WSABI Summer Research Lab **Summer 2024**

Lecture 14: Regularization and Ridge Regression

Instructor: Ryan Brill *Scribe: Jonathan Pipping*

14.1 Motivating Example: MLB Park Effects

We want to estimate the park effect α at each MLB ballpark, which represents the expected runs scored in one half-inning at that park above/below that of an average park, if an average offense faces an average defense. A full analysis is included in [BW], which we will replicate here.

14.1.1 Data

Our training data includes all half-innings from 2017-2019, each row i is a half-inning with the following features:

- park(i): the ballpark of half-inning i
- ot(i) the offensive team-season of half-inning i
- dt(i) the defensive team-season of half-inning i
- y_i: the number of runs scored in half-inning i

From here, we write out our model.

14.1.2 The Model

$$y_i = x_i^T \boldsymbol{\beta} + \epsilon_i, \quad \mathbb{E}[\epsilon_i] = 0$$

where $\boldsymbol{X}$ is a matrix whose i^{th} row is defined by

$$x_i = [1, \overset{\text{park}(1)}{\bullet}, \overset{\text{park}(2)}{\bullet}, \cdots, \overset{\text{park}(30)}{\bullet}, \overset{\text{ot}(1)}{\bullet}, \overset{\text{ot}(2)}{\bullet}, \cdots, \overset{\text{ot}(30)}{\bullet}, \overset{\text{dt}(1)}{\bullet}, \overset{\text{dt}(2)}{\bullet}, \cdots, \overset{\text{dt}(30)}{\bullet}]$$

where each park $\bullet$ is a 1 at park(i) and 0 otherwise, each offensive team $\bullet$ is a 1 at ot(i) and 0 otherwise, and each defensive team $\bullet$ is a 1 at dt(i) and 0 otherwise.

and $\boldsymbol{\beta}$ is a vector of parameters:

$$\boldsymbol{\beta} = [\beta_0, \ \alpha_1, \ \alpha_2, \ \cdots, \ \alpha_{30}, \ \beta_1, \ \beta_2, \ \cdots, \ \beta_{30}, \ \gamma_1, \ \gamma_2, \ \cdots, \ \gamma_{30}]$$

14.1.3 The Problem of Multicollinearity

When we set up the model this way, we run into a problem of multicollinearity, or linear dependence between the columns of $\boldsymbol{X}$. In this case, this occurs because when the home team is on offense, park(i) = ot(i), and when the road team is on offense park(i) = dt(i). This makes it difficult to disentangle $\alpha_{\text{park}(i)}$ from $\beta_{\text{ot}(i)}$

and $\gamma_{\text{dt}(i)}$. Practically, this equates to confusion over whether the runs scored in each half-inning are due to the offensive home team being good or the park being easy.

To disentangle these effects, we need a huge number of instances of **road teams on offense** to figure out each $\beta_{ot(i)}$ well and the same for **home teams on offense** to figure out each $\gamma_{dt(i)}$. Then, with β_{ot} and γ_{dt} well estimated, we can disentangle α_{park}.

Our current dataset consists of 123,252 half-innings. This may seem like a lot of data, but due to the multicollinearity this actually isn't a huge amount of data. It's easy to overfit to noise and get the coefficients wrong. To demonstrate, we run a simulation study.

14.2 Simulation Study

14.2.1 Setting Up the Simulation

The idea behind a simulation study is to **pretend** we know the true parameters (in our case, the $\boldsymbol{\alpha}, \boldsymbol{\beta}$, and $\boldsymbol{\gamma}$ coefficients), generate fake historical data y (in our case, 123,252 fake half-inning outcomes), and then see how well we estimate the coefficients from this synthetic data. Our hope in this study is to understand how much multicollinearity affects our park effect estimates, and how well OLS recovers the true effects.

To do this, suppose the true coefficients are those in 14.1, which are chosen to have a reasonable scale.

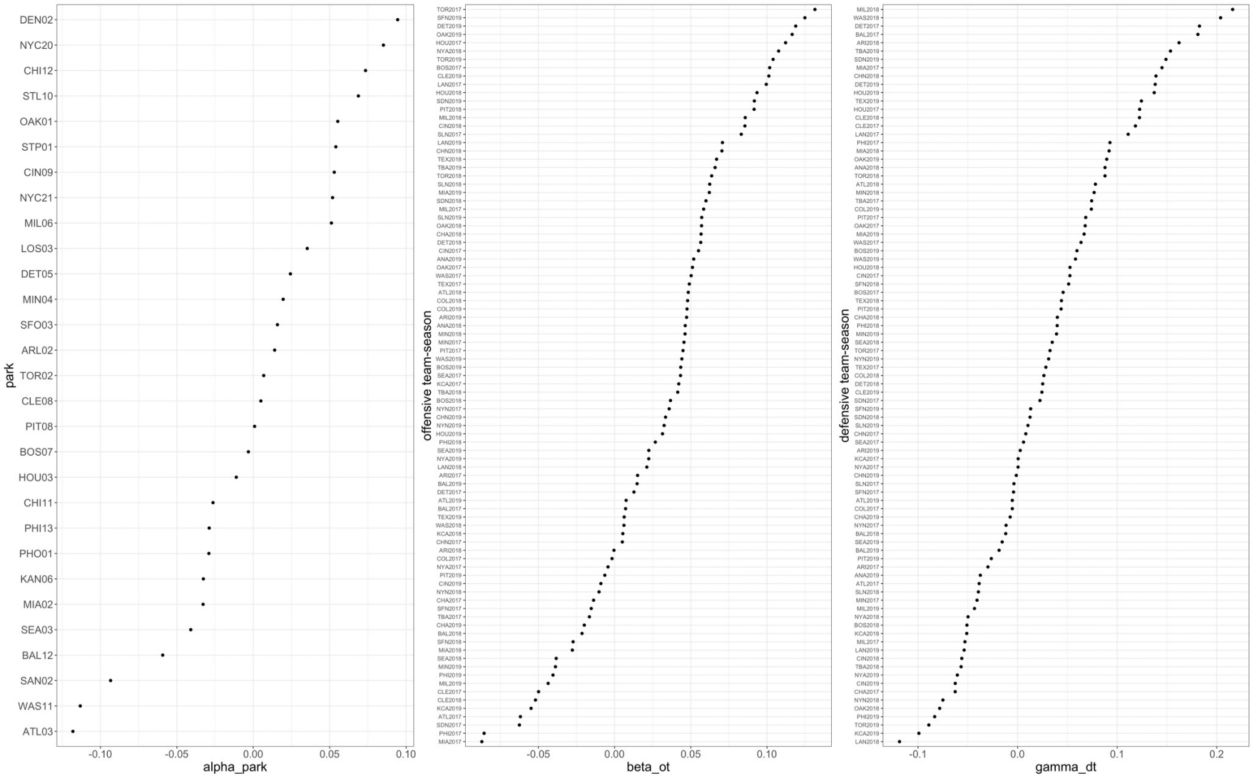

Figure 14.1: The true coefficients used in our simulation study.

Then, assuming our model is **true**, let's generate the response vector $\boldsymbol{y}$ (runs scored in a half-inning) $M = 5$ times according to

$$y_i = \text{Round}\left(\mathcal{N}_+(x_i^T\boldsymbol{\beta}, 1)\right)$$

where x_i^T is the i^{th} half-inning from our observed data matrix $\boldsymbol{X}$ of all 123,252 **real-life** half-innings from 2017-2019. Note the following:

- $\mathcal{N}_+$ is a Normal distribution truncated at 0
- We round to the nearest integer because runs scored in a half-inning are non-negative integers $\mathbb{Z} \geq 0$.
- $\mathbb{E}[y_i] \approx x_i^T\boldsymbol{\beta}$, or equivalently

  $$y_i \approx x_i^T\boldsymbol{\beta} + \epsilon_i, \quad \mathbb{E}[\epsilon_i] = 0$$

  So our original model assumption holds even if we don't explicitly model the noise ϵ_i.

14.2.2 Estimating the Coefficients

Now, let's use **linear regression** to estimate the coefficients $\boldsymbol{\beta}$ from each of our $M = 5$ simulated datasets $(\boldsymbol{X}, \boldsymbol{y})$ and see how well we recover the park effects! We know from Lecture 2 that the OLS estimator is

$$\hat{\boldsymbol{\beta}} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y}$$

And we implement this in `R` using the `lm()` function.

We plot our estimated park effect coefficients $\hat{\boldsymbol{\alpha}}$ in 14.2. What do we observe?

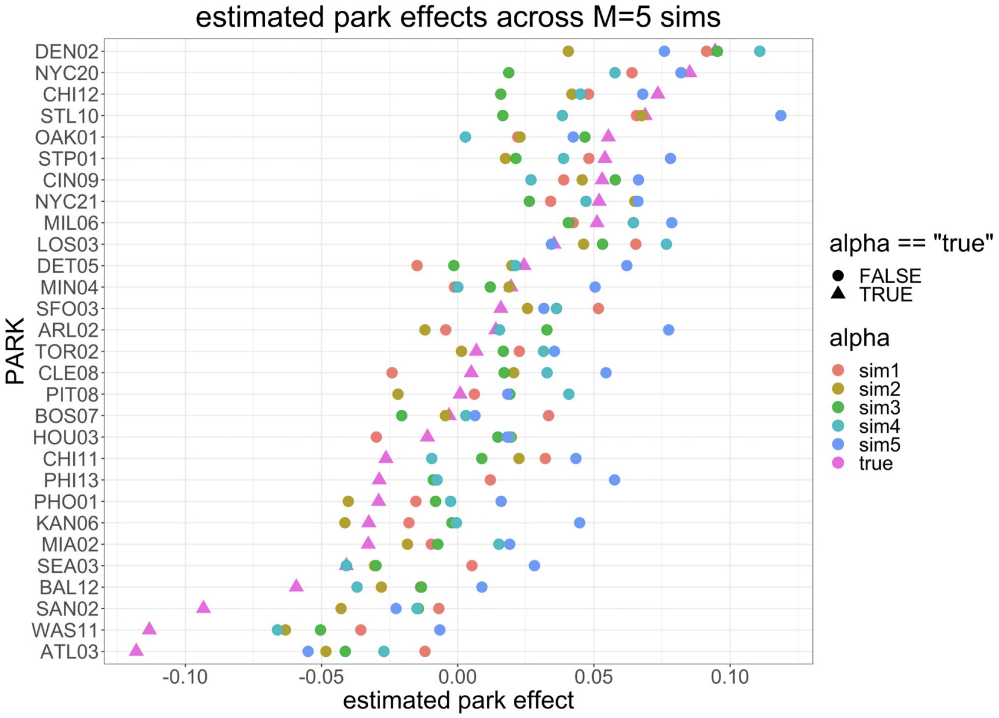

Figure 14.2: OLS estimates of park effect coefficients $\hat{\boldsymbol{\alpha}}$ from each of our $M = 5$ simulated datasets.

Due to **randomness** in the training data, each simulation yields **very different** park effect estimates $\hat{\boldsymbol{\alpha}}$, even though the "true" park effects $\boldsymbol{\alpha}$ are the same across the $M = 5$ simulations. This indicates that the OLS coefficients $\hat{\boldsymbol{\alpha}}^{(OLS)}$ are **unstable** and sensitive to the noise of the training set. How can we make the coefficients **less sensitive** to the random idiosyncracies of our data?

14.2.3 Looking for a Less-Sensitive Estimator

What's the least-sensitive estimator we could think of? That would be some constant value, like 0 or the overall mean park effect $\overline{\overline{\alpha}}$. These estimators may not be to sensitive to randomness in the training set, but they are **wrong** for many parks! We visualize this in 14.3.

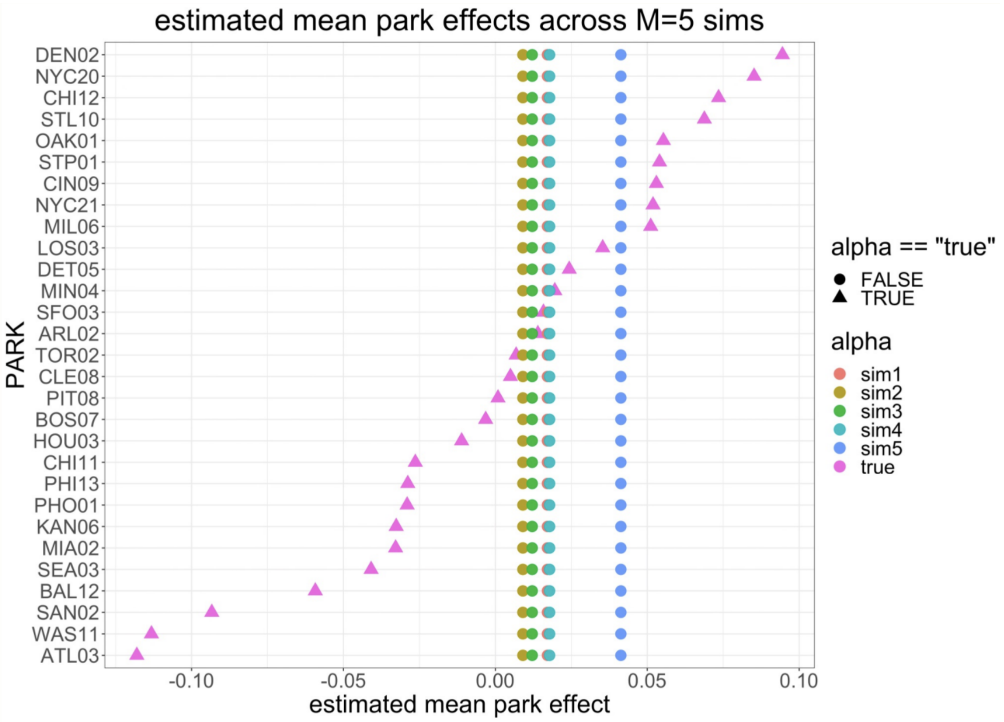

Figure 14.3: Constant park effect coefficients $\overline{\overline{\alpha}}$ from each of our $M = 5$ simulated datasets.

On the other hand, we have the OLS estimates $\hat{\boldsymbol{\alpha}}^{(OLS)}$, which are **unbiased** but **very sensitive** to randomness in the training set. There is a clear tradeoff between unbiasedness and sensitivity to randomness. How can we blend the strengths of both estimators? By **shrinking** the OLS estimates towards some constant value, like the overall mean park effect or 0!

14.2.4 Bayesian Shrinkage

So far, we are familiar with the **Bayesian** approach to shrinkage estimation. This involves shrinking our coefficients with a prior, such as:

$$\begin{aligned}
&\beta_j \sim \mathcal{N}(0, \sigma^2) \text{ or } \sim \mathcal{N}(\mu, \sigma^2)\ \forall\ j \\
&\alpha_j \sim \dots \\
&\quad \vdots \\
&\gamma_j \sim \dots \\
&\quad \vdots
\end{aligned}$$

However, this methodology has some drawbacks, chief among them being that it can be extremely computationally expensive to estimate the posterior distribution. We'd like a quicker (and frequentist) way to accomplish the same thing. How would we do that? **By altering the loss function to be optimized.**

14.3 Regularization

14.3.1 Ridge Regression

In ordinary linear regression, we estimated the coefficients $\boldsymbol{\beta}$ by minimizing the **Residual Sum of Squares (RSS)**:

$$\begin{aligned}
\hat{\boldsymbol{\beta}}^{(OLS)} &= \underset{\boldsymbol{\beta}}{\text{argmin}}\ \text{RSS}(\boldsymbol{\beta}) \\
&= \underset{\boldsymbol{\beta}}{\text{argmin}}\ \sum_{i=1}^{n} (y_i - x_i^T \boldsymbol{\beta})^2
\end{aligned}$$

In Ridge Regression , we alter the loss function to include a regularization term that encourages the estimated coefficients $\hat{\boldsymbol{\beta}}$ to be smaller (i.e., to lie closer to 0). We formally define it as follows:

Definition 14.1 (Ridge Regression)**.** *The Ridge Regression estimator is defined as the minimizer of the following loss function:*

$$\begin{aligned}
\hat{\boldsymbol{\beta}}^{(Ridge)} &= \underset{\boldsymbol{\beta}}{\arg\min}\ RSS(\boldsymbol{\beta}) + \lambda \sum_{j=1}^{p} \beta_j^2,\ \ \lambda > 0 \\
&= \underset{\boldsymbol{\beta}}{\arg\min}\ \sum_{i=1}^{n} (y_i - x_i^T \boldsymbol{\beta})^2 + \lambda \sum_{j=1}^{p} \beta_j^2,\ \ \lambda > 0 \\
&= \underset{\boldsymbol{\beta}}{\arg\min}\ (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})^T (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}) + \lambda \boldsymbol{\beta}^T \boldsymbol{\beta},\ \ \lambda > 0
\end{aligned}$$

where λ is a hyperparameter that controls the amount of shrinkage.

Note that the RSS term incentivizes $X_i^T \hat{\boldsymbol{\beta}}$ to be close to y_i, while the regularization term incentivizes $\hat{\boldsymbol{\beta}}$ to be close to 0 by **penalizing** large coefficients. The technique of adding a penalty term to the loss function is known as **Regularization**.

Note that the hyperparameter $\lambda > 0$ controls by how much we are penalized for having large coefficients β_j. This value is generally chosen by **cross-validation**, which involves evaluating the performance of our Ridge estimator on a held-out dataset and adjusting λ to minimize the MSE on the held-out dataset.

- Large λ: there is a significant penalty for having large coefficients β_j, which forces them to be closer to 0.
- $\lambda = 0$: there is no penalty for having large coefficients β_j, which is equivalent to OLS.

14.3.2 Solving for the Ridge Estimator

To solve for the Ridge estimator, we set the gradient of the loss function equal to 0 and solve for $\hat{\boldsymbol{\beta}}$. Our loss function is

$$\begin{aligned}\mathcal{L}(\boldsymbol{\beta}) &= (\boldsymbol{y} - \boldsymbol{X\beta})^T(\boldsymbol{y} - \boldsymbol{X\beta}) + \lambda\boldsymbol{\beta}^T\boldsymbol{\beta} \\ &= \boldsymbol{y}^T\boldsymbol{y} - 2\boldsymbol{\beta}^T\boldsymbol{X}^T\boldsymbol{y} + \boldsymbol{\beta}^T\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\beta} + \lambda\boldsymbol{\beta}^T\boldsymbol{\beta}\end{aligned}$$

Then taking the gradient with respect to $\boldsymbol{\beta}$ and setting it equal to 0, we get

$$\begin{aligned}\nabla\mathcal{L}(\boldsymbol{\beta}) &= -2\boldsymbol{X}^T\boldsymbol{y} + 2\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\beta} + 2\lambda\boldsymbol{\beta} = 0 \\ \implies &(\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I}_p)\boldsymbol{\beta} = \boldsymbol{X}^T\boldsymbol{y} \\ \implies &\hat{\boldsymbol{\beta}}^{\text{Ridge}} = (\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I}_p)^{-1}\boldsymbol{X}^T\boldsymbol{y}\end{aligned}$$

Let's make some notes abotu this estimate:

- A solution always exists when $\lambda > 0$ because $\boldsymbol{X}^T\boldsymbol{X}$ is a positive semi-definite matrix.
- When we add the $\lambda\boldsymbol{I}_p$ term to $\boldsymbol{X}^T\boldsymbol{X}$, we are essentially adding a "ridge" (or diagonal) of λ's: this is where Ridge Regression gets its name.
- $\boldsymbol{X}^T\boldsymbol{X}$ is like multiplying by $\frac{1}{\bullet}$, and $\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I}_p$ is like multiplying by $\frac{1}{\bullet} + \lambda$. Adding this $\lambda > 0$ term to the denominator **shrinks** the estimated coefficients $\hat{\boldsymbol{\beta}}$ towards 0.

In your homework, you will prove that the Bayesian regression model

$$\begin{cases} y_i \overset{i.i.d.}{\sim} \mathcal{N}(x_i^T\boldsymbol{\beta}, \sigma^2) \\ \beta_i \overset{i.i.d.}{\sim} \mathcal{N}(0, \frac{\sigma^2}{\lambda}),\ \ \lambda > 0 \end{cases}$$

has a maximum a posteriori (MAP) estimate (the Bayesian analog of the MLE), equal to the Ridge Estimator.

$$\begin{aligned}\widehat{\boldsymbol{\beta}}^{\text{MAP}} &= \arg\max_{\boldsymbol{\beta}} \mathbb{P}(\boldsymbol{\beta} \mid \boldsymbol{X}, \boldsymbol{y}) \\ &= \left(\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I}_p\right)^{-1}\boldsymbol{X}^T\boldsymbol{y} \\ &= \widehat{\boldsymbol{\beta}}^{\text{Ridge}}\end{aligned}$$

14.3.3 Back to Our Simulation Study

We fit the Ridge estimator to our simulation study and compare the results to the OLS estimates in Figure 14.4. These results are consistent with our theoretical results: the Ridge park effect estimates are indeed more stable than the OLS estimates across the $M = 5$ simulations. This indicates that Ridge Regression is less sensitive to the noise of the training set.

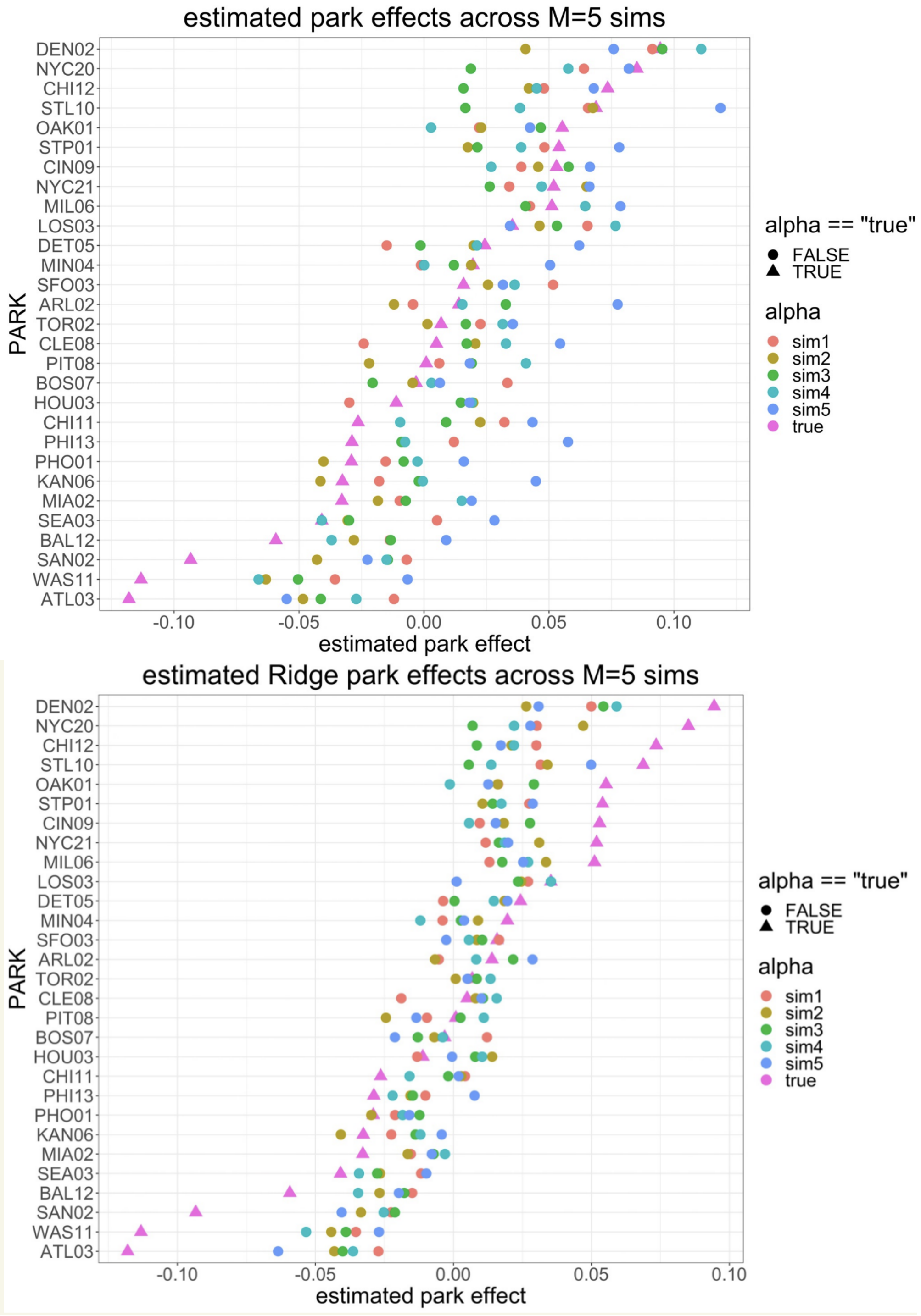

Figure 14.4: OLS estimates (top) and Ridge estimates (bottom) of park effect coefficients $\hat{\boldsymbol{\alpha}}$ from each of our $M = 5$ simulated datasets.

Note: Shrinking outliers isn't always a great idea! Due to its reduced sensitivity to the data, Ridge Regression is generally outperformed by OLS on outliers.

14.3.4 Back to the Real Data

Now that we have a better understanding of Ridge Regression, let's apply it to our real data to estimate park effects! We fit the Ridge estimator to our real data and compare the results to the OLS estimates in Figure 14.5. We see that Ridge Regression shrinks our park effects towards 0, which is consistent with our theoretical results.

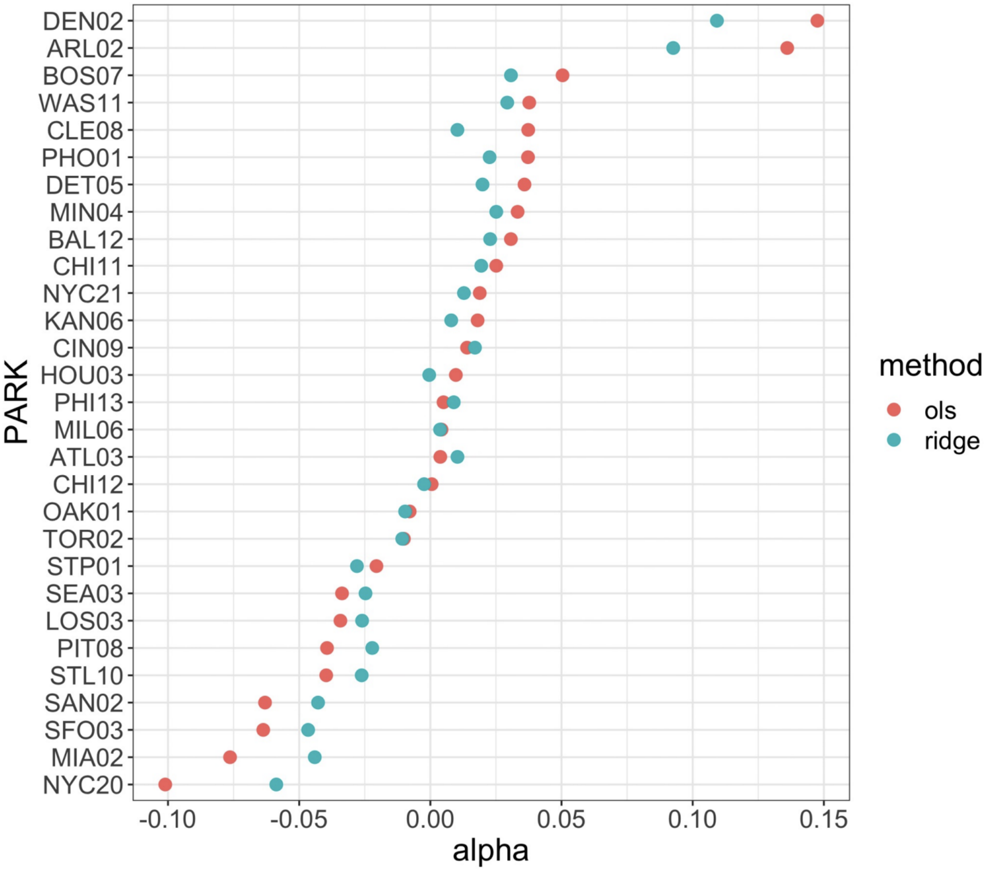

Figure 14.5: OLS and Ridge estimates of park effect coefficients $\hat{\boldsymbol{\alpha}}$ from our real data.

Note: It turns out that the shrunken Ridge park effects outperforms OLS **everywhere** out-of-sample. This is because OLS overfits to noise in the data!

References

[BW] Brill, R. S., & Wyner, A. J., *Introducing Grid WAR: Rethinking WAR for Starting Pitchers*, arXiv preprint, 2022.