Exercise set 2

your candidate number

High-dimensional regression

Background reading. Delete this section before submitting Wikipedia: Singular value decomposition. Read the introduction, Example, and Pseudoinverse sections.

Wikipedia: Pseudoinverse. Read the introduction and sections on Projectors, Examples, Linearly independent rows, and Applications.

ISLR: See section 12.5.2 for some example use of the `svd` function.

Delete from here to the previous line about deletion.

```
set.seed(1) # change this to some other number
# Generate a matrix of predictors X
# with 3 rows, 10 columns, and i.i.d. normal entries
# X <- ...

# Create a sparse coefficient vector beta
# with only 1 or 2 nonzero entries
beta <- rep(0, 10) # change this

# Compute outcome y from the noise-free linear model
y <- 0 # change this
```

Generate data (3 points)

Compute pseudoinverse (3 points) Use output from the `svd` function to compute a right pseudoinverse of `X`. (Note: if you use a source aside from the Wikipedia articles above to figure out how to do this you should cite your source and include a link if it's a website)

```
# S <- svd(X) ...
```

Verify right-inverse property (3 points) **Explanation**: (1 point) *Replace this sentence with one where you comment on the output above to say whether you expect it or find it surprising and why.*

Note: delete this comment after reading. If you are unable to compute the pseudoinverse using the svd function, you can increase the sample size to p+1 and use OLS to estimate beta instead to receive partial credit.

```
beta_svd <- beta # change this
cbind(beta, beta_svd) |> kable()
```

Compare estimated beta to true beta (3 points)

beta	beta_svd
0	0
0	0
0	0
0	0
0	0
0	0
0	0
0	0
0	0
0	0

Explanation: (1 point) ...

```
y_svd <- y # change this
# mean(...)
```

Compute MSE for predicting y (3 points) **Explanation**: (1 point) ...

Generate a new sample of test data and compute the (in-distribution) test MSE (3 points) **Explanation**: (1 point) ...

Use penalized regression to estimate beta (4 points) Compute the ridge and lasso estimates using `lambda = 0.1`, and compare these with the estimate from using `svd`. You may want to read the documentation for `?glmnet` and `?coef.glmnet`

```
beta_ridge <- c(NA, beta) # change this
beta_lasso <- c(NA, beta) # change this
```

```
# Comparison
betas <- cbind(beta, beta_lasso[-1], beta_ridge[-1], beta_svd)
colnames(betas) <- c("beta", "lasso", "ridge", "svd")
betas |>
  kable()
```

beta	lasso	ridge	svd
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0

Explanation: (1 point) ...

Compute test MSE using penalized regression estimates (3 points) **Explanation**: (1 point) ...

What are the first two variables to have nonzero coefficients in the lasso solution path as lambda decreases? (3 points) (Note: be careful about whether there is an intercept estimate)

```
# lasso_path <- ...
```