Regularization & Ridge Regression

Q (Park Effects) Estimate the park effect α of each MLB ballpark, which represents the expected runs scored in one half-inning at that park above that of an average park, if an average offense faces an average defense.

→ Read my full analysis in Appendix of "Grid WAR" paper

training data all half innings from 2017–2019

Variables i indexes the i^{th} half inning in our dataset
park(i) is the ballpark of half-inning i
ot(i) is the offensive team-szn of half inning i
dt(i) is the defensive team-szn of half-inning i
y_i is the Runs scored in half-inning i

Model $y_i = \beta_0 + \alpha_{park(i)} + \beta_{ot(i)} + \gamma_{dt(i)} + \varepsilon_i$

where ε_i is mean zero noise, $\mathbb{E}\,\varepsilon_i = 0$

The park effects α and team quality coefficients β, γ are unknown parameters which need to be estimated from data.

Equivalently, $y_i = x_i^T \beta + \varepsilon_i$

where X is a matrix whose i^{th} Row is defined by

$$\begin{array}{rcccc} & \text{intercept} & \text{park 1} \;\; \text{park 2} \; \cdots \; \text{park 30} & \text{ot1} \;\; \text{ot2} \; \cdots \; \text{ot30} & \text{dt1} \;\; \text{dt2} \; \cdots \; \text{dt30} \\ x_i^T = \Big[& 1 & \underbrace{\bullet \quad \bullet \qquad\quad \bullet}_{\substack{= 1 \text{ at } \mathrm{Park}(i) \\ 0 \text{ else}}} & \underbrace{\bullet \quad \bullet \qquad \bullet}_{\substack{1 \text{ at } \mathrm{ot}(i) \\ 0 \text{ else}}} & \underbrace{\bullet \quad \bullet \qquad \bullet}_{\substack{1 \text{ at } \mathrm{dt}(i) \\ 0 \text{ else}}} \Big] \end{array}$$

$$\beta^T = [\beta_0 \; \alpha_1 \cdots \alpha_{30} \; \beta_1 \cdots \beta_{30} \; \gamma_1 \cdots \gamma_{30}]$$

Problem: Multicollinearity

When home team is on offense, $\mathrm{park}(i) = \mathrm{ot}(i)$.
When road team is on offense, $\mathrm{park}(i) = \mathrm{dt}(i)$.
So, it is tough to disentangle $\alpha_{\mathrm{park}(i)}$ from $\beta_{\mathrm{ot}(i)}$ and $\gamma_{\mathrm{dt}(i)}$.

Are the Runs scored in those half-innings 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 offence to figure out $\beta_{ot(i)}$ well, and a huge number of instances of Home teams on offence to figure out $\gamma_{dt(i)}$ well. Then, with β_{ot} and γ_{dt} good, we can figure disentangle α_{park}.

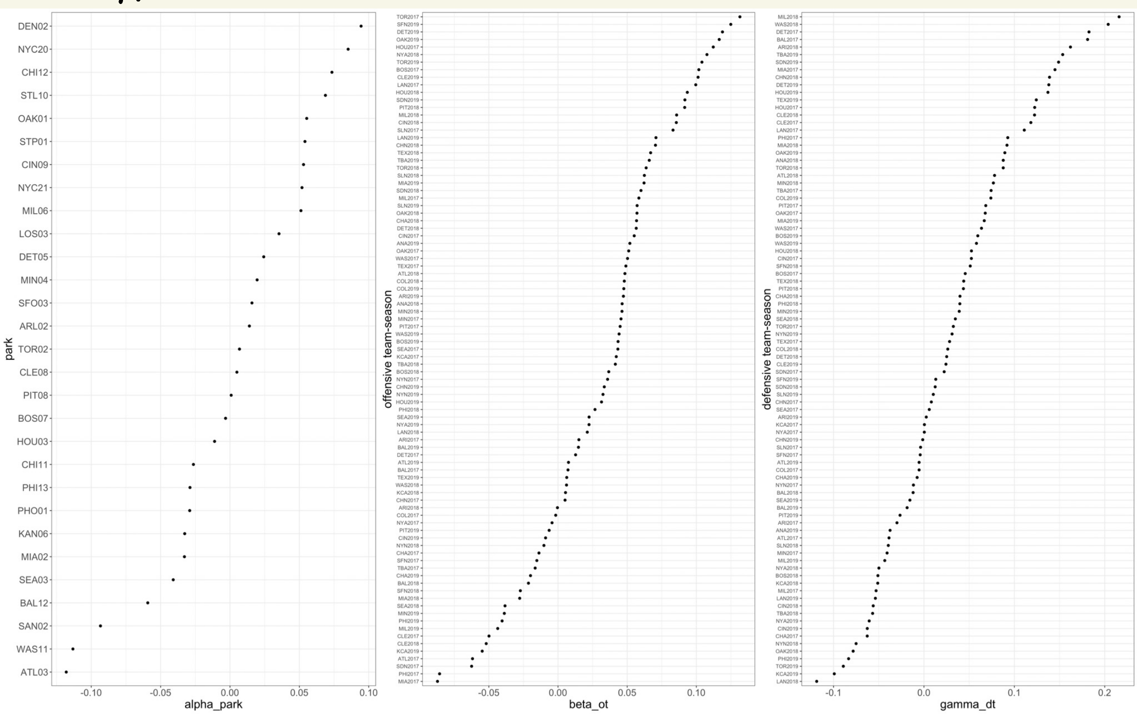
Our current dataset consists of 123,252 half-innings. This may seem like a lot of data, but due to our multicollinearity issue this actually isn't a huge amount of data. It's easy to overfit to noise and get the coefficients wrong. To demonstrate, we run a simulation study.

Q How much does multicollinearity affect our park effect estimates?
How well does OLS recover the park effects?

Simulation Study

Idea Pretend we knew the true coefficients α, β, γ, generate fake historical data y (121,000 fake inning outcomes), and see how well we estimate the coefficients from this synthetic data.

* Suppose the "true" coefficients are



which are chosen to have a "reasonable" scale.

* Then, assuming our model is <u>true</u>, let's generate the response y vector (runs scored in a half inning) M times according to

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

where x_i^T is the i^{th} half inning from our observed data matrix of all 123,252 half-innings from 2017 to 2019.

example snippet of simulated y

```
      [,1]
 [1,]    1
 [2,]    1
 [3,]    1
 [4,]    2
 [5,]    1
 [6,]    2
 [7,]    1
 [8,]    0
 [9,]    1
[10,]    1
```

Here,

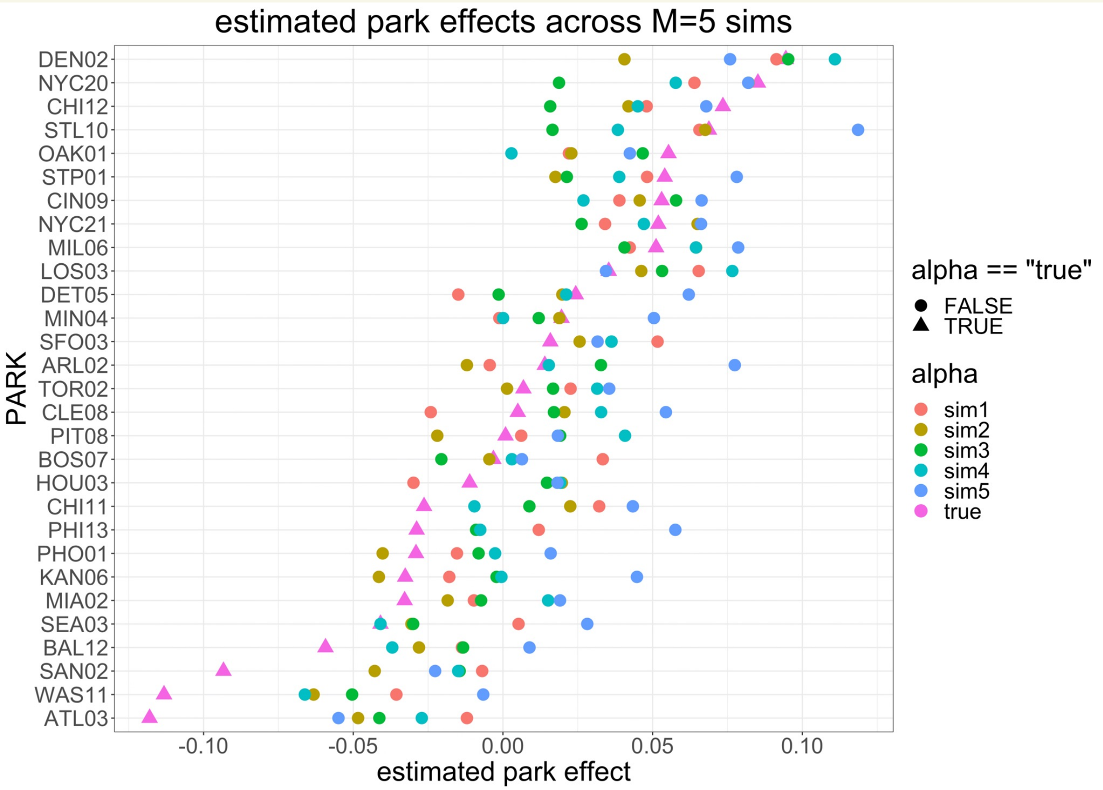
- $\mathcal{N}_+$ means normal dist. conditional on it being ≥ 0
- "Round" because runs scored is an integer ≥ 0
- $\mathbb{E}y_i \approx x_i^T\beta$

  equivalently, $y_i \approx x_i^T\beta + \varepsilon_i$, $\mathbb{E}\varepsilon_i = 0$

  so our original model assumption holds true even if we don't explicitly write ε_i here

* Then, let's use <u>linear regression</u> to estimate the coefficients $\hat{\beta}$ on each of our M simulated datasets (X, y) and see how well we Recover the park effects! $\hat{\beta} = (X^T X)^{-1} X^T y$.

We can do this because it's a Simulation and we know the "true" park effects.



* Due to <u>Randomness</u> in the training dataset, from the noise in generating y, each simulation yields very <u>different</u> park effects estimates $\hat{\alpha}$, even though the "true" park effects are the same.

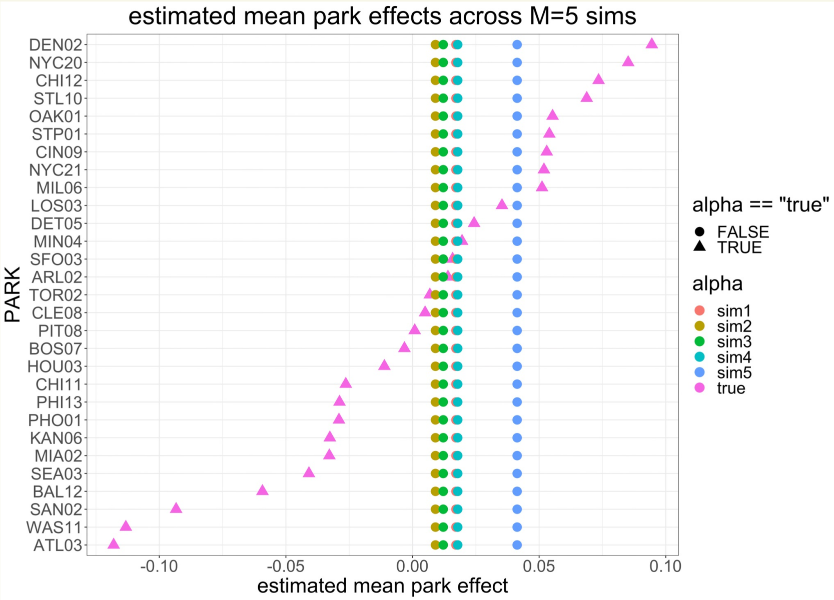
* The OLS (ordinary least squares = ordinary linear regression) coefficients $\hat{\alpha}^{(OLS)}$ change quite significantly across different simulations; they are quite sensitive to the noise of the training set

* How can we make the coefficients <u>less sensitive</u> to the random idiosyncracies of our training set?

Q What's the least sensitive estimator you can think of?

overall mean $\hat{\bar{\alpha}}$ estimated mean park effect

zero the constant value 0



* Constant values like zero, or overall mean — not too sensitive to the random idiosyncracies of the training set, but are wrong for many parks

* OLS park effect estimates — very sensitive to the randomness of the training set, but are unbiased (on average, i.e. averaged over many training set generations, they are in the right spot)

* there is a tradeoff between sensitivity and unbiasedness

Q How can we blend the strengths of OLS with the strengths of the overall mean?

Idea **Shrink** the OLS estimates towards a constant value, like the overall mean or zero.

* The Bayesian way to shrink the coefficients is via a prior:

$$\begin{cases} \beta_j \sim N(\mu, \sigma^2) & \forall j \\ \text{or} \quad \beta_j \sim N(0, \sigma^2) & \forall j \\ \text{same for } \alpha_j, \gamma_j & \end{cases}$$

* The drawback of fitting bayesian models is that it can be extremely computationally intensive...

* Is there a quicker (and frequentist) way of accomplishing this?

Idea: Make the coefficients less sensitive to noise by shrinking the OLS estimates towards zero by altering the loss function to be optimized.

* In ordinary linear regression, we estimate the coefficients β by minimizing the Residual sum of squares,

$$\hat{\beta}^{(OLS)} = \underset{\beta}{\operatorname{argmin}} \; RSS(\beta)$$

$$= \underset{\beta}{\operatorname{argmin}} \sum_{i=1}^{n} (y_i - x_i^T \beta)^2$$

* In Ridge Regression we instead minimize the RSS with a penalty term that encourages the estimated coefficients $\hat{\beta}$ to be smaller (i.e., to lie closer to 0),

$\lambda > 0$ is a hyperparameter

$$\hat{\beta}^{(Ridge)} = \underset{\beta}{\operatorname{argmin}} \underbrace{\sum_{i=1}^{n} (y_i - x_i^T \beta)^2}_{\text{want } x_i^T\hat{\beta} \text{ to be close to } y} + \underbrace{\lambda \sum_{j} \beta_j^2}_{\text{want } \hat{\beta} \text{ to be smaller, or closer to } 0}$$

* This technique of adding a penalty term to the loss function we are minimizing is called Regularization.

The hyperparameter $\lambda > 0$ describes by how much we are penalized for having large β_j.

λ is simply a number, which is tuned using cross-validation.

Large λ → large penalty for large β_j
→ forces β_j to be smaller.

$\lambda = 0$ → equivalent to OLS
→ no shrinkage of β.

$$\hat{\beta}^{(Ridge)} = \underset{\beta}{\operatorname{argmin}} \sum_{i=1}^{n} (y_i - X_i^T \beta)^2 + \lambda \sum_j \beta_j^2$$

$$= \underset{\beta}{\operatorname{argmin}} (y - X\beta)^T (y - X\beta) + \lambda \beta^T \beta$$

in matrix notation.

Calulus: Set gradient equal to 0 and solve!

$$L(\beta) = (y - X\beta)^T (y - X\beta) + \lambda \beta^T \beta$$

$$= y^T y - 2\beta^T X^T y + \beta^T X^T X \beta + \lambda \beta^T \beta$$

$$\nabla_\beta L(\beta) = -2X^T y - 2X^T X \beta + 2\lambda\beta = 0$$

$$\Longrightarrow (X^T X + \lambda I)\beta = X^T y$$

$$\Longrightarrow \boxed{\hat{\beta}^{(ridge)} = (X^T X + \lambda I)^{-1} X^T y}$$

Solution always exists when $\lambda > 0$.

Ridge Regression — add matrix

$$\lambda I = \begin{pmatrix} \lambda & & 0 \\ & \lambda & \\ 0 & & \ddots & \lambda \end{pmatrix} \text{ to } X^T X$$

prior to inverting. This is a "ridge" of λ's.

$(X^T X + \lambda I)^{-1}$ is like multiplying by $\frac{1}{\bullet + \lambda}$,

$(X^T X)^{-1}$ is like multiplying by $\frac{1}{\bullet}$

adding $\lambda > 0$ to the denominator

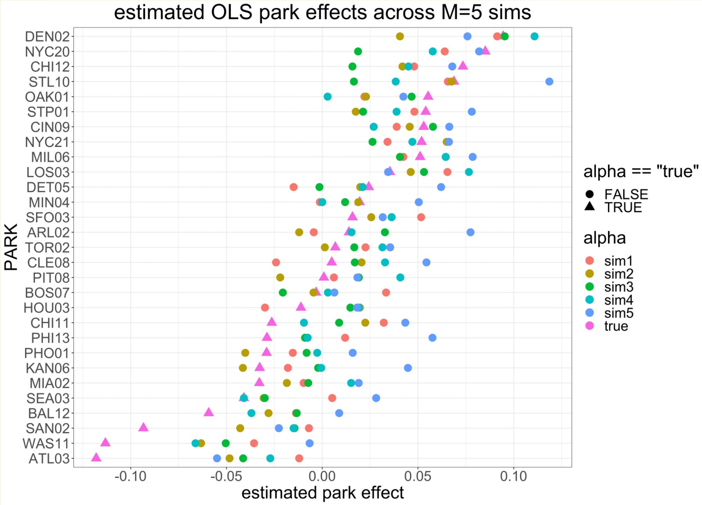
<u>shrinks</u> the estimates $\hat{\beta}_\bullet$!

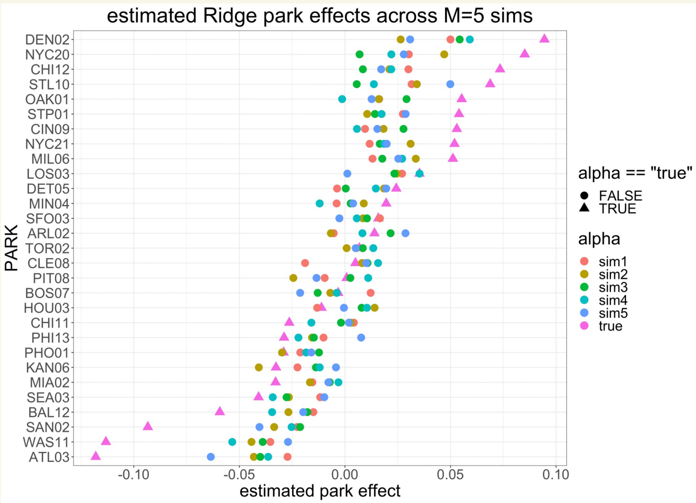
* Math HW: the Bayesian Regression Model $\begin{cases} y_i \overset{iid}{\sim} N(x_i^T \beta, \sigma^2) \\ \beta_i \overset{iid}{\sim} N(0, \frac{\sigma^2}{\lambda}) \end{cases}$

has maximum a posteriori (MAP) estimate, which is like a Bayesian version of MLE, equal to

$$\hat{\beta} = \underset{\beta}{\operatorname{argmax}}\ \mathbb{P}(\beta \mid X, y) = (X^T X + \lambda I)^{-1} X^T y$$

and so is equivalent to Ridge Regression.



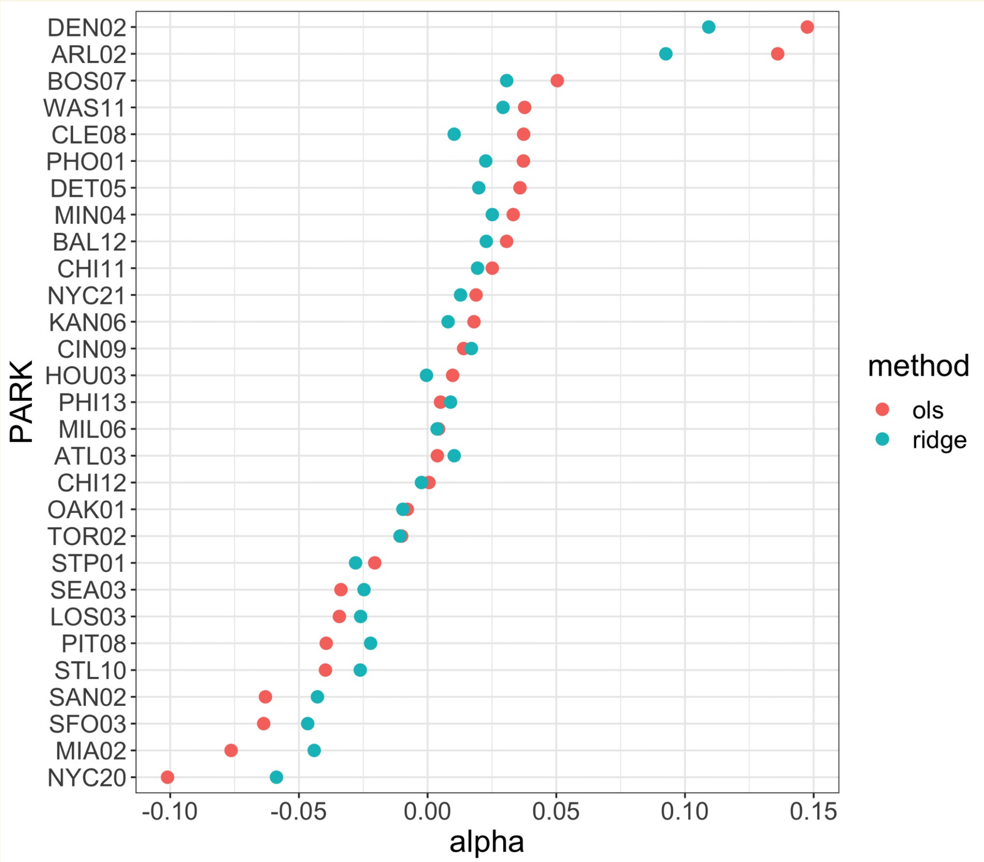


* Ridge regression park effect estimates indeed are more stable across simulations, i.e. are less sensitive to the noise of the training set!

```
> ### error
> err(beta.pk.df.sim)
[1] 0.03528335
> err(beta.pk.df.sim_ridge)
[1] 0.03804942
> ### error on non-outliers
> err(beta.pk.df.sim %>% filter( abs(beta.pk.true) < 0.05) )
[1] 0.02533202
> err(beta.pk.df.sim_ridge %>% filter( abs(beta.pk.true) < 0.05) )
[1] 0.01690153
> ### error on outliers
> err(beta.pk.df.sim %>% filter( abs(beta.pk.true) >= 0.05) )
[1] 0.04406852
> err(beta.pk.df.sim_ridge %>% filter( abs(beta.pk.true) >= 0.05) )
[1] 0.05359246
```

* Shrinking outliers isn't always a great idea; OLS outperforms on outliers

* Park effects on Real MLB data, 2017–2019



* We see that Ridge indeed shrinks the park effects towards zero!

* On this Real data, it turns out that the Ridge shrunken park effects are everywhere better than OLS since OLS overfits...
(based on out-of-sample predictive performance)