Week 5

Cogs 109: Data Analysis and Modeling

Fall 2017

Prof. Eran Mukamel

Model selection and regularization

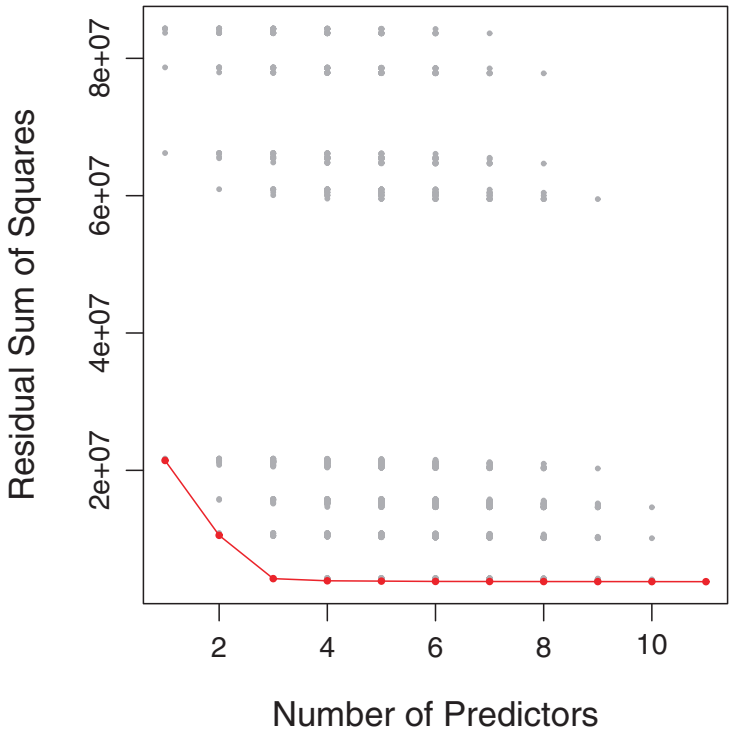
- So far we have assumed that we can just try out several models and choose the best one (“brute force search”)
- In real data sets, there may be an *huge* number of possible models, making the brute force method impractical
- Need methods for more efficiently searching for good models

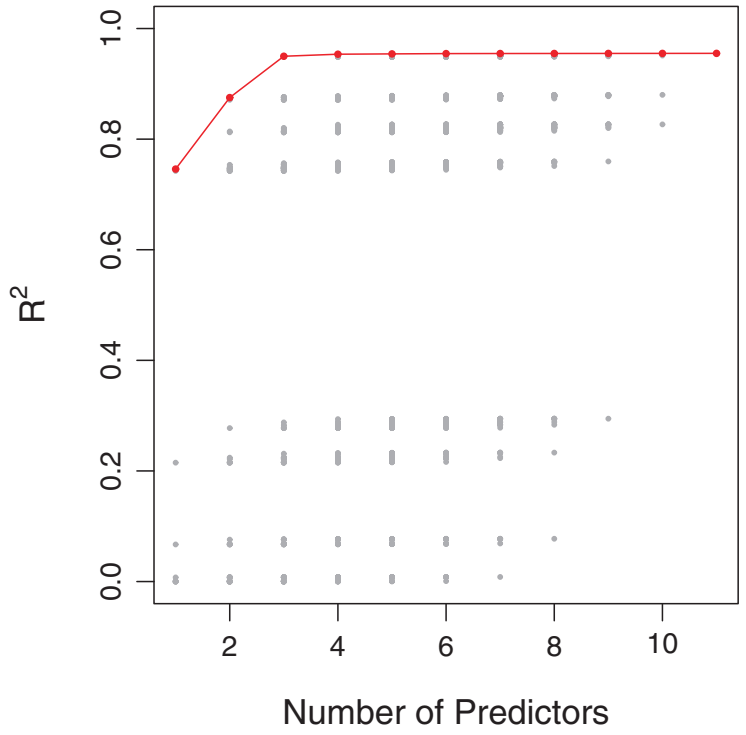
Method 1: Best subset

Algorithm 6.1 *Best subset selection*

1. Let $\mathcal{M}_0$ denote the *null model*, which contains no predictors. This model simply predicts the sample mean for each observation.
2. For $k = 1, 2, \ldots p$:
   (a) Fit all $\binom{p}{k}$ models that contain exactly k predictors.
   (b) Pick the best among these $\binom{p}{k}$ models, and call it $\mathcal{M}_k$. Here *best* is defined as having the smallest RSS, or equivalently largest R^2.
3. Select a single best model from among $\mathcal{M}_0, \ldots, \mathcal{M}_p$ using cross-validated prediction error, C_p (AIC), BIC, or adjusted R^2.

- We will always use cross-validation in Step 3; you may ignore the other approaches (Cp, AIC, BIC, etc.)


Residual Sum of Squares
2e+07
4e+07
6e+07
8e+07
2
4
6
8
10
Number of Predictors


R^2
0.0
0.2
0.4
0.6
0.8
1.0
2
4
6
8
10
Number of Predictors

When to use cross-validation (MSEtest) and when to use training data (MSEtrain)

- In step 2 of the selection, we are comparing many models with the same number of parameters:

$k = 2$:

- Model 1: $Blood\ pressure = \beta_0 + \beta_1 Sex + \beta_2 Age \rightarrow MSE_{train}(1)$
- Model 2: $Blood\ pressure = \beta_0 + \beta_1 Sex + \beta_2 Exercise \rightarrow MSE_{train}(2)$
- Model 3: $Blood\ pressure = \beta_0 + \beta_1 Age + \beta_2 Diet \rightarrow MSE_{train}(3)$

- These models are all equally flexible

- Although they may overfit the data (MSEtrain < MSEtest), they will all overfit by approximately the same amount. Thus we can use MSEtrain to compare them and select the best one (lowest MSEtrain)

- In step 3 we are comparing models with different numbers of parameters. Therefore, cross-validation is essential so we can choose a model with low MSEtest.

- Recall that MSEtrain is always lower for more flexible models, so MSEtrain cannot be used to compare models with different levels of flexibility.

- Model 1, $k = 1$: $Blood\ pressure = \beta_0 + \beta_1 Sex \rightarrow MSE_{test}(1)$
- Model 2, $k = 2$: $Blood\ pressure = \beta_0 + \beta_1 Sex + \beta_2 Age \rightarrow MSE_{test}(2)$
- Model 3, $k = 3$: $Blood\ pressure = \beta_0 + \beta_1 Sex + \beta_2 Age + \beta_3 Exercise \rightarrow MSE_{test}(3)$

Algorithm 6.1 *Best subset selection*

1. Let $\mathcal{M}_0$ denote the *null model*, which contains no predictors. This model simply predicts the sample mean for each observation.

2. For $k = 1, 2, \ldots p$:

   (a) Fit all $\binom{p}{k}$ models that contain exactly k predictors.

   (b) Pick the best among these $\binom{p}{k}$ models, and call it $\mathcal{M}_k$. Here *best* is defined as having the smallest RSS, or equivalently largest R^2.

3. Select a single best model from among $\mathcal{M}_0, \ldots, \mathcal{M}_p$ using cross-validated prediction error, C_p (AIC), BIC, or adjusted R^2.

No cross-validation; compare models using MSEtrain

Use cross-validation; compare models using MSEtest

The problem: Too many models

- Complete model: $Y = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p + \epsilon$

y = Blood pressure
x_1 = Age
x_2 = Weight
x_3 = Sex
x_4 = Vegetarian?
x_5 = Amount of exercise
...

1-variable models

$y = \beta_0 + \beta_1 x_1$
$y = \beta_0 + \beta_1 x_2$
$y = \beta_0 + \beta_1 x_3$
$y = \beta_0 + \beta_1 x_4$
$y = \beta_0 + \beta_1 x_5$

2-variable models

$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2$
$y = \beta_0 + \beta_1 x_1 + \beta_2 x_3$
$y = \beta_0 + \beta_1 x_1 + \beta_2 x_4$ ...

3-variable models

$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3$
$y = \beta_0 + \beta_1 x_1 + \beta_2 x_3 + \beta_3 x_4$ ...

...

Number of possible subsets grows exponentially (2^P)

$$\sum_{k=0}^{P} \binom{P}{k} = 2^P$$

P=5, k=0, 1 models
P=5, k=1, 5 models
P=5, k=2, 10 models
P=5, k=3, 10 models
P=5, k=4, 5 models
P=5, k=5, 1 models
Total: 2^5 = 32 models

P=10, k=0, 1 models
P=10, k=1, 10 models
P=10, k=2, 45 models
P=10, k=3, 120 models
P=10, k=4, 210 models
P=10, k=5, 252 models
P=10, k=6, 210 models
P=10, k=7, 120 models
P=10, k=8, 45 models
P=10, k=9, 10 models
P=10, k=10, 1 models
Total: 2^10 = 1024 models

P=20, k=0, 1 models
P=20, k=1, 20 models
P=20, k=2, 190 models
P=20, k=3, 1140 models
P=20, k=4, 4845 models
P=20, k=5, 15504 models
P=20, k=6, 38760 models
P=20, k=7, 77520 models
P=20, k=8, 125970 models
P=20, k=9, 167960 models
P=20, k=10, 184756 models
…
Total: 2^20 = 1048576 models

Method 2: Stepwise selection

Algorithm 6.2 *Forward stepwise selection*

1. Let $\mathcal{M}_0$ denote the *null* model, which contains no predictors.
2. For $k = 0, \ldots, p-1$:
   (a) Consider all $p-k$ models that augment the predictors in $\mathcal{M}_k$ with one additional predictor.
   (b) Choose the *best* among these $p-k$ models, and call it $\mathcal{M}_{k+1}$. Here *best* is defined as having smallest RSS or highest R^2.
3. Select a single best model from among $\mathcal{M}_0, \ldots, \mathcal{M}_p$ using cross-validated prediction error, C_p (AIC), BIC, or adjusted R^2.

$$k = 1: \quad Blood\ pressure = \beta_0 + \beta_1 Sex$$
$$k = 2: \quad Blood\ pressure = \beta_0 + \beta_1 Sex + \beta_2 Age$$
$$k = 3: \quad Blood\ pressure = \beta_0 + \beta_1 Sex + \beta_2 Age + \beta_3 Exercise$$

Stepwise selection is much more efficient than best subset selection

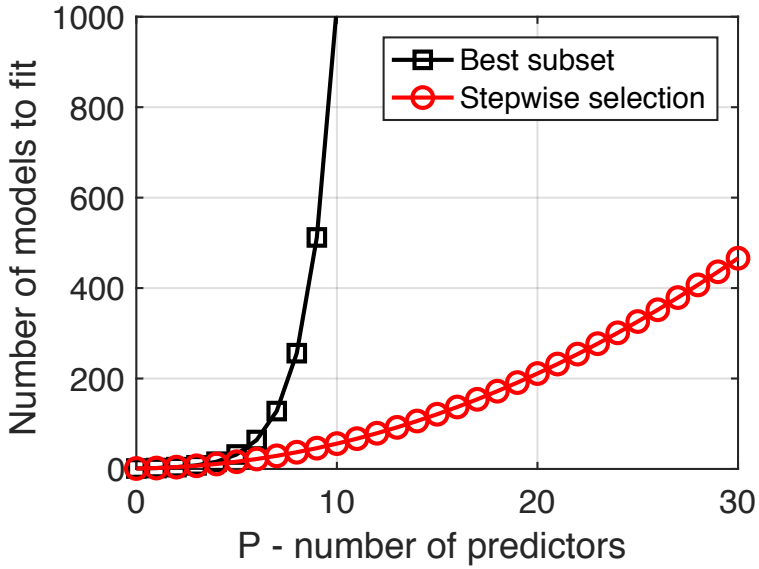
- Instead of exhaustively trying all 2^P possible subsets of parameters, at each stage we just try P subsets.

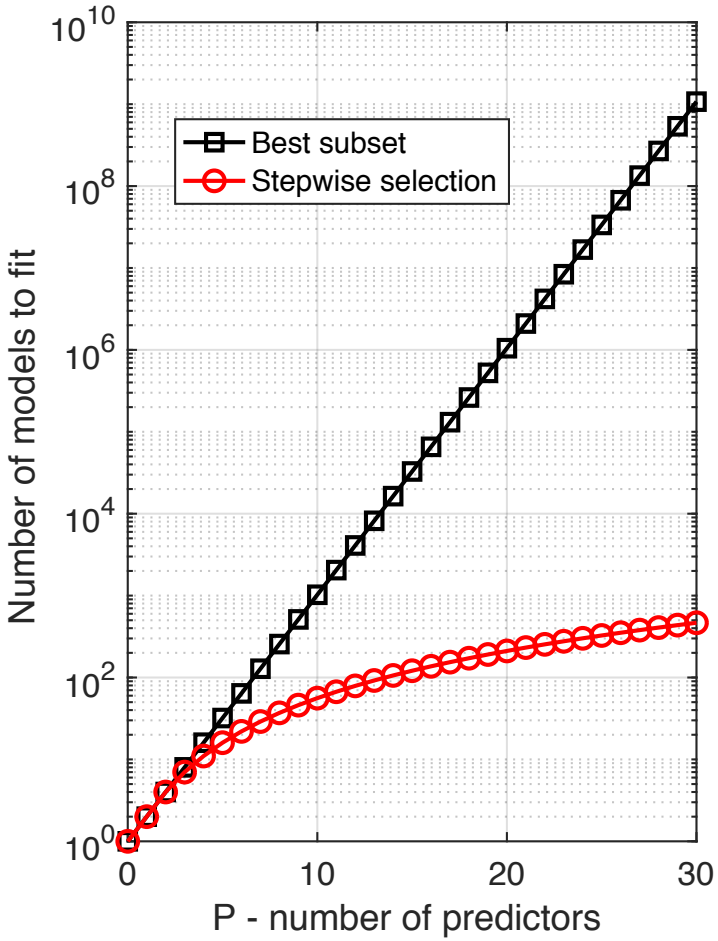
- The total number of models we end up fitting is:

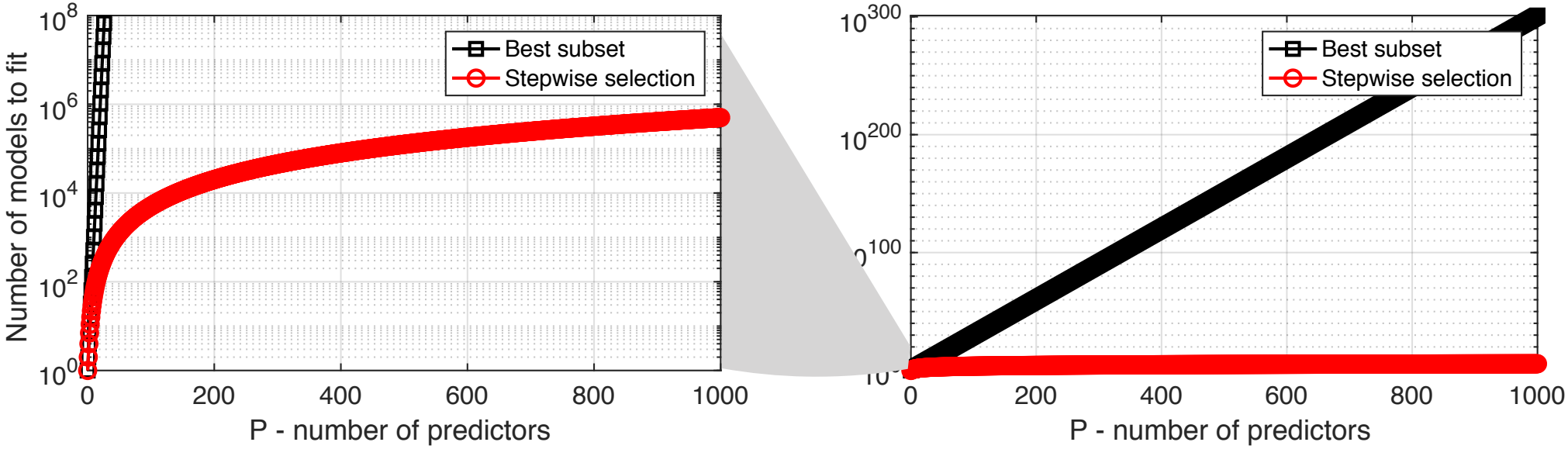
$$1 + \sum_{k=0}^{p-1}(p-k) = 1+p(p+1)/2$$

P=20, k=0, 20 models
P=20, k=1, 19 models
P=20, k=2, 18 models
P=20, k=3, 17 models
P=20, k=4, 16 models
P=20, k=5, 15 models
P=20, k=6, 14 models
P=20, k=7, 13 models
P=20, k=8, 12 models
P=20, k=9, 11 models
....
Total: 1+P(P+1)/2 = 211 models

Same data on a logarithmic scale for the y-axis






Number of models to fit
10^8
10^6
10^4
10^2
10^0
0
200
400
600
800
1000
P - number of predictors
Best subset
Stepwise selection
10^{300}
10^{200}
10^{100}
0
200
400
600
800
1000
P - number of predictors
Best subset
Stepwise selection

Stepwise selection may not find the absolute best model

# Variables	Best subset	Forward stepwise
One	`rating`	`rating`
Two	`rating`, `income`	`rating`, `income`
Three	`rating`, `income`, `student`	`rating`, `income`, `student`
Four	`cards`, `income`	`rating`, `income`,
	`student`, `limit`	`student`, `limit`

TABLE 6.1. *The first four selected models for best subset selection and forward stepwise selection on the* `Credit` *data set. The first three models are identical but the fourth models differ.*

Backward stepwise selection

Algorithm 6.3 *Backward stepwise selection*

1. Let $\mathcal{M}_p$ denote the *full* model, which contains all p predictors.
2. For $k = p, p-1, \ldots, 1$:
   (a) Consider all k models that contain all but one of the predictors in $\mathcal{M}_k$, for a total of $k-1$ predictors.
   (b) Choose the *best* among these k models, and call it $\mathcal{M}_{k-1}$. Here *best* is defined as having smallest RSS or highest R^2.
3. Select a single best model from among $\mathcal{M}_0, \ldots, \mathcal{M}_p$ using cross-validated prediction error, C_p (AIC), BIC, or adjusted R^2.

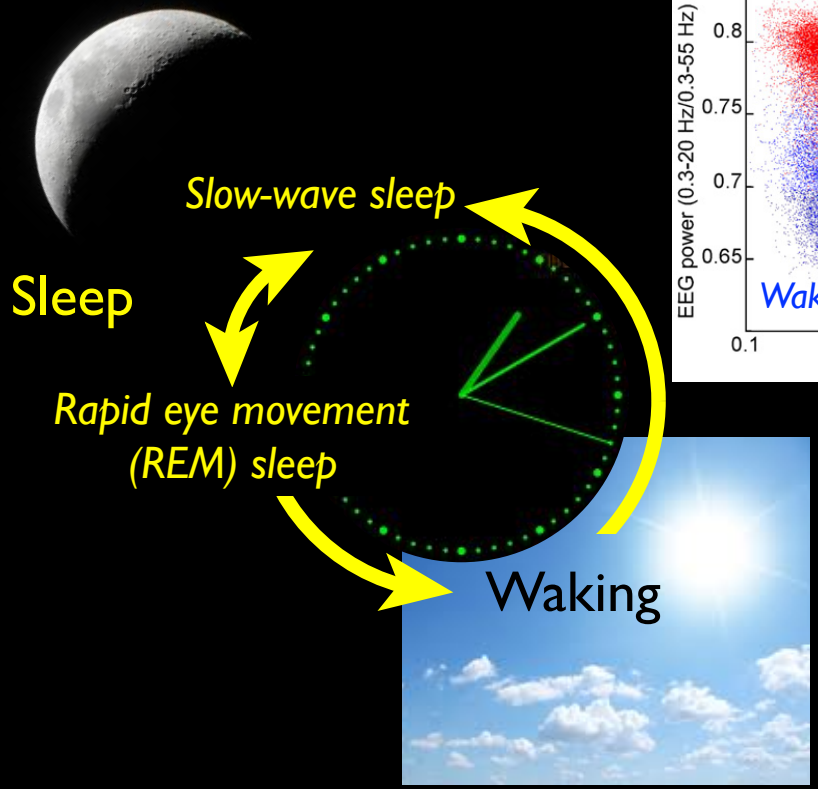
Regularization

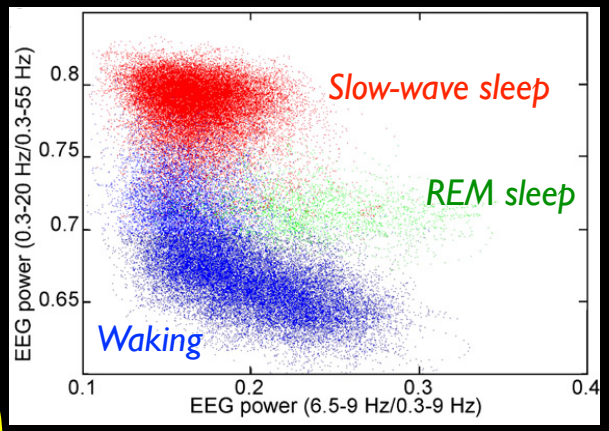
- When models are too flexible we run into a set of interrelated problems:
    - High variance on the estimated parameters
    - Overfitting
- One sign that this is happening is that regression coefficients take large, nonsensical values
- Best Subset and Stepwise variable selection deal with this by restricting the number of parameters
- **Regularization** is another approach that tries to constrain the wild swings in model parameters

Global brain state transitions are essential sleep and anesthesia

Brain state transition: A rapid, reversible switch between distinct, stable patterns of behavior and physiology

Sleep





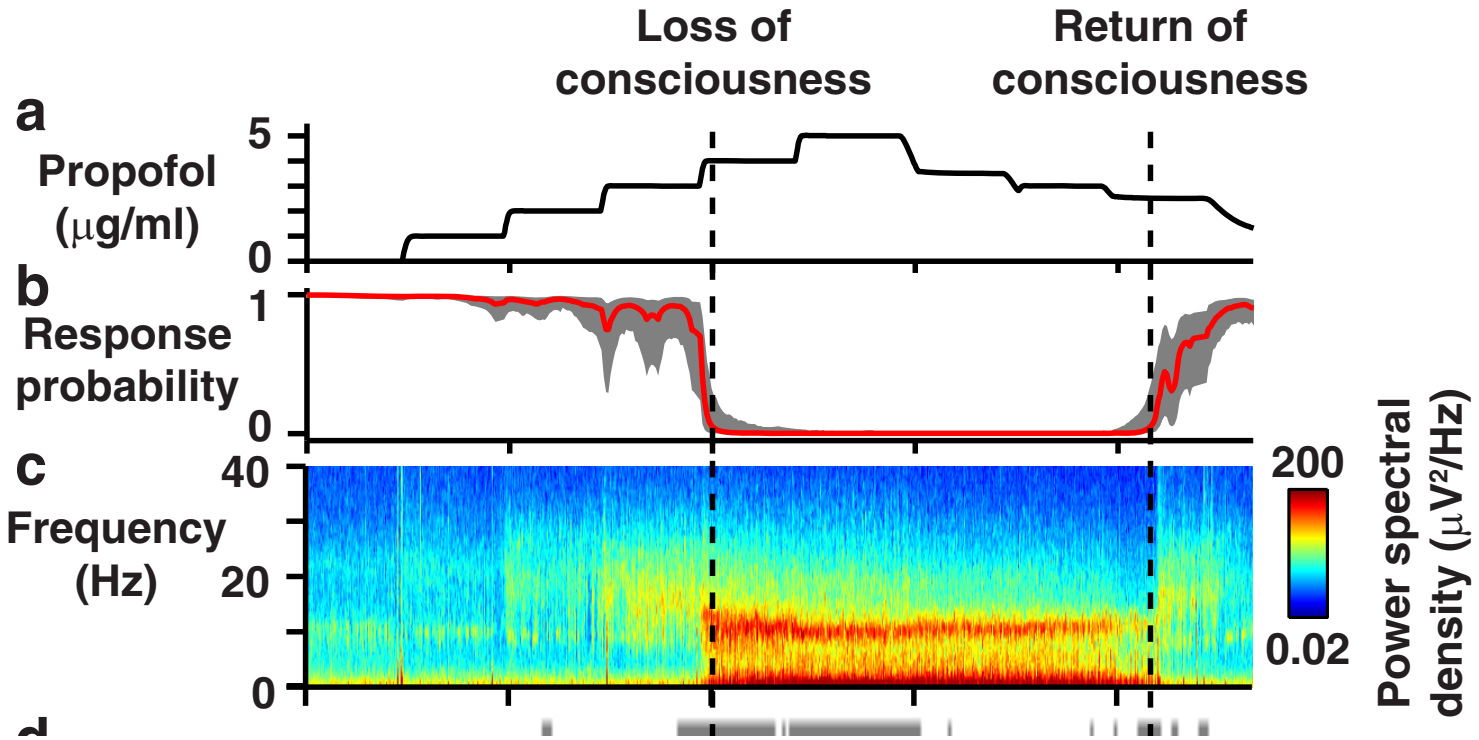
Saper *et al. Neuron* (2010)

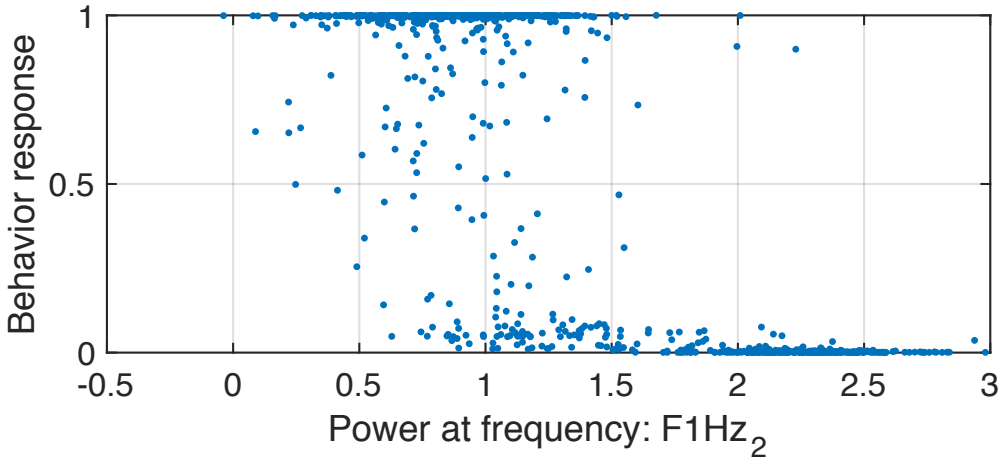
General anesthesia

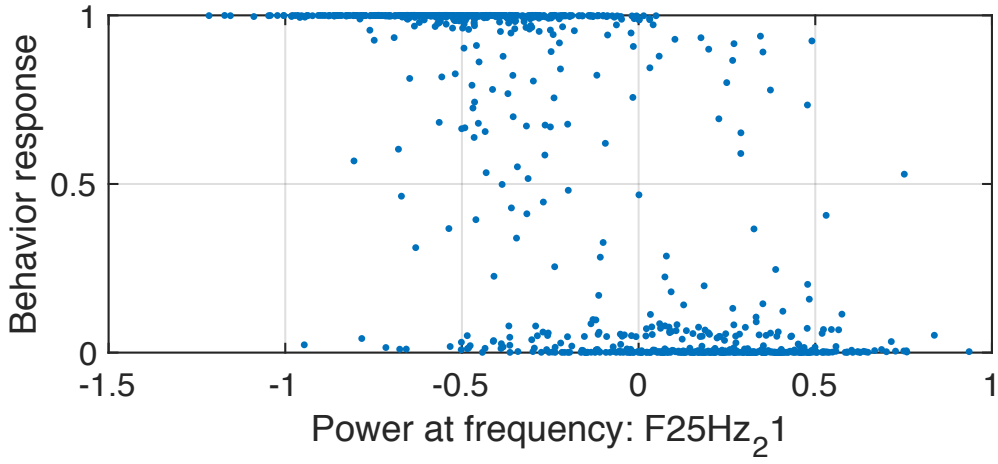
Monument to the discovery of general anesthesia in 1847 (Boston)

Example: General anesthesia

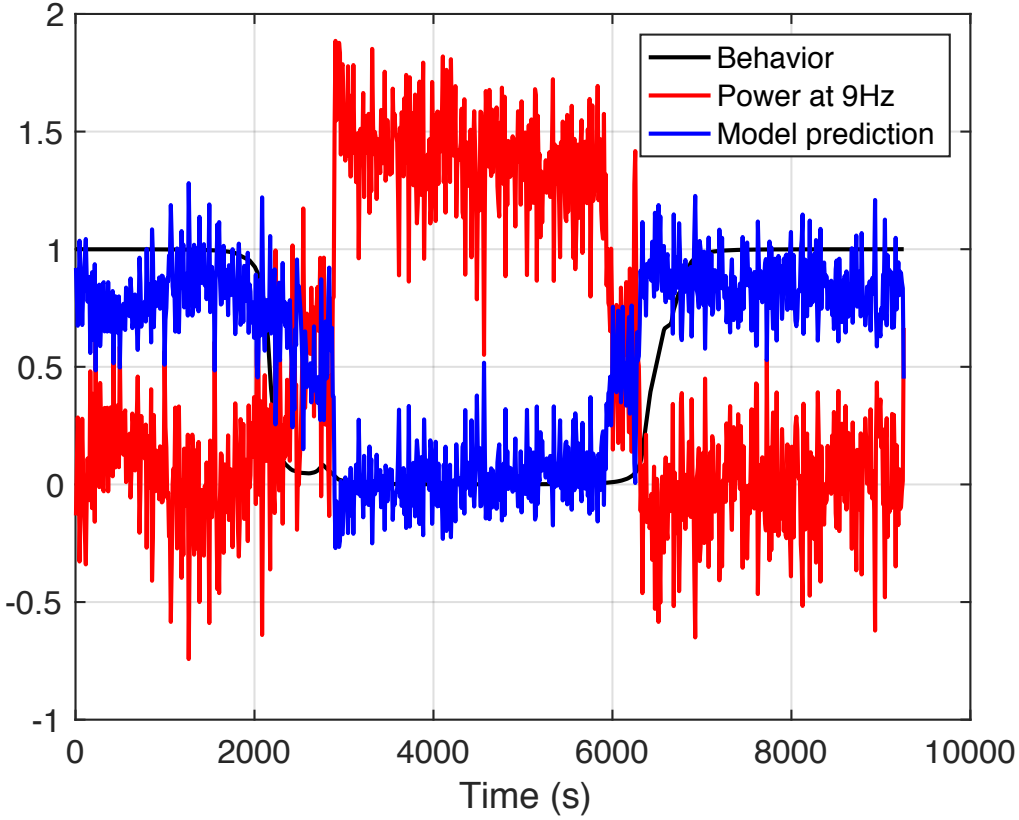
Question: Can we predict subject's loss of consciousness from the power of specific EEG frequency bands?



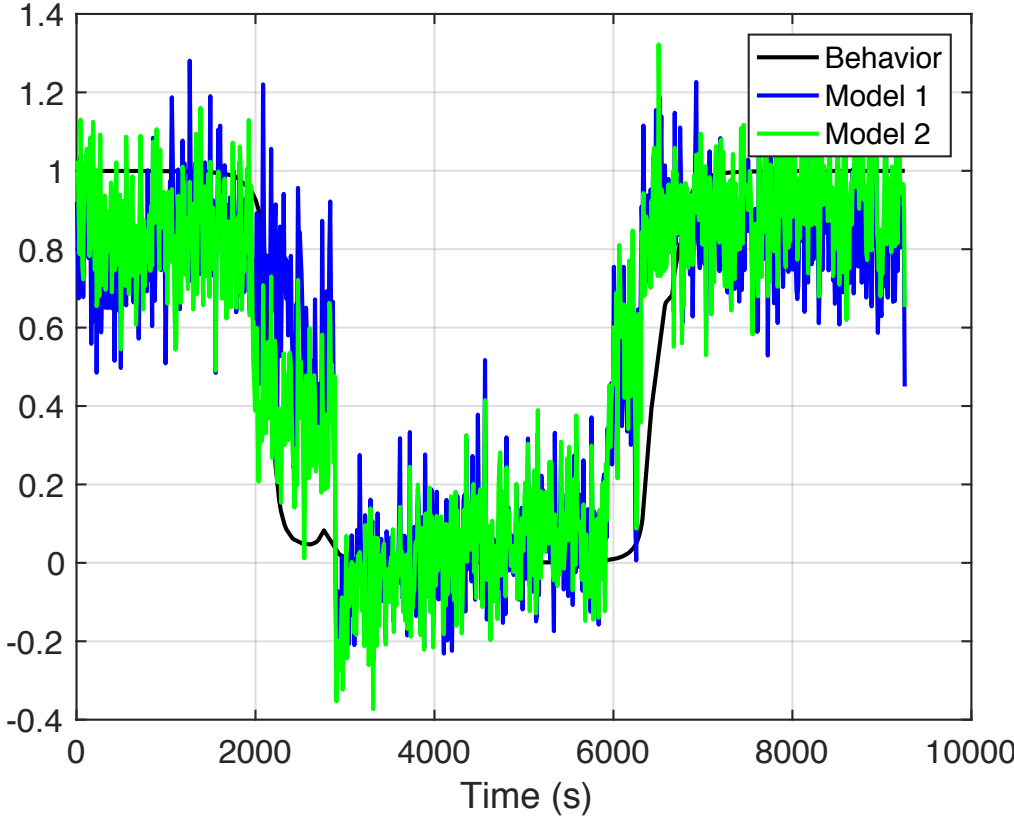




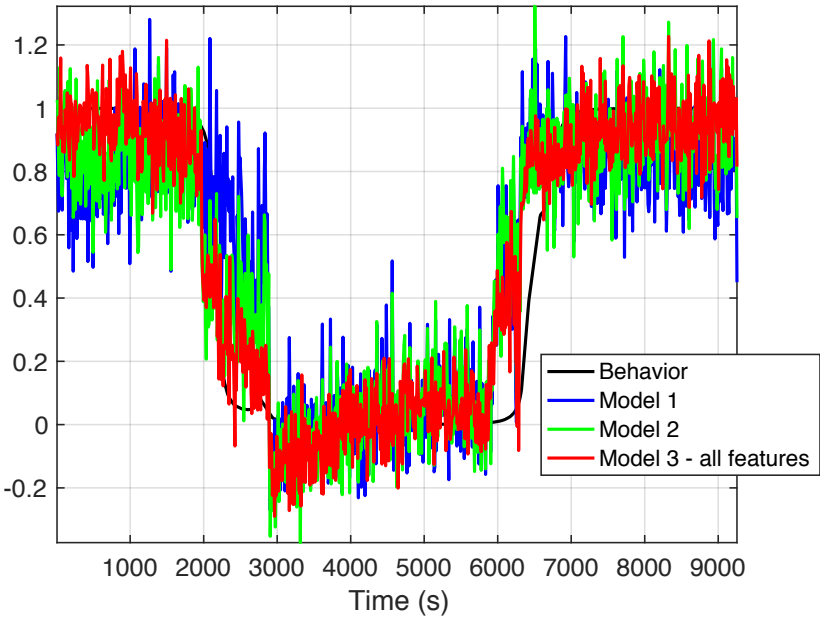
Model 1: Use only a single frequency (9Hz)



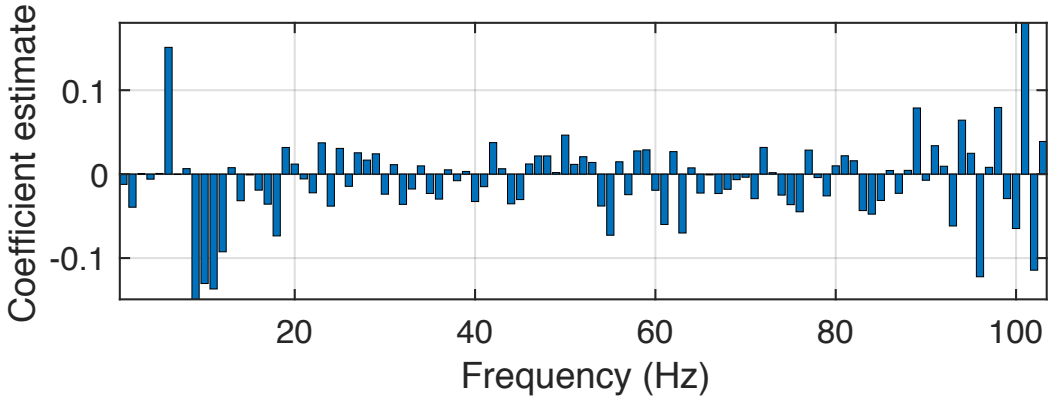
Model 2: Use two frequencies (9Hz)

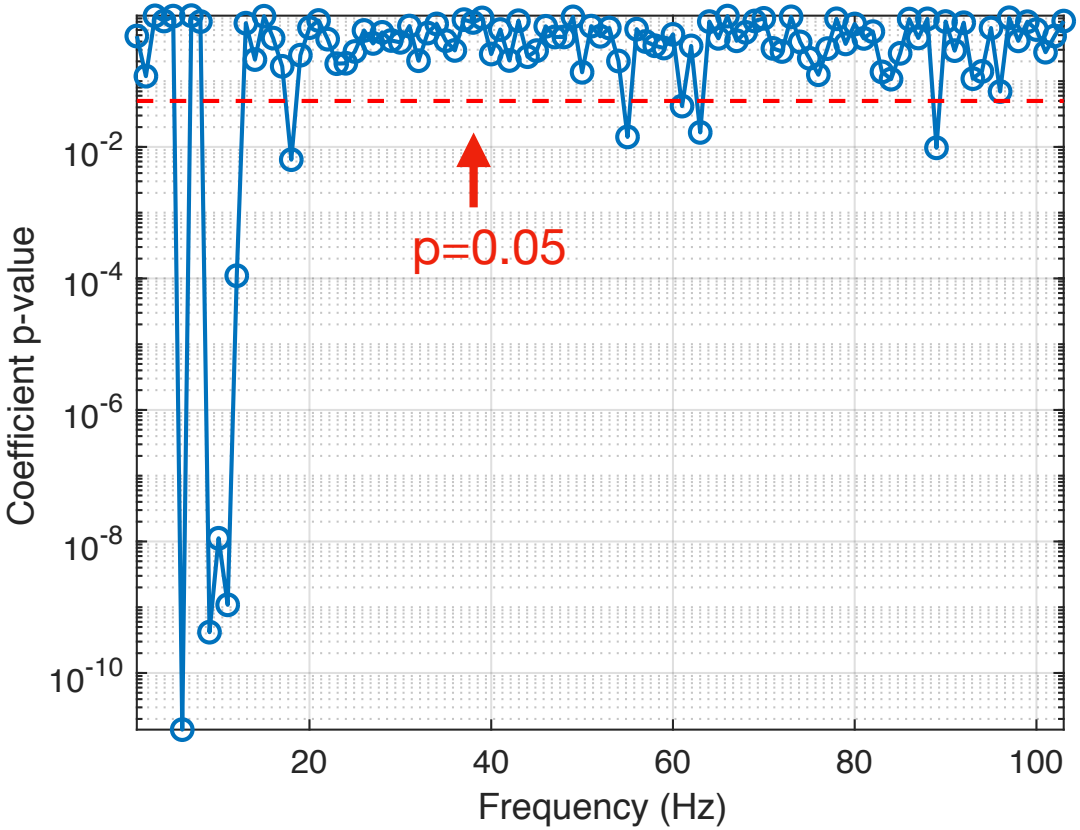


Model 3: Use all 103 frequencies



Note that the coefficients take wildly varying values. Most of the parameters are not statistically significant (p>0.05)





Regularization and Shrinkage

- Regularization tries to prevent overfitting in a different way from variable selection, by “shrinking” the coefficient values
- The idea is to modify least squares by adding a penalty for large coefficient values:

Cost function for least squares:

$$Cost(\beta_1, \beta_2, ...) = RSS = \sum_i (y - \hat{y}_i)^2$$

Cost function for *regularized* least squares:

$$Cost(\beta_1, \beta_2, ...) = RSS + Penalty$$

$$= \sum_i (y - \hat{y}_i)^2 + \lambda F(\beta_1, \beta_2, ...)$$

This term controls how well the model fits the data

This term controls model complexity by forcing the parameters

Ridge Regression

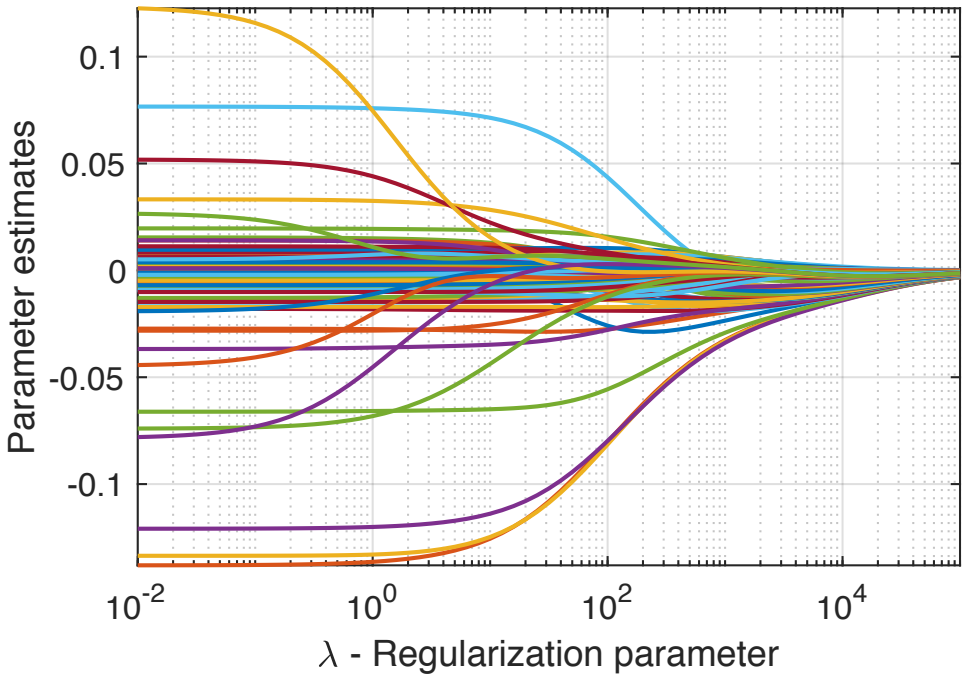
$$Cost(\beta_1, \beta_2, ...) = RSS + Penalty$$

$$= \sum_i (y - \hat{y}_i)^2 + \lambda F(\beta_1, \beta_2, ...)$$

$$= \sum_i (y - \hat{y}_i)^2 + \boxed{\lambda \sum_p (\beta_p)^2}$$

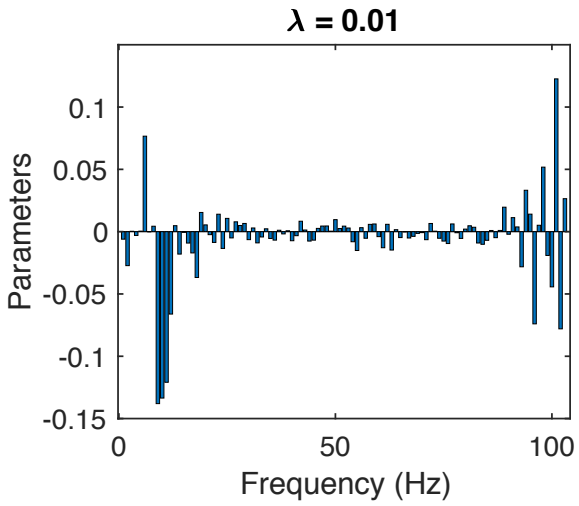
$$= \sum_i (y - \hat{y}_i)^2 + \boxed{\lambda \|\beta\|_2}$$

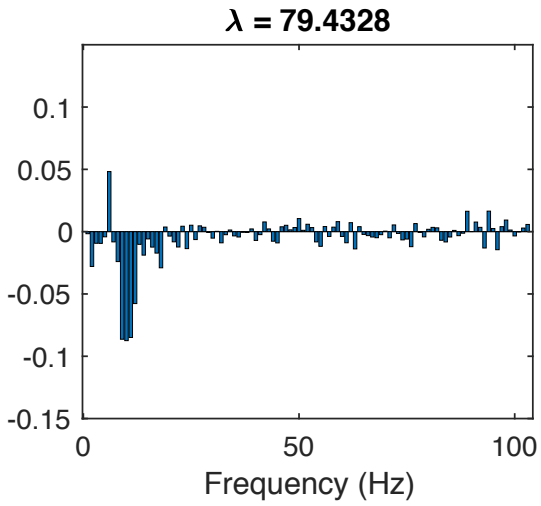
Shrinkage penalty

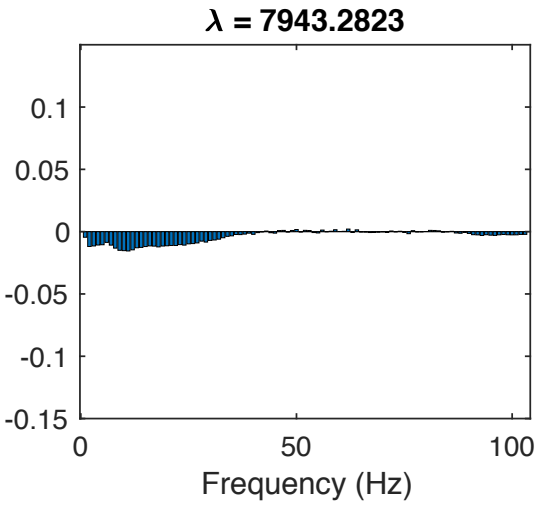


Weak shrinkage

Strong shrinkage





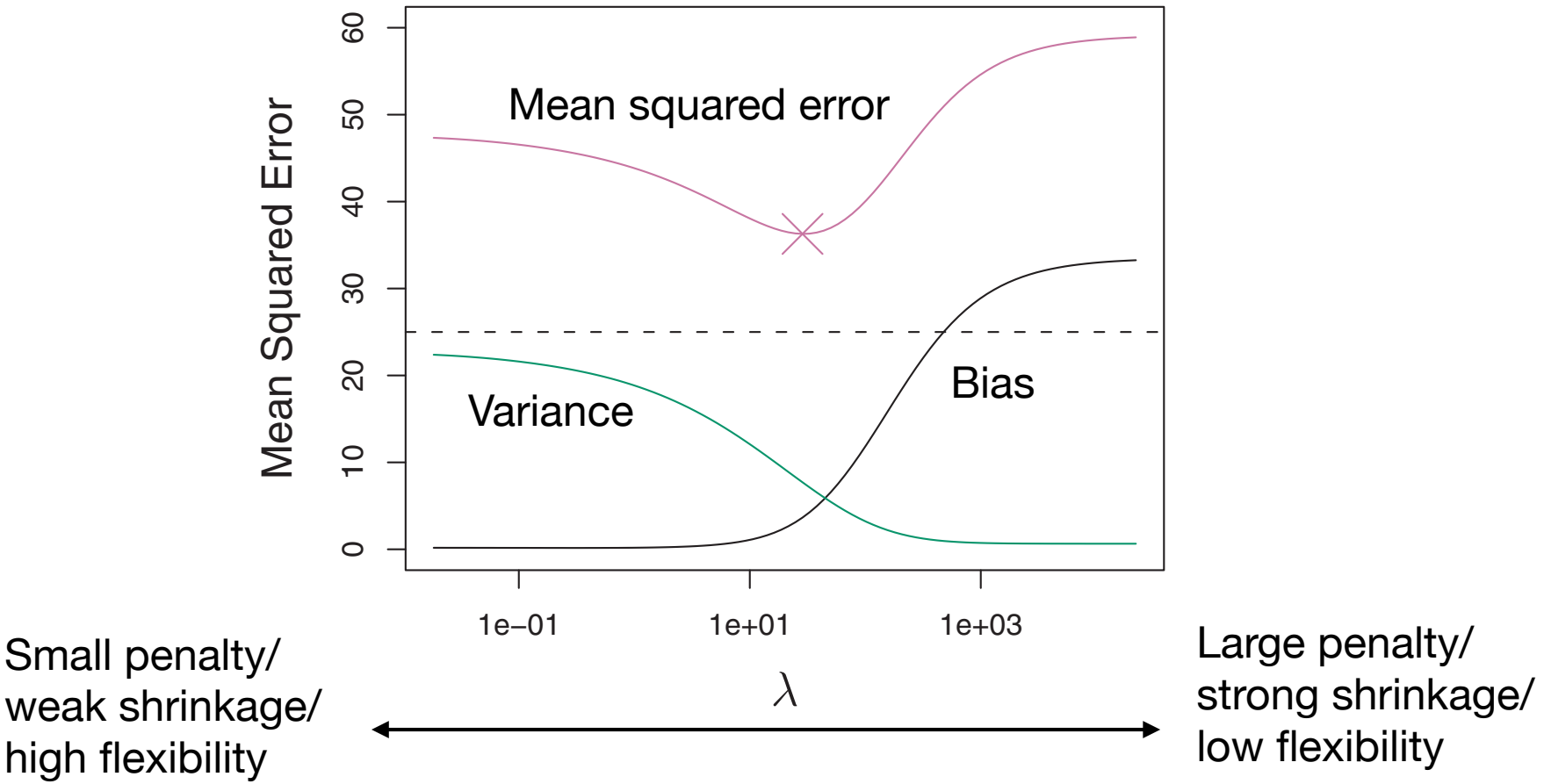


Weak shrinkage:
Coefficients have large, wildly fluctuating values

Intermediate shrinkage:
Keep the most important parameters but reduce the variation overall

Strong shrinkage:
Coefficients all take low values, no wild fluctuations

Why does shrinkage work? By controlling model flexibility!



The penalty parameter, lambda, effectively controls model flexibility/complexity.

It does so by limiting the range of values that the regression coefficients can take.

Ridge regression is equivalent to a constrained optimization

Constrained optimization:

$$\underset{\beta}{\text{minimize}} \left\{ \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 \right\} \quad \text{subject to} \quad \sum_{j=1}^{p} \beta_j^2 \leq s,$$

Loss function

Constraint

Solving a constrained optimization using *Lagrange multipliers*:

$$\min_{\beta} L(\beta) \quad \Leftrightarrow \quad solve: \ \frac{d}{d\beta} L(\beta) = 0$$

$$\min_{\beta} L(\beta) \quad subject\ to \quad f(\beta) \leq s \quad \Leftrightarrow \quad solve: \ \frac{d}{d\beta} [L(\beta) + \lambda f(\beta)] = 0$$

Loss function

Constraint

By choosing the value of lambda, we can solve this problem for different s values:

lambda = 0 corresponds to s=Infinity

lambda = Infinity corresponds to s=0

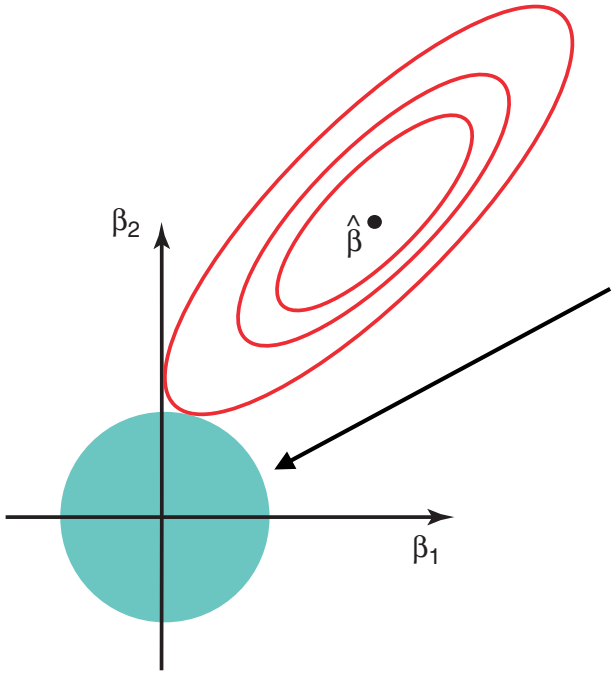
Ridge regression is equivalent to a constrained optimization

Constrained optimization:

$$\underset{\beta}{\text{minimize}} \left\{ \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 \right\} \quad \text{subject to} \quad \sum_{j=1}^{p} \beta_j^2 \leq s,$$

Loss function

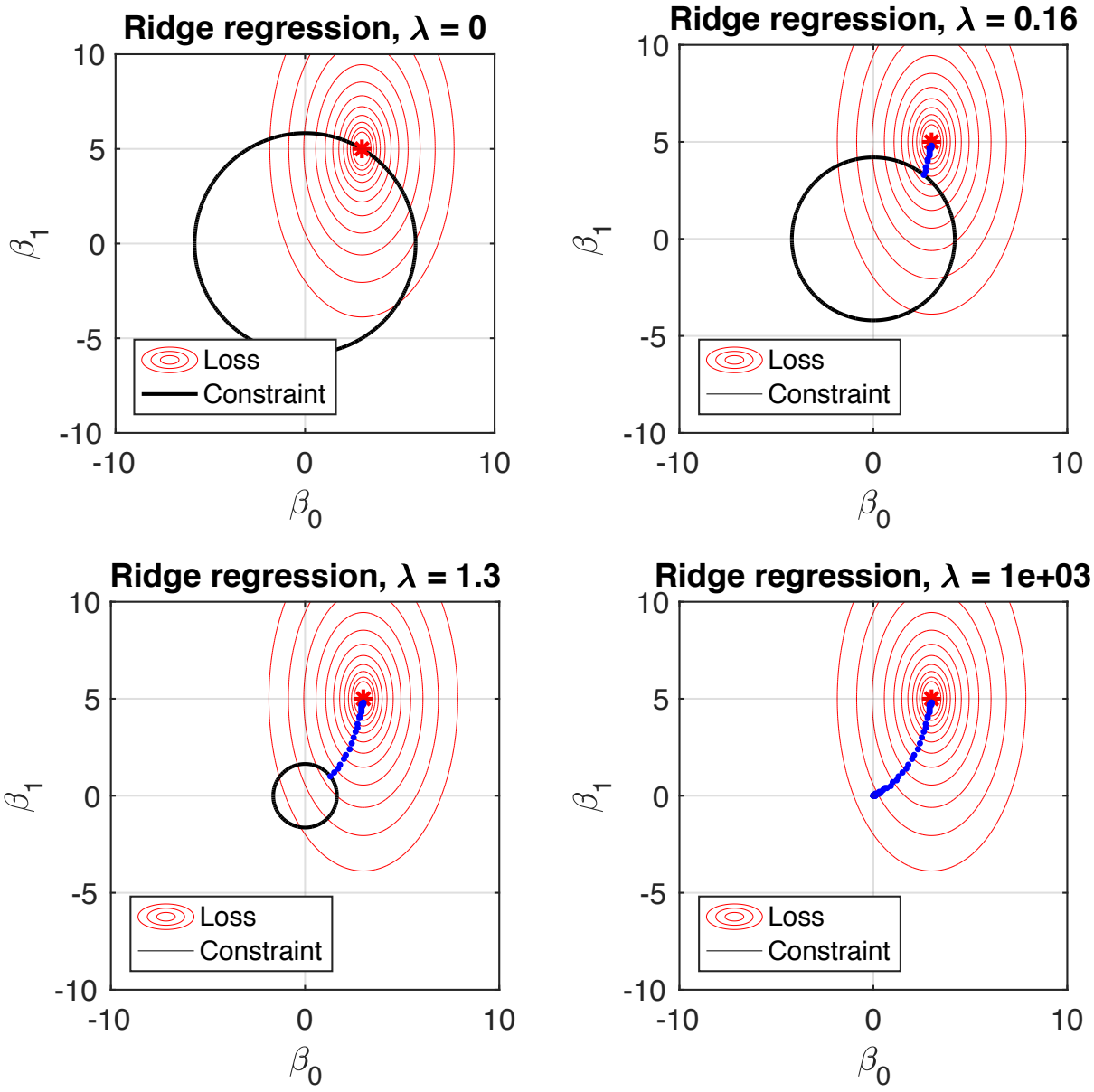
Constraint



$$f(\beta) = \|\beta\|_2 = \beta_1^2 + \beta_2^2 \leq s$$

This “L2” constraint corresponds to a circle centered at (0,0) with radius $s^{1/2}$

Visualizing ridge regression



Use standardized predictors with regularized regression

- In ordinary least squares regression, the scale of the predictors is irrelevant; the

$$y = \beta_0 + \beta_1 x_1 = \beta_0 + (\beta_1/5)\,(5x_1)$$

- When we have a penalty term, the scale matters:

$$(y - \beta_0 + \beta_1 x_1)^2 + \lambda f(\beta_0, \beta_1) \neq (y - \beta_0 + (\beta_1/5)\,(5x_1))^2 + \lambda f(\beta_0, \beta_1/5)$$

- It's best to apply ridge regression after *standardizing* the predictors — divide each predictor by its standard deviation:

$$\tilde{x}_{ij} = \frac{x_{ij}}{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_{ij} - \overline{x}_j)^2}},$$

Workflow for ridge regression

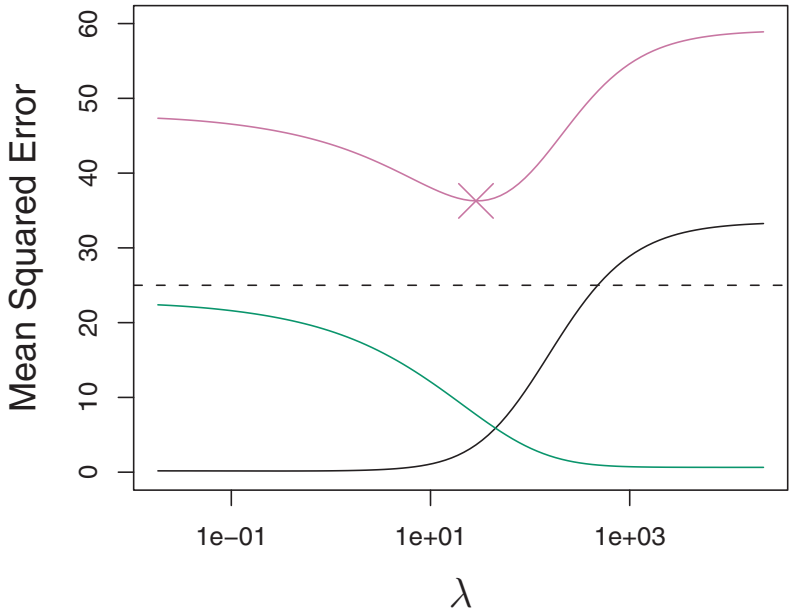
1. First standardize all the predictors

2. For each value of lambda:

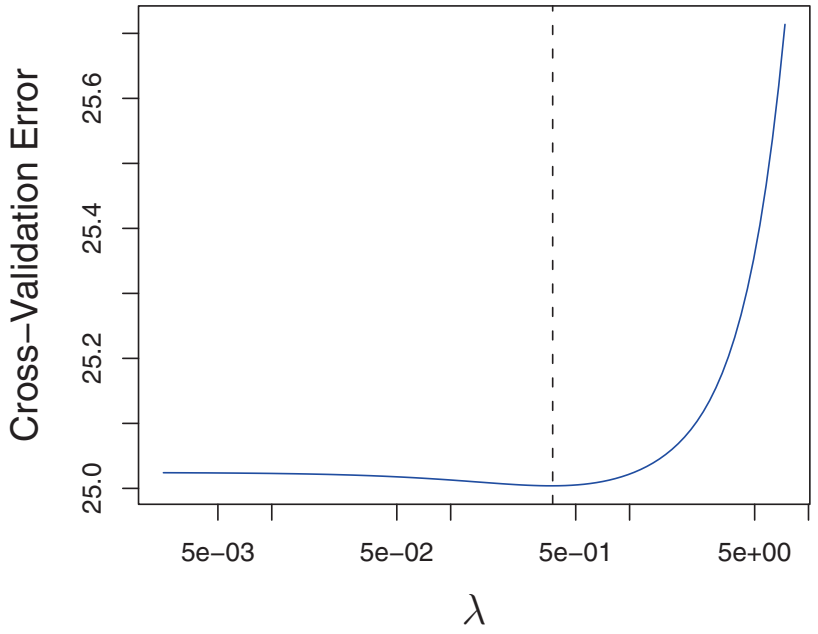
   2.1. Find the best fit (least cost) coefficients using training data

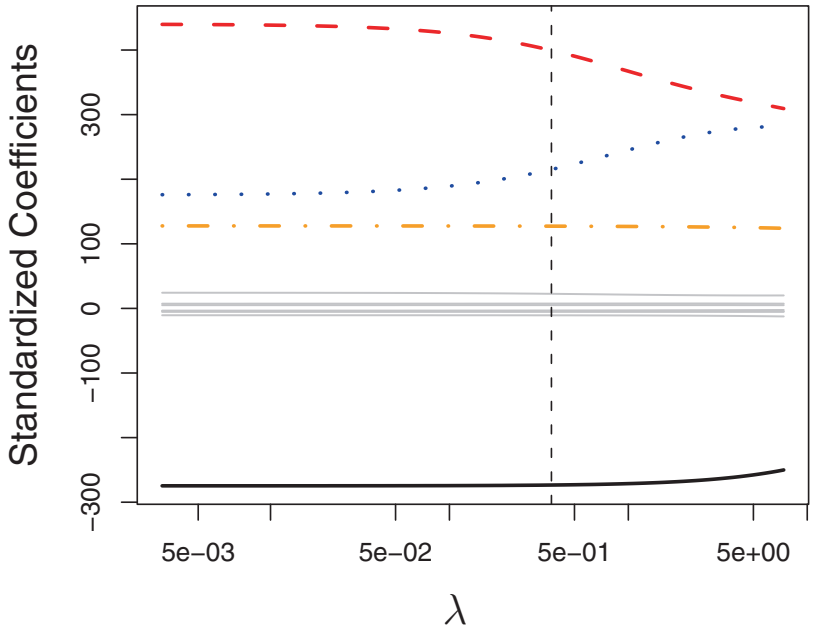
   2.2. Compute the MSE on test data and store it

3. Select the lambda value with the smallest MSE_{test}



Ridge regression with cross-validation





The LASSO

- LASSO = “Least Absolute Shrinkages and Selection Operator”

https://www.youtube.com/watch?v=ANJgTk0x7_Y

Goal of the LASSO: Sparse regression

- Ridge regression uses ALL the predictors, it just shrinks the parameters closer to zero
- LASSO shrinks some of the predictors all the way to zero
- This makes the model more interpretable: It only uses a subset of the variables
    - LASSO is thus similar to subset selection, but it uses a continuous regularization parameter (lambda) rather than trying out all of the subsets

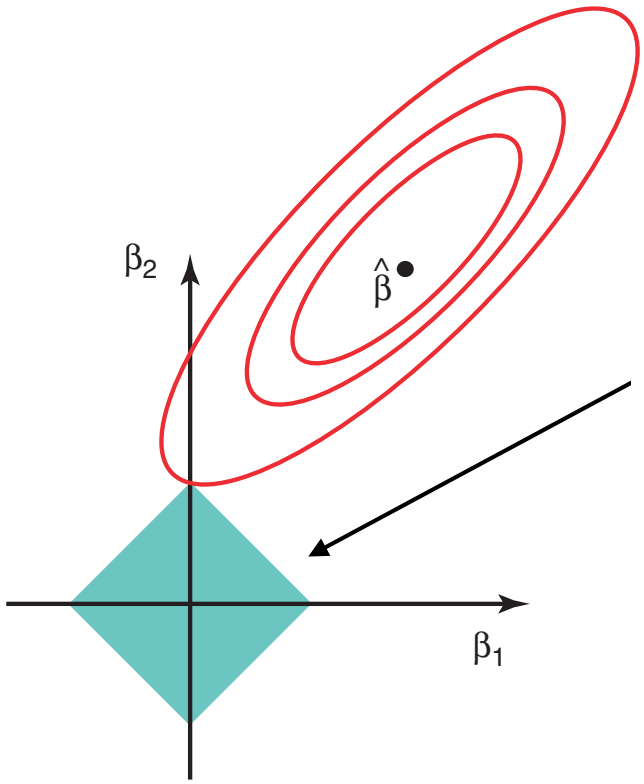
LASSO as a constrained optimization

Constrained optimization:

$$\underset{\beta}{\text{minimize}} \left\{ \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 \right\} \quad \text{subject to} \quad \sum_{j=1}^{p} |\beta_j| \leq s$$

Loss function

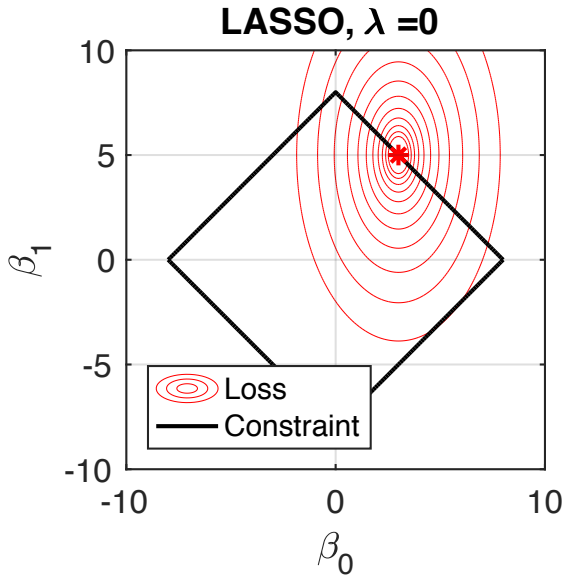
Constraint

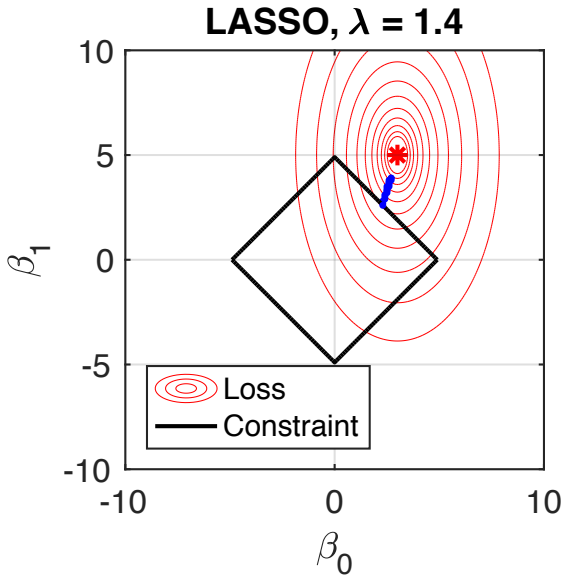


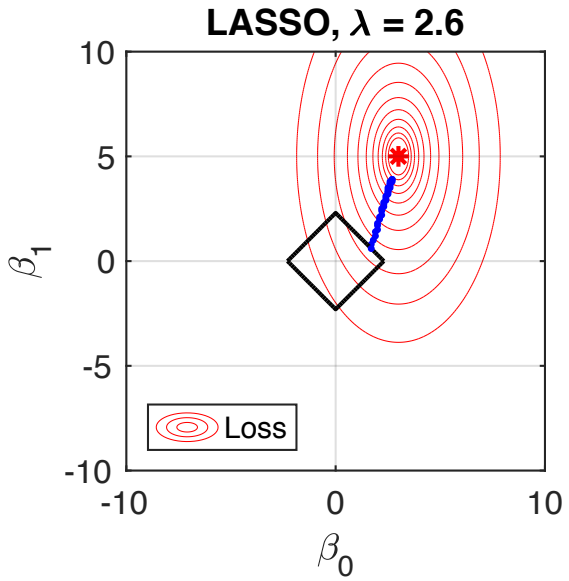
$$f(\beta) = \|\beta\|_1 = |\beta_1| + |\beta_2| \leq s$$

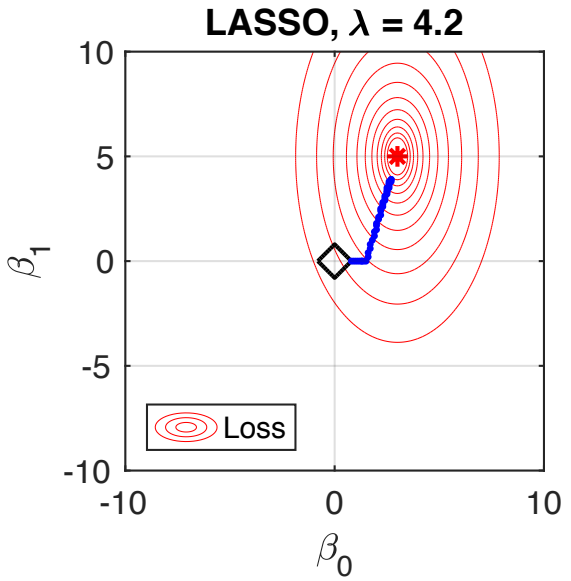
This “L1” constraint corresponds to a diamond centered at (0,0) with diagonal 2s

Visualizing LASSO



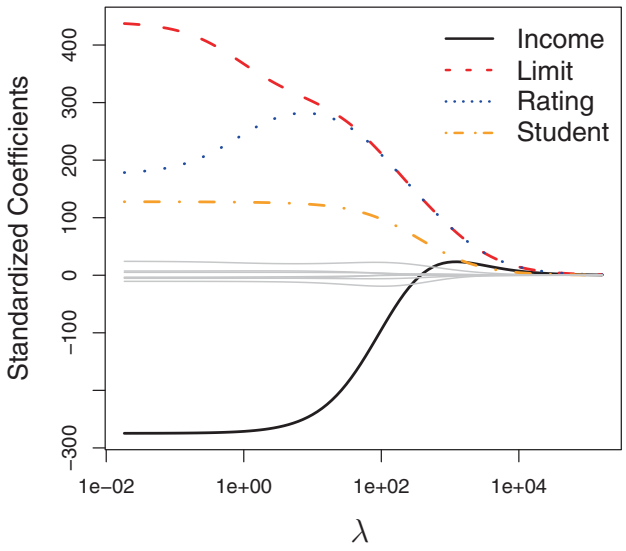




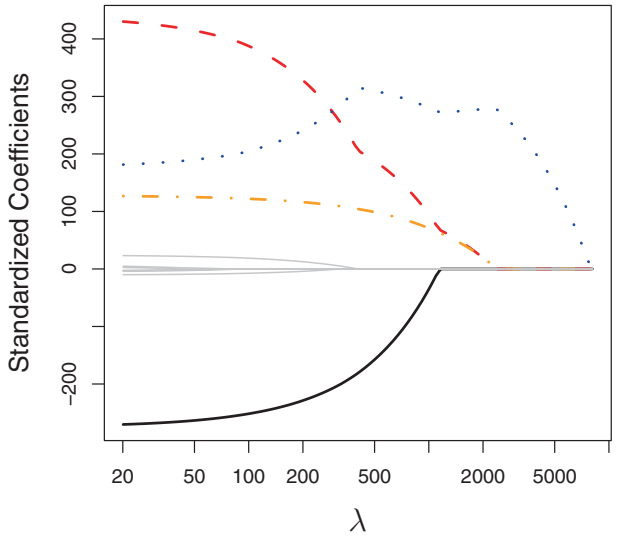


LASSO sets some of the coefficients to zero, Ridge regression does not

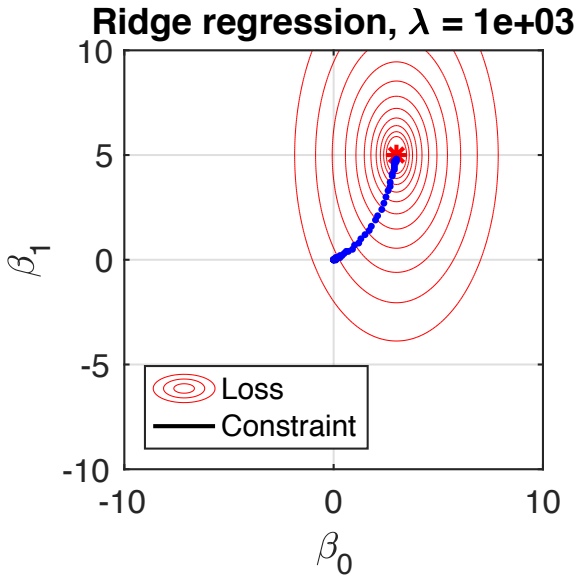
Ridge regression

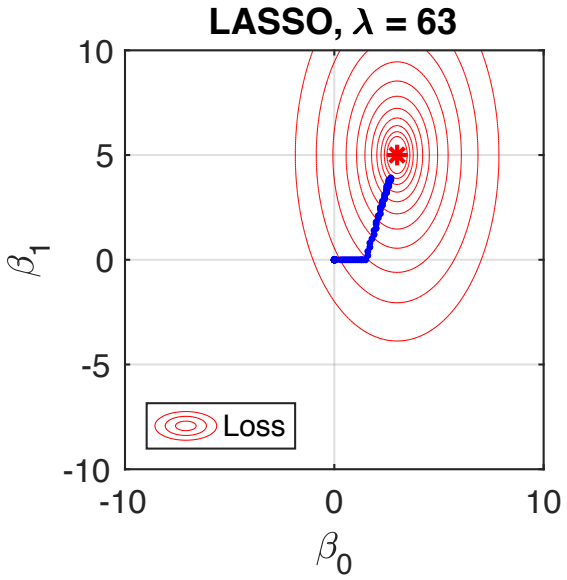


LASSO



LASSO vs. Ridge regression





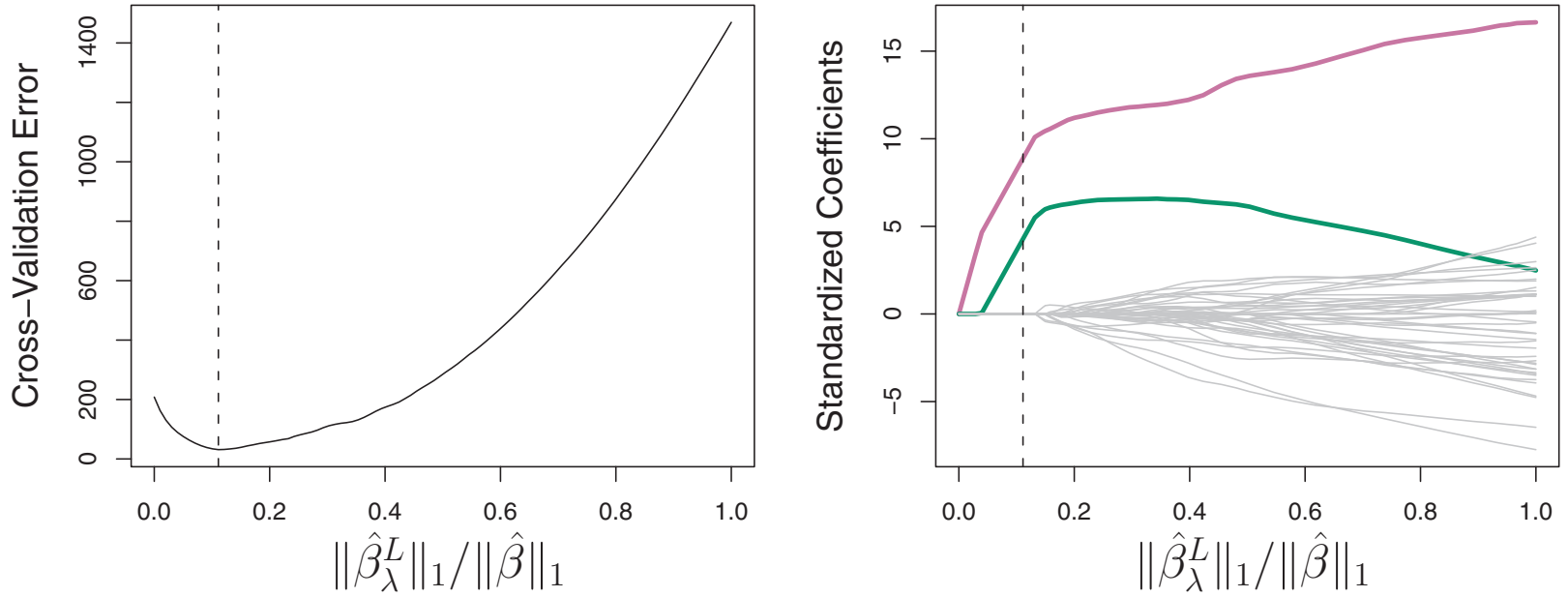


FIGURE 6.13. Left*: Ten-fold cross-validation MSE for the lasso, applied to the sparse simulated data set from Figure 6.9.* Right: *The corresponding lasso coefficient estimates are displayed. The vertical dashed lines indicate the lasso fit for which the cross-validation error is smallest.*