Lab 9: Naive Post-LASSO

In the Monte Carlo simulations below, we illustrate the bias of the naive post-Lasso estimator. The bias comes from the failure of Lasso to reliably detect small non-zero coefficients.

```
library(glmnet)
```

```
##      Matrix
```

```
## Loaded glmnet 4.1-2
```

```
n=100 #sample size
R=300 #number of Monte Carlo repetitions
```

The data generating process for simulations:

$$
\begin{aligned}
Y_i &= \beta_1 X_{i,1} + \beta_2 X_{i,2} + \beta_3 X_{i,3} + U_i, \\
\beta_1 &= 0, \\
\beta_2 &= 0.35, \\
\beta_3 &= 0, \\
X_{i,2} &= \rho X_{i,1} + Z_{i,2}, \\
X_{i,1}, U_i, Z_{i,2}, X_{i,3} &\sim \text{ iid } N(0,1) \text{ and independent from each other.}
\end{aligned}
$$

Three potential regressors:

- Regressor 1 is the main regressor.
- Regressor 2 has a small coefficient and its correlation with Regressor 1 depends on the magnitude of ρ.
- Regressor 3 is irrelevant.

```
beta1=0
beta2=0.35
```

"Large" ρ: $\rho = 1$

We assume that Regressor 1 is strongly correlated with controls.

```
rho=1
```

We write a function for generating data:

```
data_sim<-function(n,beta1,beta2,rho){
  X=matrix(rnorm(n*3),ncol=3)
  X[,2]<-rho*X[,1]+X[,2]
  Y=beta1*X[,1]+beta2*X[,2]+rnorm(n)
  data<-list(Y=Y,X=X)
}
```

We'll use LASSO with cross validation to select the controls, and then estimate the effect of Regressor 1 on Y. We set the penalty weight of Regressor 1 to 0 to always include it.

```
w=rep(1,3)
w[1]=0
```

```
data<-data_sim(n,beta1,beta2,rho)
CV.Lasso=cv.glmnet(data$X,data$Y,family="gaussian",alpha=1,penalty.factor=w)
Included=which(coef(CV.Lasso,s=CV.Lasso$lambda.1se)[-1]!=0)
Included
```

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

We generate data, select covariates using LASSO (with Regressor 1 being always in) and store the t-statistic for the coefficient of the first regressor.

```
rho=1
set.seed(42,sample.kind = "Rejection")
IN2=0 # counter for inclusion of X2
T_Beta1_post=rep(0,R) # Vector to store T-stats for the main regressor
for (r in 1:R){
  data<-data_sim(n,beta1,beta2,rho)

  CV.Lasso=cv.glmnet(data$X,data$Y,family="gaussian",alpha=1,penalty.factor=w)
  Included=which(coef(CV.Lasso,s=CV.Lasso$lambda.1se)[-1]!=0)

  Post_OLS=lm(data$Y~data$X[,Included])
  T_Beta1_post[r]=coef(summary(Post_OLS))[2,3] #Selects the t-statistic on X1

  IN2=IN2+(coef(CV.Lasso,s=CV.Lasso$lambda.1se)[3]!=0)
}
print("Prob. of X2 included")
```

```
## [1] "Prob. of X2 included"
```

```
IN2/R
```

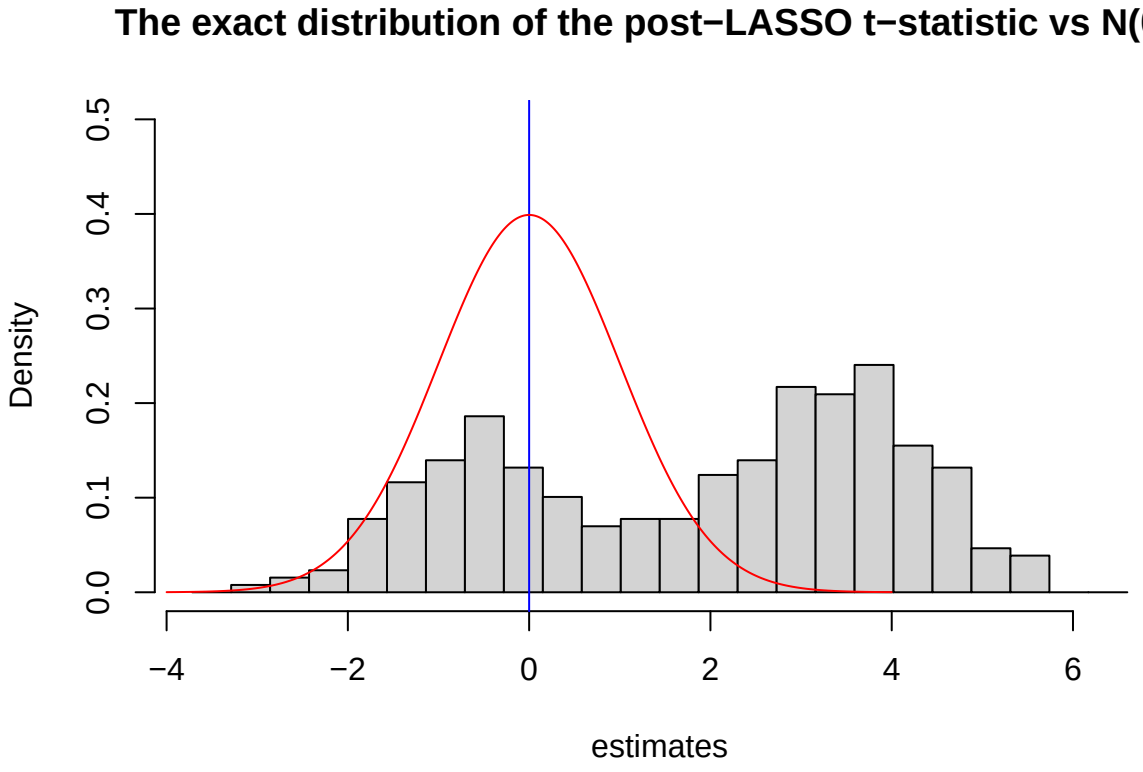
```
## [1] 0.3733333
```

We plot the histogram of the post-LASSO t-statistic for the first regressor. Its asymptotic distribution should be centered around zero since the true coefficient is zero. We compare it with the $N(0,1)$ distribution. Because Regressor 2 is omitted with high probability and correlated with Regressor 1, the distribution is distorted.

```
low=min(T_Beta1_post)
high=max(T_Beta1_post)
step=(high-low)/20
hist(T_Beta1_post,breaks=seq(low-2*step,high+2*step,step),xlab="estimates",main="The exact distribution

# add a vertical line at the true value
abline(v=beta1,col="blue")

# add the plot of the N(0,1) pdf
x=seq(-4,4,0.01)
f=exp(-x^2/2)/sqrt(2*pi)
lines(x,f,col="red")
```



Repeat with a "small" ρ: $\rho = 0.1$

We rerun the simulations with a smaller ρ. Regressor 2 is still dropped with a high probability.

```
rho=0.25
set.seed(42,sample.kind = "Rejection")
IN2=0 # counter for inclusion of X2
T_Beta1_post=rep(0,R) # Vector to store T-stats for the main regressor
for (r in 1:R){
  data<-data_sim(n,beta1,beta2,rho)

  CV.Lasso=cv.glmnet(data$X,data$Y,family="gaussian",alpha=1,penalty.factor=w)
  Included=which(coef(CV.Lasso,s=CV.Lasso$lambda.1se)[-1]!=0)

  Post_OLS=lm(data$Y~data$X[,Included])
  T_Beta1_post[r]=coef(summary(Post_OLS))[2,3] #Selects the t-statistic on X1

  IN2=IN2+(coef(CV.Lasso,s=CV.Lasso$lambda.1se)[3]!=0)
}
print("Prob. of X2 included")
```

```
## [1] "Prob. of X2 included"
```

```
IN2/R
```

```
## [1] 0.3833333
```

The exact distribution is less skewed. When the second regressor is only weakly correlated with Regressor 1, there is less distortion.

```
low=min(T_Beta1_post)
high=max(T_Beta1_post)
```

```r
step=(high-low)/20
hist(T_Beta1_post,breaks=seq(low-2*step,high+2*step,step),xlab="estimates",main="The exact distribution

# add a vertical line at the true value
abline(v=beta1,col="blue")

# add the plot of the N(0,1) pdf
x=seq(-4,4,0.01)
f=exp(-x^2/2)/sqrt(2*pi)
lines(x,f,col="red")
```