Modelling Baseball with Markov Chains

Valeria Paolucci

Stochastic Modelling and Simulation

Project work presentation

Introduction

Baseball is in some countries a very popular sport for both participants and spectators.

People have long sought ways to compare baseball players and teams and have used voluminous statistics in this quest. Today, many baseball teams employ staticians to collect and analyze players data.

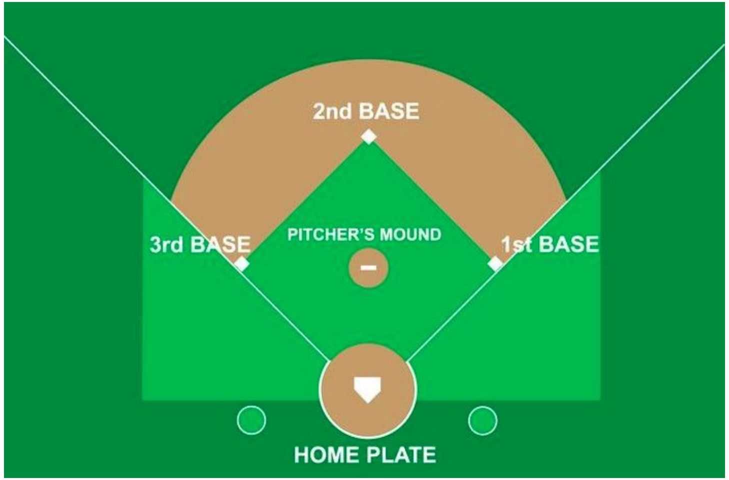
Perhaps not everyone knows that this field has been studied also in academia, and a number models have been suggested for this purpose.

In the following, we present a Markov Chain approach to the study of baseball with the aim of evaluating the performance of baseball teams.

Our main reference is a paper by Buckiet, Harold and Palacios, who were able to introduce one of the most accredited models in this field.

The rules of Baseball

- The game consists of nine *innings*; in each inning, the two opposing teams of nine players take turns batting and fielding.
- The game field, also called *baseball diamond*, includes the home plate and three other bases disposed as a square; near the center of this square, there is an artificial hill known as the *pitcher's mound*.



The rules of Baseball

- Each play starts when the *pitcher* (a player on the fielding team) throws a ball which the *batter* (a player on the batting team) tries to hit.
- A batter who hits the ball into the field of play must drop the bat and begin running toward first base (at this point, the player is referred to as a *runner*).
- The objective of the batting team is having its players advance around the four bases to score what are called *runs*.
- A batter-runner who reaches a base without being put out by the fielding team is said to be safe and can elect not to move until the next play.
- The objective of the fielding team is to prevent batters from becoming runners (by striking out the batters, i.e. when they are unable to hit the ball for three times), and prevent runners from advancing around the bases (by putting them out).
- The half-inning ends when the fielding team succeeds in eliminating three players from the batting team. When three *outs* are recorded, the inning is over.

The Markov Model for Baseball

When a player comes up to bat, he finds himself in one of 8×3=**24 possible states**.

Each state represents:

- the current locations of runners on bases (2^3=8 possibilities);
- the number of outs so far recorded in the inning (3 possibilities).

1. Bases are empty
2. One runner, on 1st base
3. One runner, on 2nd base
4. One runner, on 3rd base
5. Two runners, on 1st and 2nd base
6. Two runners, on 1st and 3rd base
7. Two runners, on 2nd and 3rd base
8. Bases are loaded

1. One out
2. Two outs
3. Three outs

The half inning ends with the third out, which constitutes the **25th** (absorbing) **state**.

The Initial Distribution

At the start of every inning, there are no outs and no runners on the bases: this is our starting state.

The initial distribution is therefore described by the vector

$$\mathbf{u_0 = (1,0,\ldots,0)}.$$

The Transition Matrix

Since there is a transition each time a new batter's at-bat comes to an end, a baseball game can be thought as a set of transitions occurring due to each player's plate appearance.

If the probability of the player changing the state of the game from any situation to any other situation is known, a **25×25 transition matrix** can be set up which will have the following block structure, in which the subscripts represent the number of outs at the start of the plate appearance:

$$\mathbf{P} = \begin{pmatrix} A_0 & B_0 & C_0 & D_0 \\ 0 & A_1 & B_1 & E_1 \\ 0 & 0 & A_2 & F_2 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

- The A, B, C blocks are 8×8 matrices
- D_0, E_1, F_2 are 8×1 column vectors
- The 0s in the middle two rows are 8×8 blocks of zeros
- The last row contains 24 zeros and a one

The Transition Matrix

- The columns 1-8 represent transitions to states with no outs;
- The columns 9-16 represent transitions to states with one out;
- The columns 17-24 represent transitions to states with two outs;

→ Each of the 8 columns in these three groups represents in particular the transition to the state having a specific location of runners on bases among the 8 possible ones;

- The 25th column represents transitions to the absorbing three-outs state.

$$\mathbf{P} = \begin{pmatrix} A_0 & B_0 & C_0 & D_0 \\ 0 & A_1 & B_1 & E_1 \\ 0 & 0 & A_2 & F_2 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Rows represent transitions *from* these states respectively.

The Transition Matrix

- The A blocks represent transitions which do not increase the number of outs of the inning;
- The B blocks represent transitions which lead from no outs to one out and from one out to outs;
- The C block represents transitions which increase the number of outs from zero to two;
- The vector D_0 represents the probability of going from zero outs to three outs;
- The vector E_1 represents the probability of going from one out to three outs;
- The vector F_2 represents the probability of going from two outs to three outs;
- The 0 blocks represent transitions which decrease the number of outs in the inning and thus have probability zero.

$$\mathbf{P} = \begin{pmatrix} A_0 & B_0 & C_0 & D_0 \\ 0 & A_1 & B_1 & E_1 \\ 0 & 0 & A_2 & F_2 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Filling a player’s Transition Matrix

If the exact probabilities of a runner changing one state of the game to another are not known, then we can fill the matrix by using a particular model combined with easily available statistics about how often a batter gets given types of hits. In our model for runners’ advancement on the basepath, we assume that, at each at-bat, only the following actions can take place:

- **Out** Runners do not advance; outs increase by one
- **Walk** Batter moves to first base; runners advance only if forced
- **Single** Batter moves to first base; runner on first moves to second base; other runners score
- **Double** Batter moves to second base; runner on first moves to third base; other runners score
- **Triple** Batter moves to third base; all base runners score
- **Home run** All base runners and batter score

Notice that in this model:

- we are not including double and triple plays (i.e. the act of making two or three outs respectively during the same continuous play), therefore the blocks C_0, D_0, E_1 are zero;
- runners are not allowed to advance on an out, hence off-diagonal elements of the B blocks are zero.

Filling a player’s Transition Matrix

So we have:

$$A = \begin{pmatrix} P_H & P_S + P_W & P_D & P_T & 0 & 0 & 0 & 0 \\ P_H & 0 & 0 & P_T & P_S + P_W & 0 & P_D & 0 \\ P_H & P_S & P_D & P_T & P_W & 0 & 0 & 0 \\ P_H & P_S & P_D & P_T & 0 & P_W & 0 & 0 \\ P_H & 0 & 0 & P_T & P_S & 0 & P_D & P_W \\ P_H & 0 & 0 & P_T & P_S & 0 & P_D & P_W \\ P_H & P_S & P_D & P_T & 0 & 0 & 0 & P_W \\ P_H & 0 & 0 & P_T & P_S & 0 & P_D & P_W \end{pmatrix}$$

$$B = P_{out} I$$

$$F = (P_{out}, \ldots, P_{out})^T$$

where $\boldsymbol{P_W}$, $\boldsymbol{P_S}$, $\boldsymbol{P_D}$, $\boldsymbol{P_T}$, $\boldsymbol{P_H}$, $\boldsymbol{P_{out}}$ are the probabilities of a player getting a walk, single, double, triple, homerun or out, respectively, and I is the 8×8 identity matrix.

All the above probabilities can be obtained by widely available individual season statistics of the players. P may be constructed for any batter if the relevant statistics about his performance are known. Once the probabilities have been entered, we normalize each row to ensure that the matrix is stochastic.

Single-Inning Run Distribution

We are interested in computing the distribution of runs that a team can score in an inning, that is the probability that *i* runs are scored, for *i* ∈ {0,1,2,…,20}.

We are assuming that the maximum number of runs scored in an inning is 20; the probability of a team obtaining more than 20 runs in an inning is indeed negligible and this choice is useful to reduce the computation run-time.

Certain transitions result in the scoring of runs. In order to determine the distribution of runs, we need to know, for any transition, whether it will cause 0, 1, 2, 3 or 4 runs to score during the course of one at-bat.

In our computations, we decompose each *P* matrix into five matrices ***P0***, ***P1***, ***P2***, ***P3***, ***P4***, called *scoring matrices*, such that P = P0 + P1 + P2 + P3 + P4, where the values in each scoring matrix correspond to transition probabilities leading to the scoring of 0, 1, 2, 3 and 4 runs respectively in a single plate appearance.

The Algorithm

We introduce a non-stochastic 21x25 matrix ***U***, called *score-keeping matrix*, in which the rows represent the runs (between 0 and 20) that a team can score, and the columns represent the 25 possible states of the game. At the beginning, before any batters have come to bat, U_0 is zero everywhere except for the first entry which equals to 1.

Then, at each at-bat, we update *U* row by row in the following way:

$$\mathbf{U_{n+1}(row\ j) = U_n(row\ j)P0 + U_n(row\ j\text{-}1)P1 + U_n(row\ j\text{-}2)P2 + U_n(row\ j\text{-}3)P3 + U_n(row\ j\text{-}4)P4}$$

In general, if *n* is the number of batters who have had an at-bat in this inning, the score-keeping matrix will only have positive entries in rows from 0 to *n*. Only after three batters can there be any positive entries in the final column of *U*; after many iterations, we find that virtually all entries of *U* outside the three-out state go to zero.

In practice, we iterate over the player transition matrices until the sum of the 25th column (that is, the probability of three outs) is within a given tolerance ε of 1. <u>**This column vector, after the stopping criterion is reached, gives the distribution of runs in an inning for the team being considered**</u>.

Nine-Inning Run Distribution

We may also want to compute the distribution of runs that a team can score in a nine-inning game.

In this case, we could simply give *U* nine times as many rows (i.e. 189 rows), with each set of 21 rows representing an inning. When the three-out state in one inning is reached, the results must be moved to the zero-out state of the following inning, with the same number of runs scored (i.e. 21 rows down and 24 columns to the left).

The computation is performed until the probability that there are 27 outs is greater than 1 - ε.

The result is the distribution of runs in a nine-inning game.

Note that this implementation of the algorithm allows for a total of 20 runs in a nine-inning game for a team (the error in truncating at 20 runs is reasonable since only twice in 20th century have both teams in a Major League game scored at least 20 runs).

Expected Number of Runs

Once we have the probabilities that a team scores *i* runs, we can easily compute the expected number of runs produced in a game.

Given a run distribution *S*, we have

$$E\left(S\right) = \sum_{i=0}^{20} i \cdot p(i) \, .$$

Implementation

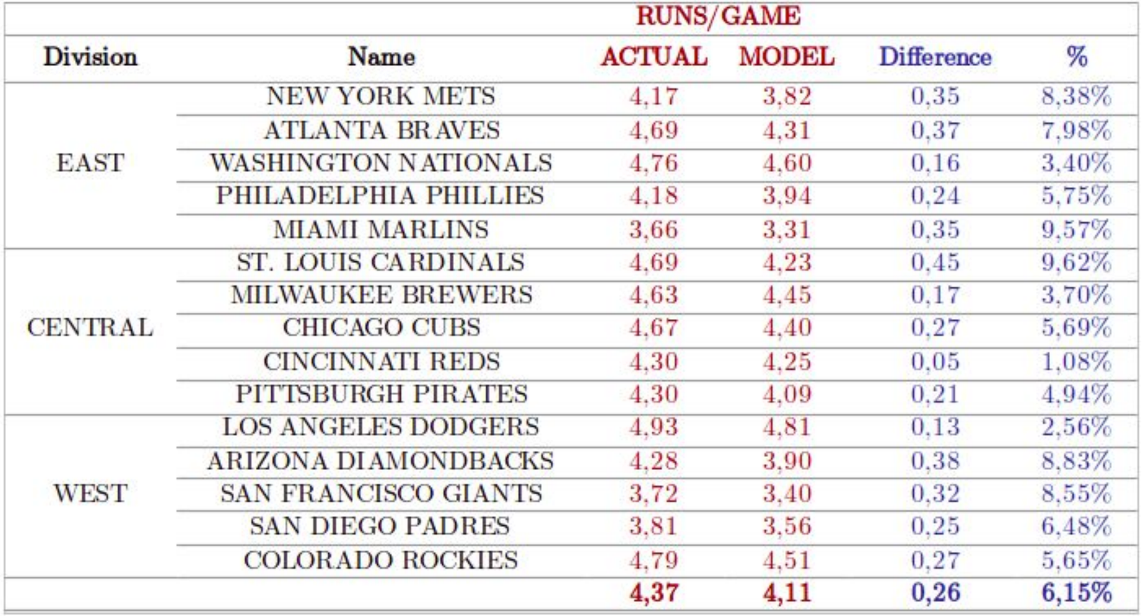
We implemented in Python a model allowing to compute the single inning-run distribution and the expected number of runs for a given team.

We ran simulations for all the 15 teams in National League Baseball for the 2018 season, using batting data to create transition matrices. In particular, we averaged over all players on a team and so we were able to use for the calculations a single constant transition matrix for each team.

Data are obtained from *www.baseball-reference.com/bullpen/Baseball_statistics*.

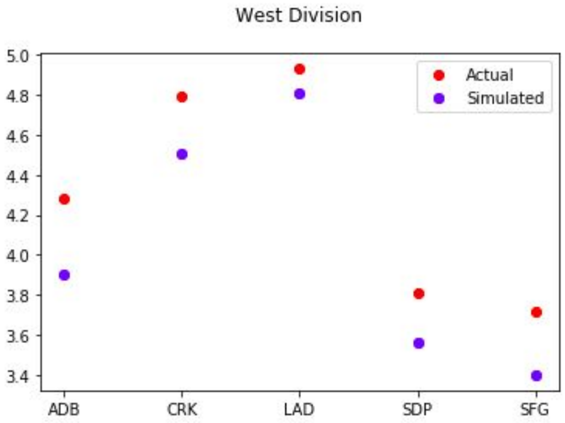
The result of the model simulations closely approximate the actual season data. The model shows a satisfactory level of accuracy for the expected number of runs produced, since the simulated data closely approximates the actual run production. We can observe that the expected number of runs produced by a team is on average lower than the actual game data, which may in part depend on the assumption about runners not advancing on an out.

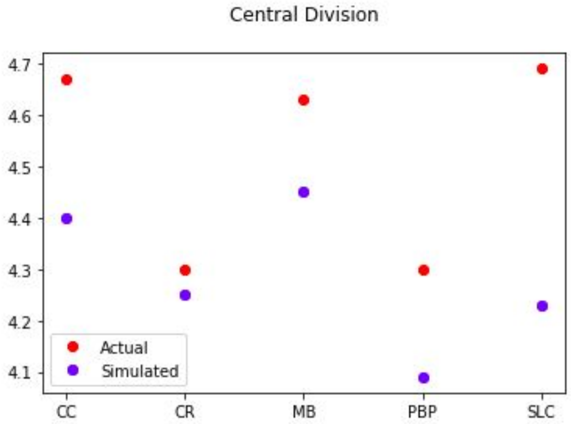
Results

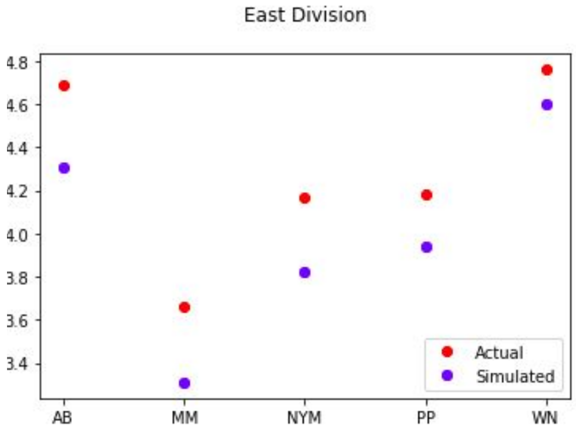


		RUNS/GAME			
Division	Name	ACTUAL	MODEL	Difference	%
EAST	NEW YORK METS	4,17	3,82	0,35	8,38%
	ATLANTA BRAVES	4,69	4,31	0,37	7,98%
	WASHINGTON NATIONALS	4,76	4,60	0,16	3,40%
	PHILADELPHIA PHILLIES	4,18	3,94	0,24	5,75%
	MIAMI MARLINS	3,66	3,31	0,35	9,57%
CENTRAL	ST. LOUIS CARDINALS	4,69	4,23	0,45	9,62%
	MILWAUKEE BREWERS	4,63	4,45	0,17	3,70%
	CHICAGO CUBS	4,67	4,40	0,27	5,69%
	CINCINNATI REDS	4,30	4,25	0,05	1,08%
	PITTSBURGH PIRATES	4,30	4,09	0,21	4,94%
WEST	LOS ANGELES DODGERS	4,93	4,81	0,13	2,56%
	ARIZONA DIAMONDBACKS	4,28	3,90	0,38	8,83%
	SAN FRANCISCO GIANTS	3,72	3,40	0,32	8,55%
	SAN DIEGO PADRES	3,81	3,56	0,25	6,48%
	COLORADO ROCKIES	4,79	4,51	0,27	5,65%
		4,37	**4,11**	**0,26**	**6,15%**

Results


West Division
5.0
4.8
4.6
4.4
4.2
4.0
3.8
3.6
3.4
ADB
CRK
LAD
SDP
SFG
● Actual
● Simulated


Central Division
4.7
4.6
4.5
4.4
4.3
4.2
4.1
CC
CR
MB
PBP
SLC
● Actual
● Simulated


East Division
4.8
4.6
4.4
4.2
4.0
3.8
3.6
3.4
AB
MM
NYM
PP
WN
● Actual
● Simulated

More complex models for *P*

The inclusion of more complete baseball statistics affects a batter's transition matrix *P*.

Almost any batting data could be incorporated in the model. Here are some examples:

- Not restricting our approach to transition matrices which remain static throughout a game or season: if the data is available, variations can be made in the values of *P*. A further improvement would be to run the simulations using live data where a player's probabilities could change as a result of actual at-bat results.
- Removing the constraint that a team cannot score on an out (that is, giving positive value to the probability of a runner advancing on an out); though, this requires to consider also hard-to-describe events.
- Allowing for double and triple plays, that is, allowing for non-zero entries in C_0, D_0, E_1 blocks.

In general, if any extra information is available, our model can take advantage of it (for instance, pitching and defense statistics can influence batter transition matrices as well).

Further Applications

Representative lineups - For each team, we can construct a representative lineup using the top 9 players (ranked according to at-bats). The model can be used to accurately compute the expected number of runs produced by a particular lineup of nine batters.

Winning the game - Once the distribution of runs in a game is known, it can be used to predict the outcome of the game. The probability of Team1 winning a game in nine innings (that is, scoring more runs than Team2) is

$$\sum_{i=1}^{20}\left[S(\text{Team1})_i \sum_{j=0}^{i-1} S(\text{Team2})_j\right]$$

Then, we could also calculate the expected number of games a team should win in a season (in this case, we assume that the lineup remains the same in every game throughout the season and that the player's transition matrix is also constant throughout the season).

Optimal batting orders - Run distributions for different batting orders can be used to determine the optimal batting order for a set of nine players. The optimal batting order can be defined as the one leading to the largest expected number of runs scored in a nine-inning game. Though, this requires exhaustive computation for the 9! (362.880) possible batting orders, therefore it may be useful to develop algorithms to find the *near-optimal* batting order, thus reducing the search space significantly. By comparing the run distributions produced by the optimal and the worst batting orders, we can quantify the effect of the batting order on the expected number of games a team will win.

References

Buckiet, Harold, Palacios, *A Markov Chain Approach to Baseball* (1997), Operations Research 45.1

Hirotsu, Bickel, *Optimal batting orders in run-limit-rule baseball*: a Markov chain approach (2014), Journal of Management Mathematics (2016) 27

Ursin, *A Markov Model for Baseball with Applications* (2014), Theses and Dissertations, Paper 964

The elegance of Markov Chains in Baseball, www.medium.com/sports-analytics/the-elegance-of-markov-chains-in-baseball-f0e8e02e7ac4

www.baseball-reference.com/bullpen/Baseball_statistics

en.wikipedia.org/wiki/Baseball_rules