Using game theory to analyze pitchouts in baseball

Nathan Kershaw

May 3, 2025
Revised November 8, 2026

Abstract

Using a game-theoretic approach to analyzing pitchouts, general solutions are computed and discussed. In doing so, we can examine which nodes favor pitchouts in our simplified model, analyzing patterns between the number of outs and different counts. Overall, we find that under our model, pitchouts are more favorable with two outs than one and no outs, and more favourable with no outs than one out. We additionally find that an increase in stolen base success rate generally incentivizes an increase in pitchout use, and that an increase in strike rate allows a pitcher to pitch out more by reducing the cost of giving up a ball. Pitchouts are generally also favored in counts where the pitcher is ahead.

Introduction

In baseball, a *pitchout* refers to a pitcher throwing the ball intentionally high and outside, conceding a ball in order to, hopefully, throw a runner out who is trying to steal. The idea is that if a runner is likely to attempt to steal, throwing a pitchout allows the catcher a clearer throwing path and reduces pop time since the catcher can receive the pitch standing or almost standing. In recent years, baseball has seen a decline in the usage of the pitchout, which for a while mirrored the decline in stolen base rates. In 2023, larger bases and restrictions on pitcher disengagements were introduced [2]. As a result, the number of stolen bases per game increased in both 2023 and 2024. One might expect pitchout rates to follow suit and also increase; however, they remained far lower than previous levels (see Figure 1).

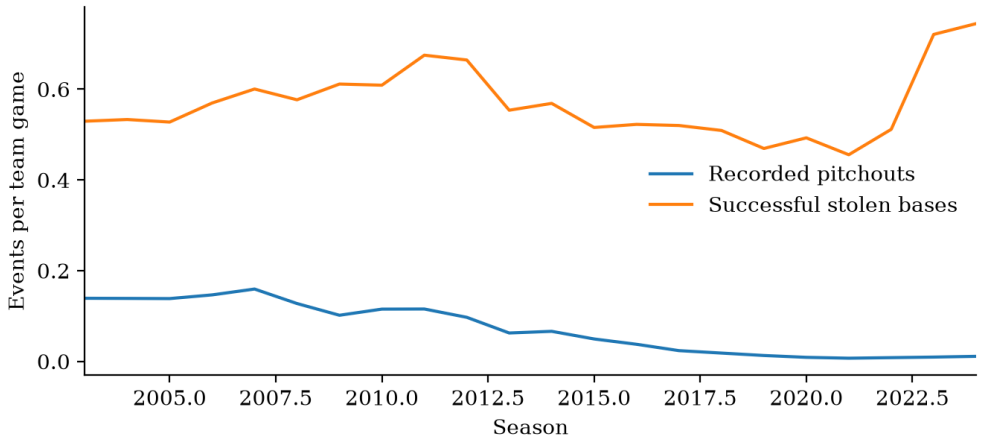


Figure 1: Recorded pitchouts and successful stolen bases per team game, 2003-2024.

Naturally, this raises the question of whether pitchouts are worth giving up a ball for, and if so, what situations are worth pitching out in? Further, what rates arise in those situations? How many runs saved does pitching out offer? These are the questions this paper aims to address.

To answer these questions, a game-theoretic approach is used. In order to isolate the effects of a pitchout, we use a simplified 2×2 zero-sum game on a given pitch: the pitcher has the option to either pitchout or pitch normally, and the runner has the option to either run or stay. Thus, on a given pitch, if we can find the expected value (EV) for each pair of pure strategies, we can calculate the Nash equilibrium to determine optimal stolen base and pitchout rates within that game. The difficulty lies in that single pitches in an at bat, or an inning, should not be considered separate: if you don't steal on the first pitch of an at bat, you will likely have the opportunity to steal on the second, and this impacts the EV on the first pitch. As such, we need to use a combination of computing Nash equilibria as well as solving a game tree backwards to arrive at proper solutions.

The contents of this paper are as follows: Section 1 discusses the methods used to arrive at the solutions. In Section 2, general solutions using average stats from the years 2000-2024 [4] and an existing run-expectancy matrix [5] are presented and discussed. In Section 3 we explore how the equilibrium rates change as things like stolen base rate and strike rate vary. Final remarks are made in Section 4.

Along the way, we will not only uncover important metrics regarding pitchouts, but also gain insight into stolen base rates as well. For simplicity, and because the majority of stolen bases are from first to second, we look at only stolen bases when there is only a runner on first. The conclusions from this case could be adapted to be used on other cases as well. The observed rates rely on the simplifications and assumptions used in the model, and should not be seen as recommendations for MLB teams to pitch out or steal at those rates. In real games, pitchers have more options available, such as a pickoff, and true optimal rates depend on specific changing factors like lead distance. Instead, the scope of this work is more modest: by using a simplified model, we may isolate the specific pitchout/no pitchout trade-off, and identify situations where this trade-off is favourable.

Related work

The idea of using game theory to analyze pitchouts is not new. Turocy studies this using an inspection game, with the defense's attention to the runner reducing the stolen base success rate at a cost against the batter [6]. Gal and Pfeffer give an example with two successive steal and pitchout decisions, where the decision on the second pitch affects the first [1].

Melville et al. [3] use a similar approach to this paper with a more complex model to study a broad range of pitching decisions. Their model includes lead distance, pickoffs, disengagements, and pitch-specific choices. Similar to this work, their model solves the count games backwards. One difference is that their model ends when a given at-bat ends, whereas our model continues the game when a strikeout occurs and the runner remains on first. While their work is not primarily concerned with pitchouts and is more concerned with a broader range of decisions, they do discuss pitchouts. The purpose of keeping our model simple is to see which parts of the decision do the work, and to isolate pitchout-specific effects. This allows us to evaluate where pitchouts tend to be more favorable and what effects are driving this.

1 Methods

The data used throughout this paper is the play-by-play data from [4]. To have enough data for the less common outcomes, data from the years 2000 through 2024 is used for the main calculation. This provides a general reference case, rather than estimating the exact probabilities for a given pitcher-hitter matchup. The main continuation values come from an existing RE288 matrix [5].

Analyzing pitchout effectiveness

The first question that must be answered is how effective pitchouts are. To answer this, we need a prediction of the stolen base success percentage when a pitchout is thrown, given the success percentage without a pitchout. This paper only focuses on stealing second from first, and for this comparison we use 2003-2024, because there is seemingly a discontinuity in the pitchout reporting from 2000-2002. This is done for the pitchout effectiveness only, we still use 2000-2024 for transition probabilities.

We look through all play-by-play data and look at instances where a pitchout was thrown, and a runner was stealing second from first, with no other runners on base. We keep track of the outcome (stolen base or caught stealing), as well as the runner. For each runner, we then need to estimate their non-pitchout stolen base success percentage. To do this, we simply look at all instances of them stealing second with no other runners on and keep track of the outcomes. This uses both past and future attempts to predict a runner's "true" stolen base rate. To ensure large enough sample sizes, we restrict to runners with at least 20 non-pitchout attempts. We group runners into ten-percentage-point bins of their non-pitchout success percentage. Within each bin, we compute the average ordinary pitch success rate, weighted by each runner's number of attempts on pitchouts. For each bin, we then compute the pitchout success rate, compare that with the ordinary success rate, and report the attempt counts and Wilson 95% intervals. We can use this to analyze the observed differences, which is done in Section 2.

Generating the game tree

We consider stealing second with no other runners on base. Each node in the game tree corresponds to a certain count, runner position, and number of outs. On a given node, the next nodes are given by:

- Ball, runner stays
- Ball, runner steals
- Ball, runner thrown out
- Strike, runner stays
- Strike, runner steals
- Strike, runner thrown out
- Hit in play, runner was not stealing
- Hit in play, runner was stealing
- Hit by pitch, with the forced advance

Note that in some cases (such as a steal and strikeout with two outs, or a walk with the runner thrown out) some of these may be impossible. The terminal nodes are ones in which there are three outs, the runner steals or is thrown out, the batter walks or is hit by a pitch, or the ball is hit in play. A strikeout with the runner staying and fewer than two outs keeps the runner on first and starts the next batter at 0-0. We use these rules to generate a game tree, with play continuing until a terminal node is reached. There are 36 nonterminal decision nodes, corresponding to 12 possible counts and three possible numbers of outs. The terminal values describe the expected runs for the remainder of the inning under the continuation convention below.

Solving the game tree

The first step in solving the game tree is to find the values of the terminal nodes. For terminal nodes not involving a ball hit in play, we use Tango's RE288 matrix at the state [5]. For nodes with 3 outs, the value is zero.

For a ball in play, we find all instances of that situation in the play-by-play data. We then take the average over all instances of the runs scored on the play, plus the RE288 matrix value of the remaining state. For simplicity, the value of a ball-in-play is taken to be the same regardless of whether a runner is moving on the pitch.

In general, these values are going to be determined both by the pitcher, the batters coming up, and the speed of any runners on base. Determining the values based on the average, however, allows us to study a broader, general case. The purpose of this is to find overarching trends rather than strategies specific to certain matchups.

Once the values of the terminal nodes are determined, we may begin solving the rest of the game tree. We first need, given two nodes A and B, the odds of reaching B from A given a pitchout and stolen base decision. To determine this, we need both pitchout and non-pitchout stolen base success percentages, which are both assumed to be given. We also need the percent a strike is thrown, as well as the percent the ball is hit in play, given the node A. To determine these percentages, we once again take an average over all occurrences of the node A, using ordinary pitches with only a runner on first. The ordinary-pitch outcome distribution is taken to be the same for a runner who goes and one who stays. A pitchout is modeled as a ball with a lower success percentage if the runner goes.

Given the ordinary-pitch outcome probabilities and the two success percentages, we can then determine the likelihood of the next nodes, given decisions on whether to pitchout and whether to steal. As an example, if the runner is stealing and no pitchout is thrown, the chance of going from 1-2 with a runner on first to 2-2 with a runner on second is

$$P(\text{ball} \mid A,\ \text{no PO}) \times P(\text{SB} \mid \text{no PO}).$$

To solve the entire game tree, we iteratively work backwards. On each iteration, we solve for the strategies on the nodes that are unsolved, but whose next nodes are all solved. This, along with the chances of reaching the next nodes given each pair of decisions, allows us to determine the EV of each pair of decisions. We can then set up a 2×2 payoff matrix and find the Nash equilibrium. This gives the strategy for that node, as well as the EV of that node. We can do this iteratively until all nodes are solved. Pure strategies and boundary cases are included. The results of doing this for different cases are discussed in the following sections.

Interpreting which nodes favor pitchouts

Recall that the purpose of our simplified model is to make the cost and benefit of different decisions easier to evaluate. Fix a node and the values under continuation. We may write the payoff matrix as:

	No pitchout	Pitchout
Stay	a	$a + C$
Steal	c	$c - B$

.

Writing it this way, C is the cost to the defense of pitching out when the runner stays, and B is its benefit of pitching out when the runner goes. If $B, C > 0$ and $c - a < B + C$ the pitchout probability q to make the runner indifferent is:

$$a + qC = c - qB, \qquad q = \frac{c - a}{B + C}.$$

Thus, if B and C are fixed, a greater benefit $c - a$ from stealing against an ordinary pitch increases the pitchout probability. With B and $c - a$ fixed, a lower cost C of pitching out when the runner stays also increases that probability. The corresponding steal probability is $C/(B + C)$.

2 The general solution

In this section, we discuss general solutions, computed using the methods of Section 1.

Pitchout effectiveness results

We begin with the results of Section 1.1. Again, we want to be able to predict the stolen base success percentage given a pitchout was thrown, using the non-pitchout success percentage. We bin the results using the non-pitchout success percentage. We then compute the success percentage for a given bin. The results are seen in Figure 2.

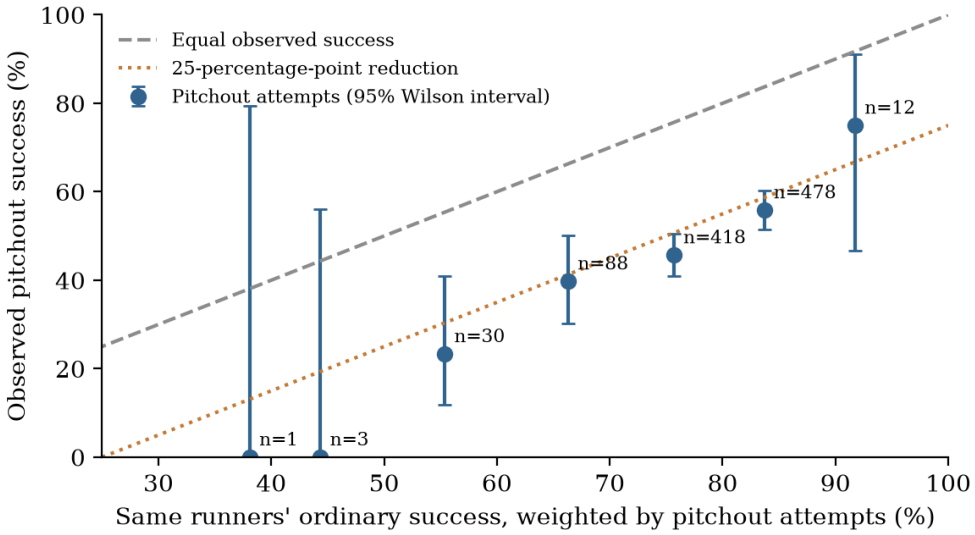


Figure 2: Observed non-pitchout and pitchout stolen base success percentages, 2003-2024.

Looking at the differences, we can examine how much lower the success percentage is when a pitchout is thrown. We see a roughly 25 percentage point decrease in success rate on steal attempts with a pitchout thrown, relatively consistent across buckets with higher sample sizes. This comparison does not account for all factors in play, such as how pitchouts are selected. We thus use an assumption that pitchouts reduce the stolen base success percentage by 25 percentage points in the reference solution, and vary this assumption separately in Section 4. We view this as an input to the model and not a claim that the 25 percentage points holds across all situations, as many factors may influence the effectiveness on a given play.

Game tree solution

Using the terminal node values from Section 1.2, we can solve for the equilibrium stealing rates as well as pitchout rates for different stolen base success percentages. The 2024 stolen base success rate was 79%, so we begin with a non-pitchout success percentage of 79%, as a reference scenario. Below is a table for the equilibrium steal and pitchout rates, as well as a table of the expected values of the nonterminal nodes. Each decision node is represented by a tuple of length four: balls, strikes, the runner's base, and the number of outs. The runner's base is one at each decision node.

Node	Go	PO	Node	Go	PO	Node	Go	PO
(0,0,1,0)	26.07%	17.36%	(0,0,1,1)	29.07%	10.17%	(0,0,1,2)	34.76%	14.33%
(0,1,1,0)	24.25%	15.60%	(0,1,1,1)	26.78%	11.23%	(0,1,1,2)	42.66%	19.56%
(0,2,1,0)	21.38%	14.68%	(0,2,1,1)	24.21%	9.55%	(0,2,1,2)	41.16%	12.85%
(1,0,1,0)	40.92%	6.46%	(1,0,1,1)	42.22%	5.22%	(1,0,1,2)	37.88%	11.24%
(1,1,1,0)	37.29%	13.23%	(1,1,1,1)	39.93%	6.49%	(1,1,1,2)	47.23%	13.74%
(1,2,1,0)	35.49%	10.68%	(1,2,1,1)	38.84%	6.82%	(1,2,1,2)	56.91%	11.78%
(2,0,1,0)	0.00%	0.00%	(2,0,1,1)	0.00%	0.00%	(2,0,1,2)	0.00%	0.00%
(2,1,1,0)	63.55%	0.84%	(2,1,1,1)	0.00%	0.00%	(2,1,1,2)	63.93%	0.92%
(2,2,1,0)	0.00%	0.00%	(2,2,1,1)	0.00%	0.00%	(2,2,1,2)	0.00%	0.00%
(3,0,1,0)	0.00%	0.00%	(3,0,1,1)	0.00%	0.00%	(3,0,1,2)	0.00%	0.00%
(3,1,1,0)	0.00%	0.00%	(3,1,1,1)	0.00%	0.00%	(3,1,1,2)	0.00%	0.00%
(3,2,1,0)	100.00%	0.00%	(3,2,1,1)	100.00%	0.00%	(3,2,1,2)	0.00%	0.00%

Figure 3: Equilibrium strategies for a 79% non-pitchout success percentage and a 25-percentage-point pitchout penalty.

Node	EV	Node	EV	Node	EV
(0,0,1,0)	0.92437	(0,0,1,1)	0.54642	(0,0,1,2)	0.23789
(0,1,1,0)	0.86359	(0,1,1,1)	0.49650	(0,1,1,2)	0.19742
(0,2,1,0)	0.78187	(0,2,1,1)	0.42980	(0,2,1,2)	0.14442
(1,0,1,0)	0.97316	(1,0,1,1)	0.58617	(1,0,1,2)	0.26394
(1,1,1,0)	0.90643	(1,1,1,1)	0.52895	(1,1,1,2)	0.21975
(1,2,1,0)	0.81571	(1,2,1,1)	0.45676	(1,2,1,2)	0.16233
(2,0,1,0)	1.06818	(2,0,1,1)	0.65570	(2,0,1,2)	0.29976
(2,1,1,0)	0.97805	(2,1,1,1)	0.58541	(2,1,1,2)	0.25212
(2,2,1,0)	0.87740	(2,2,1,1)	0.50447	(2,2,1,2)	0.19129
(3,0,1,0)	1.26937	(3,0,1,1)	0.78457	(3,0,1,2)	0.36848
(3,1,1,0)	1.15461	(3,1,1,1)	0.71529	(3,1,1,2)	0.32570
(3,2,1,0)	1.04976	(3,2,1,1)	0.62698	(3,2,1,2)	0.26477

Figure 4: Equilibrium expected values for the reference scenario.

Note that with a 3-2 count and two outs, the model steals at a 0% rate, whereas in practice the dominant strategy is a 100% steal rate. This is due to the assumptions made in our model, namely that a ball in play with a runner stealing is given the same value as a runner not stealing. With a full count and two outs, the only non-ball-in-play continuations either end the inning or force the runner to second anyway. Thus, the EV from stealing is the same as not stealing, and the model opts not to steal.

There are a few other things to note here. The first is the average pitchout rate, which across all nodes equally weighted, is 5.91%. The second thing to look at is which nodes are most favourable for pitchouts, which we divide by the number of outs, as well as the count.

Grouping by outs, we see an average pitchout rate of 6.57% with 0 outs, 4.12% with 1 out, and 7.04% with two outs. There are seemingly two factors in play here. One is the cost of a walk: with fewer outs, there is more opportunity to score, making the cost of a walk higher. This can be seen using the RE288 matrix: with no outs and a 0-0 count, a runner on first gives an expectation of 0.90, whereas runners on first and second give an expectation of 1.50, for a difference of 0.60. With one out, the difference is 0.40, and with two outs the difference is 0.21. This disincentivizes giving away a free ball with fewer outs, which is likely why we see the highest pitchout rate with two outs.

The pitchout rate is not monotone with out count; however, the lowest pitchout rate is with 1 out. This is due to the second factor, which is the cost of a stolen base. In particular, a successful

stolen base is more valuable to the offense with fewer outs: Using the same analysis as before, the difference between a runner on first and a runner on second with no outs and a 0-0 count is 0.25, with 1 outs it is 0.16, and with 2 outs it is 0.09. This warrants more protection for stolen bases with fewer outs, which, as seen in Section 1, results in a higher pitchout rate. This is likely why we see a higher rate with 0 outs than with 1 out.

The second thing to look at is the count. Grouping by balls minus strikes, the averages for differences -2, -1, $+0$, $+1$, $+2$, $+3$ are 12.36%, 12.61%, 8.37%, 2.74%, 0.00%, 0.00%, respectively. Intuitively, when the pitcher is ahead in the count, they can afford to give up a ball. On the other hand, if the pitcher is behind, giving up a ball is more costly. This explains the roughly monotone negative relationship between balls minus strikes and the pitchout rate.

We can also look at the stolen base rates. The equal weight average of the model's steal probabilities is 27.07%. With no outs, the average is 29.08%, with 1 out the average is 25.09 %, and with two outs the average is 27.04%. The two competing factors leading to these numbers are the added benefit of stealing a base with fewer outs, but also the added cost of getting caught with fewer outs. Grouping by balls minus strikes, the averages for differences -2, -1, $+0$, $+1$, $+2$, $+3$ are 28.91%, 37.49%, 23.82%, 49.83%, 0.00%, and 0.00% respectively. These numbers vary significantly within each group, so the differences between the -2, -1, $+0$, and $+1$ groups are mostly due to specific count instances. For example the high $+1$ rate is largely driven by 3-2 counts. The interesting part is the $+2$ and $+3$ counts, where the model decides to wait for a future opportunity. In this case, the odds of reaching second due to a walk are already high, and the model deems stealing not worth the risk of being thrown out.

3 Other solutions

In this section, we vary different parts of the model to see how they affect equilibrium pitchout rates and equilibrium stolen base rates.

We can also look at solutions for varying the stolen base success percentage and the pitchout reduction. The overall averages change, and individual nodes can move between pure and mixed solutions. This is shown in the two figures below:

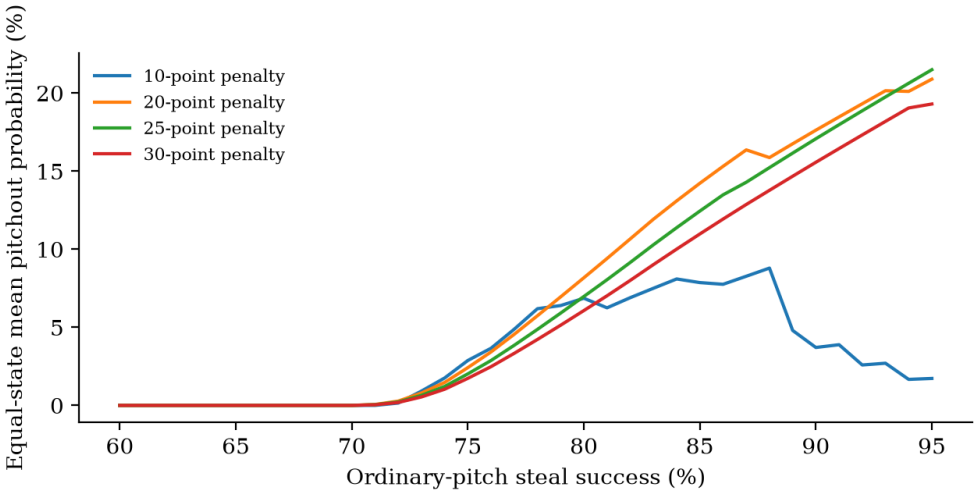


Figure 5: Equilibrium pitchout probabilities as the non-pitchout success percentage varies.

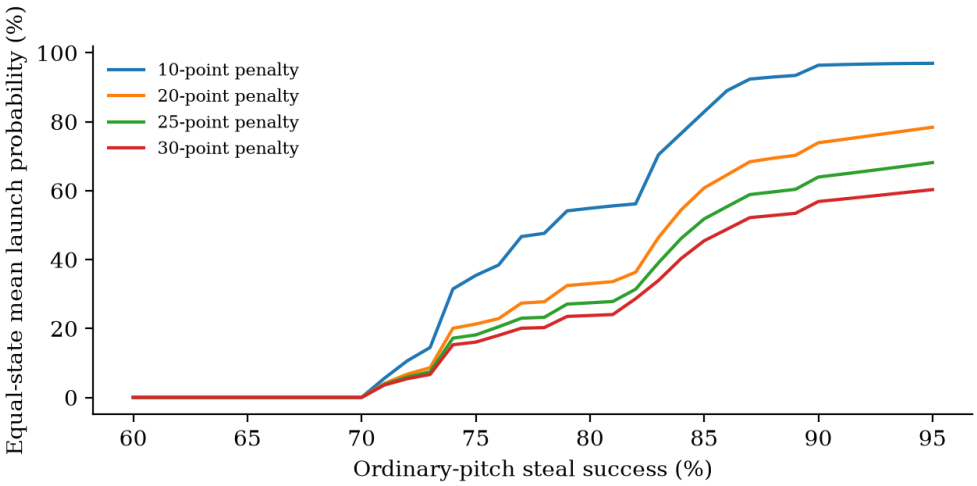


Figure 6: Equilibrium steal probabilities as the non-pitchout success percentage varies.

As one would expect, steal rate increases with both a decrease in pitchout effectiveness and an increase in success rate. For 20-, 25-, and 30-point pitchout penalties, pitchout rate also increases with success rate. Interestingly, for a 10-point penalty, the pitchout rate decreases after around an 88% success rate. In this situation, there is a high probability of success even with a pitchout, which can make pitching out not worth the cost of a ball.

Next, we alter the pitcher's strike rate. Intuitively, a pitcher with a higher strike rate can afford to pitch out more, since they are less likely to walk a batter even after giving up a ball. To test this, we add x percentage points to the pitcher's strike rate across all nodes for $x \in [-10, 10]$, take x percentage points from the ball probability, and re-compute the equilibrium. The results can be seen in the following two figures:

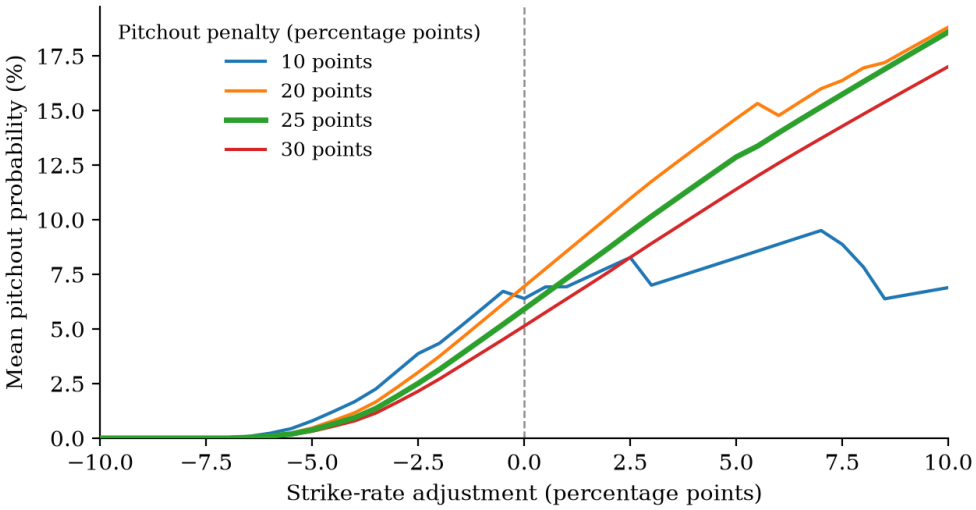


Figure 7: Equilibrium pitchout probabilities as the strike rate varies.

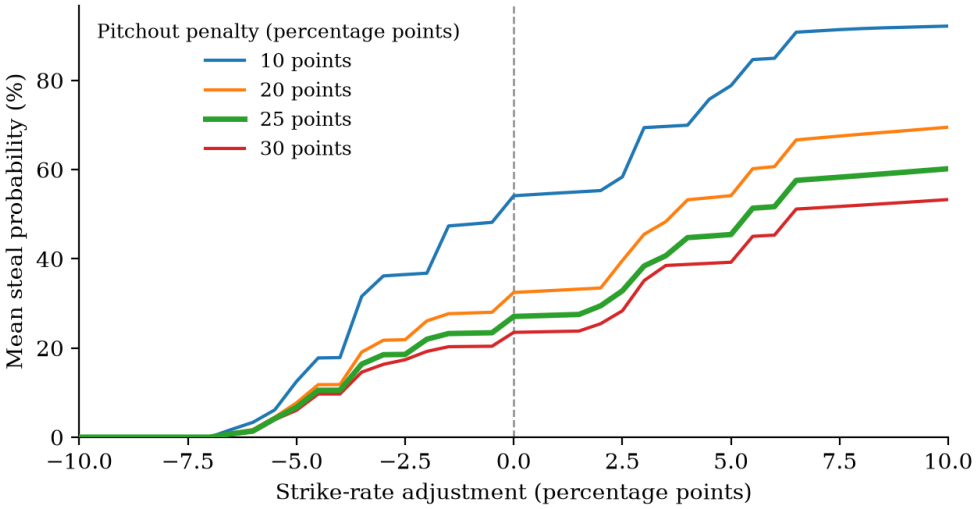


Figure 8: Equilibrium steal probabilities as the strike rate varies.

We see that generally, an increase in strike rate allows the pitcher to throw more pitchouts. Interestingly, stolen base also increases with strike rate, even with the increased pitchout rate. This is likely due to the decrease in walk rate incentivizing more stolen bases.

4 Discussion and conclusion

In this paper, we use a game-theoretic approach to analyze pitchouts. We build a game tree and use data from 2000-2024, together with an existing RE288 matrix, to solve for equilibrium stolen base and pitchout rates in the model. In doing so, we uncover which situations may be favorable to pitchout in and why that is the case.

In Section 2 we provide and analyze a general solution assuming a 79% stolen base success rate and a 25-point pitchout penalty. We find that pitchouts are favoured the most with two outs (7.04%) followed by no outs (6.57%) and one out (4.12%). This is likely due to two competing factors: the cost of a walk is more pronounced with fewer outs, but the cost of allowing a stolen base also increases with fewer outs. Pitchouts are generally also favored in counts where the pitcher is ahead.

In Section 3 we analyze two alternate solutions: varying the stolen base success rate and varying the pitcher's strike rate. We find that both an increase in stolen base rate as well as an increase in strike rate incentivize both more pitchouts and more stolen bases.

The exact percentages depend on the simplified model as well as several assumptions and estimates, and thus should not be seen as a recommendation for in-game strategy where more options are available, and the simplifications do not hold. To assess whether these solutions hold up in real game scenarios, more study is needed.

Stability checks on the general solution

The solution in Section 2 uses Tango's RE288 matrix, uses 2000-2024 data for transitions, and uses a 25-point penalty for pitchouts. To assess how much the results depend on these choices, we compare in Figure 9. These checks change the transition sample, the run-expectancy, or the pitchout-effectiveness input.

Calculation	Mean PO	Pitcher ahead	Pitcher behind	Rank corr.
Main calculation	5.91%	12.53%	1.37%	1.000
2023-2024 transitions	7.05%	12.47%	2.97%	0.962
Empirical RE288	6.22%	13.62%	1.44%	0.944
2023-2024 transitions and RE288	6.49%	10.44%	2.98%	0.882
20-point penalty	6.95%	14.90%	1.56%	0.998
30-point penalty	5.13%	10.79%	1.22%	0.998

Figure 9: Checks on the reference solution. Probabilities are unweighted averages over decision nodes; rank comparisons are with the reference node pitchout probabilities.

The rank correlation is the Spearman correlation between two vectors of pitchout probability across all states. The lowest rank correlation is 0.882, indicating that the ordering of the nodes (i.e., which favor pitchouts more or less) remains relatively similar across all choices. The mean pitchout percent also varies at most 1.14 percentage points from the reference solution.

Limitations and future work

There are a few drawbacks to the methods used, which could be improved upon for future work. One is that we do not include pickoffs in our calculations. Pickoffs and disengagements can change both a runner's willingness to go and the defense's alternatives to giving up a ball. Those effects can depend on the count, so omitting them can change which nodes are most favourable, as well as the overall level of pitchouts. A model that includes these choices is available in [3], the purpose here is to keep the two-choice interaction easier to inspect.

Secondly, we assume the stolen base success percentage is constant across all counts. The true percentage depends on the runner, pitcher, catcher, lead, and pitch. Changing these inputs can affect the stolen base rates and which counts are favoured, and hence the pitchout rates as well. Similarly, while contact values vary across nodes, the contact value is modelled as being the same regardless of if the runner is moving on the pitch. The pitchout effectiveness could also be better modeled, perhaps using things like lead distance, speed, and catcher's pop time.

Another simplification is ignoring foul balls, which affects decisions and the transition probabilities. Modelling this more accurately may alter results.

Finally, we say a node is terminal whenever the runner leaves first, a walk or hit batter forces the advance, or the ball is put in play. The continuation value is then given by the run-expectancy. A non-inning-ending strikeout with the runner staying continues within the game. Solving a full inning equilibrium would require solving decisions at other nodes where a pitchout is possible, such as when the runner is on second. This affects the expected value of some of the current terminal nodes, and may alter the given solution. The model also uses run expectancy as the measure to optimize, whereas in-game strategies should use win probability. The model gives a way to examine mechanisms, compare strategies, and develop broad insights, but using it for a particular game would require both player and situation inputs, and a fuller treatment of the choices that are available in game.

Data acknowledgment

The information used here was obtained free of charge from and is copyrighted by Retrosheet. Interested parties may contact Retrosheet at https://www.retrosheet.org.

References

[1] Ya'akov Gal and Avi Pfeffer. Networks of influence diagrams: A formalism for representing agents' beliefs and decision-making processes. *Journal of Artificial Intelligence Research*, 33: 109–147, 2008. doi: 10.1613/jair.2503. URL https://arxiv.org/pdf/1401.3426.

[2] Major League Baseball. All of your questions about the new baseball rules, answered, February 2023. URL https://www.mlb.com/news/mlb-new-rules-for-2023-faq. February 6.

[3] William Melville, Robert Greathouse, Tristan Mott, Christopher Archibald, and David Grimsman. The pitcher's dilemma: A game-theoretical model of pickoffs and stolen bases. In *MIT Sloan Sports Analytics Conference*, 2026. URL https://www.sloansportsconference.com/research-papers/the-pitchers-dilemma-a-game-theoretical-model-of-pickoffs-and-stolen-bases.

[4] Retrosheet. Parsed play-by-play data, 2026. URL https://www.retrosheet.org/downloads/plays.html. Data used in this study are restricted to the 2000–2024 seasons.

[5] Tom Tango. RE288: Run expectancy by the 24 base-out states x 12 plate-count states, recursively, November 2018. URL https://tangotiger.com/index.php/site/comments/re288-run-expectancy-by-the-24-base-out-states-x-12-plate-count-states-recu. November 22; underlying input seasons are unspecified.

[6] Theodore L. Turocy. An inspection game model of the stolen base in baseball: A theory of theft. Working paper, August 22 version, August 2014. URL https://tturocy.github.io/papers/theft-20140822.pdf.