Building an expected runs model to perform lineup analysis for Major League Baseball rosters

Technical Report

By: Cole Dillaplain, Patrick Ffrench, Tyler Gorecki, and Jack Penney

Table of Contents

1. Introduction

In the game of baseball, the team's manager makes a decision that will impact the whole game before the players even set foot on the field - creating the batting order. Once the game begins, the batting order is fixed and can only be slightly modified by substitutions. This is a key decision in baseball that needs to be heavily examined through data analytics.

To determine which lineup composition and order is best, we set out to create an expected runs simulation. Typically, an expected runs simulation is a Monte-Carlo simulation of a single half-inning to determine the long-run average number of runs a team would earn in an inning given a situation of batters due up, baserunners, and outs. However, this would need to be heavily modified for the batting order use case which impacts the entire nine or more innings of a game.

2. The Data

The data collection process was rather straightforward. To make this Monte-Carlo simulation, pitch-by-pitch data was collected across the entirety of the 2022 Major League Baseball season. The data was gathered using a function within the *baseballr* package to get the data from the MLB Stats API.

In the 2022 season, 30 teams played a total of 162 games each, totaling 2430 games worth of data. The large dataset allowed for rare and unique situations to appear, which later would be needed to build accurate conditional distributions on the varying counts, number of outs, hitter types, and baserunner locations. Furthermore, baseball is a game with many niche outcomes and rules so a large dataset was important for finding these cases and adjusting probabilities accordingly. Finally, using a whole season made the data less influenced by factors like a team's record, early vs late season hot/cold streaks, and weather, which are all hypothesized to make an impact on performance.

3. Classifying Hitter Types

We decided to break up hitters into categories for lineup analysis. This part of the project was a large discussion point for us, due to the significant effect these categories would have on the results of the project. With how much conditioning we were doing for each sample portion of the model, we didn't want to make too many hitter groups and overcomplicate our model, while

hurting its efficiency as well. Therefore, we settled on three groups based on contact percentage. If a hitter had a contact percentage in the 2022 season of at least 78%, then they were labeled a contact hitter. If their contact percentage was below 70%, then they were labeled a power hitter. The rest in between were tagged as middle hitters. We felt like this was the simplest way to split the hitters while not having one hitter group just be better overall than the others. If we had clustered based on hitting analytics like on base percentage or max exit velocity, we would have had hitter groups based on talent which was the main thing we were trying to avoid. This might work for another situation, but MLB teams don't have the luxury of just getting the best hitters they can - instead, they have their best nine and are looking to get the most production out of them by creating their strongest lineup. If we were doing this for a specific organization, many of the hitters would have enough data from years past to just sample using those numbers. For the rest (most likely younger hitters), we would consult with coaches and physical metrics to have a couple of hitters to model after and perform the same analysis. But, for ours, we used the three main hitter types which still show the effectiveness of this modeling process.

Good examples of traditional contact hitters in the contact-specific group include Stephen Kwan, Jeff McNeil, and Whit Merrifield, but there are also some bad examples like Daniel Vogelbach who is not known for being that type of hitter. Juan Soto is also included as he is more of just a strong bat-to-ball hitter than someone who should be linked with the previous names mentioned. On the other side, true power hitters like Eloy Jimenez and Joey Gallo are countered with a rare bad example like Paul Dejong who just didn't hit well during the 2022 season, including the highest strikeout rate of his career (which explains the low contact rate). Because of this, it was hard to balance who truly belongs in each group and who just performed better or worse that year leading to a higher or lower contact rate.

This summarization of statistics below is the beginning of the basic analysis we can do to prove our case for why these hitter groups are a reasonable grouping of MLB batters for our analysis. It can be seen that the batting average on balls in play (babip) is higher for our power hitter type, with the middle and contact groups following. While it might be hard to generalize completely, power hitters overall typically make contact on fewer swings but are expected to do more damage when they do hit the ball, so this trend isn't surprising to see from our data. It could be more helpful to look at specific hitters from our groupings as well.

groups	babip
contact	0.2889918
middle	0.2955487
power	0.2982194

4. Conditional Distributions

4.1. Pitch Result

In our expected runs simulation, we sample an outcome for each pitch of the simulated inning. This meant conditioning the result based on the count (number of balls and strikes) and the hitter type (one of the three contact, middle, or power groups as previously described). From these, there are 36 possible combinations, which are used to sample the five possible different pitch outcomes. These outcomes include in-play, ball, strike, foul, and hit-by-pitch. If the outcome is in-play, then we go on to sample hit results in a similar fashion, which is explained more in the next section. The other outcomes either adjust the count or call on other functions if a walk, strikeout, or hit-by-pitch ends the count, either completing the inning or adjusting the overall situation at hand. The following is a sample from our pitch result table used to perform the explained sampling:

access	InPlay	Ball	Strike	Foul	HitByPitch
contact_0_0	0.1102094	0.3945797	0.3882679	0.1049184	0.0020245
contact_0_1	0.1959471	0.4025615	0.2105499	0.1877654	0.0031760
contact_0_2	0.2173646	0.4287861	0.1359382	0.2130610	0.0048501

This portion of the data interestingly provides an accurate distribution of what you would expect in a contact hitter at the plate. Early in the count, they are less likely to put the ball in-play because they typically work the count a little more, and the strikeout rate is very low in 0-2 counts, with them putting the ball in-play, fouling it off, or taking a ball a large majority of the time. These were small things we kept an eye on throughout the process to make sure our estimations made sense from a baseball-minded perspective as well.

4.2. Hit Result

The in-play hit result function is a very similar sampling technique as was used in the pitch result simulation. It again conditions on the count and the hitter type, but this time samples from a distribution containing nine possible outcomes. The possible outcomes are single, double, triple, homerun, error, popout, flyout, lineout, and groundout. For each of these outcomes, we built separate functions to take into account how baserunners will advance, including possibly taking extra bases. We will go further into how we advance baserunners in section 5.3.

access	singles	doubles	triples	hrs	error	popout	flyout	lineout	groundout
contact_0_0	0.2127200	0.0673047	0.0049398	0.0361223	0.0074097	0.0713183	0.2037666	0.0836678	0.3127508
contact_0_1	0.2128259	0.0574348	0.0035236	0.0251938	0.0079281	0.0658915	0.1911557	0.0865046	0.3495419
contact_0_2	0.2313011	0.0493400	0.0028284	0.0153991	0.0075424	0.0597109	0.1744186	0.0983658	0.3610937

5. Building the Simulation

5.1. Master Functions

To determine the best lineup composition or lineup order, it is important to simulate full nine-inning games because the batters due up each inning are continuously changing based on the results of the previous at-bats. Therefore, our simulation is built for complete nine-inning games but can be adjusted depending on motivation.

The expected runs simulation determines the result of each pitch. Three major functions are used to structure the code to run over a full game, Sim Outcome and Sim Inning..

5.1.1. Sim Outcome

The Sim Outcome function determines what happens on each pitch. There are five possible outcomes. These are ball, strike, foul ball, in-play, or hit-by-pitch. The probability of each of these results changes based on the hitter type as well as the number of balls and strikes because a hitter's strategy changes significantly with the count. The proper conditional distribution is accessed from these variables, then the result is sampled from these probabilities, and then it is returned as a string.

5.1.2. In-Play

Similarly to the Sim Outcome function, the In-Play function accesses a conditional distribution of hit results based on the number of balls, the number of strikes, and the hitter type. This function is called directly after the Sim Outcome function if the pitch is sampled to be hit in play. The possible results are types of hits, types of outs, and errors. Hit types can be singles, doubles, triples, and home runs. Out types include groundouts, popouts, flyouts, and lineouts. These results also return as a string.

5.1.3. Sim Inning

The Sim Inning function simulates one half-inning of baseball using the prior two functions. First, it updates which hitter in the lineup is up to bat if necessary. The index of the at-bat number is used to access the hitter type from an array of the hitter types in lineup order. Then, the Sim Outcome function is run using the proper conditional distribution. Each outcome is sent into a minor function that updates the baserunners, outs, count and runs scored. This function is recursively called updating the number of runs scored until there are three outs recorded.

5.2. Minor Functions

We created individual functions for each of the simulated in-play outcomes: single, double, triple, homerun, error, popout, flyout, lineout, and groundout. These were created separately because of unique situations depending on the runners on base and how they would advance, the possibility of double plays and fielders' choices, and various other considerations outside of strictly moving baserunners generically (or not moving them if there was an out).

To go in-depth on one specifically, an explanation of our doubles function can be used to give insight into how we built these functions.

```
double <- function() {
  #print(baserunners)
  # 35% firstscores, 60% base, 5% out
  runners <- sample(c('first', 'base', 'out'), size = 1, prob = c(.35,.6,.05))
  if (baserunners[3] == 1) {
    runs <<- runs + 1
    baserunners[3] <<- 0
  }
  if (baserunners[2] == 1) {
    runs <<- runs + 1
    baserunners[2] <<- 0
  }
  if (baserunners[1] == 1) {
    if (runners == 'first') {
      runs <<- runs + 1
      baserunners[1] <<- 0
    } else if (runners == 'base') {
      baserunners[3] <<- 1
      baserunners[1] <<- 0
    } else if (runners == 'out') {
      outs <<- outs + 1
      baserunners[1] <<- 0
    }
  }
  baserunners[2] <<- 1
  #print(runners)
  #print(runs)
  #print(baserunners)
}
```

This function is only called when the Sim Outcome function returns 'inplay' and the In-Play function returns 'double' for reference. Regardless of which bases are occupied, we first define what we believe are the probabilities that a runner on first will score, the baserunners will advance only two bases, or a runner on first will be out at home. From this point, we proceed into the if-else statement where we first check whether there was a runner on third. If there is, we score him and clear third base. We do the same with the runner on second because he should always be able to score on a double.

Finally, we check if there is a runner on first base. If there is, we use the sampling output performed earlier in the function to simulate what happens. If it returns ‘first’ (meaning the guy on first scores), we add a run. If it returns ‘base’ (meaning he advances the normal two bases), we send the runner to third. Finally, if it returns ‘out’ (meaning the guy on first was thrown out at home), we add an out and remove that runner from the basepath. At the end of the function, we also make sure to have second base occupied. This process was tweaked for all types of hits like singles, triples, and home runs, but it can also be applied to outs, such as groundouts and flyouts where runners may advance as well.

5.3. Advance Baserunners

For most of these situations, there are instances where the players on base will move more than just the same number of bases as the hitter. For example, on singles, it is pretty common for the runner on second base to score, so we needed a way to factor this in. We said there is a 55% chance of a runner scoring from second base, with other probabilities split between runners being thrown out or just not trying to advance extra bases at all. In each of the individual in-play functions mentioned above, we determined probabilities that we believed were accurate representations of the distribution of runners advancing and/or being thrown out. These distributions were self-created based on what we would expect from our baseball knowledge background and to make the expected runs function more realistic (our expected runs were well lower than traditionally expected before adding in the ability for runners to advance extra bases). We also added in the possibility of fielder’s choice, sacrifice flies, and different types of errors, to mention a few other modifications.

While we did create these generic probabilities on our own to be able to tinker with the model and make small adjustments to improve it, this also could have been something we gathered using the full pitch-by-pitch data. We believe this may have been tedious compared to what we wanted to look at in terms of focusing on the modeling process and analysis portions, but this is something that could be used to further tweak the model to a better representation of MLB run scoring in the future.

6. Lineups

6.1. Common Lineup Compositions

We completed some exploratory analysis to determine which lineups to look closer at. First, we had to find out what the team's starting lineups were. We merged hitter-type data onto our pitch-by-pitch data to determine what type of hitter was up to bat. Then, we subset to the first

three innings and only the first pitch of the at-bat to see which hitters came up. We ran a for loop that went game by game taking the first nine unique hitters of the home team and the away team. We saved the lineup of hitter types and grouped them. Below are the top results of batting order compositions.

order	n
contact, contact, contact, middle, contact, contact, middle, power, power	27
contact, contact, contact, contact, middle, middle, contact, middle, contact	26
contact, middle, middle, middle, middle, middle, middle, middle, middle	23
middle, power, middle, middle, middle, middle, middle, middle, middle	20
middle, middle, power, middle, contact, power, power, contact, middle	19
middle, middle, power, contact, middle, middle, middle, middle, middle	18
middle, power, middle, middle, middle, power, middle, middle, middle	18
middle, middle, middle, middle, contact, middle, middle, middle, middle	16
middle, power, middle, middle, middle, middle, power, middle, middle	16
middle, contact, contact, middle, middle, middle, middle, power, power	15

With over 4,000 starting lineups in a year, the most common lineup order only having 27 appearances was unexpectedly low. This means that we cannot run an analysis of common lineup orders, because there seems to be little commonality across the season.

Because of these results, we decided to look at the composition of hitter types in these lineups rather than the order itself. We took a count of contact, middle, and power hitters and ran a group by function to find new counts. Below is the result in the order of contact, middle, and power respectively.

counter	n
3 4 2	385
4 4 1	379
2 4 3	343
2 5 2	341
3 5 1	338
4 3 2	301
2 6 1	254
3 3 3	251
5 3 1	185
2 3 4	162

We decided to look at some of these common lineup compositions - the types of hitters that make up a team - and look at all possible combinations of each to order them for our analysis.

6.2. Lineup Permutations

We created the list of all possible permutations for each common combination of hitter types. We then saved each list of permutations as "lineup(cmp)," where c is the number of contact hitters in the combination, m is the number of middle hitters, and p is the number of power hitters. For example, we ran the below code to create the list of all permutations for lineups with three contact hitters, four middle hitters, and two power hitters. We ran this code for each combination.

```
counts <- c(3, 4, 2)
all_permutations <- permn(rep(items, times = counts))
unique_permutations <- unique(sapply(all_permutations, function(x) paste(x, collapse = ",")))
lineups342 <- lapply(unique_permutations, function(lineup) unlist(strsplit(lineup, ",")))
```

7. Running Simulations

We ran the simulation 1,000 times for every lineup permutation within each combination of interest. We ran it separately for each combination, saving a full data frame and a summary data frame. For each game simulated, we stored the within-lineup iteration number, the total runs scored, the runs scored per inning, and the leadoff hitter of each inning in the full data frame. We decided to store the leadoff hitter for each inning because it would allow us to determine the full list of batters for any inning and perform analysis on the lists. For example, we could determine the ideal "due up" batters for an inning given the current game conditions. In the summary data frame, we stored the mean runs per inning for each lineup type.

For example, below are the first ten rows of the full data frame for lineups with two contact hitters, five middle hitters, and two power hitters. The runs scored per inning are denoted by r1 through r9 and the leadoff hitter for each inning is denoted by h1 through h10. While we did not simulate ten innings, h10 marks the hitter who would have been the leadoff hitter in a tenth inning. This is important to understand how many hitters, and which hitters, batted in the ninth inning.

lineup	simNum	totalRuns	r1	r2	r3	r4	r5	r6	r7	r8	r9	h1	h2	h3	h4	h5	h6	h7	h8	h9	h10
contact,contact,middle,middle,middle,middle,middle,power,power	1	8	0	1	0	3	0	1	3	0	0	1	4	2	6	3	7	2	9	3	7
contact,contact,middle,middle,middle,middle,middle,power,power	2	0	0	0	0	0	0	0	0	0	0	1	6	1	5	9	3	6	9	3	7
contact,contact,middle,middle,middle,middle,middle,power,power	3	4	0	0	2	0	0	2	0	0	0	1	4	8	5	9	3	2	5	1	5
contact,contact,middle,middle,middle,middle,middle,power,power	4	0	0	0	0	0	0	0	0	0	0	1	6	9	4	8	4	7	1	4	8
contact,contact,middle,middle,middle,middle,middle,power,power	5	0	0	0	0	0	0	0	0	0	0	1	4	7	1	5	8	4	9	4	8
contact,contact,middle,middle,middle,middle,middle,power,power	6	2	2	0	0	0	0	0	0	0	0	1	6	2	5	8	2	6	1	4	10
contact,contact,middle,middle,middle,middle,middle,power,power	7	6	0	0	0	2	0	0	0	0	4	1	4	7	2	8	2	7	1	5	4
contact,contact,middle,middle,middle,middle,middle,power,power	8	0	0	0	0	0	0	0	0	0	0	1	5	8	2	5	8	2	6	9	5
contact,contact,middle,middle,middle,middle,middle,power,power	9	3	0	0	0	3	0	0	0	0	0	1	4	8	4	5	8	2	5	9	4
contact,contact,middle,middle,middle,middle,middle,power,power	10	3	1	0	0	0	1	1	0	0	0	1	6	9	5	9	5	1	4	7	5

8. Comparison to Existing Expected Runs Function

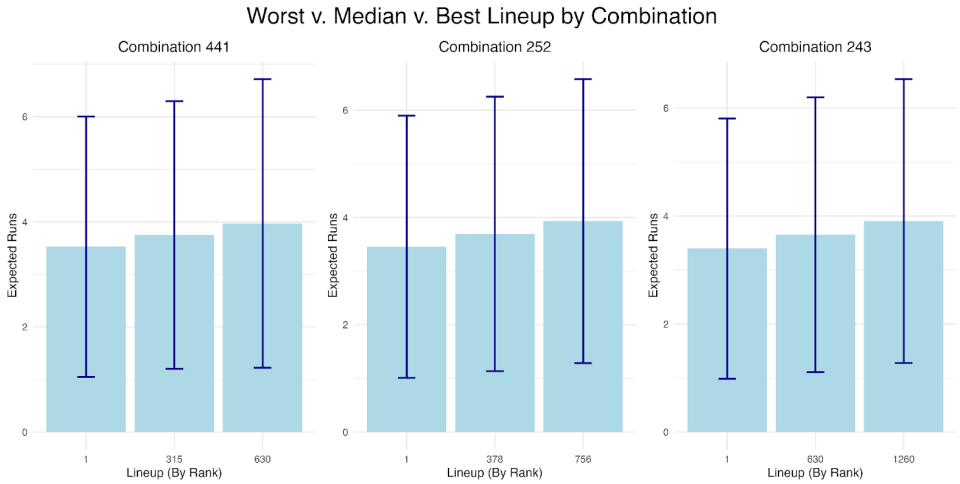
As a sort of proof of concept that our model would be useful for future analysis, we decided to compare the expected runs output to an existing model developed by the well-known baseball analytics company, Fangraphs. Something we struggled with, which can be seen in the table below, was our consistent underestimation of runs scored. We believed that even if we slightly underestimated the magnitude of runs scored, we could still derive meaning from the model if it had a similar trend of runs scored based on the situation. The columns that are labeled 'multiple of empty bases' are calculated by dividing the expected runs value for each baserunner combination by the expected runs output with the bases empty. This would give us a relative measure to compare with the Fangraphs model's output.

Baserunners	*Fangraphs runs*	*Fangraphs multiple of empty bases*	*Our model*	*Our model multiple of empty bases*
Empty	0.461	1	0.381	1
1 _ _	0.831	1.80	0.697	1.83
_ 2 _	1.068	2.32	0.829	2.18
1 2 _	1.373	2.98	1.147	3.01
_ _ 3	1.426	3.09	0.923	2.42
1 _ 3	1.798	3.90	1.216	3.19
_ 2 3	1.920	4.16	1.371	3.60
1 2 3	2.282	4.95	1.764	4.63

There are some positives and some negatives to this comparison. First, we were encouraged that the results seemed to indicate our model was somewhat accurate, given that the expected runs scored followed the general trend of the Fangraphs model with the number of runs scored increasing as we moved down the table, except for the situation starting with a runner on third only. All of the 'multiple of empty bases' values prior to adding a man on third were nearly identical to the Fangraphs values. When we did add a runner on third, however, we did begin to underestimate more. We could not seem to figure out what specifically was causing this, as runners could score from third on all hits and also on some groundouts and flyouts. We had considered adding passed balls and wild pitches to help with this but it would not have made up for the large dip in magnitude. All of our simulations were also started with empty bases for consistency, so we feel as if our results will be reliable given the insights we are looking for.

9. Expected Runs Model Analysis

9.1. Lineup Quality Comparison



Within each combination type, we determined the highest-performing, lowest-performing, and median-performing lineups. Comparing the means and confidence intervals of the worst, median, and best lineups within each combination, we can see that there is a steady increase in expected runs as rank increases. More importantly, the floor and ceiling for runs scored increases with rank. While results certainly vary by game, a higher run floor and ceiling will result in higher mean runs scored in the long run. Even a small increase in mean runs scored could mean a large increase in point differential come the end of a season.

9.2. Statistical Testing

Our next goal was to test the significance of the trends we found in our exploratory data analysis. Using the worst, median, and best lineups, we conducted pairwise testing. One issue we

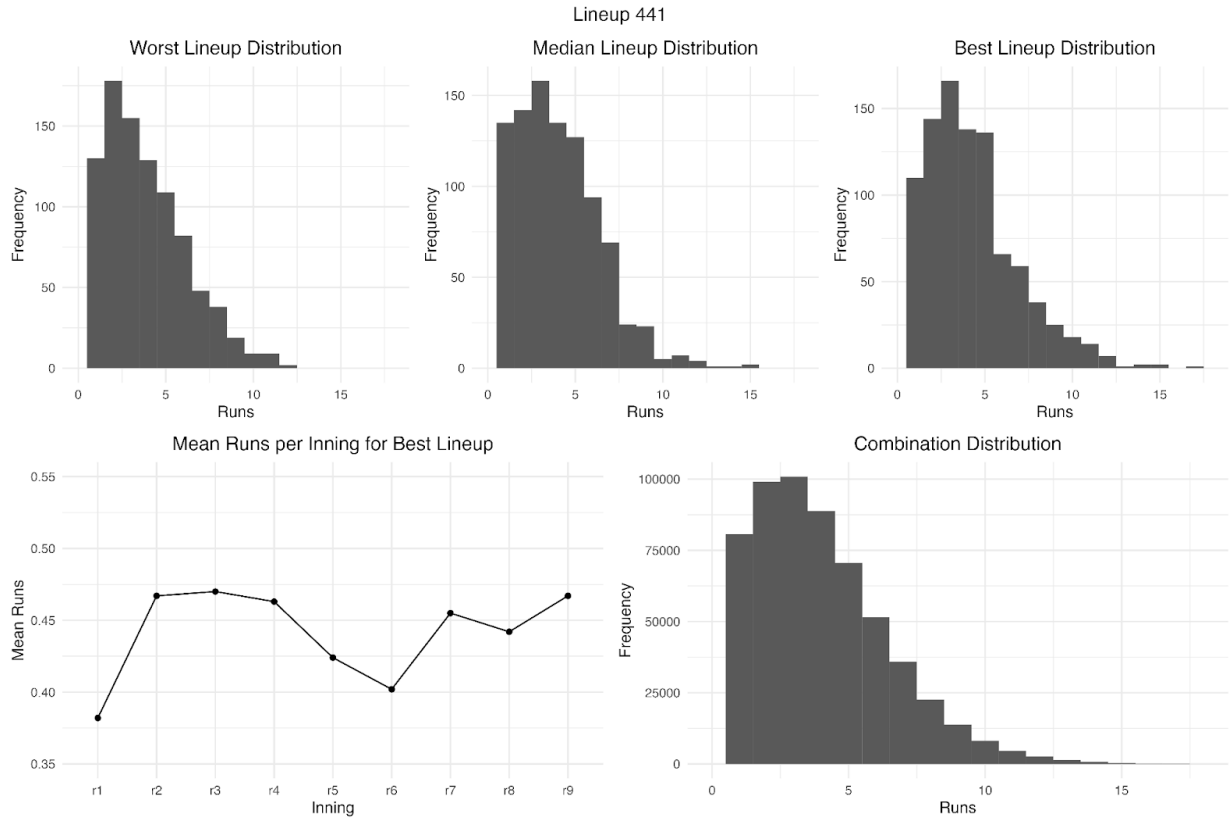
encountered with significance testing was that the distribution of total runs was positively skewed for every combination. This makes sense since the total runs scored in most games of a sample tend to be between zero and six runs, with higher-scoring games occurring occasionally. Our samples therefore violated the two-sample t-test assumption of normality, so we instead used the right-tailed Wilcoxon Rank Sum test since it is more robust to skewed data. Our goal with statistical testing was to determine whether the difference in lineup performance within combinations was significant or due to chance.

Combination	Combination Variance	Combination Expected Runs	Best Lineup	P vs. Median	P vs. Worst
441	6.709992	3.741943	CMMCMMCCP	0.09843	0.0003561***
252	6.539752	3.694005	PMMPMCCMM	0.03401*	8.123e-06***
243	6.425641	3.646728	CCMPMMPMP	0.01796*	5.924e-06***

Displays the Wilcoxon Rank Sum p-value in "P vs. Median" and "P vs. Lowest"

Within combinations "252" and "243", we found that the expected runs value of the highest performing lineup was significantly greater than both the median performing and lowest performing lineups. Combination "441" broke this trend. Within this combination, the highest-performing lineup was significantly greater than the lowest-performing lineup, but not the median-performing lineup. We could not conduct significance testing on combination "342" due to an issue with data storage.

Interestingly, combination "441" has the highest within-combination variance and the highest expected runs. The variance was calculated using the array of total runs from individual games, rather than the array of expected runs (each lineup's mean total runs). While our goal was not to compare across combinations, this finding points to a potential trend between the number of contact hitters in a lineup and expected runs.



The distributions for combination “441” indicate a high variance, with many lineups scoring 15 or more runs in some games. Furthermore, the distributions of the worst, median, and best lineups differ in shape, demonstrating that teams with this personnel could see noticeable changes in runs scored as a result of small alterations in their batting order. The line plot displaying mean runs per inning for the best lineup shows that the combination variance can also affect the performance of a lineup on an inning-by-inning basis.

Since we determined that the differences in lineup performance were significant, we can conclude that there is value in optimizing the batting order. Our following analyses dive deeper into optimizing the batting order by examining both traditional strategies and exploring smaller subsets within the lineup.

10. Traditional Lineup Analysis

As to reflect the effects and motivations of the analytics movement in baseball, much of our analysis focused on confirming or denying conventional baseball knowledge relating to lineup construction. As with all areas of baseball strategy, analytics sometimes butts heads with traditional thinking in lineup construction, but in many cases, may also affirm conventional

wisdom with statistics. We identified four conventionally used components of a lineup and used our model to test whether that strategy results in higher expected runs.

10.1. Contact Hitter Leadoff

A typical lineup consists of a contact hitter in the first, or leadoff spot, in order to get on base and set up the "heart" of the lineup to score them. We found that in our model, lineups that include a contact hitter in the leadoff spot do have higher expected runs per inning. Despite this, the difference is extremely small (0.00024 in the 3 power, 4 middle, 2 contact lineup combination), suggesting that the type of hitter in the leadoff position does not significantly influence lineup success. We hypothesized that successful MLB lineups typically have an above-average hitter in the leadoff spot, rather than necessarily a generic contact hitter.

Type of hitter in 1st lineup spot	*Mean runs*
Contact	0.41054
Middle	0.41035
Power	0.41030

10.2. Power Hitter Cleanup

Another widely-used component of lineup construction is placing a power hitter in the 4th spot, meant to drive in the hitters ahead of them in the lineup, typically contact or middle hitters. We found that lineups with a power hitter in the 4th spot actually have lower expected runs per inning than lineups with a middle and contact hitter in that spot, as evidenced by the table below, showing our findings from the 342 lineup combination. Therefore, we found conventional wisdom to be misguided in this instance as well.

Type of hitter in 4th lineup spot	*Mean runs*
Contact	0.41025
Middle	0.41087
Power	0.40971

10.3. Contact Hitter in 9th Spot

Another common, yet much less universal, component of traditional lineup strategy is placing a contact hitter in the 9th spot, often called the "second leadoff" due to the fact that the 9th and

leadoff hitters are both contact hitters that bat consecutively. We found that lineups with a contact hitter in the 9th spot have lower expected runs per inning than lineups with a middle and contact hitter in that spot, again using the "342" combination. Therefore, we found conventional wisdom to be once again misguided.

Type of hitter in 9th lineup spot	*Mean runs*
Contact	0.40991
Middle	0.41026
Power	0.41143

10.4. Contact-Middle-Power Opening

Our final traditional lineup analysis compares lineups that begin with the sequence "contact, middle, power" (CMP) to those that do not. This CMP combination is traditionally the sequence of types of the first three hitters in a lineup. We found that lineups that begin with CMP have lower expected runs per inning than lineups with a middle and contact hitter in that spot. Therefore, our findings once again do not affirm conventional analysis.

First three hitters in lineup	*Mean runs*
Contact, middle, power	0.40991
Not contact, middle, power	0.41026

10.5. Traditional Lineup Conclusions

In all of our traditional lineup analyses, we made conclusions that conventional lineup wisdom is not universally correct. As testing for lineups with a certain type of hitter in a specific position did not produce significant differences in expected runs, yet our model itself did produce such significant differences, we are led to believe that lineups should be formulated with sequences of hitter types in mind rather than slotting certain types of hitters independently in specific spots in a lineup. Although MLB lineups are commonly ordered with sequences in mind, our findings affirm the nuances in lineup construction and the merit in analytical thinking, since there is no single clear and obvious way to order hitters in a lineup. In short, we conclude that, for example, placing a power hitter in the 4th spot no matter what is often misguided and represents a flawed way of thinking about lineup construction.

11. Due Up Analysis

The next output we wanted to look at was how teams perform based on the first three hitters of an inning and if there's a trend regardless of lineup composition after those hitters. We performed this using all possible combinations of three hitters from each lineup type.

Due Up	243 Runs	243 Rank	342 Runs	342 Rank	252 Runs	252 Rank	441 Runs	441 Rank	Mean Runs	Mean Rank
CCC	-	-	0.421	3	-	-	0.416	12	0.419	5.1
CCM	0.429	1	0.411	13	0.420	1	0.422	4	0.421	3.9
CCP	0.412	3	0.409	16	0.408	16	0.425	2	0.414	7.5
CMC	0.411	5	0.409	15	0.409	13	0.418	9	0.412	8.5
CMM	0.409	7	0.416	5	0.417	4	0.421	5	0.416	4.3
CMP	0.411	4	0.414	6	0.404	19	0.411	15	0.410	8.9
CPC	0.412	2	0.424	24	0.402	22	0.432	1	0.418	9.9
CPM	0.410	6	0.408	18	0.416	6	0.410	18	0.411	9.7
CPP	0.400	20	0.412	10	0.409	14	-	-	0.407	11.1
MCC	0.403	16	0.412	9	0.418	2	0.418	8	0.413	7.1
MCM	0.407	8	0.411	11	0.412	9	0.415	13	0.411	8.3
MCP	0.404	15	0.410	14	0.412	11	0.412	14	0.410	10.9
MMC	0.407	12	0.413	8	0.414	7	0.416	11	0.413	7.7
MMM	0.406	13	0.417	4	0.413	8	0.417	10	0.413	7.1
MMP	0.399	21	0.405	22	0.417	3	0.419	6	0.410	10.5
MPC	0.403	17	0.401	25	0.404	20	0.403	20	0.403	16.5
MPM	0.407	10	0.413	7	0.410	12	0.410	17	0.410	9.3
MPP	0.402	19	0.424	1	0.401	23	-	-	0.409	10.9
PCC	0.395	24	0.411	12	0.397	24	0.410	19	0.403	15.9
PCM	0.403	18	0.409	17	0.412	10	0.419	7	0.411	10.5
PCP	0.397	23	0.408	23	0.402	21	-	-	0.402	16.9
PMC	0.395	25	0.407	20	0.408	15	0.410	16	0.405	15.3
PMM	0.407	9	0.408	19	0.406	18	0.423	3	0.411	9.9
PMP	0.398	22	0.423	2	0.416	5	-	-	0.412	7.4
PPC	0.407	11	0.390	26	0.395	25	-	-	0.397	15.6
PPM	0.405	14	0.406	21	0.408	17	-	-	0.406	13.1
PPP	0.392	26	-	-	-	-	-	-	0.392	13.2

This is a great example of the type of analysis we can do using this data. Here we are comparing the first three hitters due up for each inning, based on lineup type, and finding the best combination of each. Here, there is a clear trend that more contact hitters in the first three hitters of an inning is better. We believe that this could either be due to the traditional thought process that early hitters get on base for the power hitters to knock them in, or it could be because hitters with better contact rates generally perform better in our model. Either way, there is promise here by looking at future lineups specifically.

An interesting outcome we noticed was the one really good due-up combination 'PMP' (power, middle, power) toward the bottom of the list. This performed really well when there were only two power hitters in the lineup, which could indicate that in those lineups where you have two strong power hitters, it might be better to hit them earlier. It could also be slightly due to chance because only a few other lineups with two power hitters due up performed well within those two lineup combinations, so that's something to keep in mind as well.

12. New York Yankees Lineup Analysis

For the team-specific application, we chose the New York Yankees because their 2022 Opening Day lineup contained one of our four hitter group combinations that we had simulated. Their General Manager, Brian Cashman, also recently bragged about having the smallest analytics department in the AL East, which received a lot of backlash, even from his own fan base, considering their recent underperformance.

12.1. Opening Day Lineup 2022

Hitter Name	*Hitter Type*
Josh Donaldson	Power
Aaron Judge	Middle
Anthony Rizzo	Middle
Giancarlo Stanton	Power
DJ LeMahieu	Contact
Joey Gallo	Power
Aaron Hicks	Middle
Kyle Higashioka	Middle
Isiah Kiner-Falefa	Contact

From this lineup, it can be seen that there are two contact hitters, four middle hitters, and three power hitters. With the assumption that this will be their best nine hitters throughout the season (obviously there is a lot of variance in baseball rosters that makes it hard to account for), we can simulate every possible lineup of these hitters to find the best order to use.

12.2. Simulation Results

As a reminder, the following results were from lineups with two contact hitters, four middle, and three power hitters. We simulated 1000 games of nine innings for each lineup and sorted them based on the average runs scored per inning.

Top three lineups:

lineup	mean_runs_per_inning
contact,contact,middle,power,middle,middle,power,middle,power	0.434
power,power,middle,middle,middle,contact,middle,power,contact	0.431
contact,middle,middle,middle,contact,power,power,power,middle	0.427

It is very interesting when looking at the top three lineups that there is not a common trend among them. From prior analysis, we know that they are statistically better than other lineups, so the interesting part now is figuring out what makes them better. Also, because we have hitter groups, we still need to figure out which hitters go in which spot, allowing the manager (or GM or anyone who wants to have an opinion) to still be able to make decisions when building the lineup as well. We will take a look at where the Yankees lineup stood specifically and also look at the worst lineup combinations.

Actual lineup:

power,middle,middle,power,contact,power,middle,middle,contact	0.407

The Yankees' actual order performed about average within our simulation, with an expected runs value that is pretty much right between the maximum and minimum mean runs per inning values. This shows that this lineup was constructed well, but there is definitely room for improvement too.

Bottom three lineups:

middle,middle,middle,power,power,contact,contact,middle,power	0.379
middle,middle,middle,power,power,middle,contact,contact,power	0.378
middle,power,middle,power,contact,middle,middle,power,contact	0.378

All three of these worst lineup compositions had a lot of middle and power hitters toward the top of the lineup, including two of the three having the exact same first five batter types. Even if the top lineups aren’t directly adopted, teams can use this analysis to know what combinations to avoid (such as ‘middle, middle, middle, power, power’).

All of the outputs from this simulation are very interesting when thinking about how this would be applied to their lineup in real life. Yes, a lineup may be expected to perform better, but that is dependent on each of the hitters buying into their role within the team. Some of the players on the Yankees are current/former stars that may be expected to drop in the lineup using this data, so implementation is another whole aspect after this modeling. If the manager and players don't commit to the data-driven decisions, then there can be no improvement regardless.

12.3. Lineup Insights

The Yankees in 2022 were very successful, finishing as the second-highest scoring team in baseball (807 runs) and with the second-best record in the American League at 99-63. However, their League Championship Series sweep to the Houston Astros felt like a disappointment. Could they have scored more runs and won more games with better use of their offensive personnel? Our analysis says they could have used a better lineup to score more, which could have led to them having the best American League record. This would then have given them home-field advantage throughout the playoffs. They won over 70% of their home games, but only just above 50% of their road games, so this may have given them a better chance to reach the World Series.

The following calculations show how using the top simulated lineup would allow the Yankees to be the highest scoring team in the MLB (all simulations are over 162 game season):

- Actual lineup simulated runs scored: 593 runs (73.5% of actual 807 runs total)
- Top lineup simulated runs scored: 632 runs
- The top lineup, if scaled relative to the real-life comparison, would have scored 860 runs, which would move them ahead of the Los Angeles Dodgers for the most in the 2022 MLB season
- Over 50 more runs scored over the course of the season would likely allow them to win more games, possibly catching the Astros who won seven more games than the Yankees did

We understand that this is a strong claim to make, especially considering our general model underestimations, but with more time to complete this analysis, develop our model further, and implement player specific statistics, we believe that this type of insight can be brought to a MLB organization to help them increase their success in the future. More simulations for each would also allow our variance in our estimations to decrease as well, but computation time was not on our side during this project.

13. Final Conclusions

Across all types of lineups, our simulations in many ways refuted traditionally-used hitter placements and sequences. We found the sequence “contact, middle, power” not to be especially common at the top of successful lineups, supporting a lack of consistent success for this traditional sequence of the top 3 hitters in a lineup. Lineups with power hitters in the 4th spot do not have a significantly higher expected runs outcome than lineups with a contact or middle hitter in that spot, which is evidence against traditional baseball strategy of placing power hitters in this position. Similarly, lineups with contact hitters in the leadoff spot, as well as separately, with a contact hitter in the 9th spot do not have a higher expected runs value than lineups with a power or middle hitter in these spots. This refutes traditional baseball thinking of placing a contact hitter in these spots.

In conclusion, our lineup analysis supports lineup experimentation and flexibility, as well as to form lineups in the context of the entire sequence of hitters rather than placing each hitter type individually in a specific spot. For example, in recent years, teams have been experimenting with placing a power hitter in the leadoff spot, sparking debate in the baseball community. Our analysis supports that this strategy will work in many cases. In addition, our analysis shows significant differences in expected runs between full lineup sequences, but few significant differences when comparing lineups differentiated by their type of hitter or hitters in smaller sequences in the lineup. Because of this, we feel that it is likely beneficial to think of a lineup sequence in its totality when ordering it, rather than placing each hitter in the lineup in a set position based on perceived success in that position. This is consistent with the analytical, modern baseball school of thought.

14. Changes and Future Ideas

While we successfully created a functional model and identified meaningful insights, we faced several inefficiencies and limitations that could be remedied in future analyses.

14.1. Efficiency: Pitch-by-pitch vs. Batter-by-batter

We began our project by creating the simulation pitch-by-pitch. We did not initially consider that this would significantly increase the simulation time. As a result, running the simulation for each combination was a long process, taking nearly a week for two of the combinations. Additionally, the pitch-by-pitch nature of our project may have affected the distribution of expected runs. A batter-by-batter approach may be more effective in future analysis, particularly for our use case, which we did not identify until later in the project. Simulation speed aside, a batter-by-batter

approach could more directly follow the outcome distributions of MLB data, because the pitch-by-pitch approach adds another layer of distributions to consider, and therefore more reducible error in our model.

14.2. Systematic Underestimation

There are many sources of variation that we did not consider in our project, such as uncommon events like wild pitches and stolen bases. We believe that ignoring these factors led our model to systematically underestimate expected runs. We are uncertain as to how the inclusion of these factors would affect the interpretability of our simulation. While our model underestimates expected runs, it could still be useful for comparing lineup types. We compared the distribution of a sample expected runs distribution to the Fangraphs' expected runs distribution and found that they were quite similar when simulated with zero outs. The distributions only noticeably differed with a runner on third base.

Therefore, a comparison between simulated expected runs of two lineups could be representative of the comparison between actual expected runs of those same lineups. Regardless, including these sources of variation in future iterations could prove useful. If an event is possible in a baseball game, then it makes sense to include it in a baseball simulation.

14.3. Other Factors

Our project could have been improved by including more factors essential to batting order strategy, specifically unique pitchers, player handiness, substitutions, runner speed, fatigue, and momentum.

While a certain batting order may output the most expected runs when called into our simulation, its real-world success will be dependent on the dominant hand of each batter and the dominant hands of the opposing team's pitchers. Not only did we exclude pitcher handiness, but we also excluded pitcher variation entirely. Incorporating unique pitchers and player handiness would create more opportunities for niche analysis of simulation results.

Our "due up" analysis could be used to determine the ideal pinch hitter for any set of game conditions, but our analysis of lineup efficacy does not consider the potential use of a pinch hitter. Consider a hypothetical lineup that performs poorly when held constant, per our simulation. This lineup could perform significantly better, relative to other lineups, when the option to substitute hitters is introduced. The existence of such a trend would hypothetically change the results of our analysis.

Runner speed, which we did not consider in our model, is essential to accurately calculating baserunner advancement. Power hitters are often assumed to be slower, on average, than both contact hitters and middle hitters, on average. We did not explore the relationship between hitting power and running speed, but if the commonly held assumption is true, controlling for this factor would likely decrease the expected runs for lineups with a greater proportion of power hitters. This is because the contact hitters would tend to advance more bases on hits and steal more bases.

Other factors to consider are fatigue and momentum. Starting pitchers often show fatigue starting at around the sixth inning of a game. Since our data was drawn from all innings and situations, we likely controlled for pitcher fatigue when calculating expected runs. Still, many useful insights could be drawn from the analysis of batter performance across varying levels of pitching fatigue. Momentum is an intriguing concept that could be further explored. Many players fall into "slumps," extended periods of time where they perform below their usual level. How do these slumps affect expected runs and lineup optimization? Conversely, how does positive momentum affect these parameters? An ideal baseball model would attempt to incorporate momentum, but the challenge at hand is determining how it should be implemented. Properly weighing momentum among more technical factors would require extensive research and optimization.

We excluded these important factors due to time limitations and computational infeasibility, but they should certainly be implemented into future projects to capture the complexity of baseball more closely.

14.4. Scope: Grouped vs. Unique Hitters

Finally, while we attempted to generalize this analysis to MLB hitters as a whole, we believe this process would be better aimed towards a specific organization with specific hitters' previous data used as input. This would give a more accurate representation of what your team would actually expect to happen as opposed to making large categorizations of there only being three hitter types in the entire league. Our focus was more on the process than the hitters specifically which was why we approached our project in this fashion.