Announcements

- Make sure to sign in on the google form (linked here)
- Pset 7 due November 4 at 5 pm
- Midterm 2 November 11 through November 18

Weighted least squares regression

This question is based on an October 29th conversation with Skyler Wu.

Consider a least squares model where, rather than weighting all residuals equally, we are going to assign different weights to different residuals. That is, we want to minimize

$$\sum_{i=1}^{n}[w_i(Y_i - \hat{Y}_i)]^2$$

Equivalently, letting $\mathbf{W}$ be a diagonal matrix of weights, letting $\vec{Y} = \mathbf{X}\vec{\beta} + \vec{\epsilon}$ with $\vec{\epsilon} \sim \text{MVN}_n(0, \sigma^2 I_n)$, and using the fact that $\hat{\vec{Y}} = \mathbf{X}\hat{\vec{\beta}}$, we want to minimize

$$||\mathbf{W}(\vec{Y} - \mathbf{X}\hat{\vec{\beta}})||^2$$

Expanding and taking the derivative gives the following:

$$\begin{aligned} 0 &= \frac{\partial}{\partial \hat{\vec{\beta}}}((\vec{Y} - \mathbf{X}\hat{\vec{\beta}})^T\mathbf{W}^T\mathbf{W}(\vec{Y} - \mathbf{X}\hat{\vec{\beta}})) \\ &= -2\mathbf{X}^T\mathbf{W}^T\mathbf{W}(\vec{Y} - \mathbf{X}\hat{\vec{\beta}}) \\ \implies \mathbf{X}^T\mathbf{W}^T\mathbf{W}\mathbf{X}\hat{\vec{\beta}} &= \mathbf{X}^T\mathbf{W}^T\mathbf{W}\vec{Y} \\ \implies \hat{\vec{\beta}} &= (\mathbf{X}^T\mathbf{W}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}^T\mathbf{W}\vec{Y} \end{aligned}$$

This is our new weighted least-squares regression $\hat{\vec{\beta}}$, which we will be studying in this problem.

Here are a few facts that will be useful here and on the homework:

- For matrices $\mathbf{A}, \mathbf{B}$, and $\mathbf{C}$, of allowable dimensions, $\mathbf{A}(\mathbf{B} + \mathbf{C}) = \mathbf{AB} + \mathbf{AC}$
- If $\mathbf{A}$ is of full column rank, $\mathbf{A}^T\mathbf{A}$ is symmetric and invertible
- If $\mathbf{A}$ is symmetric, and $\mathbf{B}$ is of allowable dimensions, $\mathbf{B}^T\mathbf{AB}$ is symmetric
- For an invertible and symmetric matrix $\mathbf{A}$, $\mathbf{A}^{-1} = (\mathbf{A}^{-1})^T$
- For $\vec{Y} = \vec{c} + \mathbf{B}\vec{X}$ with $\vec{c}$ and $\mathbf{B}$ constant and $\vec{X}$ random, $E(\vec{Y}) = \vec{c} + \mathbf{B}E(\vec{X})$
- $\text{Cov}(\vec{X})$ is an $n \times n$ matrix whose i, j entry is $\text{Cov}(X_i, X_j)$
- For $\vec{Y} = \vec{c} + \mathbf{B}\vec{X}$ with $\vec{c}$ and $\mathbf{B}$ constant and $\vec{X}$ random, $\text{Cov}(\vec{Y}) = \mathbf{B}\text{Cov}(\vec{X})\mathbf{B}^T$

1. Verify that using the usual weights for least squares regression, this formula reduces to the usual estimator for $\hat{\vec{\beta}}$.

In our usual least squares regression, $\mathbf{W} = \mathbf{I}$, so

$$\hat{\vec{\beta}} = (\mathbf{X}^T\mathbf{I}^T\mathbf{I}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{I}^T\mathbf{I}\vec{Y} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\vec{Y}$$

as expected.

2. Find the bias of $\hat{\vec{\beta}}$ for $\vec{\beta}$.

$$\begin{aligned} E(\hat{\vec{\beta}}) - \vec{\beta} &= E\left[(\mathbf{X}^T\mathbf{W}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}^T\mathbf{W}\vec{Y}\right] - \vec{\beta} \\ &= (\mathbf{X}^T\mathbf{W}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}^T\mathbf{W}E\left[\vec{Y}\right] - \vec{\beta} \\ &= (\mathbf{X}^T\mathbf{W}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}^T\mathbf{W}\mathbf{X}\vec{\beta} - \vec{\beta} \\ &= \vec{0} \end{aligned}$$

3. Find the variance-covariance matrix of $\hat{\vec{\beta}}$ in matrix form. When will this equal the variance-covariance matrix of $\hat{\vec{\beta}}$ in OLS regression?

$$\begin{aligned} \text{Cov}(\hat{\vec{\beta}}) &= \text{Cov}\left[(\mathbf{X}^T\mathbf{W}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}^T\mathbf{W}\vec{Y}\right] \\ &= \left[(\mathbf{X}^T\mathbf{W}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}^T\mathbf{W}\right]\text{Cov}\left(\vec{Y}\right)\left[(\mathbf{X}^T\mathbf{W}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}^T\mathbf{W}\right]^T \\ &= \left[(\mathbf{X}^T\mathbf{W}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}^T\mathbf{W}\right]\text{Cov}\left(\vec{\epsilon}\right)\left[(\mathbf{X}^T\mathbf{W}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}^T\mathbf{W}\right]^T \\ &= \sigma^2\left[(\mathbf{X}^T\mathbf{W}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}^T\mathbf{W}\right]\mathbf{I}\left[(\mathbf{X}^T\mathbf{W}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}^T\mathbf{W}\right]^T \\ &= \sigma^2(\mathbf{X}^T\mathbf{W}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}^T\mathbf{W}\mathbf{W}^T\mathbf{W}\mathbf{X}(\mathbf{X}^T\mathbf{W}^T\mathbf{W}\mathbf{X})^{-1} \end{aligned}$$

This is equal to the normal variance-covariance matrix when $\mathbf{W}^T\mathbf{W} = I$, which gives $\sigma^2(\mathbf{X}^T\mathbf{X})^{-1}$. In principle, any orthogonal $\mathbf{W}$ would work, but we have defined $\mathbf{W}$ to be a diagonal matrix, so all the entries must be 1 or -1. Note that -1 is not an issue because we square the weights, so really we're just back to our ordinary least squares regression.

Ridge, LASSO, optimizing λ, and β trajectories

This question will deal with a data set of country-level statistics from this source with an explanation of the data encoding found here.

A few useful columns:

- `spi_ospi`: Overall social progress index on 0-100 scale
- `mad_gdppc`: GDP per capita
- `wdi_internet`: Percent of population using the internet
- `wdi_birth`: Birth rate per 1000 people
- `wdi_chexppgdp`: Current health expenditures as percent of GDP
- `wdi_elerenew`: Percent of total electricity output that's renewable
- `wdi_lifexp`: Life expectancy at birth
- `wdi_wip`: Proportion of seats held by women in national parliaments
- `wdi_popurb`: Percentage of total population that is urban
- `wdi_imig`: Proportion of people born outside the country in which they live

1. Find a well-tuned Ridge regression model via `cv.glmnet` for predicting `spi_ospi`: consider all main predictors above and all 2-way interactions of these predictors.

```
library(glmnet)
set.seed(139)

# Variables to be used
columns <- c("mad_gdppc", "wdi_internet", "wdi_birth", "wdi_chexppgdp",
             "wdi_elerenew", "wdi_lifexp", "wdi_wip", "wdi_popurb", "wdi_imig")

# Model matrix for glmnet
X = model.matrix(spi_ospi ~ (mad_gdppc + wdi_internet + wdi_birth + wdi_chexppgdp +
```

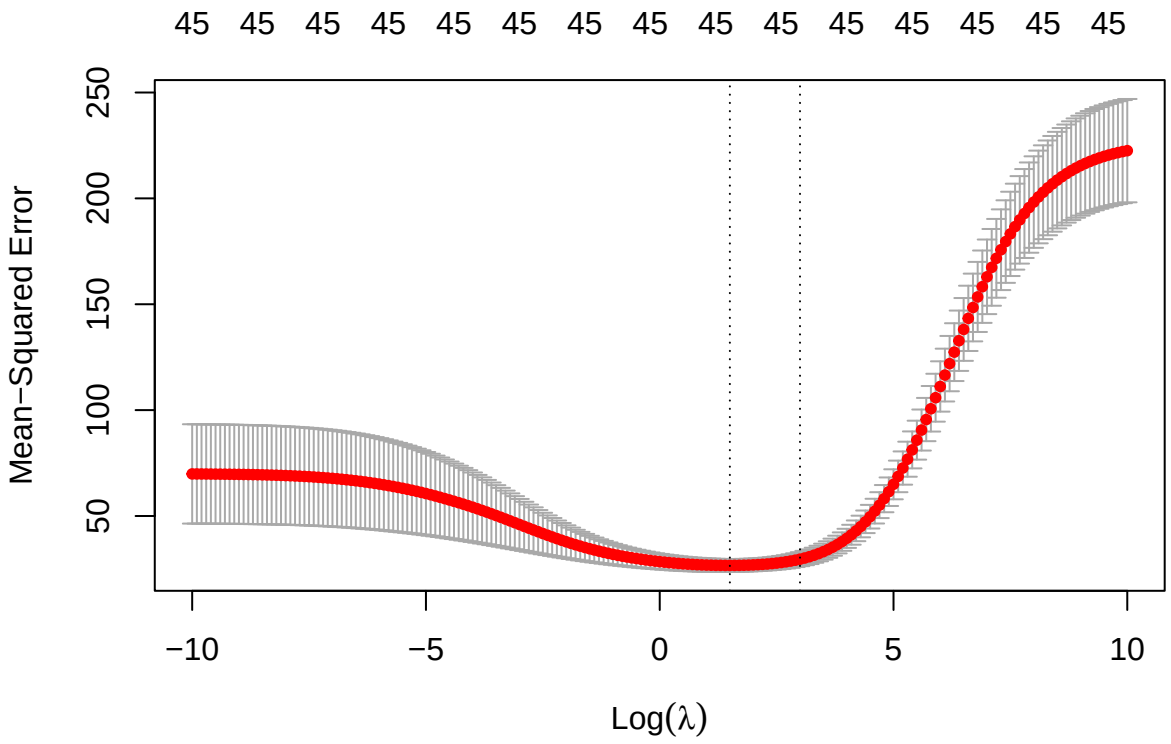
```
                                wdi_elerenew + wdi_lifexp + wdi_wip + wdi_popurb +
                                wdi_imig)^2, countries)[,-1] # Drop the intercept

# Scaling X is usually a good idea when penalizing betas
X <- scale(X)

# Run cross validation
ridges=cv.glmnet(X, unlist(lm1$model["spi_ospi"]), alpha=0, lambda = exp(seq(-10,10,0.1)))

print(ridges)
```

```
##
## Call:  cv.glmnet(x = X, y = unlist(lm1$model["spi_ospi"]), lambda = exp(seq(-10,      10, 0.1)), alpl
##
## Measure: Mean-Squared Error
##
##     Lambda Index Measure    SE Nonzero
## min  4.482    86   26.65 3.027      45
## 1se 20.086    71   29.61 3.303      45
```

The best ridge regression model is one with a λ of 4.48.

2. Plot the average MSE on the validation sets against the λ's you considered in the previous part. Report the best λ and justify this choice using this plot.

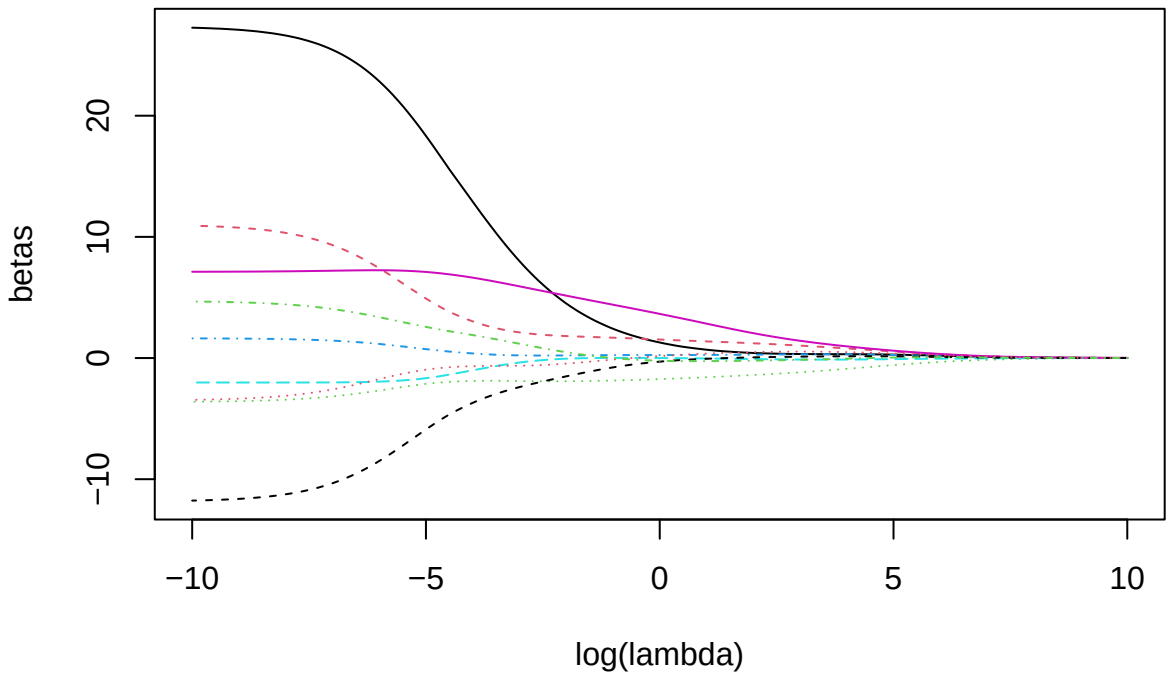
```
plot(ridges)
```



The best λ is 4.48, which corresponds to $\log(4.55) = 1.50$ in the plot. Values of λ slightly above this might also help in preventing overfitting.

3. Provide the $\hat{\beta}$ trajectory plot of the main effects from this model (plot each β_j as a function of λ as a line, and do this for all 11 main effects). Interpret what you see in 2-3 sentences.

```
# Fit for many lambdas
ridges_for_plot=glmnet(X, unlist(lm1$model["spi_ospi"]), alpha=0, lambda=exp(seq(-10,10,0.1)))

# Plot trajectories
matplot(log(ridges_for_plot$lambda),
        t(ridges_for_plot$beta[-grep(":", rownames(ridges_for_plot$beta)),]), # Remove interactions
        type="l",
        xlab = "log(lambda)",
        ylab = "betas")
```



We can see that the estimates vary a lot for low values of λ, but coefficients shrink to zero asymptotically. However, as λ grows, the $\hat{\beta}$s never reach zero exactly as expected in ridge regression.

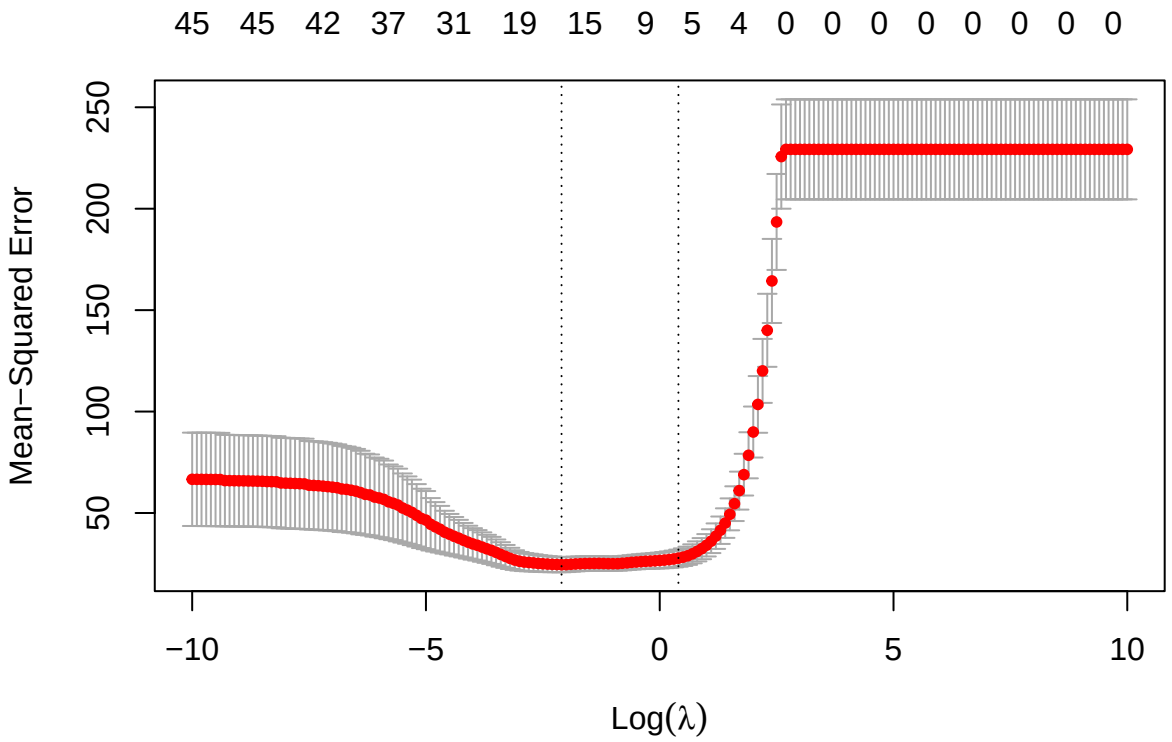
4. Fit a well-tuned LASSO regression model: examine main effects of predictors and all their 2-way interactions.

```
# Run cross validation
lassos=cv.glmnet(X, unlist(lm1$model["spi_ospi"]), alpha=1, lambda = exp(seq(-10,10,0.1)))

print(lassos)
```

```
##
## Call:  cv.glmnet(x = X, y = unlist(lm1$model["spi_ospi"]), lambda = exp(seq(-10,      10, 0.1)), alpl
##
## Measure: Mean-Squared Error
##
##     Lambda Index Measure    SE Nonzero
## min 0.1225   122   24.50 3.707      13
## 1se 1.4918    97   27.73 4.225       5
```

5. Plot the average MSE on the validation sets against the λ's you considered in the previous part. Report the best λ and justify this choice using this plot.

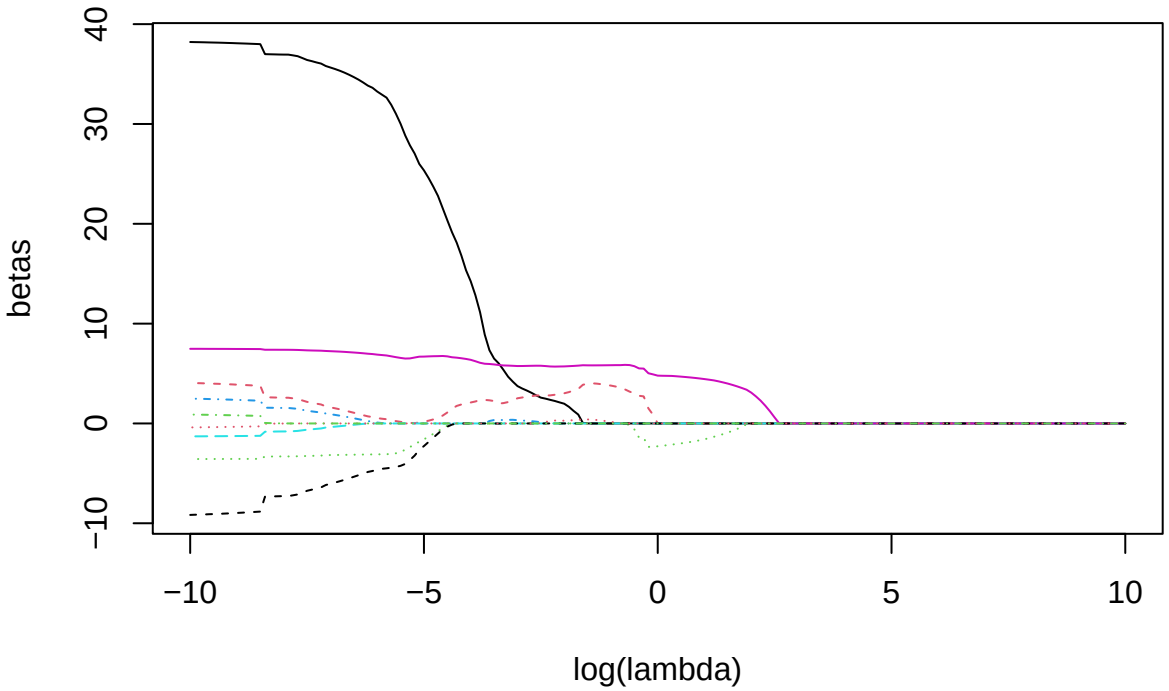
```
plot(lassos)
```



The best λ is 0.100, which corresponds to $\log(0.100) = -2.302$ in the plot. Values of λ slightly above this might also help in preventing overfitting.

6. Provide the $\hat{\beta}$ trajectory plot of the main effects from this LASSO model (plot each β_j as a function of λ as a line, and do this for all 11 main effects). Compare this to the ridge trajectories.

```
lasso_for_plot = glmnet(X, unlist(lm1$model["spi_ospi"]), alpha=1, lambda=exp(seq(-10,10,0.1)))
matplot(log(lasso_for_plot$lambda),
        t(lasso_for_plot$beta[-grep(":", rownames(lasso_for_plot$beta)),]), # Remove interactions
        type="l",
        xlab = "log(lambda)",
        ylab = "betas")
```



The estimates vary a lot for low values of λ, and they even increase in magnitude occasionally, indicating collinearity. Note that occasionally the coefficients actually are 0 but then become non-zero. However, as λ grows, generally more and more snap to 0. Compared to ridge regression, these coefficients decrease more sporadically but actually become 0 rather than just approaching 0.

7. Choose a best regularized/penalized regression model and briefly justify your choice.

```
data.frame(ridge=min(ridges$cvm), lasso=min(lassos$cvm))
```

```
##      ridge    lasso
## 1 26.64561 24.50046
```

These are the minimum means of cross validated error (the cross validated MSE using the optimal λ). LASSO has a slightly lower MSE on cross-validation, so LASSO with a λ of 0.100 is the best regularized model.

Penalization functions

Recall that for both Ridge and LASSO, we are trying to minimize something of the form:

$$\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 + p(\hat{\vec{\beta}})$$

State whether the following functions could or couldn't be used as penalization functions for $\hat{\vec{\beta}}$. If so, provide a context in which this might be a useful penalization function; if not, explain why it would give undesired behavior.

1. $p(\hat{\vec{\beta}}) = \sum_{i=1}^{k} \hat{\beta}_i$

No: the $\hat{\beta}_i$ would just get more and more negative.

2. $p(\hat{\vec{\beta}}) = \sum_{i=1}^{k} \hat{\beta}_i^{\,4}$

Yes: you could use this if you wanted to very strongly penalize $\hat{\beta}$s with large magnitudes.

3. $p(\hat{\vec{\beta}}) = \sum_{i=1}^{k} \log(\hat{\beta}_i)$

No: negative $\hat{\beta}_i$ would break this.

4. $p(\hat{\vec{\beta}}) = \sum_{i=1}^{k} \log(|\hat{\beta}_i|)$

No: the $\hat{\beta}_i$ would always go to 0 since that would give $-\infty$ loss.

5. $p(\hat{\vec{\beta}}) = \sum_{i=1}^{k} 1/|\hat{\beta}_i|$

No: larger $\hat{\beta}$s give a smaller loss.

6. $p(\hat{\vec{\beta}}) = -\sum_{i=1}^{k} 1/|\hat{\beta}_i|$

Still no: the $\hat{\beta}_i$ would always go to 0 since that would give $-\infty$ loss.

7. What general requirements do we need for a penalization function?

The loss should increase (or remain 0) with coefficients of increasing magnitude, and it should have a finite (preferably global) minimum.

8. Write a valid penalization function that we haven't studied before.

One example would be

$$p(\hat{\vec{\beta}}) = \sum_{i=1}^{k} f(\hat{\beta}_i) \text{ with } f(\hat{\beta}_i) = \begin{cases} 0 & |\hat{\beta}_i| \leq \delta \\ |\hat{\beta}_i| & \text{Otherwise} \end{cases}$$

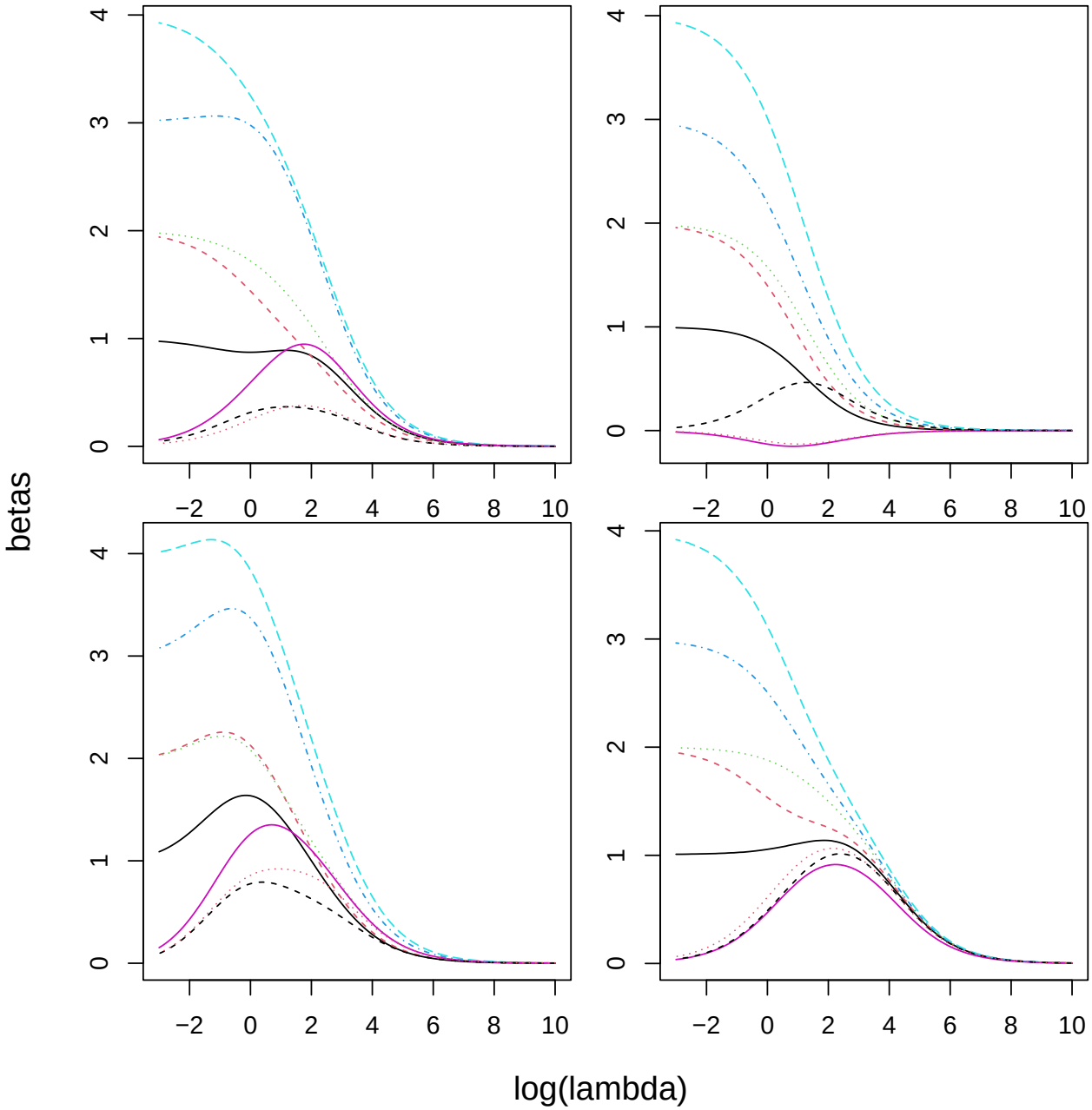
for some δ, which applies no penalty once a $\hat{\beta}_i$ is within δ of 0.

Miscellaneous

1. For what λs would LASSO and Ridge give the same model?

$\lambda = 0, \infty$ since there would be either no penalization or the model would just be $\bar{Y}$.

2. Below are four $\hat{\beta}$ trajectory plots. Each comes from a data set with 50 data points. One trajectory comes from data with no built-in correlation between the predictors; one comes from data with moderate and equal correlation among all the predictors; one comes from data with moderate random (but fixed) correlation among the predictors; and one is fake (and impossible). Determine which is which.



The first plot is the random correlation as seen by its peaks rising and falling more sporadically. The second is the no correlation plot because the slopes mostly fall independently of each other. The third is impossible because all the slopes rise at the beginning (but at least one must fall for the other to rise). The fourth is the strong correlation as seen by the fact that all the slopes are about the same after $\lambda = e^4$.