Hyperparameter specification experiment results

Summary

In comparing the current and a new method for specifying hyperparameters to `covdepGE`, I found that although the current method produces greater ELBO, the new method consistently outperforms the current one in terms of accuracy. I include visualizations for the differences in metrics, as well as a section analyzing the distribution of the candidate hyperparameters proposed from `varbvs`.

Experiment Overview

In this experiment, I compared two methods of hyperparameter specification across 500 trials. In each trial, the data was generated by setting a seed and then following the continuous data paradigm. That is, all of the individuals in the first and third interval of the covariate space had the same precision matrix (Ω_1 and Ω_3), while the precision matrix for individuals in the second interval continuously shifted from Ω_1 to Ω_3.

The first method (`M1`) follows the methodology for hyperparameter specification that we have already been utilizing. For each variable, a grid of 20 slab variances σ_β^2 are generated independent of the data and the other hyperparameters. The best σ_β^2 is selected based on ELBO. σ^2 is chosen by taking the mean across the candidates fit by `varbvs` and π is chosen by taking the mean across the grid generated by `varbvs`.

To demonstrate `M1`, below are the results for the first two variables from one trial. For each variable, the selected value of the hyperparameters are listed, followed by a value `hyperparameters` that lists each of the elements of the hyperparameter space for which a model was fit, along with the resulting `ELBO`.

```
out1 <- covdepGE(X, Z, var_min = 1e-3, var_max = 1, n_sigma = 20, CS = T)
```

```
##   |                                                                      |
```

```
out1$CAVI_details[1:2]
```

```
## [[1]]
## [[1]]$sigmasq
## [1] 0.7221278
##
## [[1]]$sigmabeta_sq
## [1] 0.1623777
##
## [[1]]$pi
## [1] 0.1398245
##
## [[1]]$ELBO
## [1] -14150.06
##
## [[1]]$converged_iter
```

```
## [1] 20
##
## [[1]]$hyperparameters
##         sigmasq sigmabetasq        pi      elbo
##  [1,] 0.7221278 1.000000000 0.1398245 -14221.08
##  [2,] 0.7221278 0.695192796 0.1398245 -14198.04
##  [3,] 0.7221278 0.483293024 0.1398245 -14177.72
##  [4,] 0.7221278 0.335981829 0.1398245 -14161.50
##  [5,] 0.7221278 0.233572147 0.1398245 -14151.36
##  [6,] 0.7221278 0.162377674 0.1398245 -14150.06
##  [7,] 0.7221278 0.112883789 0.1398245 -14161.37
##  [8,] 0.7221278 0.078475997 0.1398245 -14190.24
##  [9,] 0.7221278 0.054555948 0.1398245 -14242.90
## [10,] 0.7221278 0.037926902 0.1398245 -14326.57
## [11,] 0.7221278 0.026366509 0.1398245 -14448.62
## [12,] 0.7221278 0.018329807 0.1398245 -14615.04
## [13,] 0.7221278 0.012742750 0.1398245 -14828.02
## [14,] 0.7221278 0.008858668 0.1398245 -15083.91
## [15,] 0.7221278 0.006158482 0.1398245 -15372.63
## [16,] 0.7221278 0.004281332 0.1398245 -15678.96
## [17,] 0.7221278 0.002976351 0.1398245 -15984.49
## [18,] 0.7221278 0.002069138 0.1398245 -16270.49
## [19,] 0.7221278 0.001438450 0.1398245 -16521.11
## [20,] 0.7221278 0.001000000 0.1398245 -16725.53
##
##
## [[2]]
## [[2]]$sigmasq
## [1] 0.5723876
##
## [[2]]$sigmabeta_sq
## [1] 0.483293
##
## [[2]]$pi
## [1] 0.1398245
##
## [[2]]$ELBO
## [1] -17084.37
##
## [[2]]$converged_iter
## [1] 33
##
## [[2]]$hyperparameters
##         sigmasq sigmabetasq        pi      elbo
##  [1,] 0.5723876 1.000000000 0.1398245 -17113.63
##  [2,] 0.5723876 0.695192796 0.1398245 -17093.11
##  [3,] 0.5723876 0.483293024 0.1398245 -17084.37
##  [4,] 0.5723876 0.335981829 0.1398245 -17092.66
##  [5,] 0.5723876 0.233572147 0.1398245 -17125.30
##  [6,] 0.5723876 0.162377674 0.1398245 -17192.39
##  [7,] 0.5723876 0.112883789 0.1398245 -17307.53
##  [8,] 0.5723876 0.078475997 0.1398245 -17488.48
##  [9,] 0.5723876 0.054555948 0.1398245 -17757.30
## [10,] 0.5723876 0.037926902 0.1398245 -18139.46
```

```
## [11,] 0.5723876 0.026366509 0.1398245 -18661.02
## [12,] 0.5723876 0.018329807 0.1398245 -19343.11
## [13,] 0.5723876 0.012742750 0.1398245 -20193.69
## [14,] 0.5723876 0.008858668 0.1398245 -21198.65
## [15,] 0.5723876 0.006158482 0.1398245 -22316.71
## [16,] 0.5723876 0.004281332 0.1398245 -23483.29
## [17,] 0.5723876 0.002976351 0.1398245 -24624.05
## [18,] 0.5723876 0.002069138 0.1398245 -25671.98
## [19,] 0.5723876 0.001438450 0.1398245 -26580.43
## [20,] 0.5723876 0.001000000 0.1398245 -27327.71
```

The second method, `M2`, relies entirely on `varbvs` to generate the grid of hyperparameters by fitting σ^2 and σ_β^2 to the data for each value of π in a grid of length 20. Again, the results for the first two variables are displayed.

```
out2 <- covdepGE(X, Z)
```

```
##   |                                                                      |
```

```
out2$CAVI_details[1:2]
```

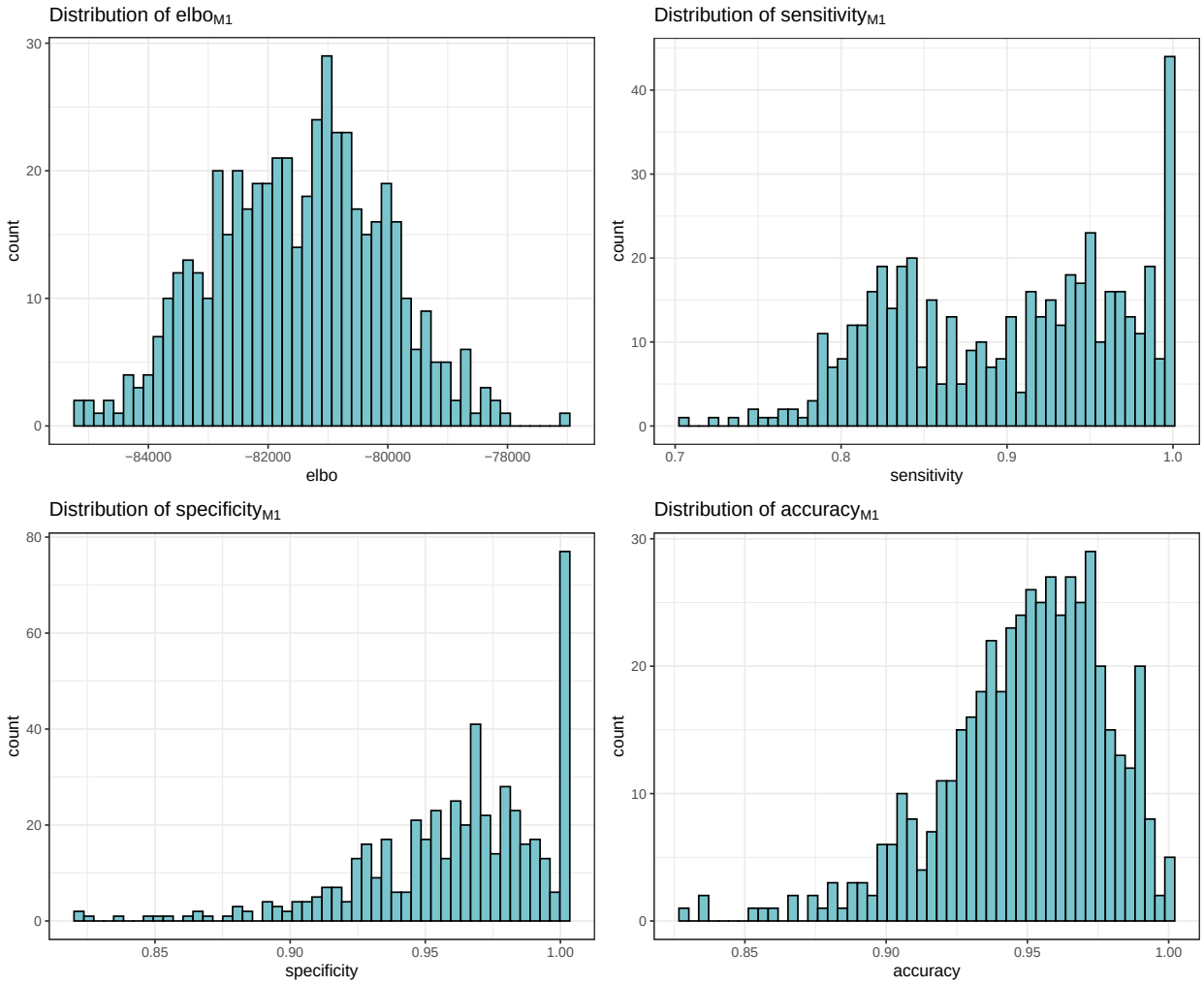
```
## [[1]]
## [[1]]$sigmasq
## [1] 0.7214097
##
## [[1]]$sigmabeta_sq
## [1] 0.9839818
##
## [[1]]$pi
## [1] 0.1923954
##
## [[1]]$ELBO
## [1] -14212.6
##
## [[1]]$converged_iter
## [1] 18
##
## [[1]]$hyperparameters
##         sigmasq sigmabetasq         pi      elbo
##  [1,] 0.7212941   0.9832076 0.20000000 -14212.90
##  [2,] 0.7214097   0.9839818 0.19239536 -14212.60
##  [3,] 0.7215196   0.9847206 0.18501301 -14212.61
##  [4,] 0.7216240   0.9854256 0.17785154 -14212.90
##  [5,] 0.7217233   0.9860982 0.17090914 -14213.47
##  [6,] 0.7218178   0.9867401 0.16418362 -14214.32
##  [7,] 0.7219076   0.9873527 0.15767243 -14215.41
##  [8,] 0.7219930   0.9879372 0.15137269 -14216.76
##  [9,] 0.7220744   0.9884950 0.14528125 -14218.34
## [10,] 0.7221518   0.9890272 0.13939466 -14220.16
## [11,] 0.7222255   0.9895351 0.13370928 -14222.19
## [12,] 0.7222956   0.9900197 0.12822124 -14224.43
## [13,] 0.7223624   0.9904824 0.12292649 -14226.88
```

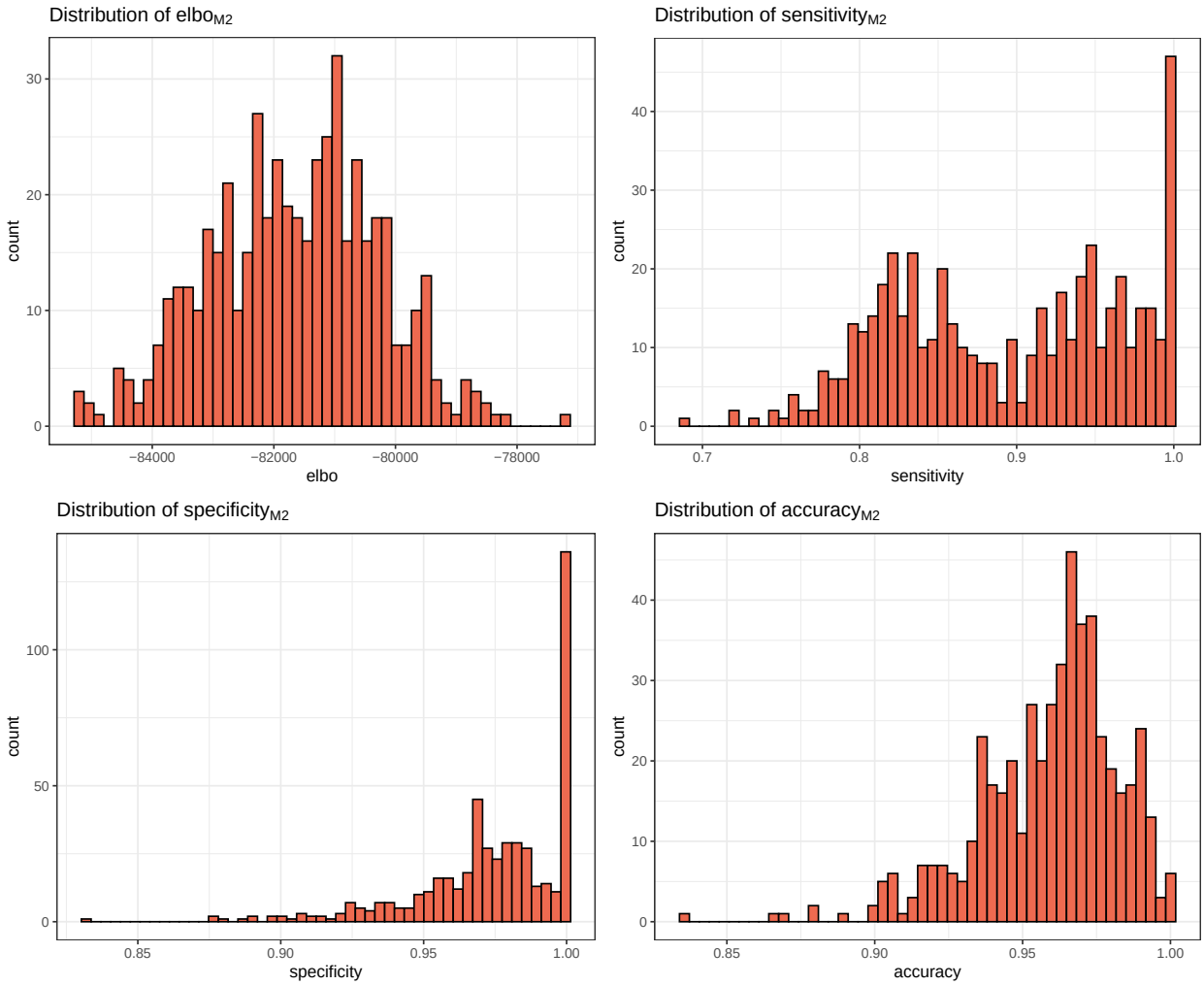
```
## [14,] 0.7224260   0.9909236 0.11782083 -14229.52
## [15,] 0.7224866   0.9913445 0.11289993 -14232.35
## [16,] 0.7225442   0.9917461 0.10815936 -14235.36
## [17,] 0.7225992   0.9921293 0.10359460 -14238.54
## [18,] 0.7226515   0.9924949 0.09920105 -14241.89
## [19,] 0.7227015   0.9928441 0.09497411 -14245.39
## [20,] 0.7227488   0.9931764 0.09090909 -14249.05
##
##
## [[2]]
## [[2]]$sigmasq
## [1] 0.5760302
##
## [[2]]$sigmabeta_sq
## [1] 0.9574463
##
## [[2]]$pi
## [1] 0.09090909
##
## [[2]]$ELBO
## [1] -17100.79
##
## [[2]]$converged_iter
## [1] 32
##
## [[2]]$hyperparameters
##         sigmasq sigmabetasq         pi      elbo
##  [1,] 0.5692560   0.9482048 0.20000000 -17129.56
##  [2,] 0.5695278   0.9486262 0.19239536 -17128.84
##  [3,] 0.5698092   0.9490533 0.18501301 -17128.06
##  [4,] 0.5701031   0.9494897 0.17785154 -17127.12
##  [5,] 0.5704050   0.9499300 0.17090914 -17126.15
##  [6,] 0.5707163   0.9503774 0.16418362 -17125.11
##  [7,] 0.5710374   0.9508319 0.15767243 -17123.97
##  [8,] 0.5713681   0.9512937 0.15137269 -17122.74
##  [9,] 0.5717084   0.9517632 0.14528125 -17121.41
## [10,] 0.5720582   0.9522406 0.13939466 -17119.99
## [11,] 0.5724174   0.9527258 0.13370928 -17118.46
## [12,] 0.5727858   0.9532191 0.12822124 -17116.84
## [13,] 0.5731632   0.9537204 0.12292649 -17115.12
## [14,] 0.5735493   0.9542296 0.11782083 -17113.31
## [15,] 0.5739437   0.9547468 0.11289993 -17111.41
## [16,] 0.5743462   0.9552717 0.10815936 -17109.44
## [17,] 0.5747562   0.9558040 0.10359460 -17107.38
## [18,] 0.5751732   0.9563435 0.09920105 -17105.27
## [19,] 0.5755969   0.9568898 0.09497411 -17103.10
## [20,] 0.5760302   0.9574463 0.09090909 -17100.79
```

Results: ELBO and accuracy

First, as a frame of reference for scale, the distribution of the ELBO, sensitivity, specificity, and accuracy for both `M1` and `M2` is visualized.

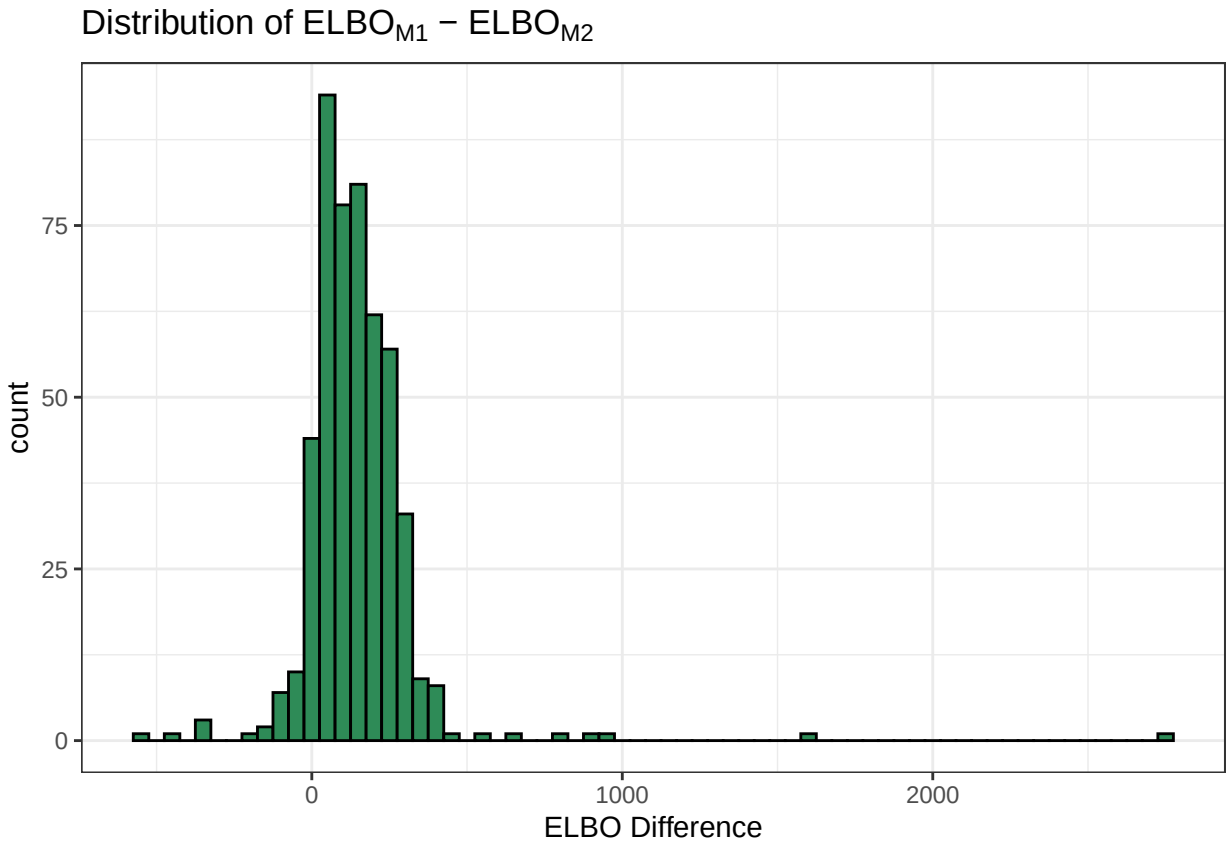
Note: One of the ELBO for `M2` *was removed, as it was on the order of* 1×10^{12}*. However, all other metrics for this model were within 3% of that of* `M1`*, and so these were not dropped*





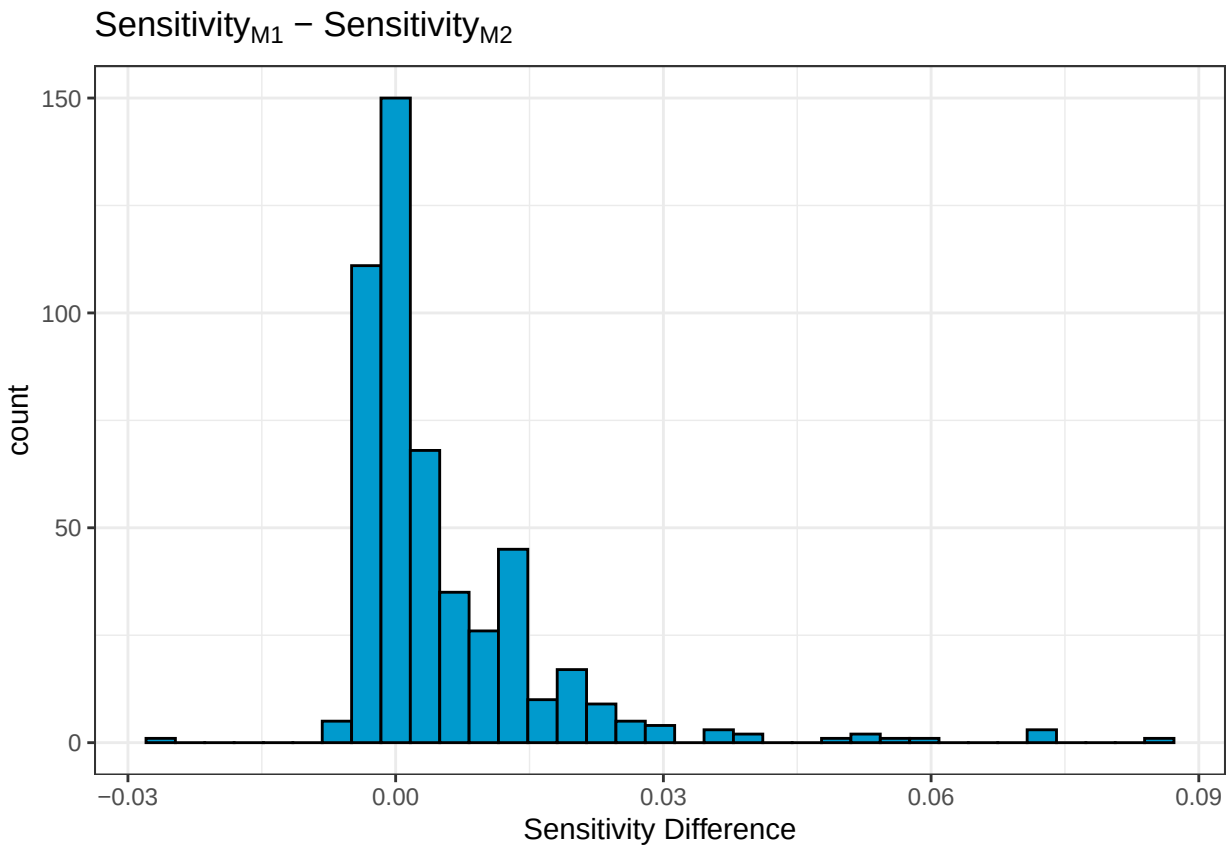
Visualized next is the pairwise difference between the ELBO of `M1` and `M2`. The ELBO of `M1` was more frequently greater than that of `M2`

Note: the large ELBO for `M2` was again removed

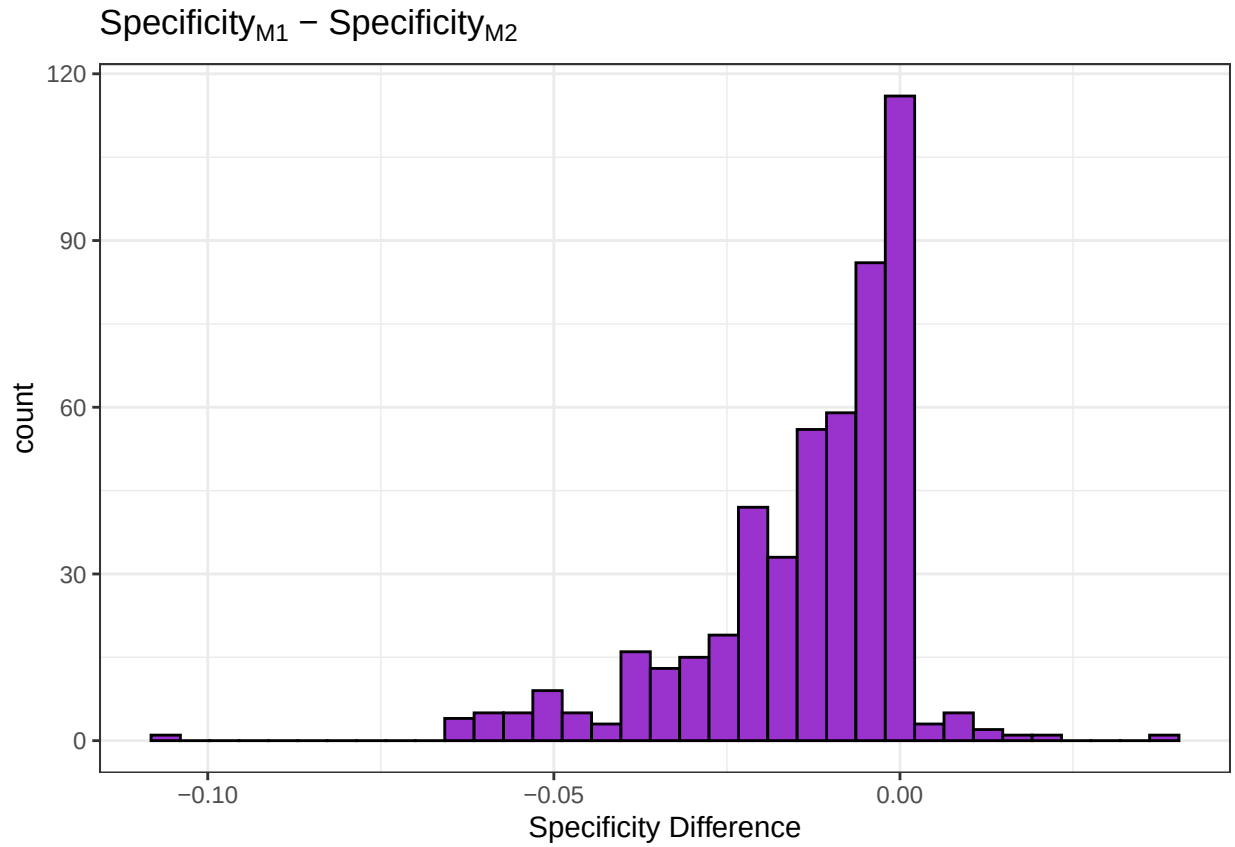



```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  -543.2    58.3   134.5   146.4   217.3  2764.3
```

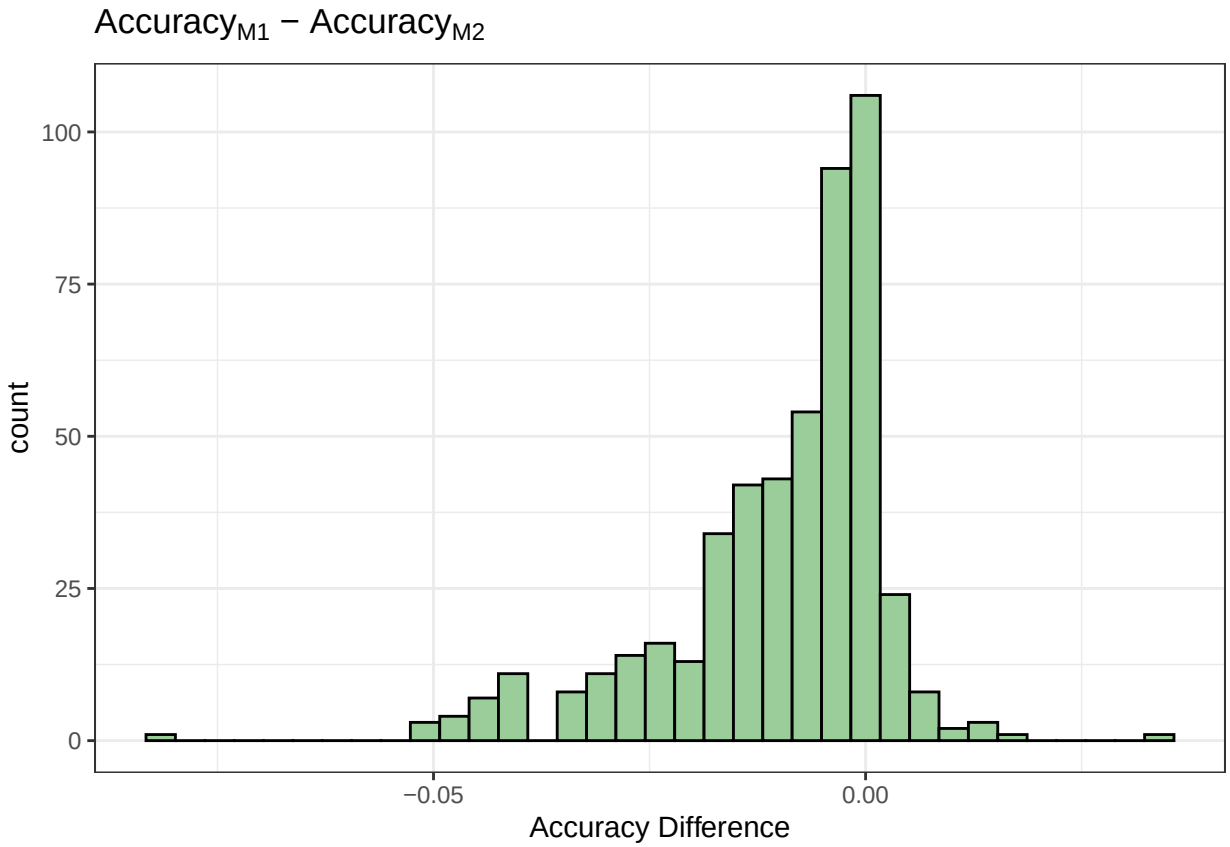
Visualized next is the pairwise differences in sensitivity, specificity, and accuracy. Although the rate at which edges were correctly detected (sensitivity) was roughly equivalent between the methods, `M2` outperformed `M1` in non-edge detection rate (specificity), which resulted in the superior accuracy of `M2`.




```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -0.026190  0.000000  0.000000  0.005333  0.009524  0.085714
```




```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -0.104372 -0.020219 -0.008743 -0.013143 -0.001639  0.039891
```




```
##       Min.    1st Qu.     Median       Mean    3rd Qu.       Max.
## -0.0813333 -0.0151111 -0.0057778 -0.0096942 -0.0004444  0.0342222
```

Distribution of Candidates

The `pi` candidates generated by `varbvs` result from a grid of log-odds spanning from $-\log_{10}(p)$ to $-\log_{10}(1)$:

```
p <- 4
log.odds <- seq(-log10(p), -1, length.out = 20)
(pi <- 1 / (1 + 10^(-log.odds)))
```

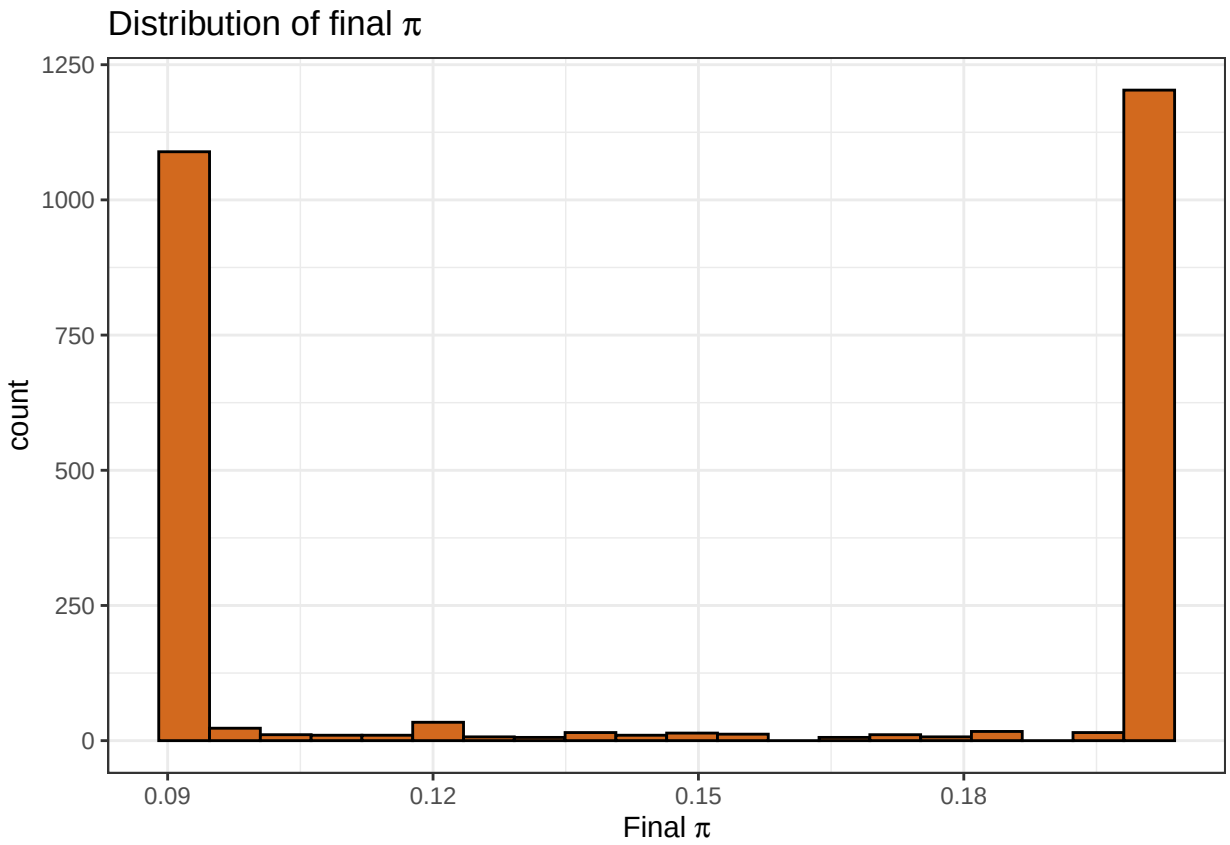
```
##  [1] 0.20000000 0.19239536 0.18501301 0.17785154 0.17090914 0.16418362
##  [7] 0.15767243 0.15137269 0.14528125 0.13939466 0.13370928 0.12822124
## [13] 0.12292649 0.11782083 0.11289993 0.10815936 0.10359460 0.09920105
## [19] 0.09497411 0.09090909
```

```
mean(pi)
```

```
## [1] 0.1398245
```

Since `M1` uses the mean of these candidates, all of the hyperparameter candidates for `M1` use $\pi \approx 0.14$, as can be seen from the demonstration of `M1` in the experiment overview section. However, `M2` fits a model for each of the candidates in the π grid.

The following is a visualization of the final values of π chosen by the `M2` models. Since each variable can have its own value of π, and there are 5 variables per model, this distribution contains 2,500 values of π. This visualization shows that ELBO favors values of π on the extremes of the distribution.



I proposed `M2` with the reasoning that since it fits σ^2 and σ_β^2 to the data for each π in the grid, it would cover a greater portion of the hyperparameter space than `M1`. It is important to note that after analyzing the spread of σ^2 and σ_β^2 fit to each variable by `varbvs`, it became apparent that the candidates were relatively similar to one another.

For reference to the magnitude of these candidates, here are summary statistics for the σ^2 candidates fit to the data by `varbvs`:

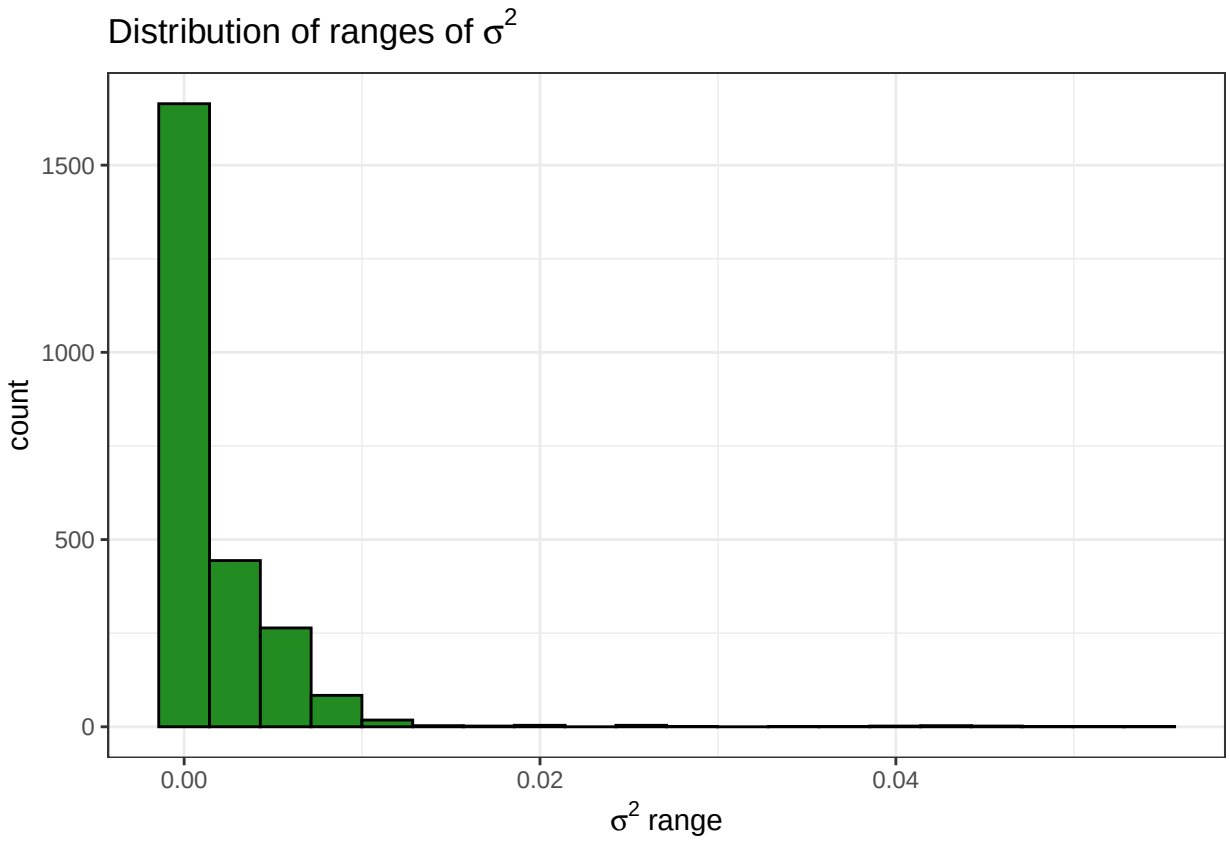
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.3268  0.4804  0.5214  0.5327  0.5722  0.9960
```

And the σ_β^2 candidates:

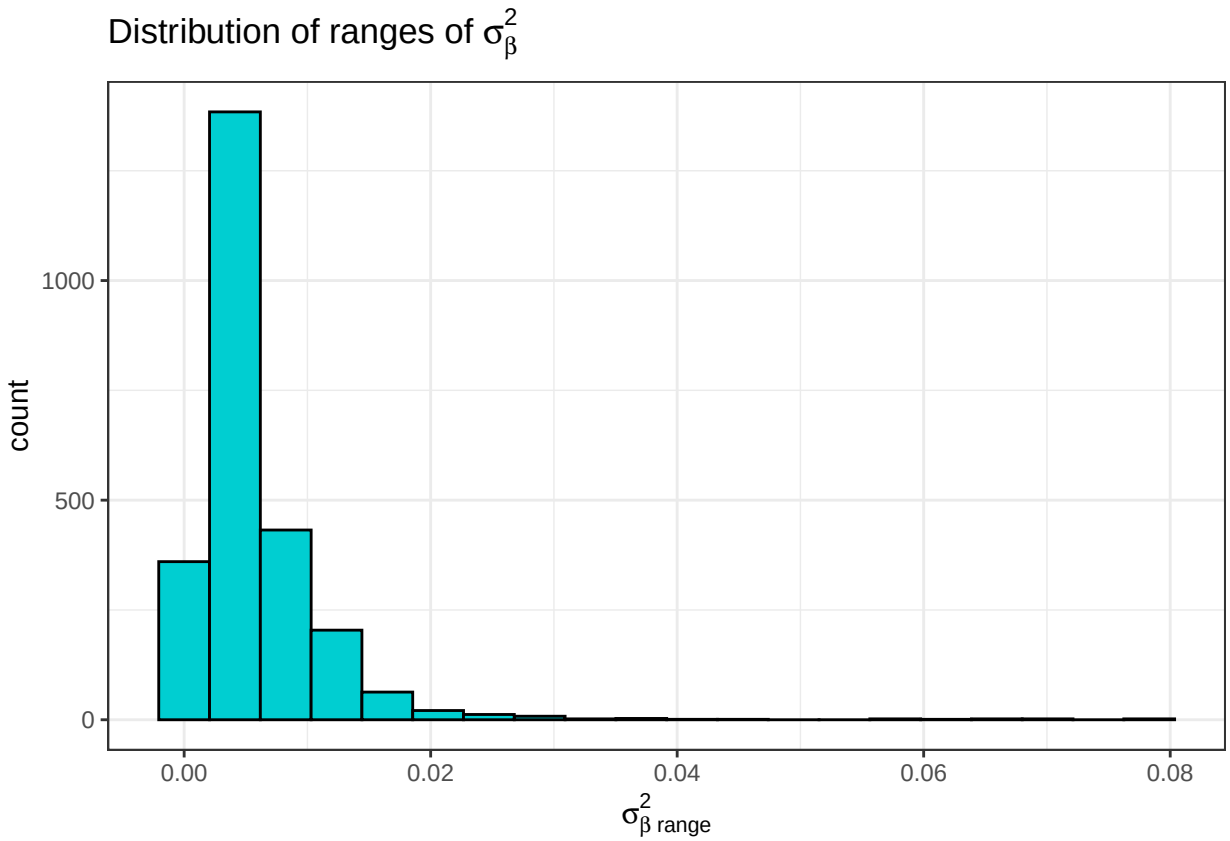
```
summary(as.numeric(res$hyperparameters$sigmabetasq_cands))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.8692  0.9401  0.9636  0.9630  0.9944  1.0039
```

The following visualizations are the distribution of the ranges of these candidates proposed by `varbvs`. For each variable in a model, `varbvs` fits σ^2 and σ_β^2 to the data for each of the 20 π values in the grid. I defined the range as the difference between the largest and smallest hyperparameter fit to the data; since there were 500 models, and 5 variables per model, both of these distributions represent 2,500 ranges.




```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 6.000e-08 1.578e-04 5.914e-04 1.994e-03 2.470e-03 5.424e-02
```




```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## 0.001105 0.002679 0.003999 0.005806 0.007253 0.079401
```