SuSiE L Sensitivity 2

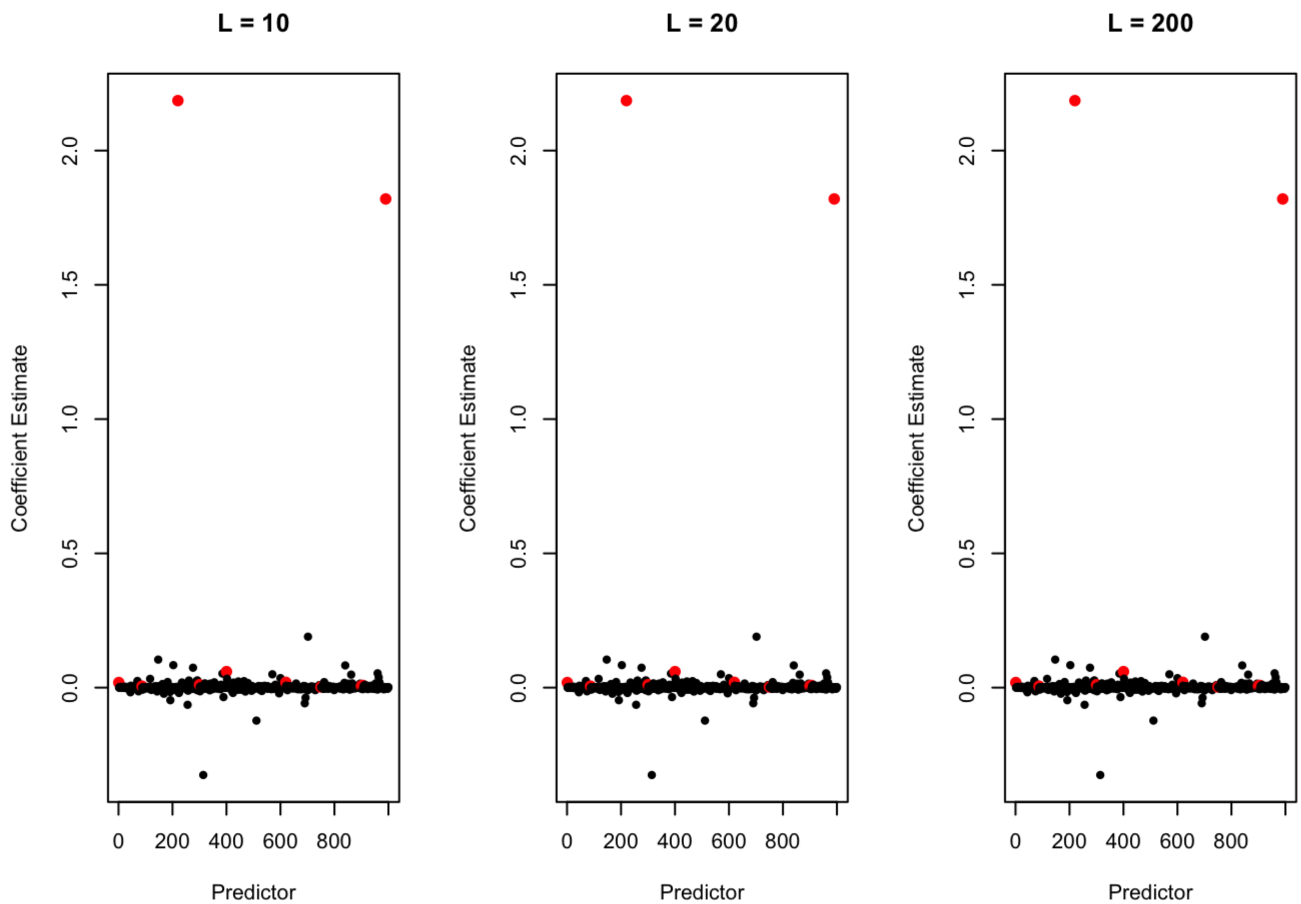
Stuart Brabbs

6/8/2021

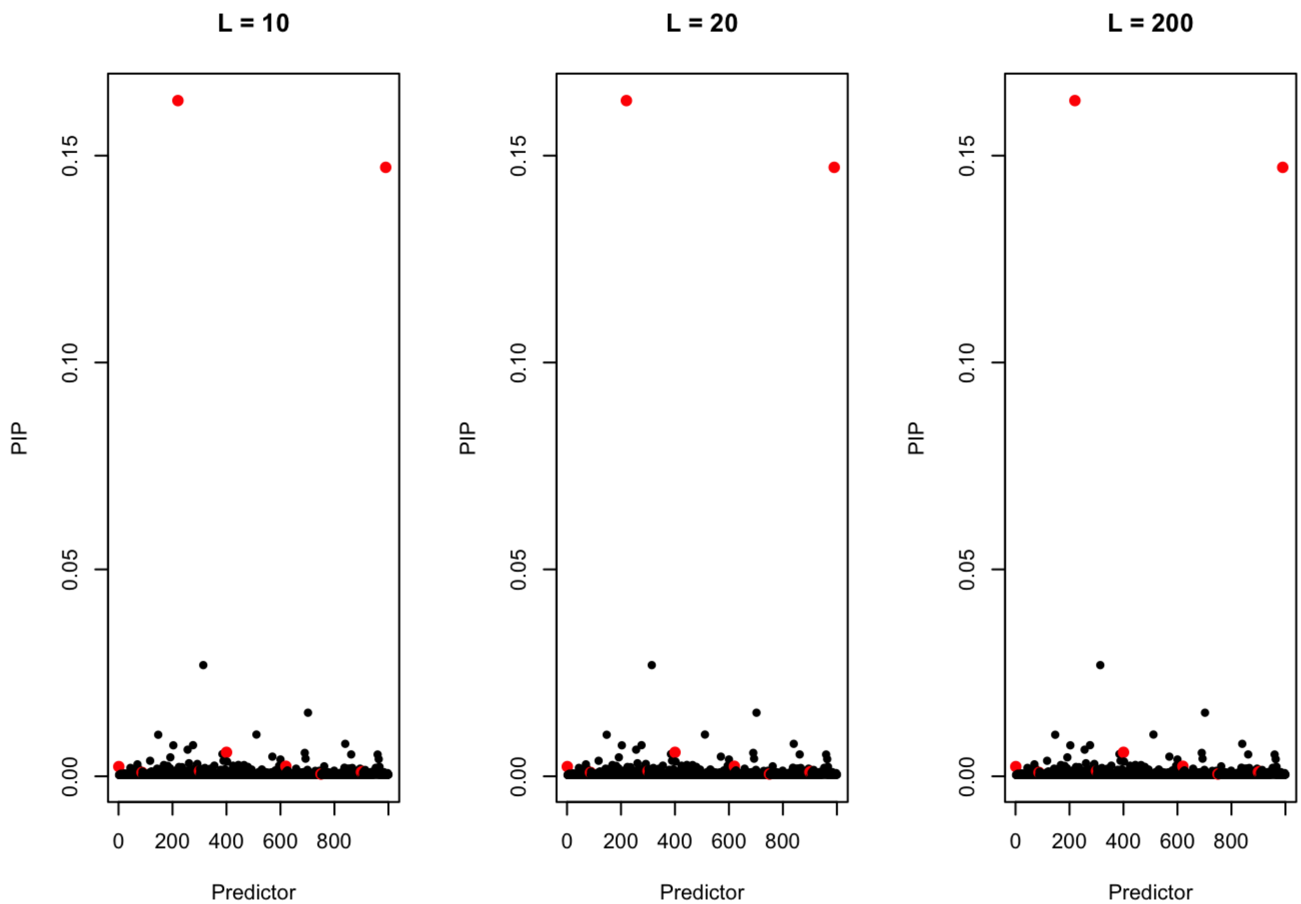
Our goal is to continue from the previous L sensitivity tests, and this time introduce greater noise and variability in signals. The previous tests used a minimal example of n = 1000 and p = 1000 with all signals equal to 1 and all other variable coefficients equal to 0. This time, we will assign varying sizes to the signals and draw coeffients for noise variables from a normal distribution, and use n = 100 and p = 1000.

We begin with an example where all signals are equal to 10, but noise coefficients are drawn from N(0,1). Note that red points are true signals. First we plot increasing L:

```
set.seed(1)
index <- c(1:1000)
n <- 100
p <- 1000
beta <- rnorm(1000, sd = 1)
signals <- c(1,87,220,300,400,620,750,900,905,990)
beta[c(1,87,220,300,400,620,750,900,905,990)] <- 10
X <- matrix(rnorm(n*p, sd = 5), nrow=n, ncol=p)
y <- X %*% beta + rnorm(n, sd = 3)
res1 <- susie(X, y, L = 10)

res2 <- susie(X, y, L = 20)

res3 <- susie(X, y, L = 200)

par(mfrow = c(1,3))
plot(index, coef(res1)[-1], xlab = "Predictor", ylab = "Coefficient Estimate", main = "L
= 10", col = ifelse(index %in% signals, "red", "black"), pch = ifelse(index %in% signal
s, 19, 20))
plot(index, coef(res2)[-1], xlab = "Predictor", ylab = "Coefficient Estimate", main = "L
= 20", col = ifelse(index %in% signals, "red", "black"), pch = ifelse(index %in% signal
s, 19, 20))
plot(index, coef(res3)[-1], xlab = "Predictor", ylab = "Coefficient Estimate", main = "L
= 200", col = ifelse(index %in% signals, "red", "black"), pch = ifelse(index %in% signal
s, 19, 20))
```

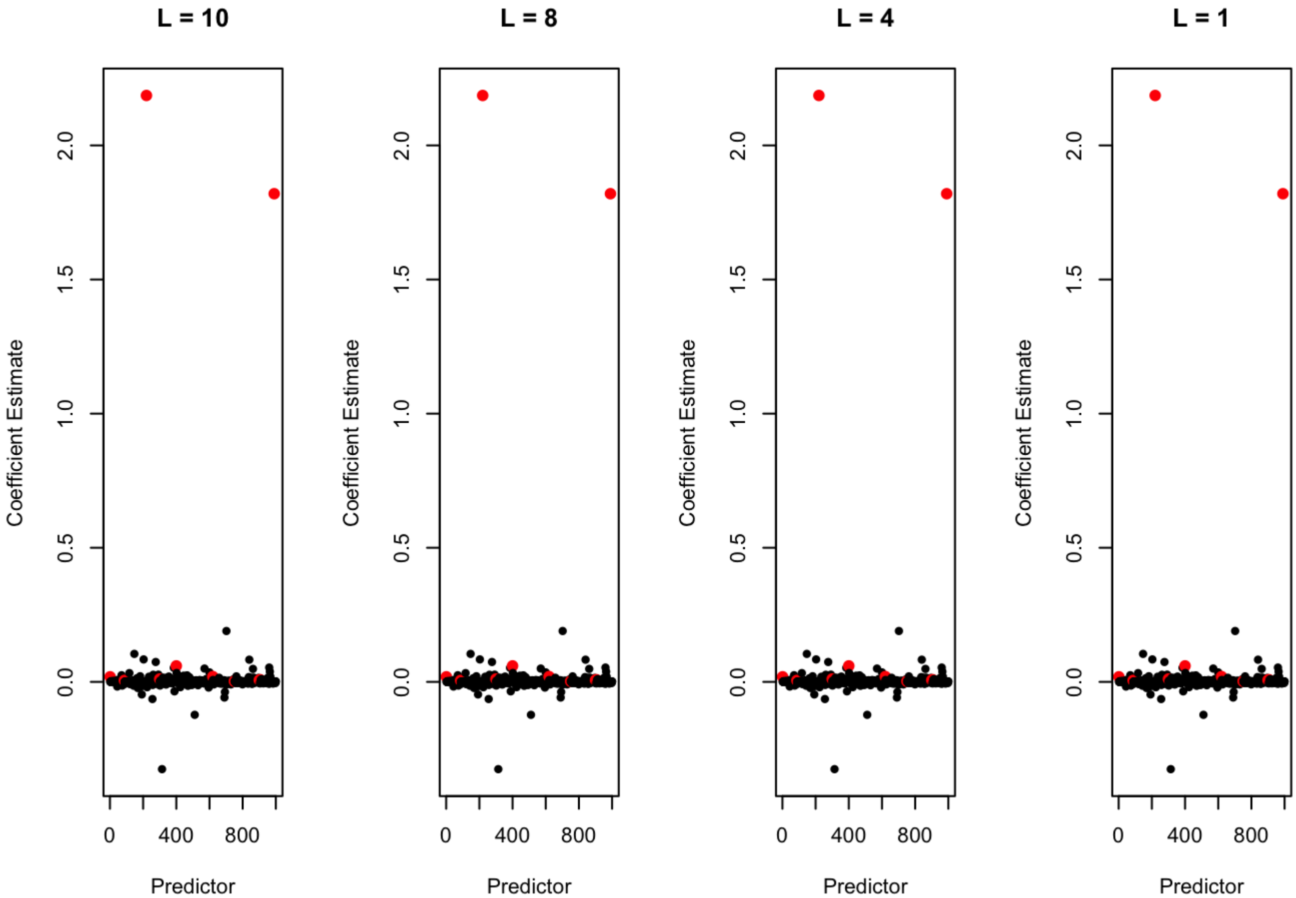



```
par(mfrow = c(1,3))
plot(index, res1$pip, xlab = "Predictor", ylab = "PIP", main = "L = 10", col = ifelse(in
dex %in% signals, "red", "black"), pch = ifelse(index %in% signals, 19, 20))
plot(index, res2$pip, xlab = "Predictor", ylab = "PIP", main = "L = 20", col = ifelse(in
dex %in% signals, "red", "black"), pch = ifelse(index %in% signals, 19, 20))
plot(index, res3$pip, xlab = "Predictor", ylab = "PIP", main = "L = 200", col = ifelse(i
ndex %in% signals, "red", "black"), pch = ifelse(index %in% signals, 19, 20))
```

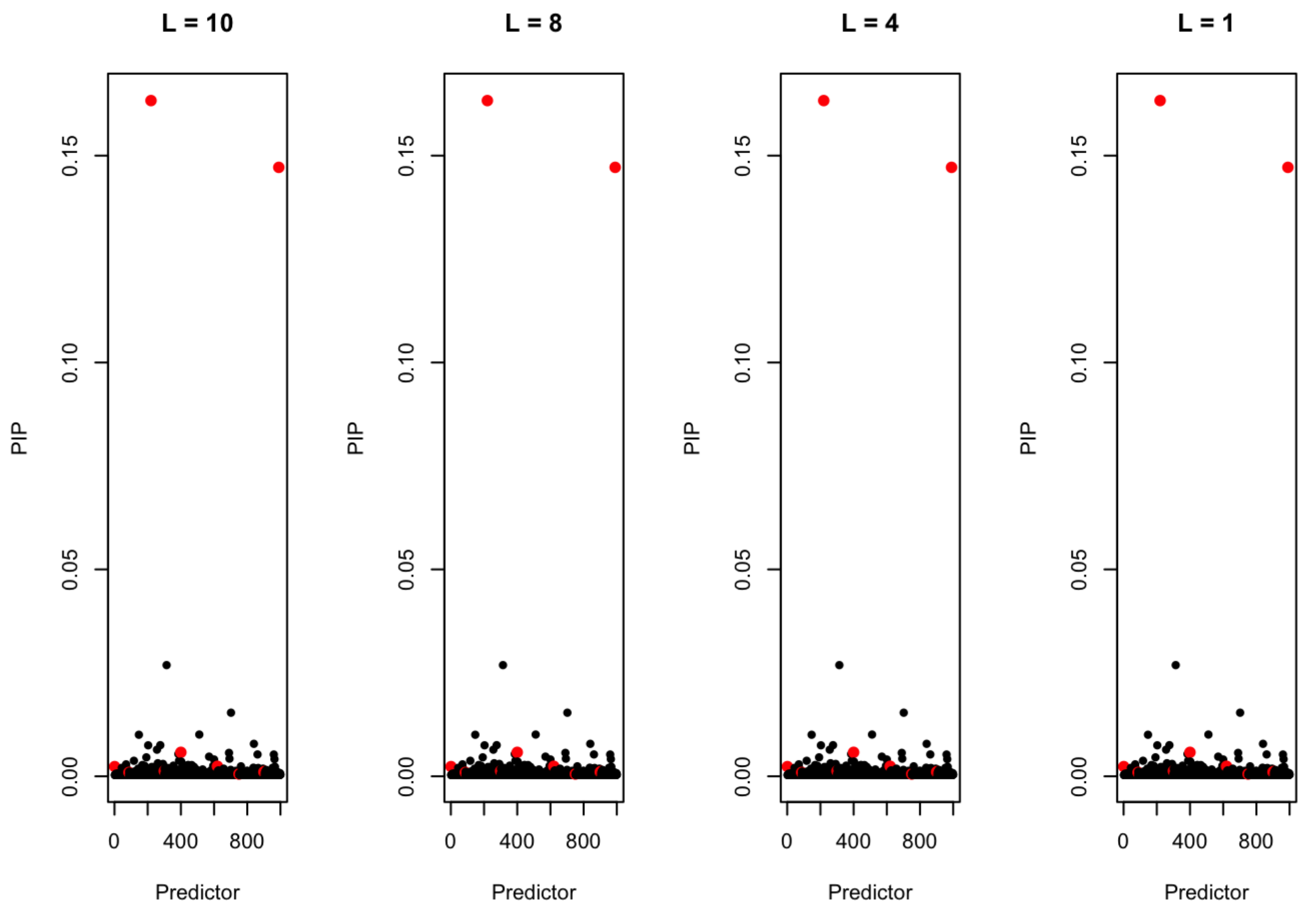


And decreasing L:

```
res4 <- susie(X, y, L = 8)

res5 <- susie(X, y, L = 4)

res55 <- susie(X, y, L = 1)

par(mfrow = c(1,4))
plot(index, coef(res1)[-1], xlab = "Predictor", ylab = "Coefficient Estimate", main = "L
= 10", col = ifelse(index %in% signals, "red", "black"), pch = ifelse(index %in% signal
s, 19, 20))
plot(index, coef(res4)[-1], xlab = "Predictor", ylab = "Coefficient Estimate", main = "L
= 8", col = ifelse(index %in% signals, "red", "black"), pch = ifelse(index %in% signals,
19, 20))
plot(index, coef(res5)[-1], xlab = "Predictor", ylab = "Coefficient Estimate", main = "L
= 4", col = ifelse(index %in% signals, "red", "black"), pch = ifelse(index %in% signals,
19, 20))
plot(index, coef(res55)[-1], xlab = "Predictor", ylab = "Coefficient Estimate", main =
"L = 1", col = ifelse(index %in% signals, "red", "black"), pch = ifelse(index %in% signa
ls, 19, 20))
```

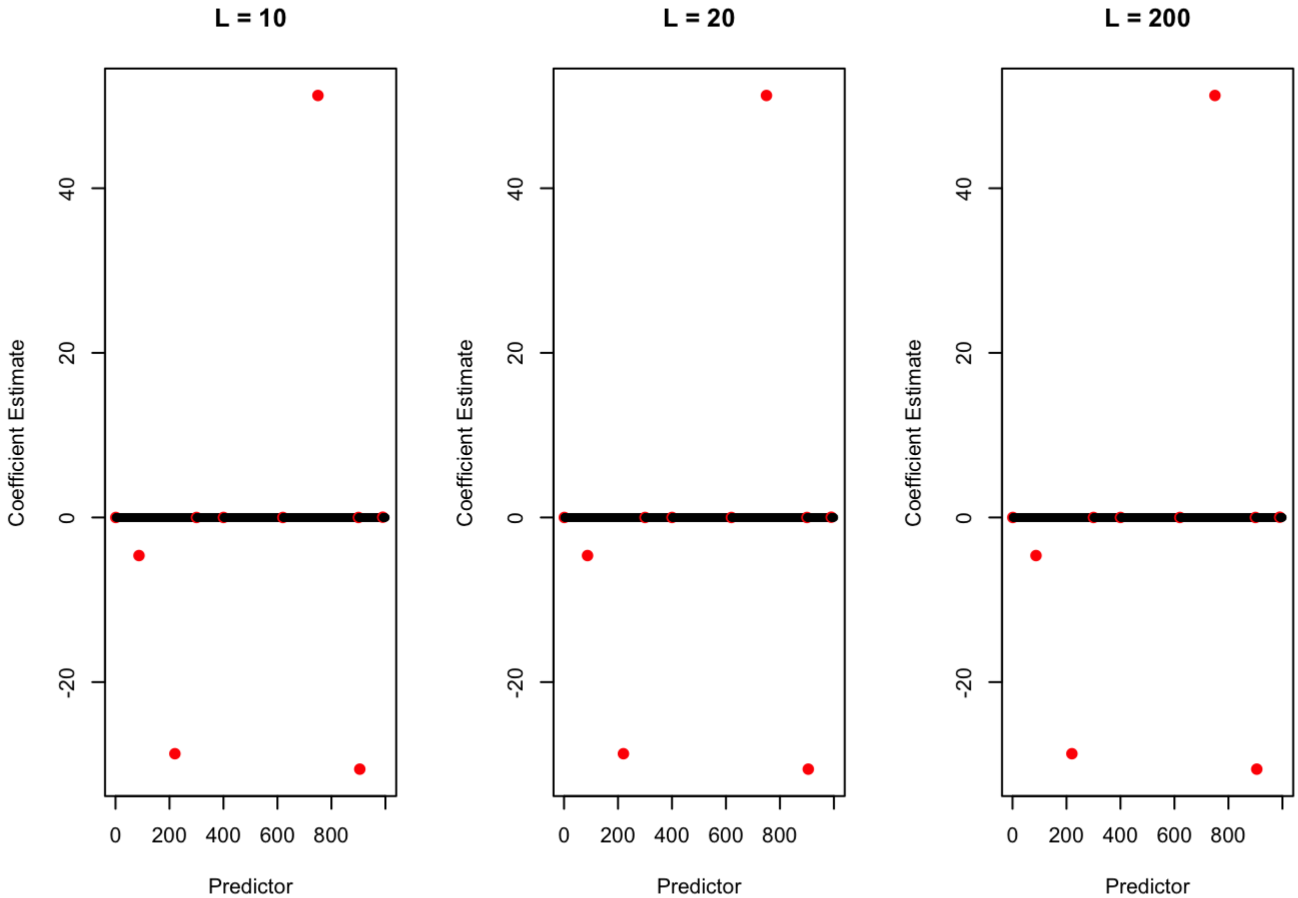



```
par(mfrow = c(1,4))
plot(index, res1$pip, xlab = "Predictor", ylab = "PIP", main = "L = 10", col = ifelse(in
dex %in% signals, "red", "black"), pch = ifelse(index %in% signals, 19, 20))
plot(index, res4$pip, xlab = "Predictor", ylab = "PIP", main = "L = 8", col = ifelse(ind
ex %in% signals, "red", "black"), pch = ifelse(index %in% signals, 19, 20))
plot(index, res5$pip, xlab = "Predictor", ylab = "PIP", main = "L = 4", col = ifelse(ind
ex %in% signals, "red", "black"), pch = ifelse(index %in% signals, 19, 20))
plot(index, res55$pip, xlab = "Predictor", ylab = "PIP", main = "L = 1", col = ifelse(in
dex %in% signals, "red", "black"), pch = ifelse(index %in% signals, 19, 20))
```

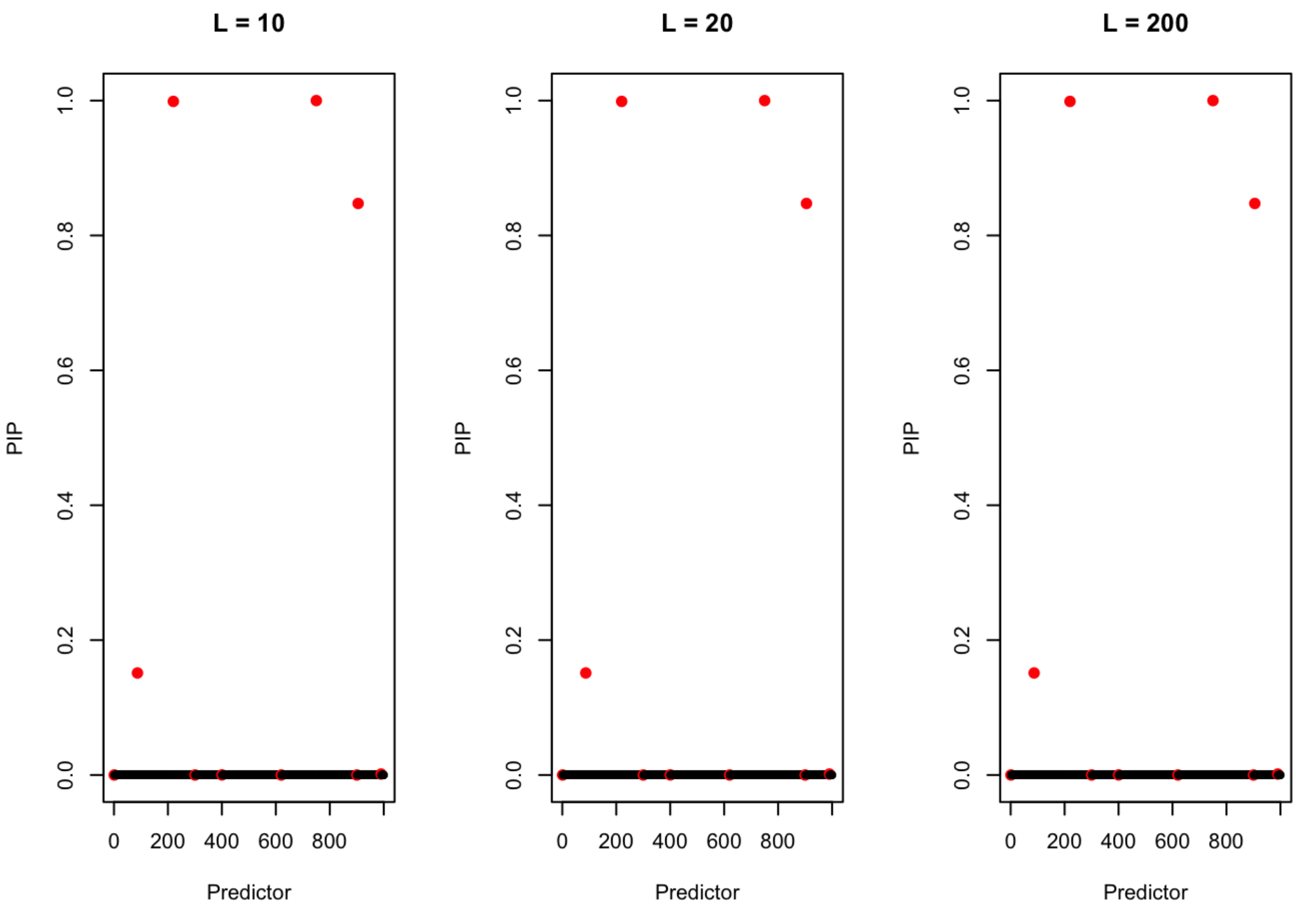


We then plot the same but with varying signal sizes:

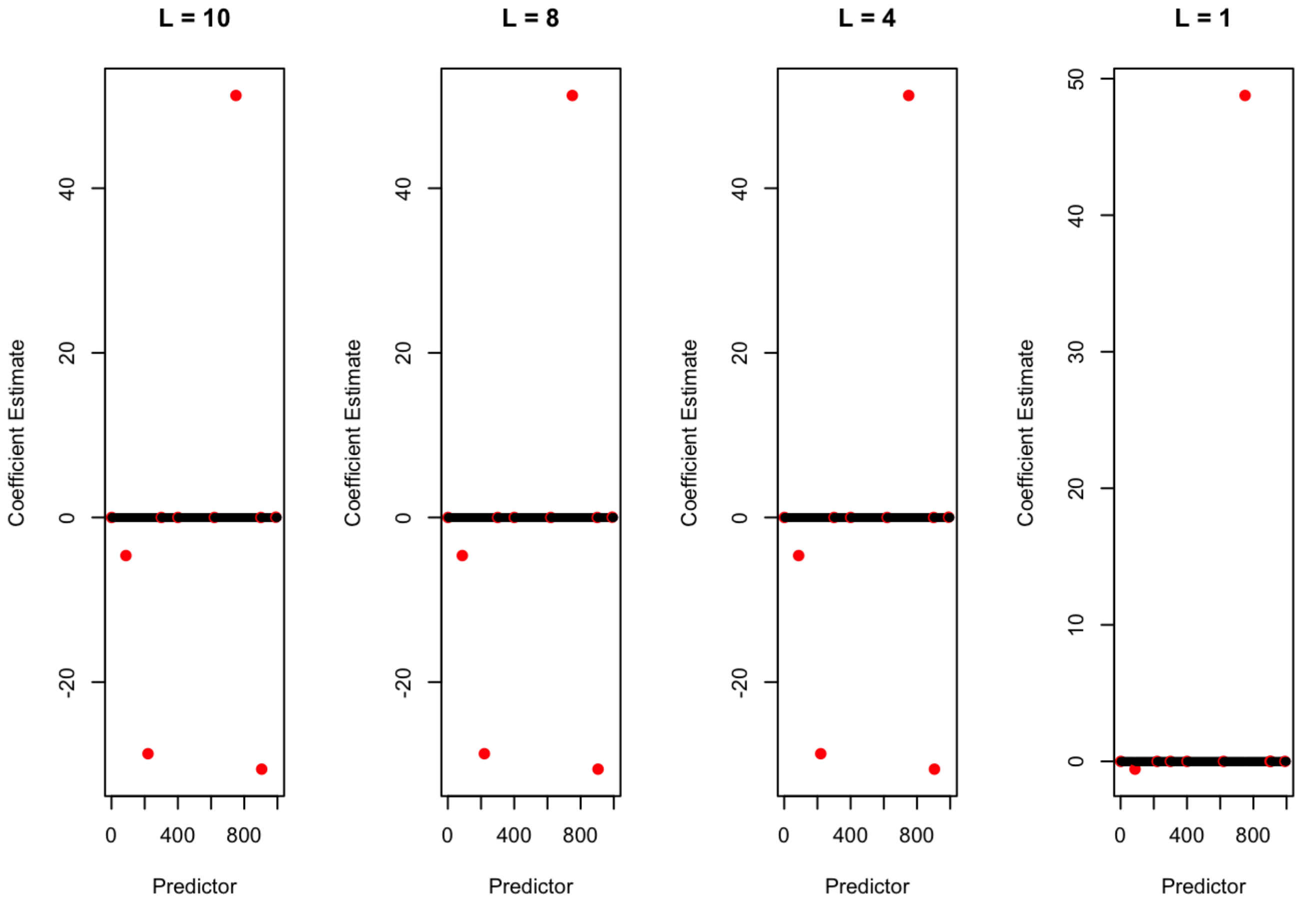
```
set.seed(1)
index <- c(1:1000)
n <- 100
p <- 1000
beta <- rnorm(1000, sd = 1)
signals <- c(1,87,220,300,400,620,750,900,905,990)
beta[c(1,400)] <- 20
beta[c(87, 905)] <- -25
beta[c(300, 620, 900)] <- 10
beta[220] <- -35
beta[750] <- 50
beta[990] <- 8

X <- matrix(rnorm(n*p, sd = 5), nrow=n, ncol=p)
y <- X %*% beta + rnorm(n, sd = 3)
res1 <- susie(X, y, L = 10)

res6 <- susie(X, y, L = 20)

res7 <- susie(X, y, L = 200)

res8 <- susie(X, y, L = 8)

res9 <- susie(X, y, L = 4)

res10 <- susie(X, y, L = 1)

par(mfrow = c(1,3))
plot(index, coef(res1)[-1], xlab = "Predictor", ylab = "Coefficient Estimate", main = "L
= 10", col = ifelse(index %in% signals, "red", "black"), pch = ifelse(index %in% signal
s, 19, 20))
plot(index, coef(res6)[-1], xlab = "Predictor", ylab = "Coefficient Estimate", main = "L
= 20", col = ifelse(index %in% signals, "red", "black"), pch = ifelse(index %in% signal
s, 19, 20))
plot(index, coef(res7)[-1], xlab = "Predictor", ylab = "Coefficient Estimate", main = "L
= 200", col = ifelse(index %in% signals, "red", "black"), pch = ifelse(index %in% signal
s, 19, 20))
```



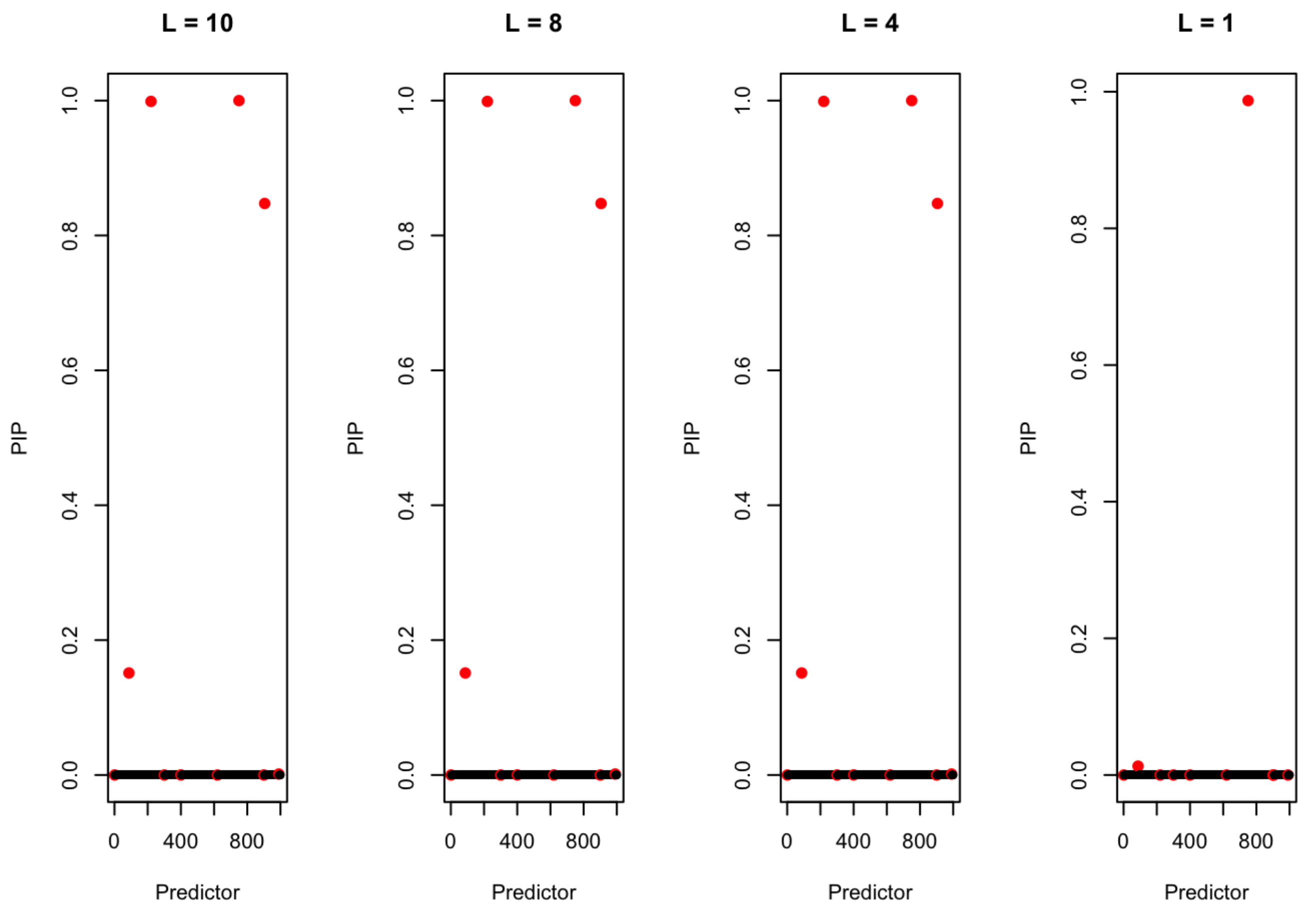

```
par(mfrow = c(1,3))
plot(index, res1$pip, xlab = "Predictor", ylab = "PIP", main = "L = 10", col = ifelse(in
dex %in% signals, "red", "black"), pch = ifelse(index %in% signals, 19, 20))
plot(index, res6$pip, xlab = "Predictor", ylab = "PIP", main = "L = 20", col = ifelse(in
dex %in% signals, "red", "black"), pch = ifelse(index %in% signals, 19, 20))
plot(index, res7$pip, xlab = "Predictor", ylab = "PIP", main = "L = 200", col = ifelse(i
ndex %in% signals, "red", "black"), pch = ifelse(index %in% signals, 19, 20))
```




```
par(mfrow = c(1,4))
plot(index, coef(res1)[-1], xlab = "Predictor", ylab = "Coefficient Estimate", main = "L
= 10", col = ifelse(index %in% signals, "red", "black"), pch = ifelse(index %in% signal
s, 19, 20))
plot(index, coef(res8)[-1], xlab = "Predictor", ylab = "Coefficient Estimate", main = "L
= 8", col = ifelse(index %in% signals, "red", "black"), pch = ifelse(index %in% signals,
19, 20))
plot(index, coef(res9)[-1], xlab = "Predictor", ylab = "Coefficient Estimate", main = "L
= 4", col = ifelse(index %in% signals, "red", "black"), pch = ifelse(index %in% signals,
19, 20))
plot(index, coef(res10)[-1], xlab = "Predictor", ylab = "Coefficient Estimate", main =
"L = 1", col = ifelse(index %in% signals, "red", "black"), pch = ifelse(index %in% signa
ls, 19, 20))
```




```
par(mfrow = c(1,4))
plot(index, res1$pip, xlab = "Predictor", ylab = "PIP", main = "L = 10", col = ifelse(in
dex %in% signals, "red", "black"), pch = ifelse(index %in% signals, 19, 20))
plot(index, res8$pip, xlab = "Predictor", ylab = "PIP", main = "L = 8", col = ifelse(ind
ex %in% signals, "red", "black"), pch = ifelse(index %in% signals, 19, 20))
plot(index, res9$pip, xlab = "Predictor", ylab = "PIP", main = "L = 4", col = ifelse(ind
ex %in% signals, "red", "black"), pch = ifelse(index %in% signals, 19, 20))
plot(index, res10$pip, xlab = "Predictor", ylab = "PIP", main = "L = 1", col = ifelse(in
dex %in% signals, "red", "black"), pch = ifelse(index %in% signals, 19, 20))
```

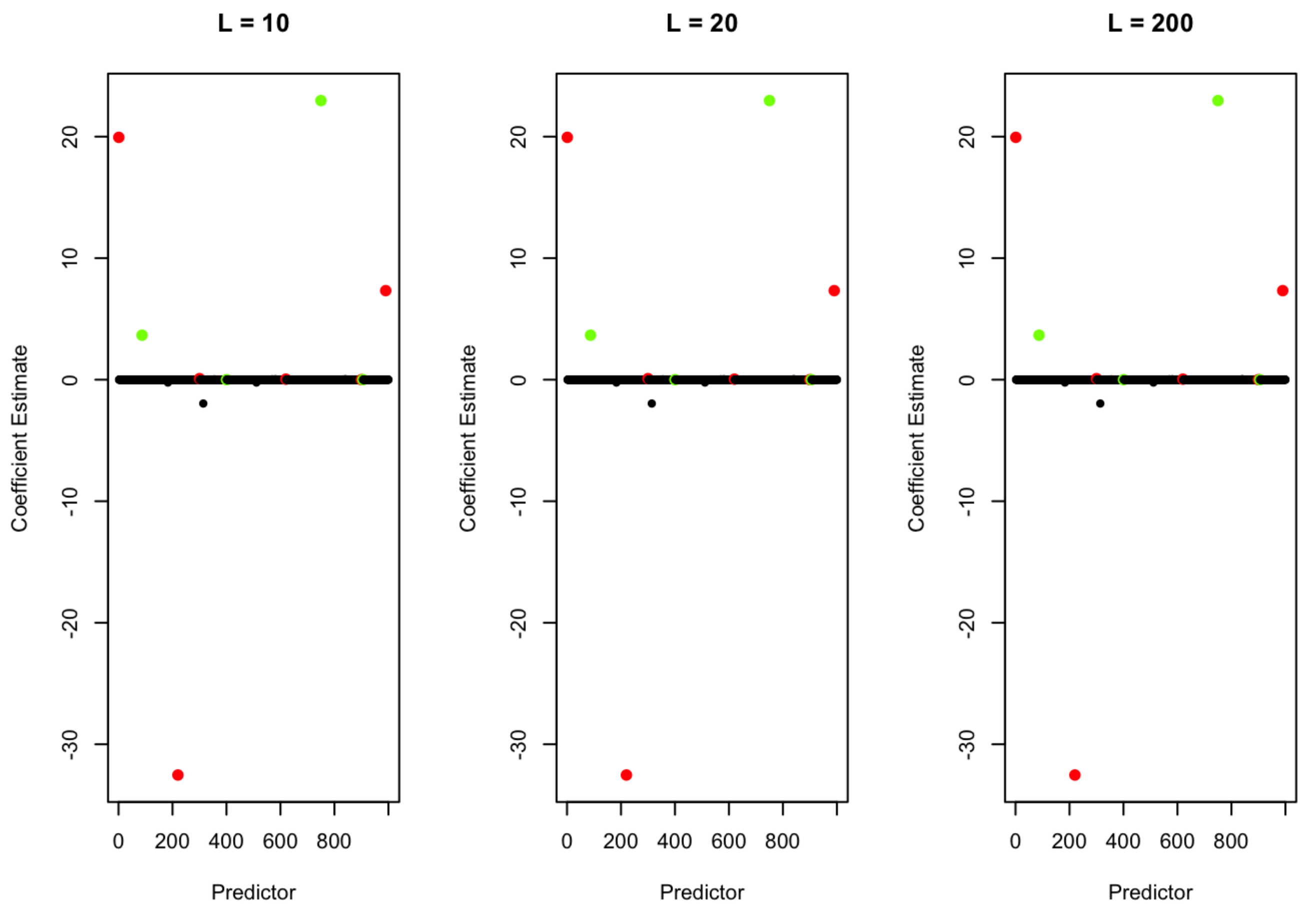


We can see that compared to the previous simulation, the outcomes for this example are relatively the same for both overstated and understated L, and not particularly sensitive to it in either direction. The only case where the plots were significantly different was when we set L = 1. However, even the L = 10 case did not identify all signals in the second data set, assigning the two largest signals PIP of near 1 and assigning the next two largest signals (which have the same size) with PIPs that sum to 1. Thus more realistic data seems less sensitive to L, but also is more difficult to identify signals on.

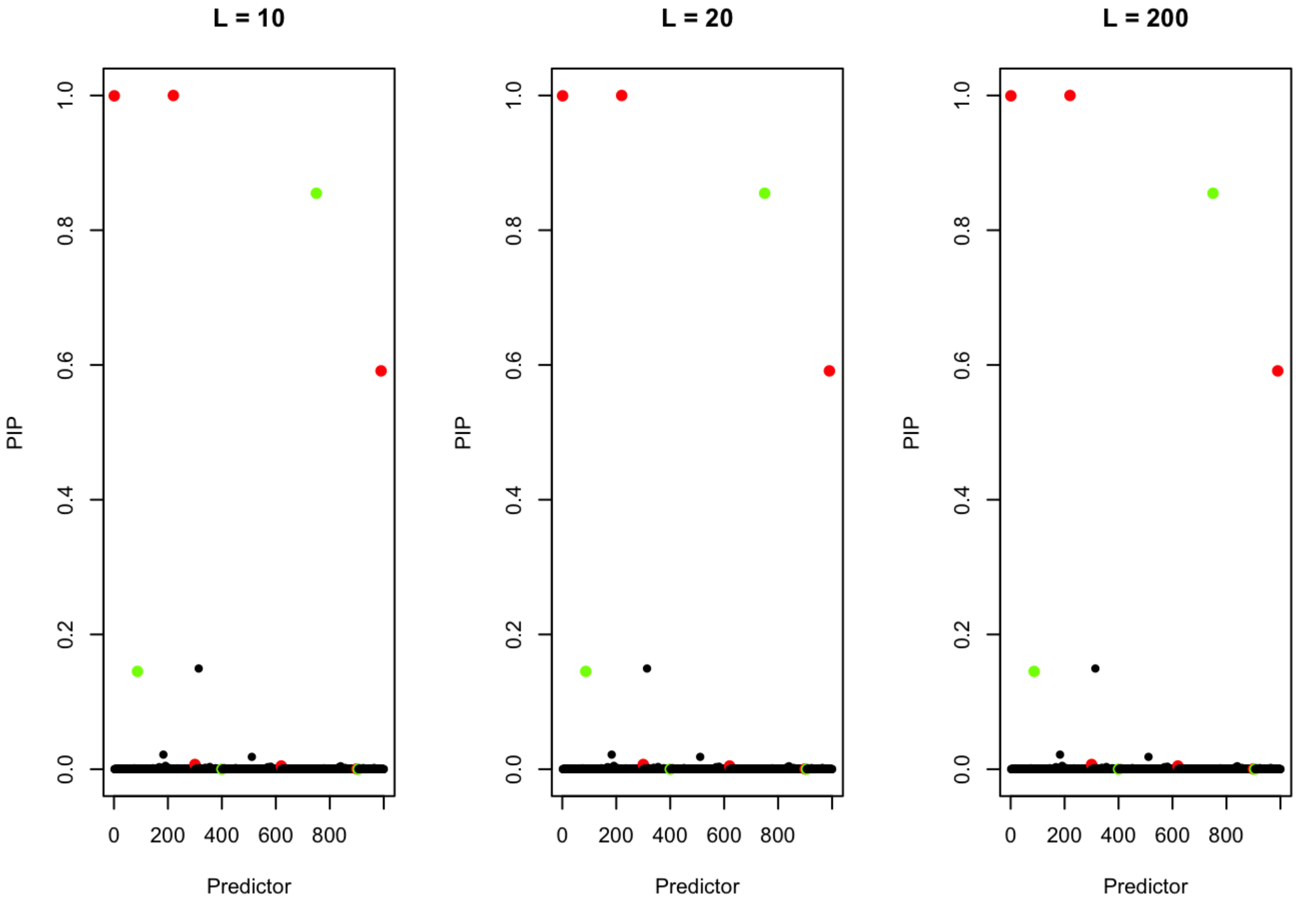
For the second data set, all tested values of L yielded the same credible sets - the two largest signals, variables {750} and {220} - except for L = 1, which produced just one credible set - {750}, the largest of the signals.

We now make a data set similar to the previous one, but with several highly correlated signal variables. Note that the highly correlated pairs are variables 87 and 750, and 400 and 905, with each pair approximately 98% correlated:

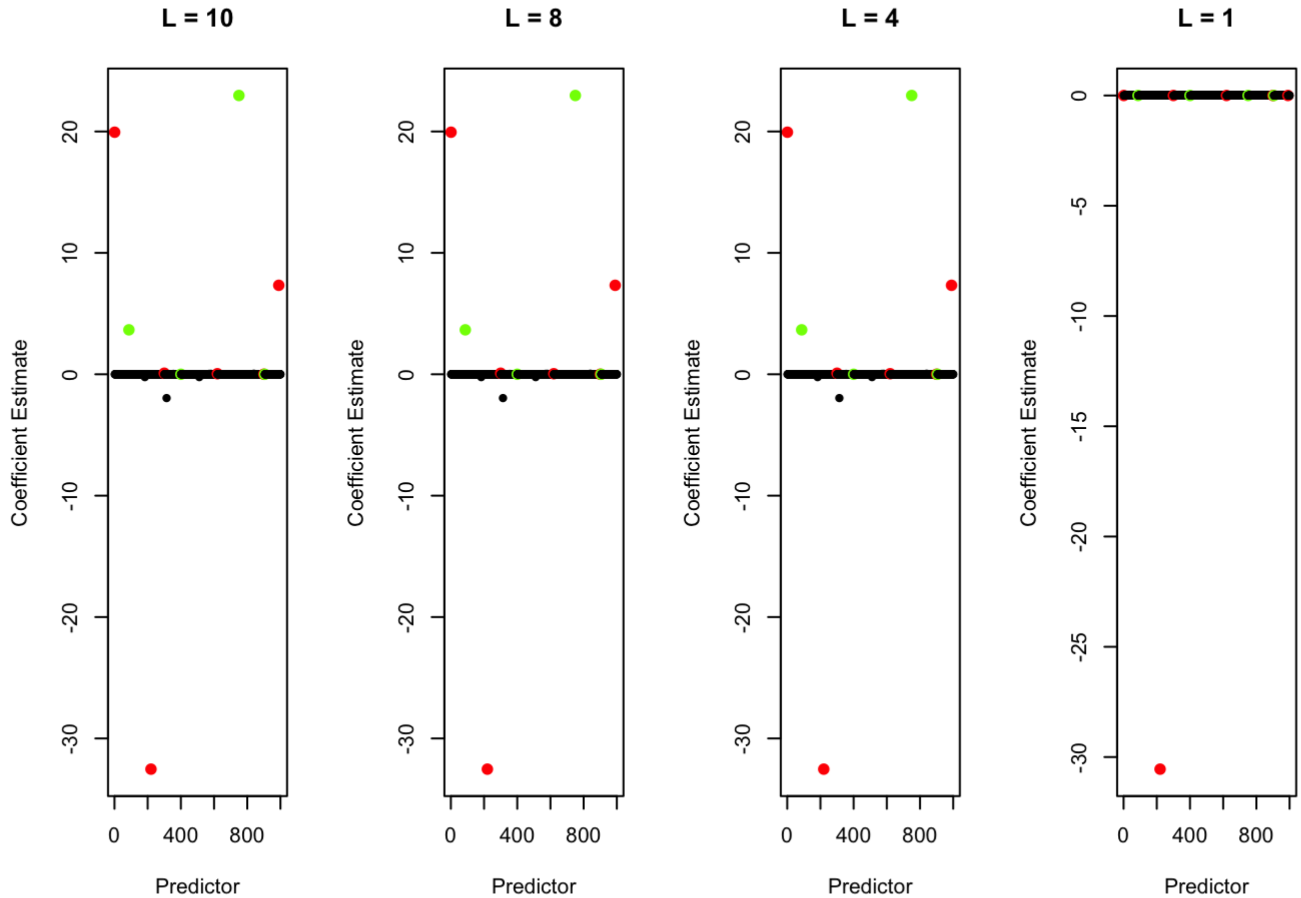
```r
set.seed(1)
index <- c(1:1000)
n <- 100
p <- 1000
beta <- rnorm(1000, sd = 1)
signals <- c(1,87,220,300,400,620,750,900,905,990)
beta[c(1,400)] <- 20
beta[c(87, 905)] <- -25
beta[c(300, 620, 900)] <- 10
beta[220] <- -35
beta[750] <- 50
beta[990] <- 8
X <- matrix(rnorm(n*p, sd = 5), nrow=n, ncol=p)
covm <- cbind(c(25,24.5), c(24.5,25))
X1 <- mvrnorm(n = 100, mu = c(0,0), Sigma = covm)
X2 <- mvrnorm(n = 100, mu = c(0,0), Sigma = covm)
#set variable pairs 87&750, 400&905 to be highly correlated
X[,87] <- X1[,1]
X[,750] <- X1[,2]

X[,400] <- X2[,1]
X[,905] <- X2[,2]
y <- X %*% beta + rnorm(n, sd = 3)

res1 <- susie(X, y, L = 10)

res11 <- susie(X, y, L = 20)

res12 <- susie(X, y, L = 200)

res13 <- susie(X, y, L = 8)

res14 <- susie(X, y, L = 4)

res15 <- susie(X, y, L = 1)

color <- rep("black", 1000)
color[signals] <- "red"
color[c(87, 400, 750, 905)] <- "chartreuse"

par(mfrow = c(1,3))
plot(index, coef(res1)[-1], xlab = "Predictor", ylab = "Coefficient Estimate", main = "L
= 10", col = color, pch = ifelse(index %in% signals, 19, 20))
plot(index, coef(res11)[-1], xlab = "Predictor", ylab = "Coefficient Estimate", main =
"L = 20", col = color, pch = ifelse(index %in% signals, 19, 20))
plot(index, coef(res12)[-1], xlab = "Predictor", ylab = "Coefficient Estimate", main =
"L = 200", col = color, pch = ifelse(index %in% signals, 19, 20))
```



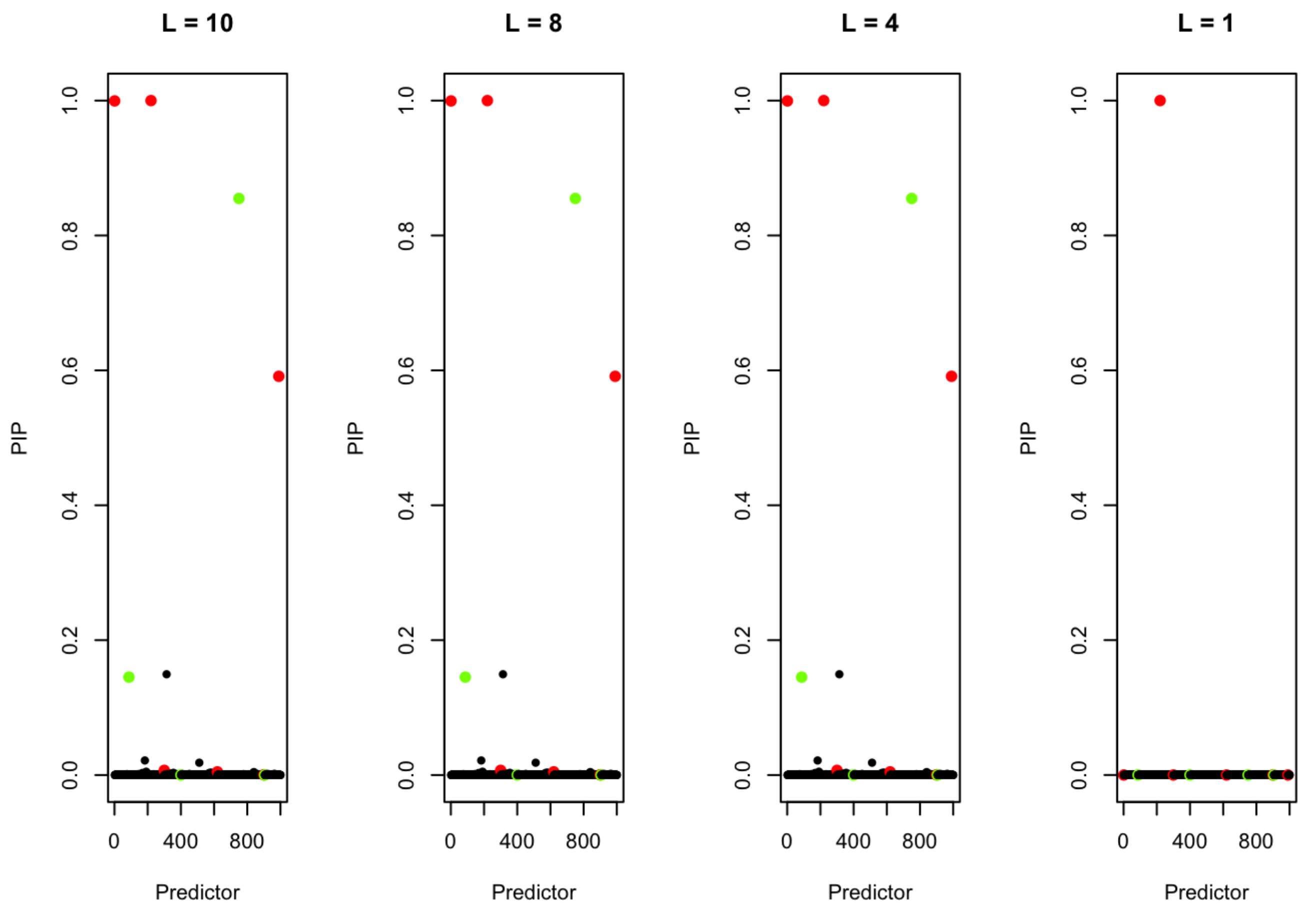

```
par(mfrow = c(1,3))
plot(index, res1$pip, xlab = "Predictor", ylab = "PIP", main = "L = 10", col = color, pc
h = ifelse(index %in% signals, 19, 20))
plot(index, res11$pip, xlab = "Predictor", ylab = "PIP", main = "L = 20", col = color, p
ch = ifelse(index %in% signals, 19, 20))
plot(index, res12$pip, xlab = "Predictor", ylab = "PIP", main = "L = 200", col = color,
 pch = ifelse(index %in% signals, 19, 20))
```




```
par(mfrow = c(1,4))
plot(index, coef(res1)[-1], xlab = "Predictor", ylab = "Coefficient Estimate", main = "L
= 10", col = color, pch = ifelse(index %in% signals, 19, 20))
plot(index, coef(res13)[-1], xlab = "Predictor", ylab = "Coefficient Estimate", main =
"L = 8", col = color, pch = ifelse(index %in% signals, 19, 20))
plot(index, coef(res14)[-1], xlab = "Predictor", ylab = "Coefficient Estimate", main =
"L = 4", col = color, pch = ifelse(index %in% signals, 19, 20))
plot(index, coef(res15)[-1], xlab = "Predictor", ylab = "Coefficient Estimate", main =
"L = 1", col = color, pch = ifelse(index %in% signals, 19, 20))
```




```
par(mfrow = c(1,4))
plot(index, res1$pip, xlab = "Predictor", ylab = "PIP", main = "L = 10", col = color, pc
h = ifelse(index %in% signals, 19, 20))
plot(index, res13$pip, xlab = "Predictor", ylab = "PIP", main = "L = 8", col = color, pc
h = ifelse(index %in% signals, 19, 20))
plot(index, res14$pip, xlab = "Predictor", ylab = "PIP", main = "L = 4", col = color, pc
h = ifelse(index %in% signals, 19, 20))
plot(index, res15$pip, xlab = "Predictor", ylab = "PIP", main = "L = 1", col = color, pc
h = ifelse(index %in% signals, 19, 20))
```



Several interesting things happen in this case. The largest signal of 750 goes from a PIP of 1 to closer to 0.8, and several noisy variables gain enough PIP to distinguish themselves on the plots. The sensitivity to L is largely the same as in the previous data set, with significant changes only at L = 1, though we should note that while the previous uncorrelated data set yielded the largest signal as the only PIP possessor when L = 1, in this case, when that signal is highly correlated with another, the second largest signal is instead assigned all the PIP, i.e. we miss the largest signal.

For this data set, most L values yielded not two but three credible sets - {220}, {87, 750}, and {1}, while L = 1 gives only {220}. We can thus see the value of credible sets as it gives us two large signals that are highly correlated in one set, and though we cannot distinguish them based on this alone, the algorithm still assigns much higher PIP to 750 over 87.