Situational Hitting: A Markov Chains Approach

SCOTT BAKER

SABERMETRICS

RHONDA HOENIGMAN

MAY 7, 2018

Contents

1 Introduction

Markov Approach Baseball can be broken up into discrete events a lot easier than a sport more dependent on flow, such as basketball. By breaking up baseball into events, it allows sabermetricians to analyze the complex relationships and subtleties in America's favorite pastime. In mathematics, a markov chain refers to a stochastic process that describes transitions between states in the state-space. In this analysis, an attempt is made to quantify offense in baseball as such process in discrete-time. This means that events in the game are based on the state of the offense before the play. Then, a one-step transition is made to the state of the game after the play happens. One important element and assumption of this type of classification is that it satisfies the markov property [3]. The basis of this property is that information of the next state of the game will be influenced by the current state and not past states. The reason this property applies because it can be assumed that batters only worry about the current state of the game[1]. Although this assumption is significant, the results from our new statistic, `marginal Situational Hitting Score (mSHS)` provide decent insight into the best batters given a certain state of the game.

Data The main source of data for the `mSHS` comes from retrosheet. For building the baseline algorithm, the years `2010` to `2013` were used. However, any standard retrosheet database is directly compatible with the `.SQL` and Python scripts found in the GitHub repository[2]. A table was built in `SQL` and exported to `.csv`[3]. This data included all events for all within the four-year span. Important elements of the event included player, starting base state, ending base state, outs generated, runs generated, inning, final score, and outcome of the play. An additional table with player names and batting average was exported and joined during Python analysis.

Design of `mSHS` There are three main elements to `mSHS`. First, a `outsFactor` is calculated based on the number of outs a player generates on a play. This is weighted with the total number of outs in the data for a certain situation. Second, a `runsFactor` is calculated based on the number of runs a player generates on a play. Similarly, a weight is added for the number of runs observed for this situation. These two factors allow us to eliminate contribution from runs and outs when there are no runs or outs generated on a play. The importance of this is to isolate the base state change for the play. The final factor is a `baseFactor`, which looks at the contribution that a player makes by advancing base state (advancing runners). This is independently calculated since a batter should be rewarded for advancing the base state and scoring runners. Sacrifice hits work in a way that the batter would be rewarded for advancing base state but penalized for getting an out. Details of the calculation will be discussed in a later section.

Limitations There are two main limitations to this approach for `mSHS`. First, isolating a player's situational hitting ability can only hold so many explicit assumptions. In our

[1]from the talk Jamie Hollowell provided in class

[2]link can be found in the appendix

[3]furthermore, the final section of this paper is a README for full instructions on use

case, we are isolating the play to just the current state. However other factors, conscious and unconscious, contribute to performance. A factor explored in *Hitting Is Contagious in Baseball: Evidence from Long Hitting Streaks* [1] was the performance of certain teams based on streaky players. Another limitation is the assumption that number of outs is independent of a player's ability to hit given a certain base configuration. It is difficult to evaluate cases in which a hitter actually contributes a sacrifice hit based on what a coach could request, for example. However, it can also be assumed that there are a comparatively low number of these cases versus normal gameplay in hopes that the weights generated from normal plays override the effect of edge cases. Additionally, as discussed with the term "clutch," there is a certain level of underlying uncertainty to the game. As Nate Silver suggests [2], " Producing wins at the plate is about 70 percent a matter of overall hitting ability, 27 percent dumb luck, and perhaps 3 percent clutch skill." `mSHS` might fail to take in account the underlying dependence of the situation such as ballpark, emotions, and wild plays. Although it does show a small relationship between skill and ability to transition the base state forward. However with most of these types of discussions: *if sabermetricians were able to perfectly model the game, why would anyone watch?*

2 Calculation for 2010 to 2013

Setup The main element to setting up the calculation is counting the number of observed events that happen in a certain data set exported from Retrosheet. This is done for three values–outs, runs, and number of one-step transitions between states. The following table is an important key-value pair of base states and their values in Retrosheet:

base config.	0B	1B	2B	1B2B	3B	1B3B	2B3B	1B2B3B
value	0	1	2	3	4	5	6	7

This will be used when looking at the number of observed events and then calculating the one-step probability transition matrix.

Probability Transition Matrix Calculating the number of outs and runs on a play requires summing up the value in the column. Therefore the average probability of an out or a run on a play is the number of total outs divided by the total number of plays. Next, the number of occurrences of each transition of base state is counted and placed in a matrix. Entry in row i and column j corresponds to the number of transitions from state i to state j. The complete matrix is seen below:

$$\begin{array}{c|cccccccc} & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ \hline 0 & 298909 & 100877 & 20322 & 0 & 2163 & 0 & 0 & 0 \\ 1 & 18680 & 70607 & 24101 & 26250 & 1800 & 6087 & 3970 & 0 \\ 2 & 2209 & 4939 & 36854 & 9555 & 10921 & 3879 & 143 & 0 \\ 3 & 1283 & 861 & 3255 & 28215 & 1391 & 5070 & 5863 & 7137 \\ 4 & 3005 & 3466 & 975 & 0 & 11283 & 3231 & 3 & 0 \\ 5 & 896 & 2157 & 1556 & 3144 & 1350 & 10823 & 2636 & 2018 \\ 6 & 359 & 1109 & 1633 & 152 & 1487 & 1177 & 7446 & 3359 \\ 7 & 452 & 85 & 381 & 1797 & 342 & 1577 & 2007 & 11275 \end{array}$$

In order to get an actual probability transition matrix, each row in the matrix must be divided by the number of total occurrences of the starting state. These values are in the following table:

start state	0	1	2	3	4	5	6	7
count	422271	151495	68499	53075	21963	24580	16722	17916

After the division, the following is the probability transition matrix. Note that each row sums to 1.0[4] [3]. This is an important property of probability transition matrices, meaning that the probability of transitioning from state i to any state is 1.

$$\begin{array}{c|cccccccc} & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ \hline 0 & 0.707 & 0.238 & 0.048 & 0.000 & 0.005 & 0.000 & 0.000 & 0.000 \\ 1 & 0.123 & 0.466 & 0.159 & 0.173 & 0.011 & 0.040 & 0.026 & 0.000 \\ 2 & 0.032 & 0.072 & 0.538 & 0.139 & 0.159 & 0.056 & 0.002 & 0.000 \\ 3 & 0.024 & 0.016 & 0.061 & 0.531 & 0.026 & 0.095 & 0.110 & 0.134 \\ 4 & 0.136 & 0.157 & 0.044 & 0.000 & 0.513 & 0.147 & 0.001 & 0.000 \\ 5 & 0.036 & 0.087 & 0.063 & 0.127 & 0.054 & 0.440 & 0.107 & 0.082 \\ 6 & 0.021 & 0.066 & 0.097 & 0.009 & 0.088 & 0.070 & 0.445 & 0.200 \\ 7 & 0.025 & 0.004 & 0.021 & 0.100 & 0.019 & 0.088 & 0.112 & 0.629 \end{array}$$

Factors The main three factors that contribute to a player's `mSHS` are the `outsFactor`, `runsFactor`, and `baseFactor`. The `mSHS` calculation follows as:

$$mSHS = (runsFactor + baseFactor) - outsFactor \tag{1}$$

First, the runs factor calculation counts the number of runs a batter has on a play and multiplies that by the expected value of a run happening given the state of the game.

$$runsFactor = runsOnPlay * expectedValueForRunsOnPlay \tag{2}$$

[4]in our example, values were truncated and row-sums come very close to 1.0

The beauty of this approach is that it "marginalizes" based on the average number of runs made on a play. In a similar fashion, `outsFactor` is calulated by:

$$outsFactor = runsOnPlay * expectedValueForOutOnPlay \tag{3}$$

However, the `baseFactor` calculation is slightly more involved. It considers two cases: no base state change and base state change. If there was no base state change on the play, `baseFactor` only considers contributions for `runsFactor` or `outsFactor`. For example, if a player was on second and the batter hits a double, the only contribtion would come from the run batted in and the lack of an out. If there was a difference in base state, the calculation for `baseFactor` is as such:

$$baseFactor = ln(changeInBaseState) * expectedValueForOneStepTransition \tag{4}$$

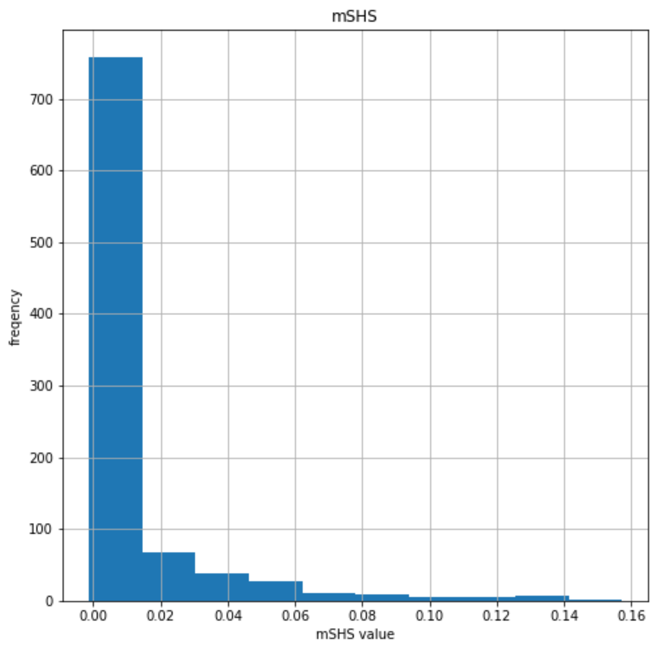
The reason for taking a natural log of the change in the base state is because a linear relationship between value of base state puts too much weight on the higher base states. For example, bases loaded does not hold seven times more value than having a player on first base. Counting occurrences of base states also found to let higher base states have too much weight compared to other factors.

3 Analysis

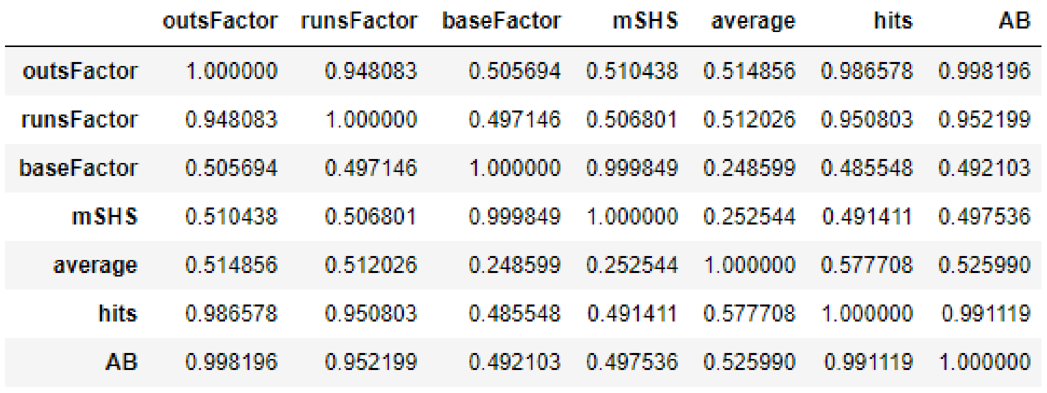
Top Players Here is a list of the top five players based on their `mSHS`.

Player	`mSHS`	`avg`	`outsFactor`	`runsFactor`	`baseFactor`
Alex Rios	0.157	0.256	0.003391	0.003810	0.156810
Ben Zobrist	0.139	0.224	0.003503	0.003833	0.138724
Josh Willingham	0.138	0.207	0.002632	0.003786	0.137257
Pedro Alvarez	0.134	0.212	0.002549	0.003384	0.133721
Yuniesky Betancourt	0.133	0.231	0.002679	0.002934	0.133574

Among first observation of this evaluation, the top five list includes a 2013 The Sporting News NL Silver Slugger (Alvarez) [5] and a 2012 The Sporting News AL Silver Slugger (Willingham) [4]. Furthermore, here is a distribution of `mSHS`.



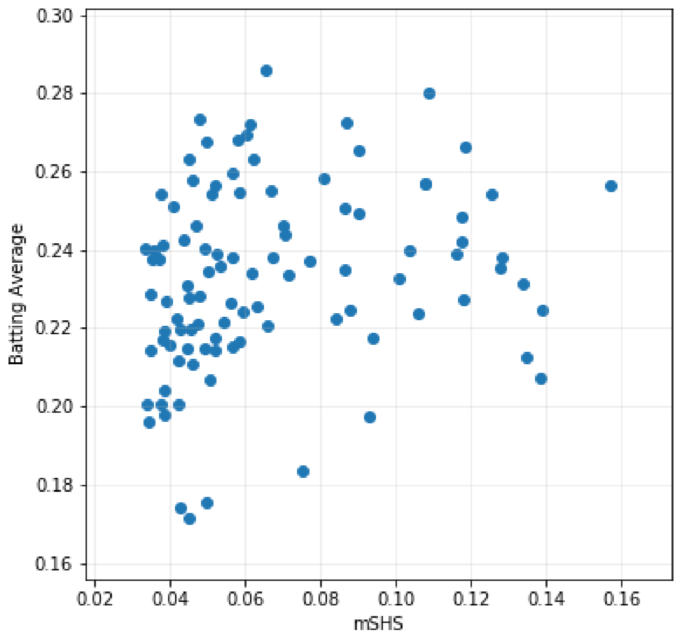
Correlations First, here is a table of standard correlation values:



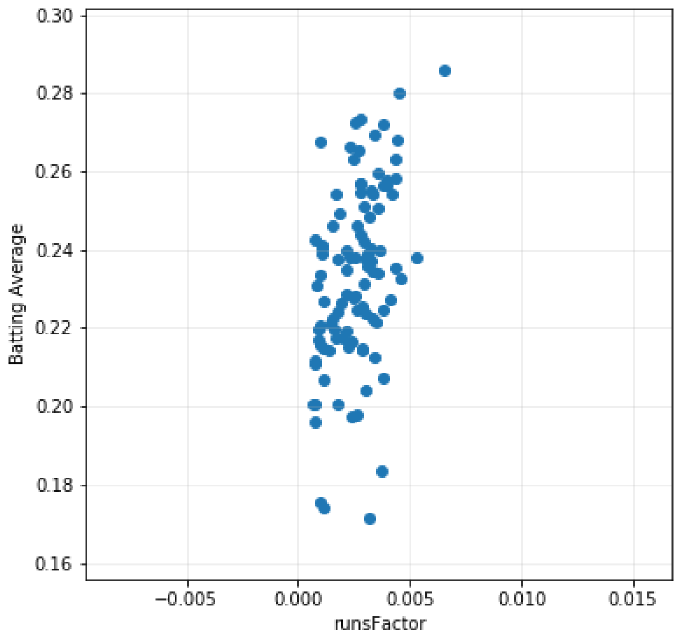
	outsFactor	runsFactor	baseFactor	mSHS	average	hits	AB
outsFactor	1.000000	0.948083	0.505694	0.510438	0.514856	0.986578	0.998196
runsFactor	0.948083	1.000000	0.497146	0.506801	0.512026	0.950803	0.952199
baseFactor	0.505694	0.497146	1.000000	0.999849	0.248599	0.485548	0.492103
mSHS	0.510438	0.506801	0.999849	1.000000	0.252544	0.491411	0.497536
average	0.514856	0.512026	0.248599	0.252544	1.000000	0.577708	0.525990
hits	0.986578	0.950803	0.485548	0.491411	0.577708	1.000000	0.991119
AB	0.998196	0.952199	0.492103	0.497536	0.525990	0.991119	1.000000

Moving in depth, it is seen that there is a weak correlation of `0.244` between `mSHS` and `average`[5].

[5]note that the correlation graphs contain the top 100 players, sorted by `mSHS`



This weak correlation could be a product of the simplicity of a batting average calculation. Additionally, a simple boolean comparison, *did they get on base?* might not be very related to *how well did they hit for the situation given to them?* Next, look at the correlation between `runsFactor` and `average`.



This is an interesting relationship since there is not a huge intuition behind getting on base and scoring runs in such a severe relationship (correlation value of `0.952`).

Conclusions Using an analysis of `mSHS`, it is apparent that part of the calculation is weighted on how the runner gets on base. Additionally, it is important that a batter has a good ratio between runs above average and outs below average while maintaining a valuable base state. Furthermore, this stat is similar to the `RE24` stat in that it looks at changes in base states. `mSHS` is a marginal statistic which puts value on being better than expectation. Whereas `RE24` calculates a raw run expectancy given a change in base state. By running 2010 to 2013 data, the top five players based on `mSHS` were observed. At first glance, these players have decent attributes when it comes to offense. Additionally, common household players such as Carlos Gonzales, Jason Heyward, Joey Votto, and Prince Fielder appear in the top fifty list.

Improvements There are several potential improvements that would offer a higher level of complexity to `mSHS`, but would take much longer to implement. Additionally, it would take further analysis to add more elements to a process based on the markov property. However, this statistic could be modified in order to observe n step transitions in a larger group of potential outcomes. This might provide better insight into the actual nature of a "situational hitter."

Another improvement to `mSHS` would be to calculate `RE24` and compare or mimic the nature of this stat to `mSHS`. This could be a dangerous addition since in `mSHS`, the assumption is that transitions within game events are independent. However, when looking at `RE24`, there is dependence on out-state of the game. It would be tricky to reason into 3-step-transitions as individual innings. This exploration could also include strike count, ballpark, and inning. These potential improvements would require a long-term investment and investigation out of scope of this project.

References

[1] Bock JR, Maewal A, Gough DA (2012) *Hitting Is Contagious in Baseball: Evidence from Long Hitting Streaks*. PLoS ONE 7(12): e51367. doi:10.1371/ journal.pone.0051367

[2] Click, J. et. al. *Baseball between the Numbers: Why Everything You Know about the Game Is Wrong*. Basic Books, 2007.

[3] Durrett, Richard. *Essentials of Stochastic Processes*. Second ed., Springer, 2012.

[4] *Josh Willingham*. http://www.retrosheet.org/boxesetc/W/Pwillj004.htm.

[5] *Pedro Alvarez*. www.retrosheet.org/boxesetc/A/Palvap001.htm.

Appendix

Code and README can be found at: `github.com/bakerscott/CU-Sabermetrics-Final`

Website link: `bakerscott.github.io/CU-Sabermetrics-Final`