# BIOS 7345: Advanced Regression Analysis I

**Andrew J. Spieker, Ph.D.**

Assistant Professor of Biostatistics
Vanderbilt University Medical Center

Set 11: Regularization

Version: 10/02/2023

# Table of Contents

**Motivation**:

- So far, we have assumed that our $N \times K$ design matrix, $\mathbf{X}$, is such that its $K$ predictors could be supported by $N$ observations ($N \gg K$).
- However, sometimes the covariate space that we're seeking to understand is extremely high-dimensional relative to any sample size we can reasonably obtain.
  - We estimate that humans have about 20,000-25,000 genes.
  - ...and about 3.2 billion base pairs.
- We have handled the case where $\mathbf{X}$ is not of full column rank for $N \gg K$. However, OLS is not equipped to handle $N \leq K$, and the case where $N \not\gg K$ is far from optimal.
  - Some rules of thumb call for at least ten independent observations for each one degree of freedom used.

**Degrees of freedom**:

- One way to define degrees of freedom used by a model (in the case of constant variance) is through the relationship between the observed and fitted values:

$$\text{df}(\widehat{\mathbf{y}}) = \frac{1}{\sigma^2}\sum_{i=1}^{N}\text{Cov}[Y_i, \widehat{Y}_i]$$

- We can use this to show that OLS uses $K$ degrees of freedom:

$$\begin{aligned}
\text{df}(\widehat{\mathbf{y}}) &= \frac{1}{\sigma^2}\sum_{i=1}^{N}\text{Cov}[Y_i, \mathbf{x}_i^T(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{x}_i Y_i] \\
&= \frac{1}{\sigma^2}\sum_{i=1}^{N}\mathbf{x}_i^T(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{x}_i\text{Cov}[Y_i, Y_i] \\
&= \sum_{i=1}^{N}\mathbf{x}_i^T(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{x}_i = \text{trace}(\mathbf{P}) = \sum_{i=1}^{N}h_i = K.
\end{aligned}$$

# Penalized regression

**Degrees of freedom**:

- Intuitively, the more degrees of freedom used by a model, the more closely correlated the fitted and observed values. Adding more degrees of freedom to a model improves the model's fit *to the data*.
- The problem is that our goal is not simply to predict observations in our own data, but to use the data to predict observations *external* to our data.
- Many machine-learning approaches focus on ways to balance the following trade-off:
  - Allowing the model to use available covariate information.
  - Controlling the model from overfitting the data.
- We'll focus in this set of notes specifically on penalized least squares.

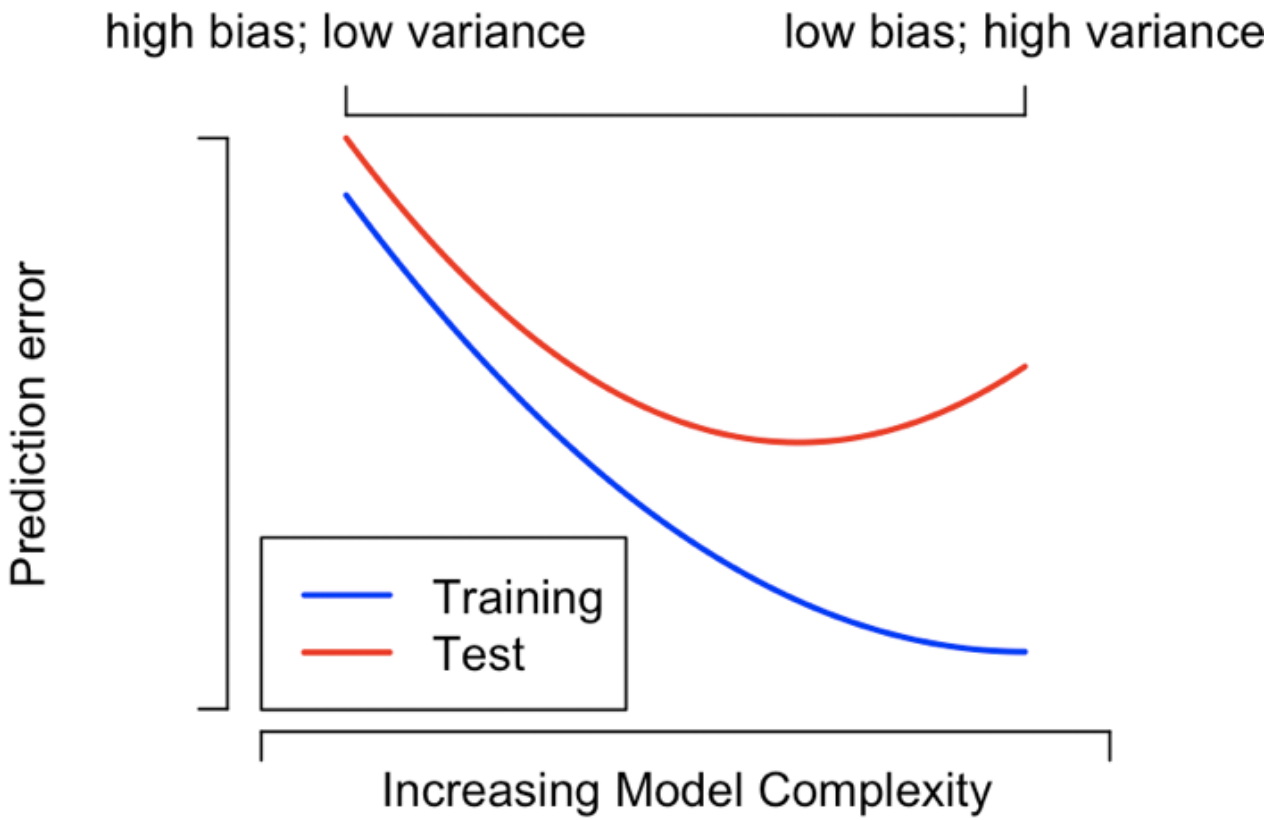
**Prediction error**: Bias-variance decomposition

- Controlling degrees of freedom is central to being able to navigate the bias-variance trade-off.
- Consider the goal of estimating: $\mathrm{E}[Y|\mathbf{X} = \mathbf{x}] = f(\mathbf{x})$.
- Estimate $f$ with $\widehat{f}$.
- Consider random out-of-sample observation, $(\mathbf{X}_0, Y_0)$.
- Mean squared error at $\mathbf{X}_0$ is:

$$\begin{aligned}\mathrm{E}[(Y_0 - \widehat{f}(\mathbf{X}_0))^2] &= \mathrm{E}[(Y_0 - \mathrm{E}[Y_0])^2] + \mathrm{E}[(\mathrm{E}[Y_0] - \mathrm{E}[\widehat{f}(\mathbf{X}_0)])^2] \\ &\quad + \mathrm{E}[(\widehat{f}(\mathbf{X}_0) - \mathrm{E}[\widehat{f}(\mathbf{X}_0)])^2] \\ &= \sigma^2_{Y_0} + \mathrm{Bias}^2(\widehat{f}(\mathbf{X}_0)) + \mathrm{Var}(\widehat{f}(\mathbf{X}_0)).\end{aligned}$$

- Prediction error: trade-off between bias and variance.
  - Bias: Model correctly captures $f$ on average?
  - Variance: Model precisely captures relationship?

**Important figure**:



# Table of Contents

# The ridge penalty

**Objective function**:

- In OLS, we seek to solve the following problem:

$$\underset{\boldsymbol{\beta}}{\text{minimize}} \; ||\mathbf{y} - \mathbf{X}\boldsymbol{\beta}||^2$$

- One way to control the degrees of freedom used by the model is to shrink some of the coefficients toward zero.
  - Including a predictor in a model in an unconstrained fashion uses one degree of freedom.
  - Leaving the predictor out of the model effectively treats the corresponding coefficient as zero and uses no degrees of freedom.
  - Intuitively, including a predictor in a model with a shrunken coefficient should use "somewhere between zero and one" degrees of freedom.

# The ridge penalty

**Objective function**:

- Suppose we add a penalty to the objective function:

$$\underset{\boldsymbol{\beta}}{\text{minimize}} \; ||\mathbf{y} - \mathbf{X}\boldsymbol{\beta}||^2 + \lambda||\boldsymbol{\beta}||^2$$

- The parameter $\lambda$ is not "estimated" per se, but rather "chosen" by the researcher in some informed way (typically data-driven, but more on this later).
  - $\lambda = 0$: this is just the OLS problem ($K$ degrees of freedom).
  - $\lambda = \infty$: $\widehat{\boldsymbol{\beta}} = 0$ (zero degrees of freedom).
  - $0 < \lambda < \infty$: $||\widehat{\boldsymbol{\beta}}||^2$ shrunken toward 0 ($>0$, $<K$ degrees of freedom).

**Objective function**:

- How do we obtain a solution to this optimization problem?

$$\underset{\boldsymbol{\beta}}{\text{minimize}} \; ||\mathbf{y} - \mathbf{X}\boldsymbol{\beta}||^2 + \lambda||\boldsymbol{\beta}||^2$$

- This is a convex loss function; the calculus is similar to the OLS case:

$$\begin{aligned} \frac{\partial}{\partial \boldsymbol{\beta}} \left( ||\mathbf{y} - \mathbf{X}\boldsymbol{\beta}||^2 + \lambda||\boldsymbol{\beta}||^2 \right) &= -2\mathbf{X}^T(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}) + 2\lambda\boldsymbol{\beta} \\ &= -2\mathbf{X}^T\mathbf{y} + 2\mathbf{X}^T\mathbf{X}\boldsymbol{\beta} + 2\lambda\boldsymbol{\beta} \\ \Rightarrow \widehat{\boldsymbol{\beta}}_\lambda &= (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}. \end{aligned}$$

**Solution to penalized least squares equations**:

- For the ridge penalty, there is a closed-form solution:

$$\widehat{\boldsymbol{\beta}}_\lambda \quad = \quad (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}.$$

- Letting $d_1, \ldots, d_K$ denote the singular values of $\mathbf{X}$, we have that:

$$\text{df}(\widehat{\mathbf{y}}) \quad = \quad \sum_{k=1}^{K} \frac{d_k^2}{d_k^2 + \lambda},$$

  which you will prove for homework!
- Quick check: if $\lambda = 0$, then $\text{df}(\widehat{\mathbf{y}}) = K$, and when $\lambda = \infty$, $\text{df}(\widehat{\mathbf{y}}) = 0$.

# The ridge penalty

**Penalized estimates**:

- It is computationally cleaner to center the variables to avoid estimating the intercept (though I can estimate it on the back end).
- The ridge regression is not scale-invariant!!
  - If one of the predictors is height, I get a different answer depending on whether I measure in inches, feet, meters, etc.
  - We can deal with this by scaling everything to have unit variance prior to estimation.
- If $\widehat{\boldsymbol{\beta}}_\lambda$ denotes the standardized (centered/scaled) estimates:
  - The penalized coefficients can be obtained as $\sigma_Y \text{diag}(1/\sigma_{x_k})\widehat{\boldsymbol{\beta}}_\lambda$.
  - $\widehat{\beta}_{\lambda,0} = \mu_y - \sigma_Y \text{diag}(1/\sigma_{x_k})\boldsymbol{\mu}_{\mathbf{X}}^T\widehat{\boldsymbol{\beta}}_\lambda$ is the penalized intercept.

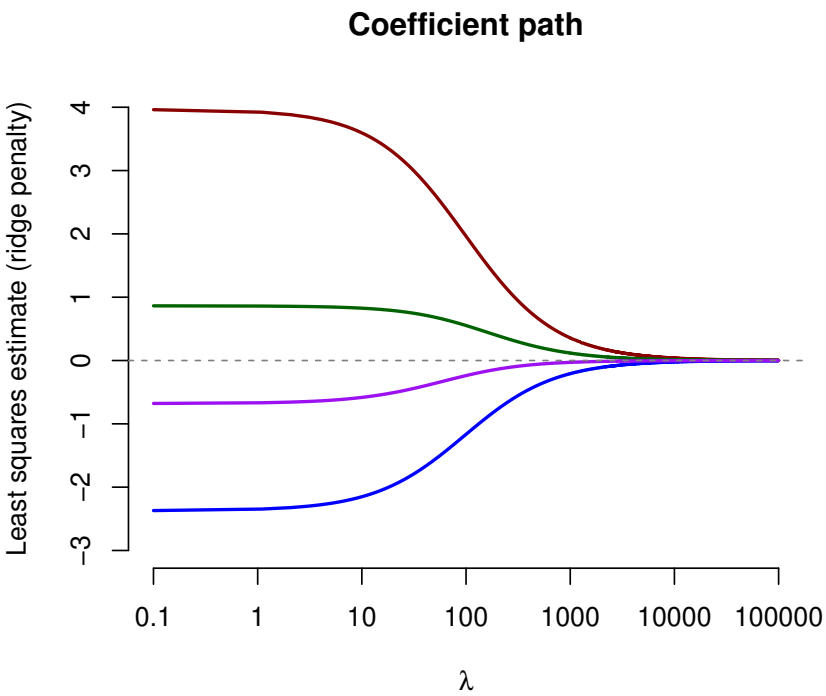
**Simulation**: Parameter setup

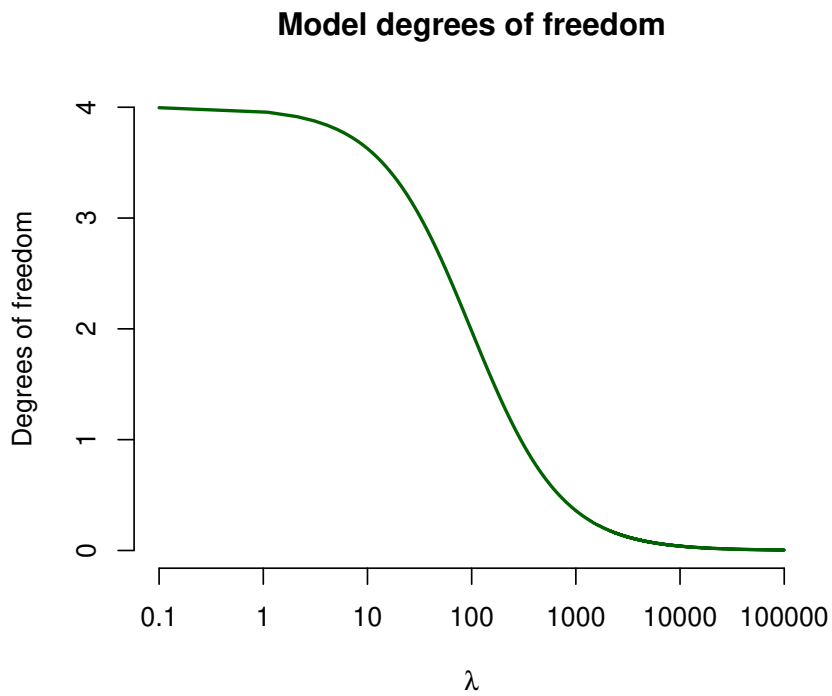
```
## Set sample size
n <- 100

## Generate predictors
x <- rmvnorm(n, mean = rep(0, 4), sigma = 0.5*diag(4))

## Generate outcome
y <- 3 + 4*x[,1] - 2*x[,2] + 0.1*x[,3] - 1*x[,4] + rnorm(n, 0, 2)

## Center/scale cariables
means <- colMeans(x); sds <- apply(x, 2, sd)
meanY <- mean(Y); sdY <- sd(Y)
x1 <- (x[,1] - means[1])/sds[1]
x2 <- (x[,2] - means[2])/sds[2]
x3 <- (x[,3] - means[3])/sds[3]
x4 <- (x[,4] - means[4])/sds[4]
y <- (y - meanY)/sdY
```

**Simulation**: Fitting models; extracting results

```
## Generate design matrix
X <- cbind(x1, x2, x3, x4)

## Choose values for tuning parameter
lambda <- seq(0.1, 100000, 1)

## Create place to store results
results <- matrix(0, nrow = length(lambda), ncol = 4)
dfres <- matrix(0, nrow = length(lambda), ncol = 1)

for (j in 1:length(lambda))
{
  ## Compute estimate, re-scale, and store results
  bhat.lambda <- solve(t(X) %*% X + diag(lambda[j], 4)) %*% t(X) %*% y
  results[j,] <- sdY*bhat.lambda/sds

  ## Determine degrees of freedom
  sv <- svd(X)$d
  dfres[j,1] <- sum(sv^2/(sv^2 + lambda[j]))
}
```

## **Simulation**: Results



# The ridge penalty

**Solution to penalized least squares equations**:

- Interestingly, $(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})$ is invertible for all choices of $\lambda > 0$, even if $\mathbf{X}^T\mathbf{X}$ is singular. You will argue this on a homework problem!

**Ridge and Bayes**:

- Suppose that $\mathbf{y} \sim \mathcal{N}(\mathbf{X}\boldsymbol{\beta}, \sigma^2\mathbf{I})$.
- Consider the Gaussian prior $\boldsymbol{\beta} \sim \mathcal{N}(0, \tau^2\mathbf{I})$.
- It is straightforward to show that the posterior mode, $\widehat{\boldsymbol{\beta}}_\pi$, is given by:

$$\widehat{\boldsymbol{\beta}} = \left(\mathbf{X}^T\mathbf{X} + \frac{\sigma^2}{\tau^2}\mathbf{I}\right)^{-1}\mathbf{X}^T\mathbf{y}.$$

- Letting $\lambda = \sigma^2/\tau^2$ makes clear the connection with the ridge penalty.
- Bayesian methods can be understood more generally as shrinkage approaches (it is arguably one of the best ways to think about Bayesian methods).

# Table of Contents

# The LASSO penalty

**Objective function**:

- There's no rule that says we *have* to penalize via the $\ell_2$-norm. Suppose we instead penalize based on the $\ell_1$-norm:

$$\underset{\boldsymbol{\beta}}{\text{minimize}} \ \frac{1}{2}||\mathbf{y} - \mathbf{X}\boldsymbol{\beta}||^2 + \lambda||\boldsymbol{\beta}||_1,$$

  where $||\boldsymbol{\beta}||_1 = \sum_{k=1}^{K} |\beta_k|$.
- While a different choice of a penalty seems as if it should just be a technicality, the LASSO penalty actually produces very different behavior as compared to the ridge penalty.
- Geometrically, the fact that $||\boldsymbol{\beta}||^2$ is "circular" and $||\boldsymbol{\beta}||_1$ "has sharp corners" is what accounts for differences in behavior.

# The LASSO penalty

**Objective function**:

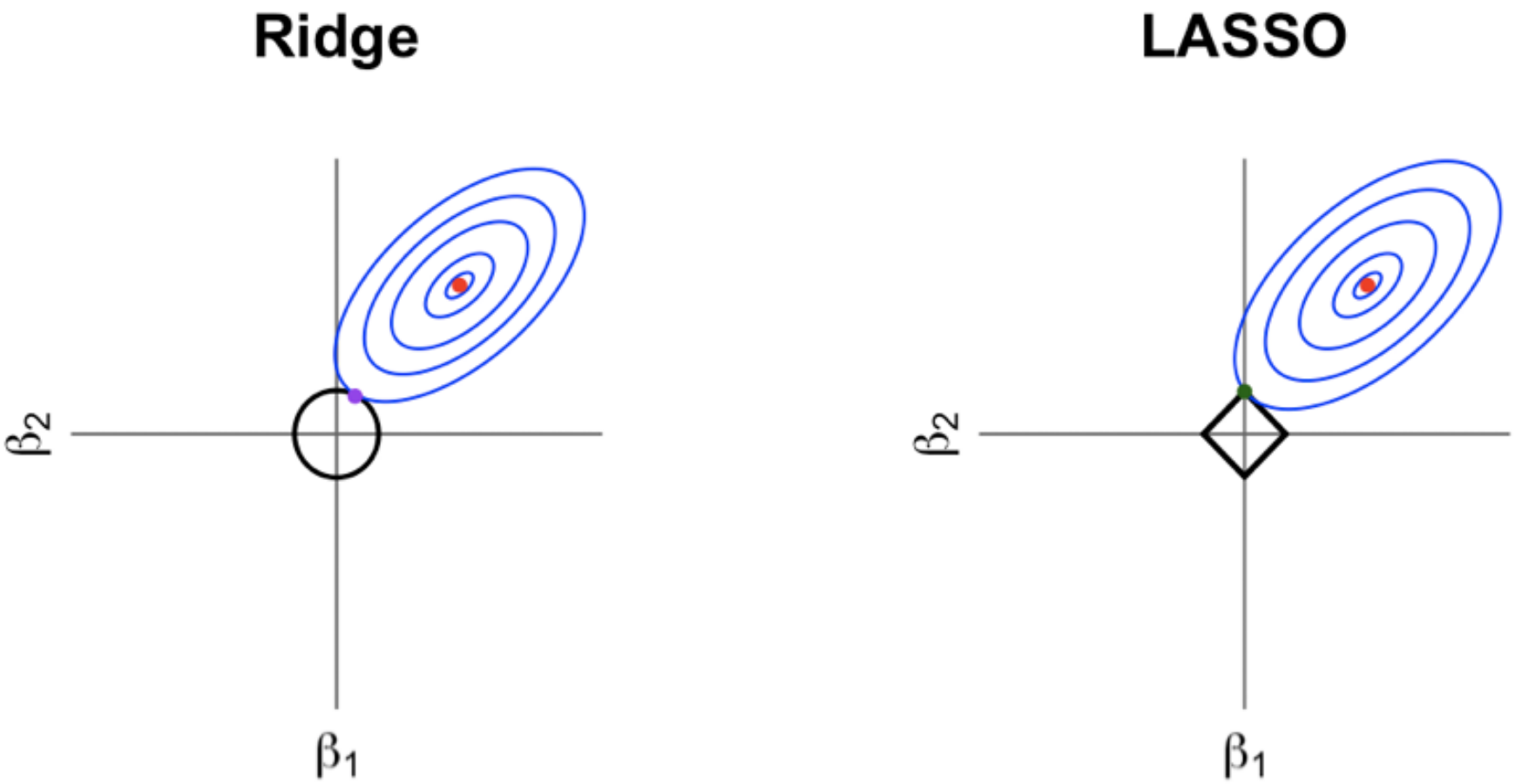
- Rephrasing the ridge penalty problem:

$$\underset{\boldsymbol{\beta}}{\text{minimize}} \; ||\mathbf{y} - \mathbf{X}\boldsymbol{\beta}||^2 \text{ subject to constraint } ||\boldsymbol{\beta}||^2 \leq c$$

- Rephrasing the LASSO penalty problem:

$$\underset{\boldsymbol{\beta}}{\text{minimize}} \; ||\mathbf{y} - \mathbf{X}\boldsymbol{\beta}||^2 \text{ subject to constraint } ||\boldsymbol{\beta}||_1 \leq c$$

**Comparing penalties**:

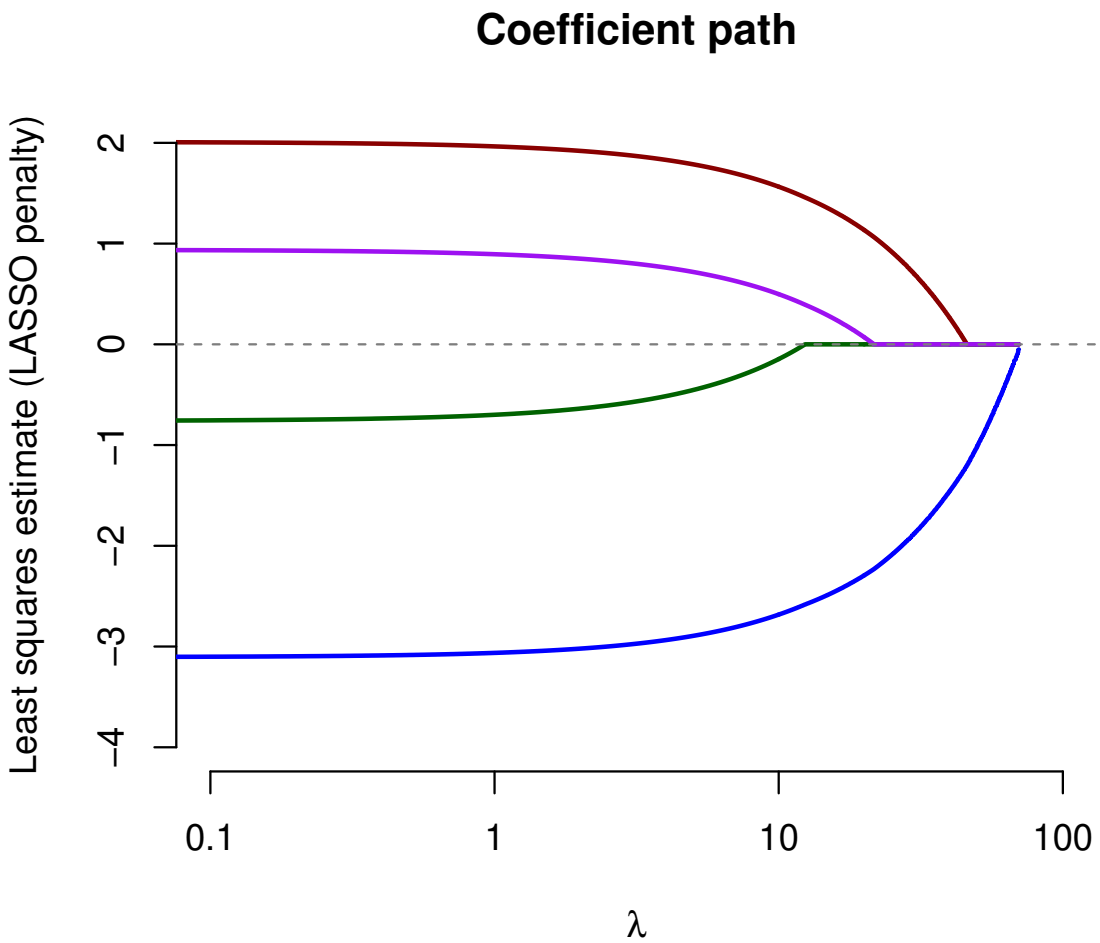


# The LASSO penalty

**Geometry**:

- While the ridge penalty shrinks coefficients *toward* zero, the LASSO penalty shrinks coefficients *to* zero.
- LASSO is said to perform "feature selection," so that it allows you to determine a smaller set of predictors that must be measured to form a prediction on an external observation. This will have good performance when there are a large number of variables having no association with the outcome.
  - This is still a source of debate and controversy.
- On the other hand, ridge will perform better when most of the predictors are associated with the outcome.

**Example**: Coefficient path



# The LASSO penalty

**Estimation**:

- Sadly, there is no closed-form expression for the LASSO-penalized least squares problem.
- Instead, we rely on coordinate-descent algorithms to obtain numerical solutions. These algorithms are lightning-fast (same computational cost as ridge), but a pain to code.
- Under certain assumptions, the number of degrees of freedom used by the LASSO model can be shown to be estimated as the number of non-zero predictors selected into the model.

# Table of Contents

# Cross-validation

**Estimating test error**:

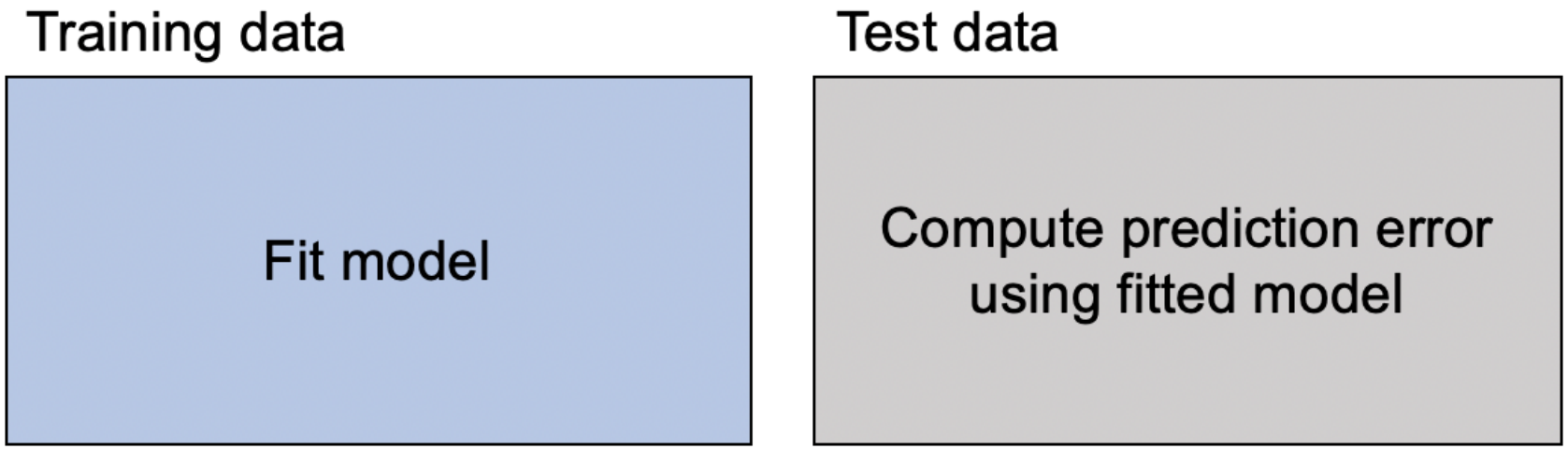
1. We haven't yet talked about how to select a tuning parameter for penalized regression.
2. Given a particular $\lambda$, how does one go about estimating the model's "external" predictive capacity?
   - We typically call this the *test error*, and it can be characterized in a number of ways (mean squared error, among other things).

# Cross-validation

**Estimating test error**: Simple method

- The basic idea is to try to mimic the process of getting two independent sets of data.
  1. Split your sample into random halves; label one of those halves as a *training* set and the other as a *test* set.
  2. Fit your model on the training set.
  3. Use fitted model to predict the values on the test set and estimate the test prediction error.
- The split doesn't have to be 50/50. Typically, we use a larger portion of the data to develop/fit the model, and a smaller portion to compute the test error.
- Training error tends to be more optimistic than the test error. This is generally true across various models (not just penalized regression).

**Estimating test error**:



# Cross-validation

**Model building**:

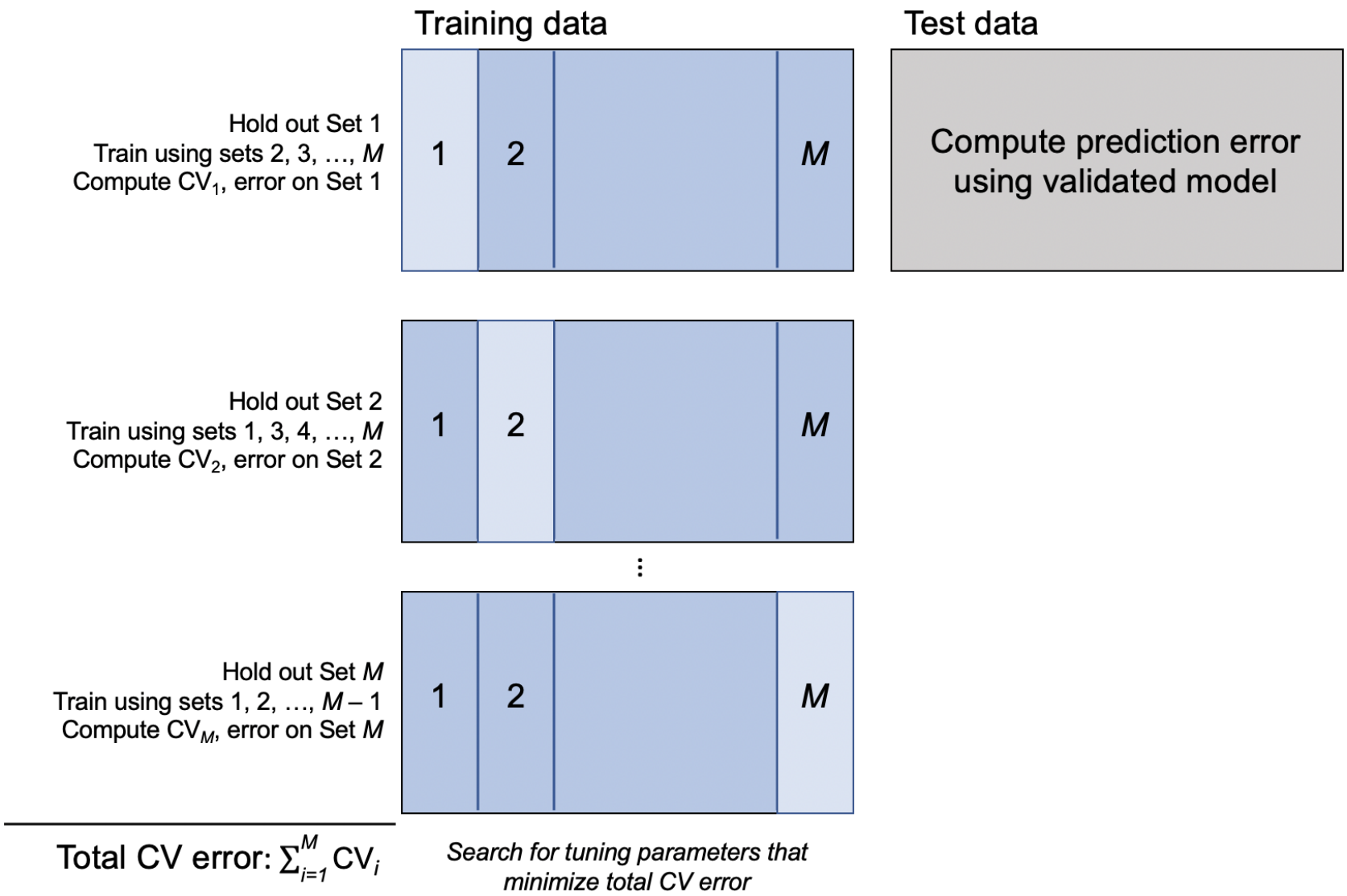
- How do we use the training data to select the tuning parameter?
- Typically, we split the training data into subsets via a procedure known as cross-validation.

# Cross-validation

**Procedure**: $M$-fold cross-validation

- For some range $\lambda$:
  1. Split training observations into $M$ equally-sized folds.
  2. For each $m = 1, \ldots, M$, fit model using the observations *not* in the $m^{\text{th}}$ fold, and estimate validation error, $\text{CV}_m$, for observations in the $m^{\text{th}}$ fold.
  3. Calculate total cross-validation error, $\text{CV}_\lambda = \sum_{m=1}^{M} \text{CV}_m$.
- Choose $\lambda$ to minimize $\text{CV}_\lambda$ (grid search).

# Cross-validation



**Cross-validation**: Special cases

- $M = 2$: Random halves.
  - Easy to explain; low computation time; aggregation of splits.
  - Variation due to random split, variation owing to fits from smaller splits of training data.
- $M = N$: Leave-one-out-cross-validation.
  - No random variation owing to random splits; use most of training data to fit model at each iteration.
  - Models highly correlated; high computation time in some settings (but not all).
- Often, we use $M = 5$ or $M = 10$ (Goldilocks situation).

# Summary

**So far**:

- Regularization.

# Summary

**Up next**:

- Generalized linear models.