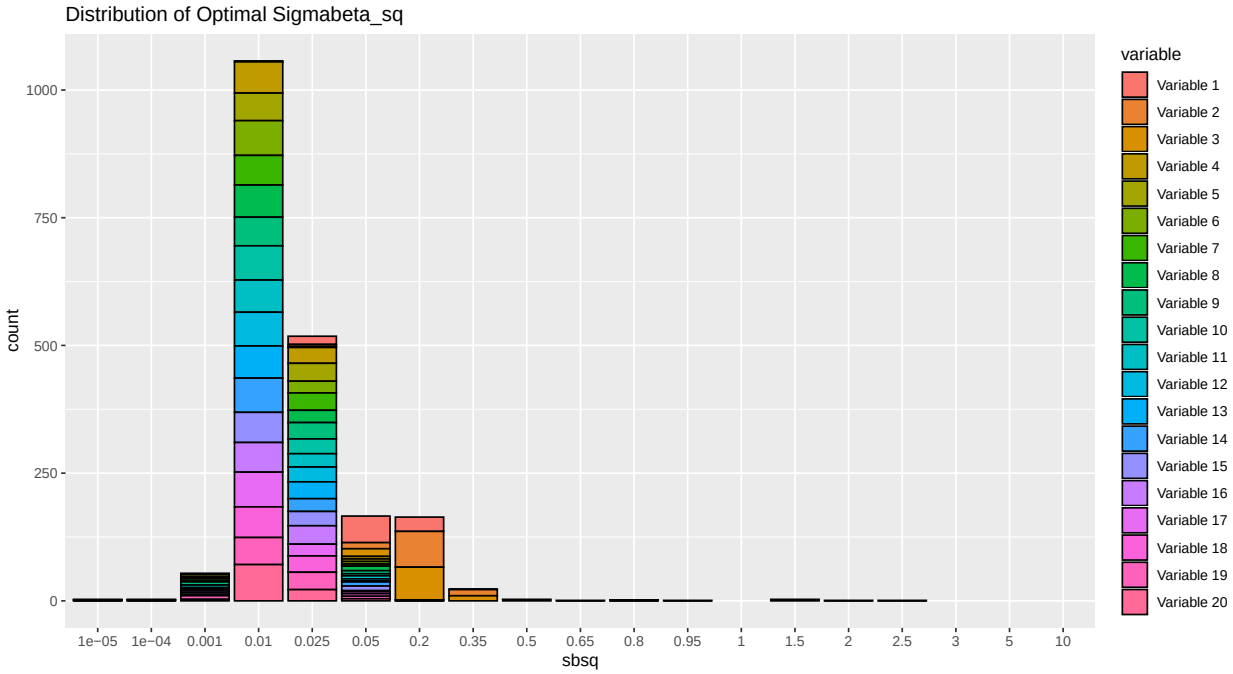
```
## [1] 6859
```

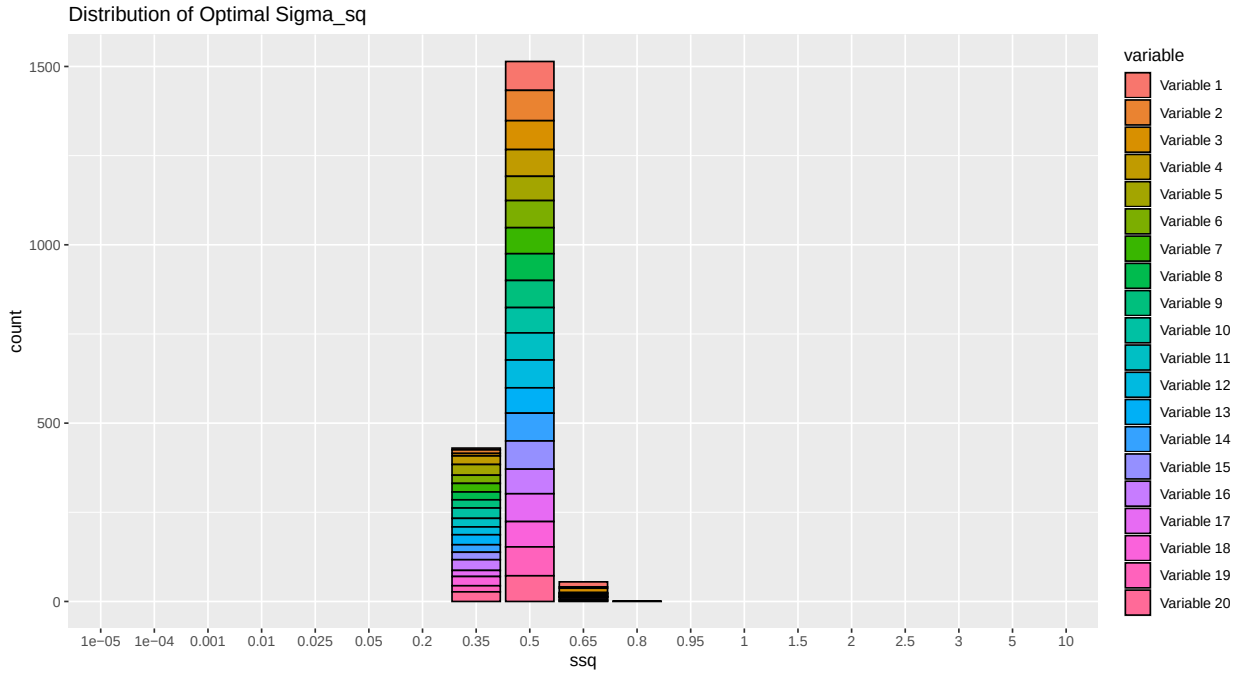
The true precision structures are given below:



Below is the distribution of the optimal hyperparameters aggregated across all variables.


Distribution of Optimal Pi
count
pi
0
100
200
300
1e−05
1e−04
0.001
0.01
0.025
0.05
0.15
0.25
0.35
0.45
0.55
0.65
0.75
0.85
0.95
0.99
0.999
0.9999
0.99999
variable
Variable 1
Variable 2
Variable 3
Variable 4
Variable 5
Variable 6
Variable 7
Variable 8
Variable 9
Variable 10
Variable 11
Variable 12
Variable 13
Variable 14
Variable 15
Variable 16
Variable 17
Variable 18
Variable 19
Variable 20

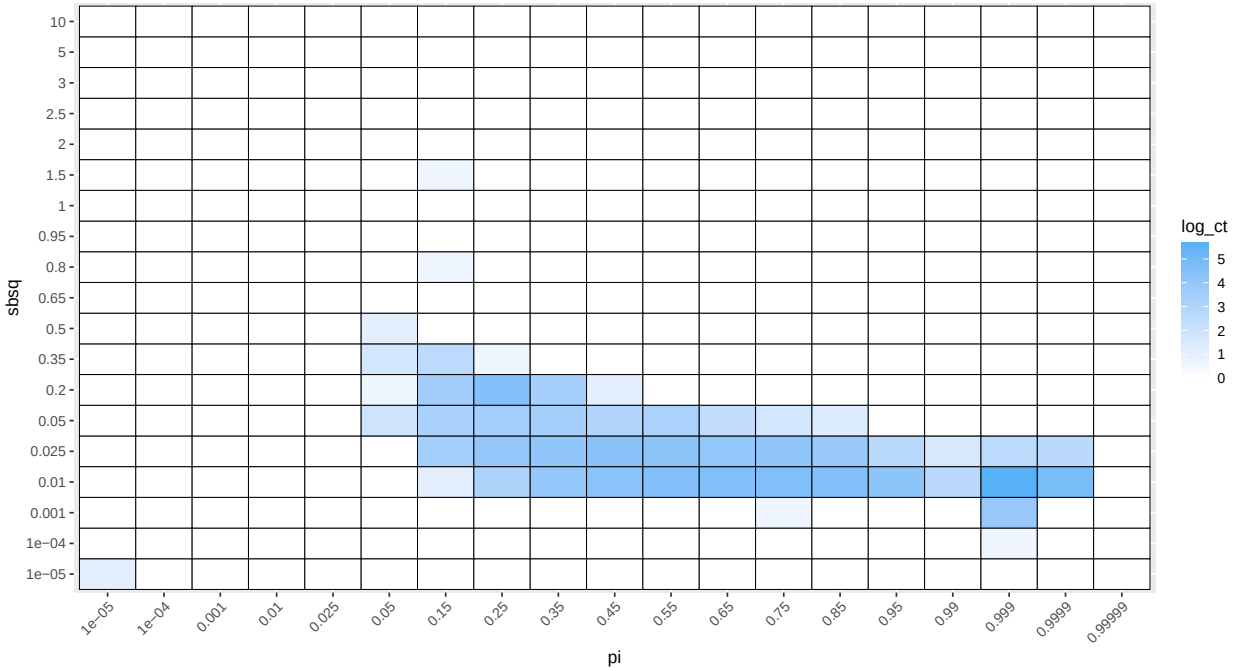

Distribution of Optimal Sigmabeta_sq
count
sbsq
0
250
500
750
1000
1e−05
1e−04
0.001
0.01
0.025
0.05
0.2
0.35
0.5
0.65
0.8
0.95
1
1.5
2
2.5
3
5
10
variable
Variable 1
Variable 2
Variable 3
Variable 4
Variable 5
Variable 6
Variable 7
Variable 8
Variable 9
Variable 10
Variable 11
Variable 12
Variable 13
Variable 14
Variable 15
Variable 16
Variable 17
Variable 18
Variable 19
Variable 20



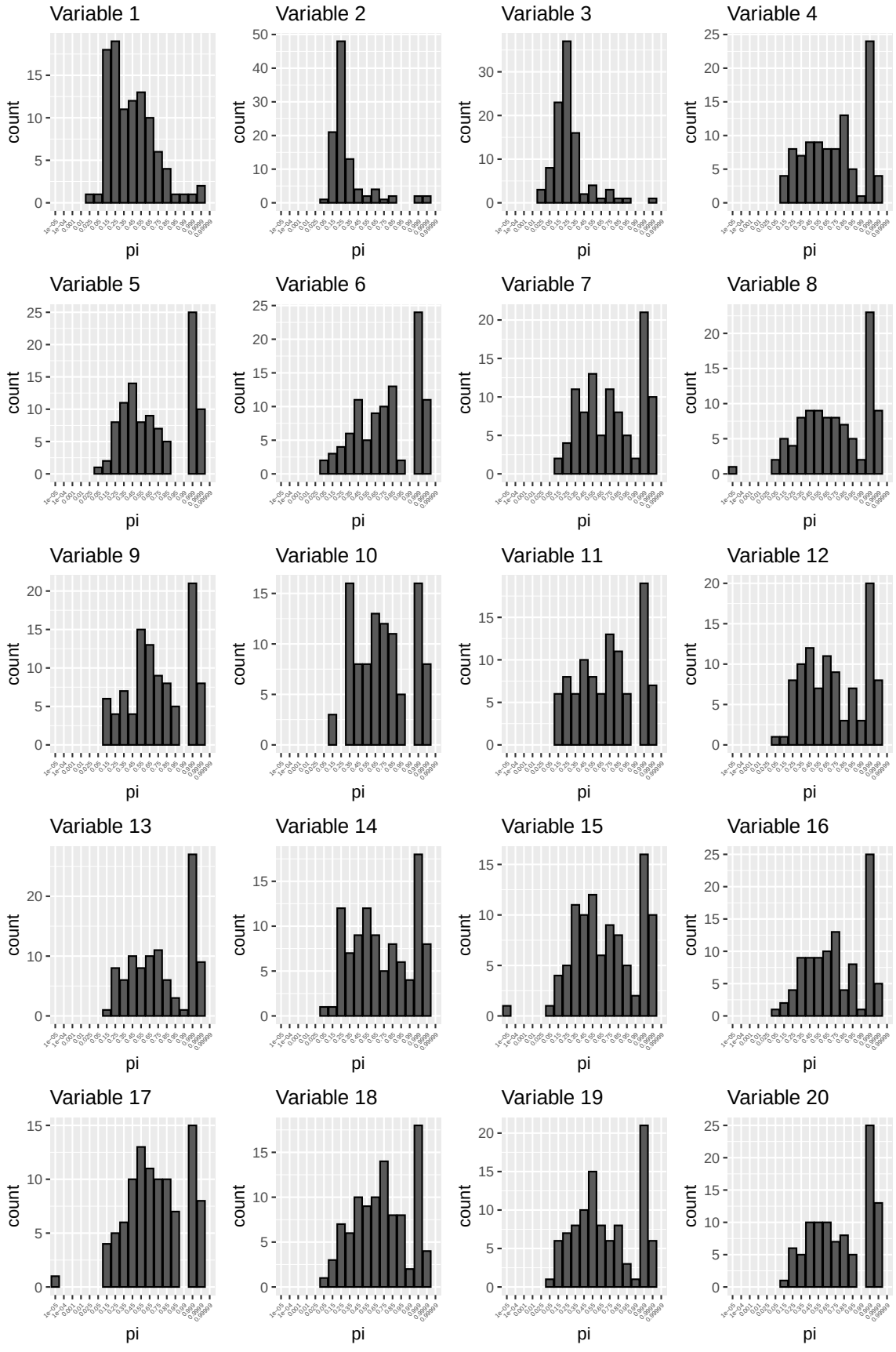
The optimal value of sbsq as a function of pi is visualized below.

```
##
##          1e-05 1e-04 0.001 0.01 0.025 0.05 0.2 0.35 0.5 0.65 0.8 0.95   1 1.5
##   1e-05      3     0     0    0     0    0   0    0   0    0   0    0   0   0
##   1e-04      0     0     0    0     0    0   0    0   0    0   0    0   0   0
##   0.001      0     0     0    0     0    0   0    0   0    0   0    0   0   0
##   0.01       0     0     0    0     0    0   0    0   0    0   0    0   0   0
##   0.025      0     0     0    0     0    0   1    1   0    0   0    0   0   1
##   0.05       0     0     0    1     1    7   2    6   3    1   0    0   0   0
##   0.15       0     0     0    3    31   27  35   14   0    0   2    1   0   2
##   0.25       0     0     1   23    57   31  92    2   0    0   0    0   0   0
##   0.35       0     0     1   55    62   32  30    0   0    0   0    0   0   0
##   0.45       0     0     0   77    81   20   3    0   0    0   0    0   0   0
##   0.55       0     1     1   94    68   25   0    0   0    0   0    0   0   0
##   0.65       0     0     1  100    58   11   1    0   0    0   0    0   0   0
##   0.75       0     0     2  101    63    6   0    0   0    0   0    0   0   0
##   0.85       0     0     0   95    47    4   0    0   0    0   0    0   0   0
##   0.95       0     0     1   69    16    1   0    0   0    0   0    0   0   0
##   0.99       0     0     0   15     5    0   0    0   0    0   0    0   0   0
##   0.999      0     2    47  297    14    1   0    0   0    0   0    0   0   0
##   0.9999     0     0     0  127    15    1   0    0   0    0   0    0   0   0
##   0.99999    0     0     0    0     0    0   0    0   0    0   0    0   0   0
##
##            2 2.5   3   5  10
##   1e-05    0   0   0   0   0
##   1e-04    0   0   0   0   0
##   0.001    0   0   0   0   0
##   0.01     0   0   0   0   0
##   0.025    0   1   0   0   0
##   0.05     0   0   0   0   0
##   0.15     1   0   0   0   0
```

```
##    0.25       0    0    0    0    0
##    0.35       0    0    0    0    0
##    0.45       0    0    0    0    0
##    0.55       0    0    0    0    0
##    0.65       0    0    0    0    0
##    0.75       0    0    0    0    0
##    0.85       0    0    0    0    0
##    0.95       0    0    0    0    0
##    0.99       0    0    0    0    0
##    0.999      0    0    0    0    0
##    0.9999     0    0    0    0    0
##    0.99999    0    0    0    0    0
```



Here the distribution of the optimal pi is visualized by variable.


Variable 1
Variable 2
Variable 3
Variable 4
Variable 5
Variable 6
Variable 7
Variable 8
Variable 9
Variable 10
Variable 11
Variable 12
Variable 13
Variable 14
Variable 15
Variable 16
Variable 17
Variable 18
Variable 19
Variable 20
count
pi

Model Averaging

Algorithm Overview

This section describes the model averaging approach used here for a spike-and-slab regression of $y \in \mathbb{R}^n$ on $X \in \mathbb{R}^{n \times p}$ to compute an importance-weighted-average of the posterior inclusion probabilities $P(\gamma_k = 1|y, X, \theta_j)$, where $\gamma_k = 1 \iff \beta_k = 1$.

Denote the importance weights by $\underset{\sim}{w} = w(\theta_1), ..., w(\theta_N)$, where $\theta_1, ..., \theta_N$ are points the points comprising the hyperparameter grid.

With the j-th column of the data as the response y and the data with the j-th column removed as the covariates X, an extraneous-covariate-weighted regression will be performed with respect to individual l for each $\theta \in \underset{\sim}{\theta}$. The importance-weighted sum of the posterior quantities $p(\gamma_k = 1|X, y, \theta_i)$ will then be computed using:

$$w(\theta_i) = \frac{p(y|X, \theta_i)p(\theta_i)}{q(\theta_i)} \tag{10}$$

Where $p(y|X, \theta_i), p(\theta_i)$, and $q(\theta_i)$ are the marginal log-likelihood, the prior density, and the importance density, respectively. The prior density and the importance density are defined as:

$$p(\theta_i) = p({\sigma^2}_i) * p(\sigma^2_{\beta_i}) * p(\pi_i) \qquad q(\theta_i) = q({\sigma^2}_i) * q(\sigma^2_{\beta_i}) * q(\pi_i) \tag{11}$$

Proposition 1. *Let $w(\theta_i) = p(y|X, \theta_i)$. Then:*

$$\lim_{N \to \infty} \frac{\sum_{i=1}^N p(\gamma_k = 1|X, y, \theta_i)w(\theta_i)}{\sum_{i=1}^N w(\theta_i)} = p(\gamma_k = 1|X, y), \ \ a.s. \tag{12}$$

The approximation of $p(y|X, \theta_i)$ is made using $\exp(ELBO_j^l(\theta_i))$, where $ELBO_j^l(\theta_i)$ is the log-lower bound of the marginal likelihood $p(y|X, \theta_i)$ in the extraneous-covariate-weighted regression of y on X with respect to individual l.

Note that the prior and importance density used in this experiment were uniform, which is equivalent to:

$$\frac{\sum_{i=1}^N p(\gamma_k = 1|X, y, \theta_i)p(y|X, \theta_i)}{\sum_{i=1}^N p(y|X, \theta_i)} \tag{13}$$

Sensitivity and Specificity

In this section, I compare the sensitivity and the specificity for the two methods, grid search and model averaging.

```
# grid sensitivity
summary(grid_sens)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.7881  0.9065  0.9571  0.9425  0.9982  1.0000
```

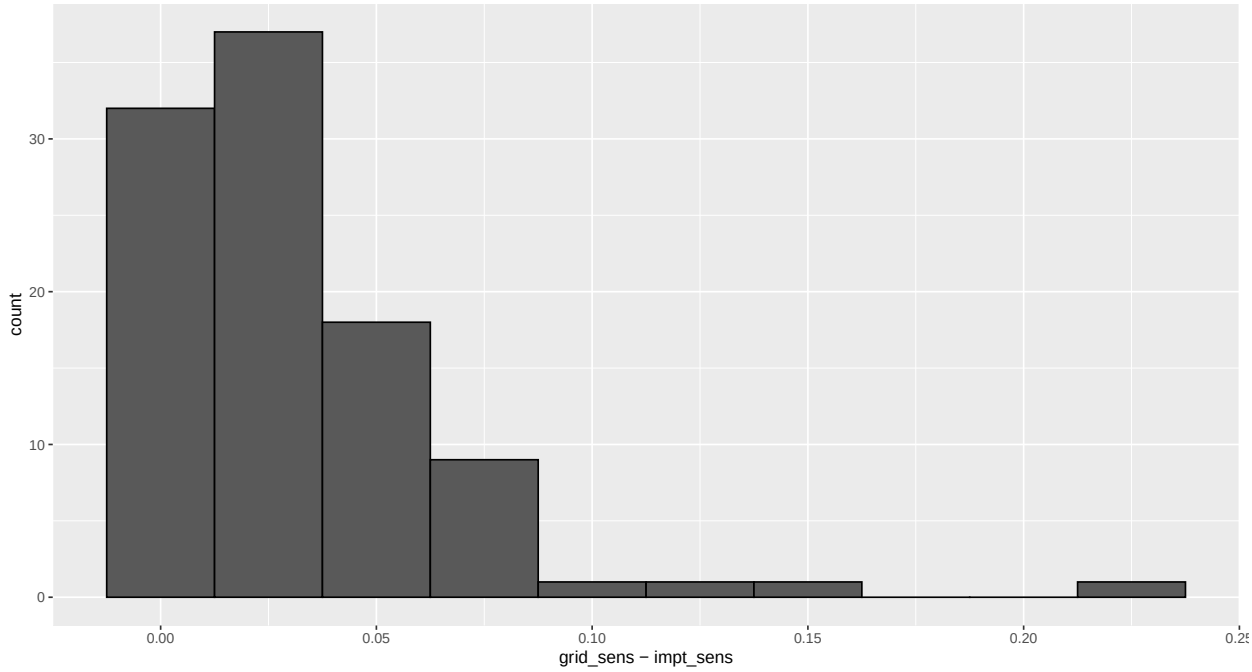
```
# importance sensitivity
summary(impt_sens)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.6833  0.8327  0.9345  0.9114  0.9786  1.0000
```

```
# difference in the sensitivities
summary(grid_sens - impt_sens)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## 0.000000 0.009524 0.021429 0.031143 0.045238 0.228571
```

```
# visualize the difference in the sensitivity
ggplot() + geom_histogram(aes(grid_sens - impt_sens), binwidth = 0.025, color = "black")
```




```
# grid specificity
summary(grid_spec)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.7273  0.8408  0.8934  0.8874  0.9361  1.0000
```

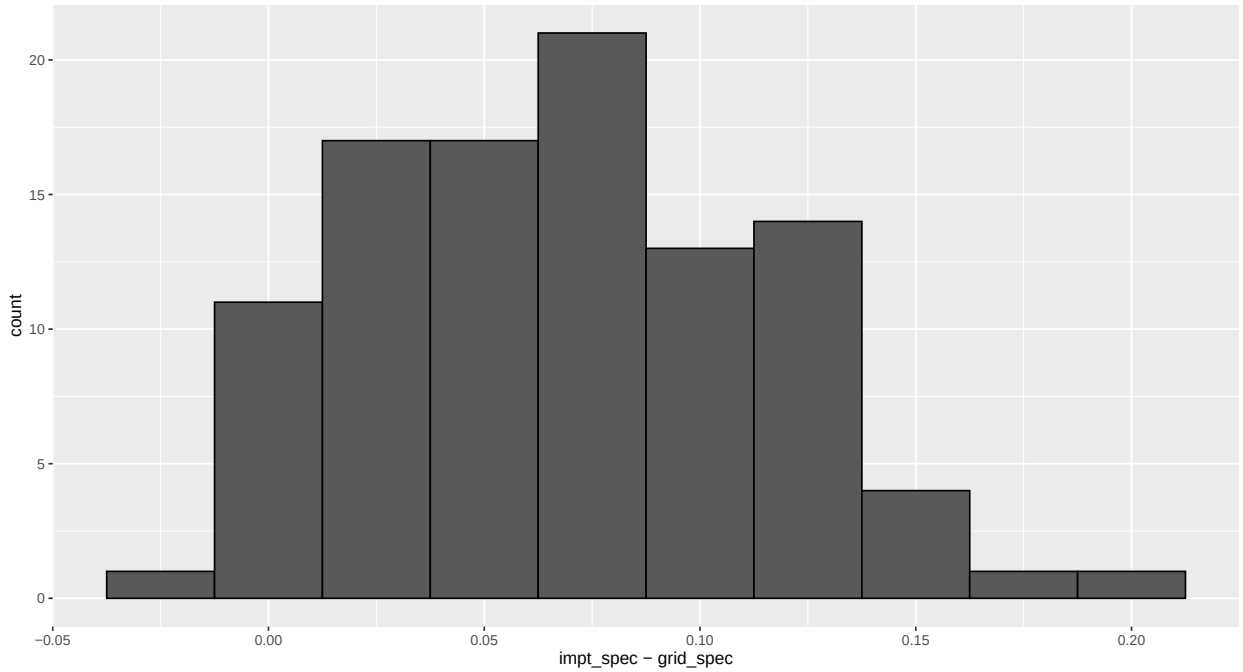
```
# importance specificity
summary(impt_spec)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.8443  0.9384  0.9626  0.9562  0.9847  1.0000
```

```
# difference in the specificities
summary(impt_spec - grid_spec)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -0.01475  0.03333  0.07268  0.06878  0.10260  0.20109
```

```
# visualize the difference in the specificity
ggplot() + geom_histogram(aes(impt_spec - grid_spec), binwidth = 0.025, color = "black")
```



Case Study

Here, I present the graphs estimated by the two methods in two cases; the first case represents the performance of the methods in the trial wherein grid search outperformed model averaging by the greatest margin in terms of sensitivity. The second case represents the performance of the methods in the trial wherein model averaging outperformed grid search by the greatest margin in terms of specificity.

Case 1: Greatest Grid Search Margin of Victory for Sensitivity

```
diff_ind <- which.max(grid_sens - impt_sens)

# sensitivity for grid search
grid_sens[diff_ind]
```

```
## [1] 0.9119048
```

```
# sensitivity for model averaging
impt_sens[diff_ind]
```

```
## [1] 0.6833333
```

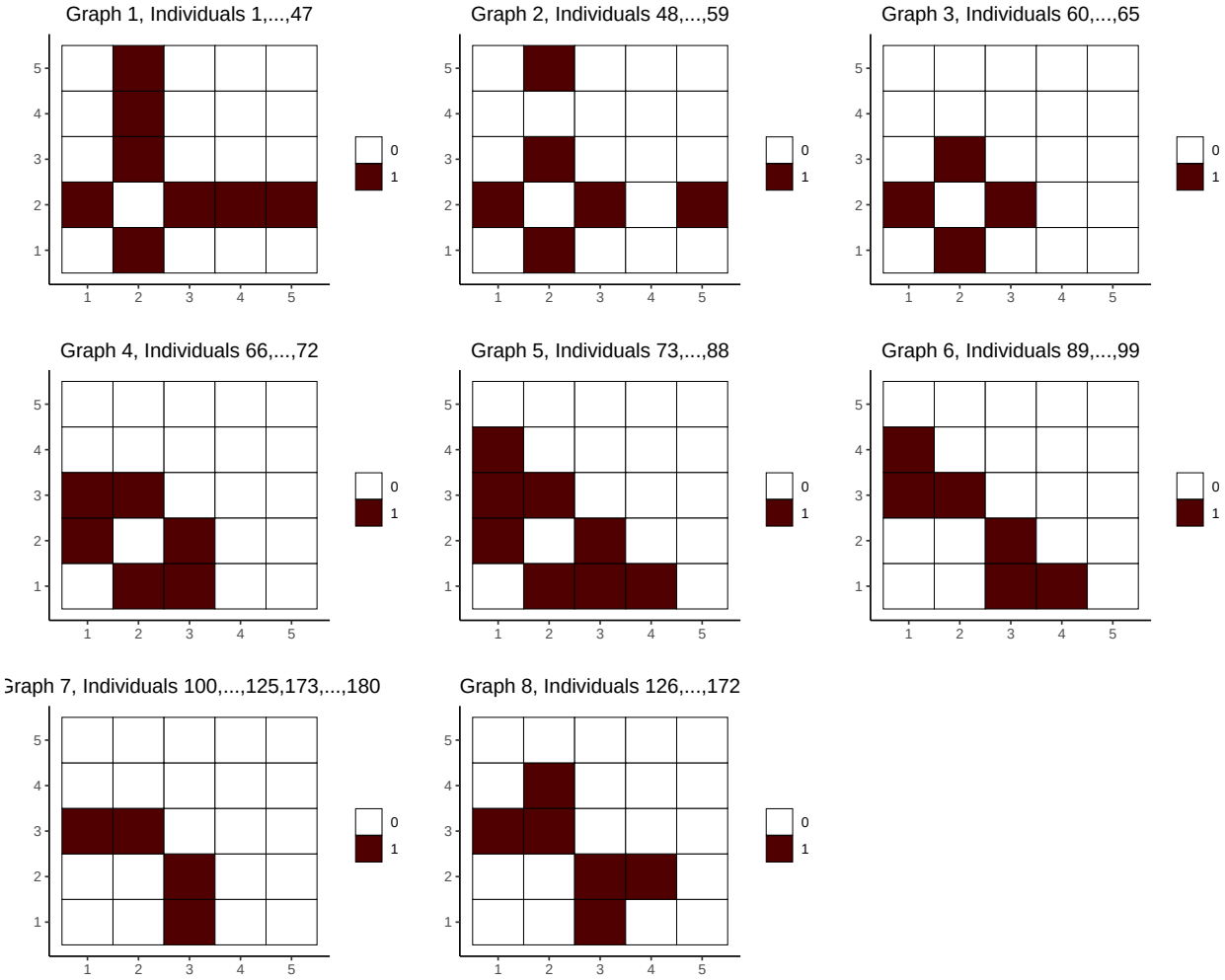
```
# specificity for grid search
grid_spec[diff_ind]
```

```
## [1] 0.9016393
```

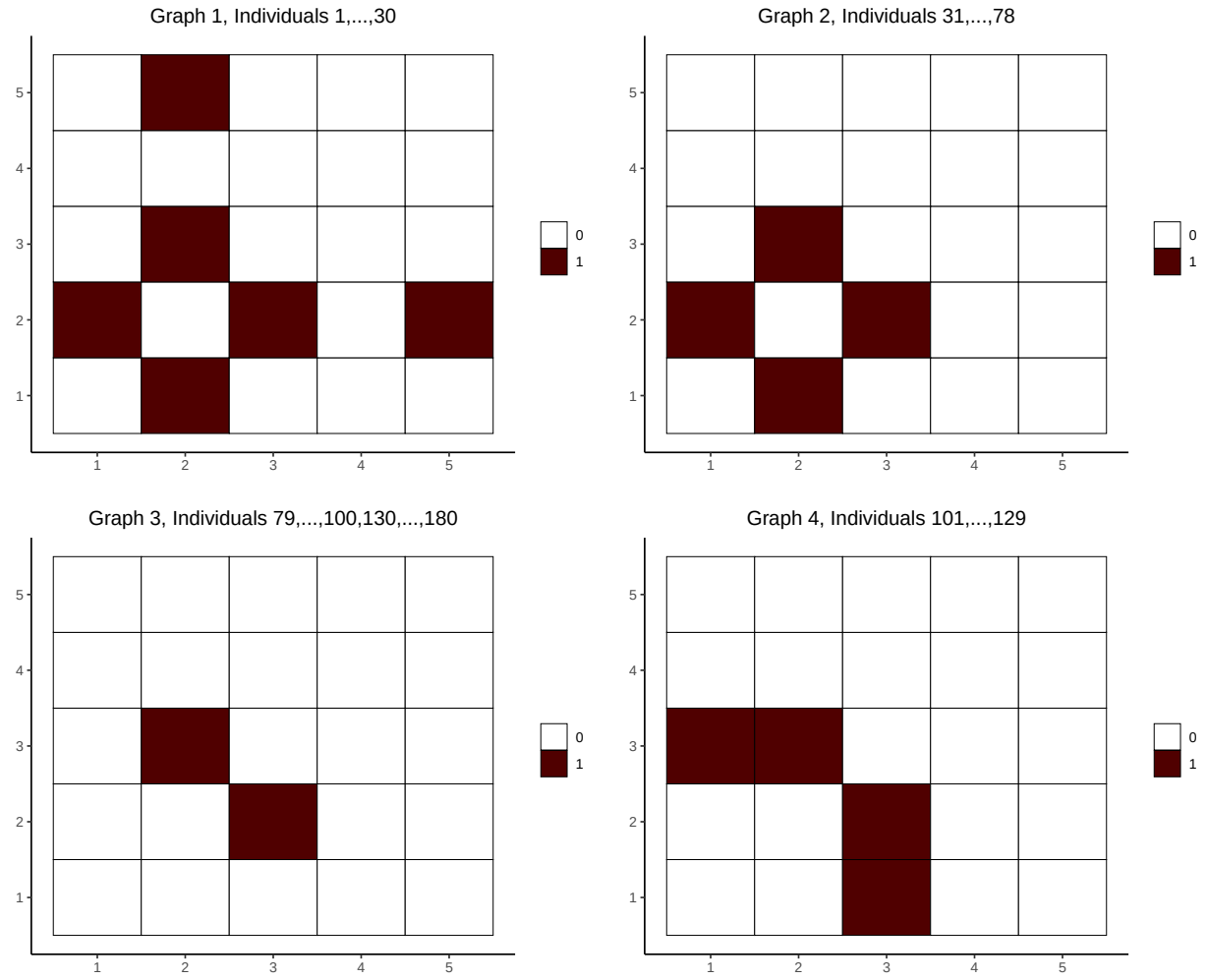
```
# specificity for model averaging
impt_spec[diff_ind]
```

```
## [1] 0.9836066
```

```
# graphs estimated by grid search
ggarrange(plotlist = plot(grid_mods[[diff_ind]]))
```




```
# graphs estimated by model averaging
ggarrange(plotlist = plot(impt_mods[[diff_ind]]))
```



Case 2: Greatest Model Averaging Margin of Victory for Specificity

```
diff_ind <- which.max(impt_spec - grid_spec)

# sensitivity for grid search
grid_sens[diff_ind]
```

```
## [1] 0.9547619
```

```
# sensitivity for model averaging
impt_sens[diff_ind]
```

```
## [1] 0.9142857
```

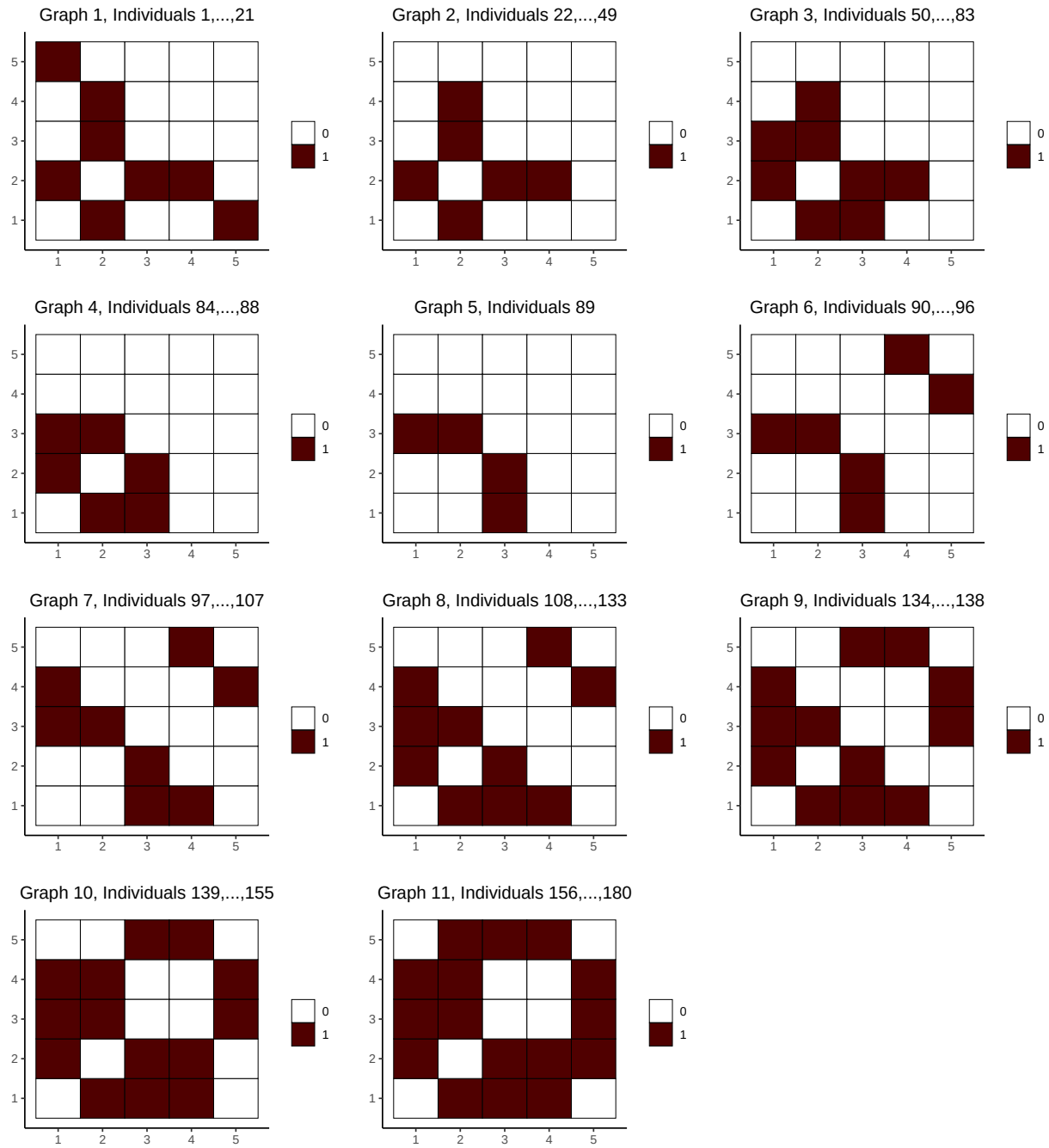
```
# specificity for grid search
grid_spec[diff_ind]
```

```
## [1] 0.7464481
```

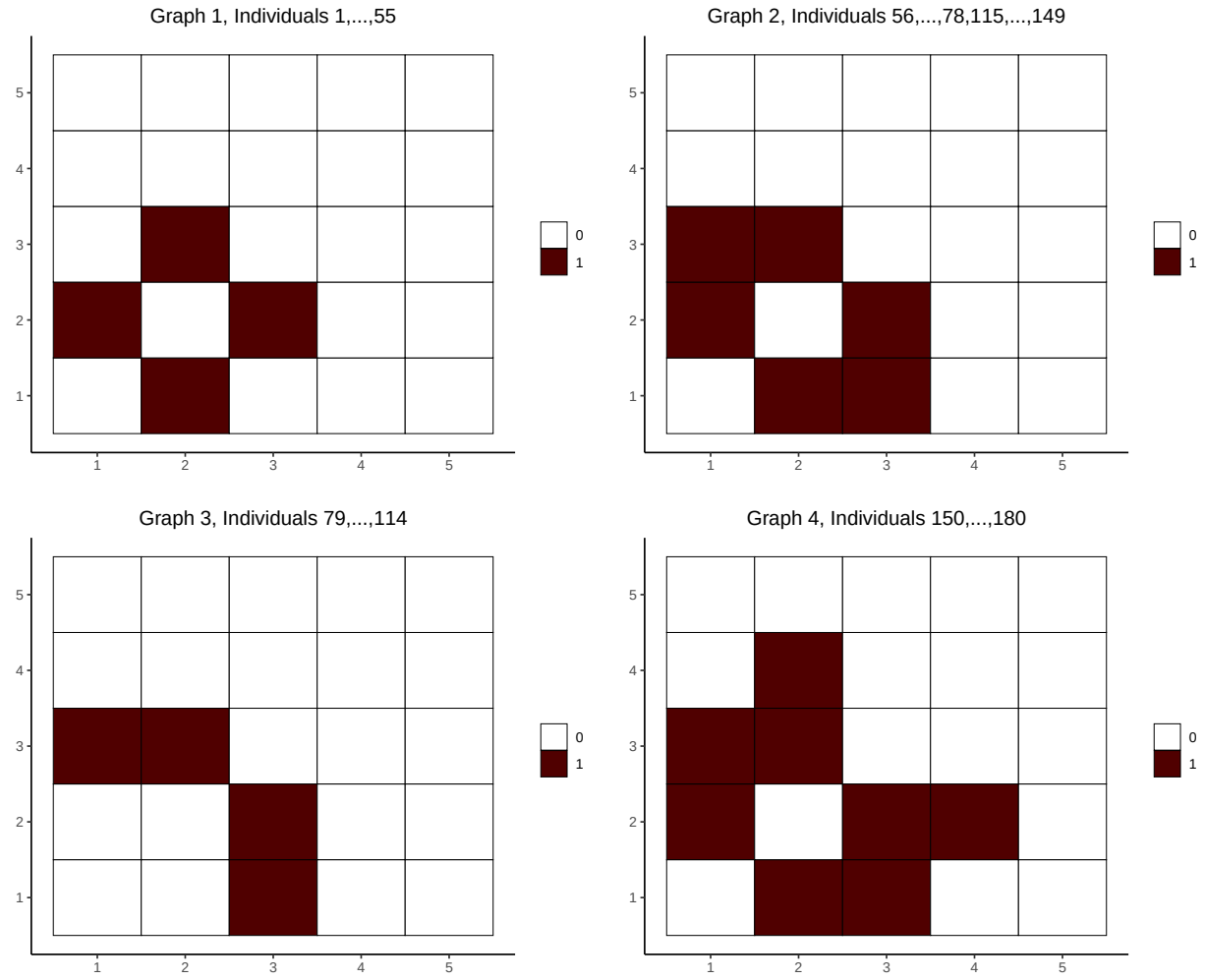
```
# specificity for model averaging
impt_spec[diff_ind]
```

```
## [1] 0.947541
```

```
# graphs estimated by grid search
ggarrange(plotlist = plot(grid_mods[[diff_ind]]), ncol = 3, nrow = 4)
```



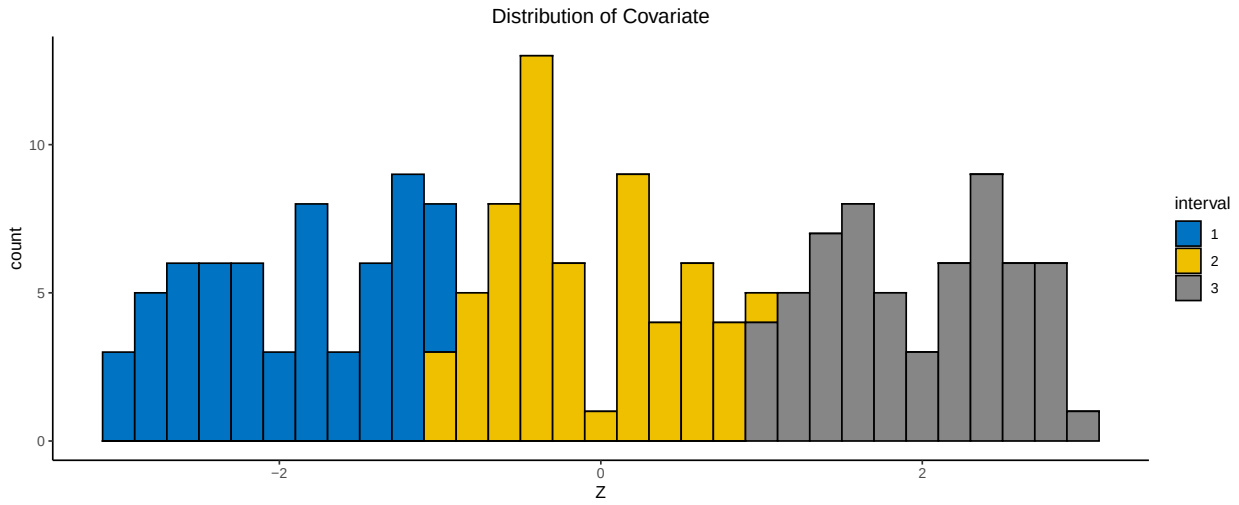
```
# graphs estimated by model averaging
ggarrange(plotlist = plot(impt_mods[[diff_ind]]))
```



Data Generation

Extraneous Covariate

I generated the covariate, Z, as the union of three almost disjoint intervals of equal measure. That is, $Z = Z_1 \cup Z_2 \cup Z_3$ with $Z_1 = (-3, -1), Z_2 = (a, b) = (-1, 1), Z_3 = (1, 3)$. Within each interval, I generated 60 covariate values from a uniform distribution. For example:



Precision Matrix

All of the individuals in interval 1 had the same precision matrix, $\Omega^{(1)}$:

$$\Omega^{(1)}_{i,j} = \begin{cases} 2 & i = j \\ 1 & (i,j) \in \{(1,2),(2,1),(2,3),(3,2)\} \\ 0 & o.w. \end{cases}$$

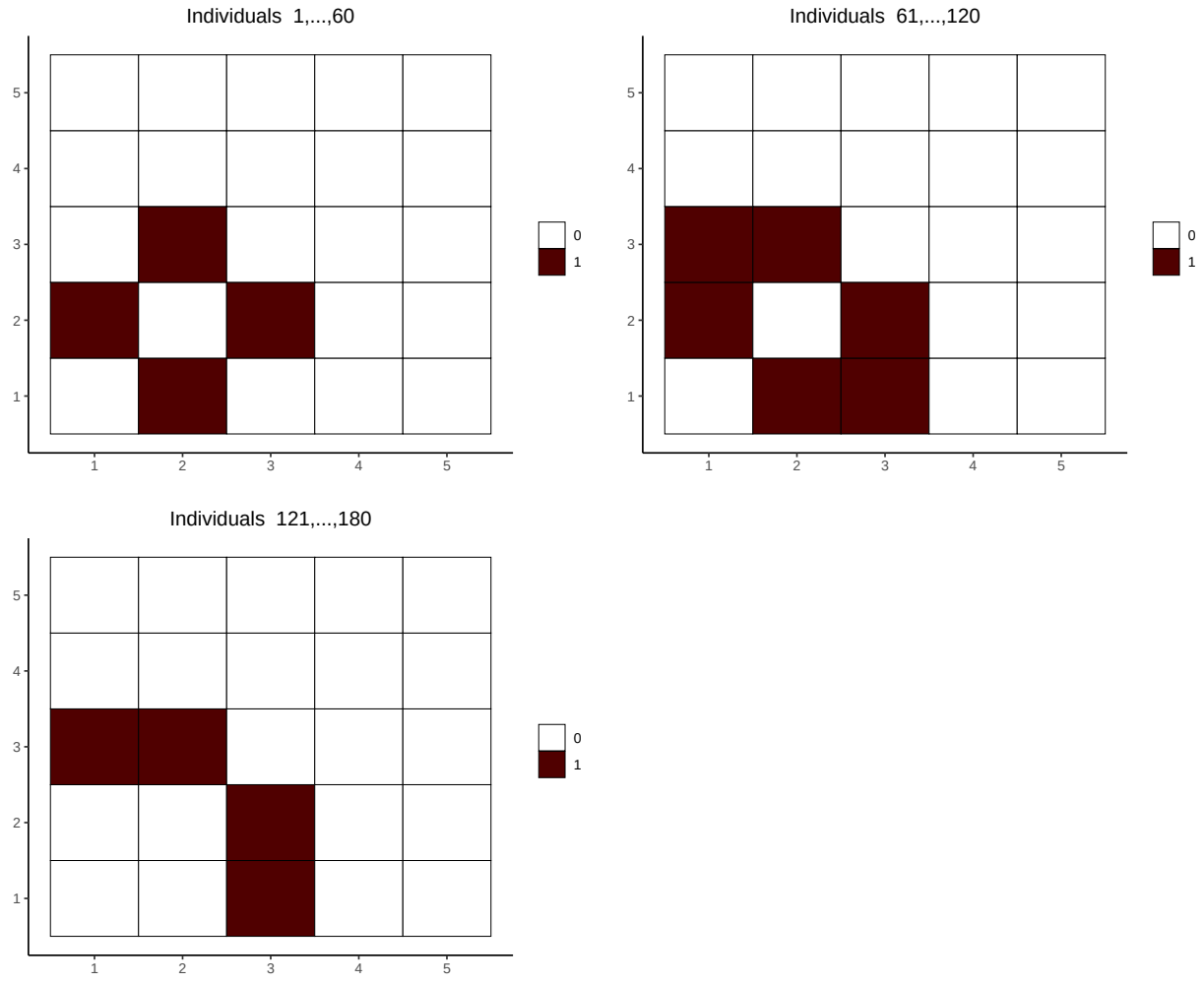
Also, all of the individuals in interval 3 had the same precision matrix, $\Omega^{(3)}$:

$$\Omega^{(3)}_{i,j} = \begin{cases} 2 & i = j \\ 1 & (i,j) \in \{(1,3),(3,1),(2,3),(3,2)\} \\ 0 & o.w. \end{cases}$$

However, the individuals in interval 2 had a precision matrix that was dependent upon Z and (a,b). Let $\beta_0 = -a/(b-a)$ and $\beta_1 = 1/(b-a)$. Then:

$$\Omega^{(2)}_{i,j}(z) = \begin{cases} 2 & i = j \\ 1 & (i,j) \in \{(2,3),(3,2)\} \\ 1-\beta_0-\beta_1 z & (i,j) \in \{(1,2),(2,1)\} \\ \beta_0+\beta_1 z & (i,j) \in \{(1,3),(3,1)\} \\ 0 & o.w. \end{cases}$$

Thus, $\Omega^{(2)}(a) = \Omega^{(1)}$ and $\Omega^{(2)}(b) = \Omega^{(3)}$. That is, an individual on the left or right boundary of Z_2 would have precision matrix $\Omega^{(1)}$ or $\Omega^{(3)}$, respectively. The conditional dependence structures corresponding to each of these precision matrices are visualized below.



Data matrix

Let z_l be the extraneous covariate for the l-th individual. To generate the data matrix for the l-th individual, I took a random sample from $\mathcal{N}(0, \{\Omega_l(z_l)\}^{-1})$, where:

$$\Omega_l(z_l) = \begin{cases} \Omega^{(1)} & z_l \in Z_1 \\ \Omega^{(2)}(z_l) & z_l \in Z_2 \\ \Omega^{(3)} & z_l \in Z_3 \end{cases}$$