Analysis of MLB Pitch Spin Rate Data Before and After 2021 Crackdown on Foreign Substances

Jacob Keck †
† Department of Mathematics, California State University, Fullerton

Abstract

Major League Baseball's crackdown on foreign substances in June of 2021 was based on the increase in spin rates from pitchers and a decrease in league-wide offensive production. However, did the enforcement of said foreign substance ban noticeably decrease the league-wide spin rate in Major League Baseball? Using the divided differences method and a cubic spline, the data of league-wide spin rate immediately before and after the enforcement of the foreign substance ban was interpolated to numerically evaluate the integrals using Romberg's method to analyze the divided differences polynomial and the composite trapezoidal rule to analyze the cubic spline polynomial. A logistic differential equation was also used to model the data. Comparing the evaluations of the integrals before the foreign substance ban to the evaluations after reveals an approximate 3.071% decrease in spin rate around when enforcement began, with the offensive production at this decreased spin rate begin significantly greater.

1 Introduction

In baseball, spin rate - measured in rotations per minute (RPM) - is one of the most important factors in recognizing a pitcher's effectiveness. It is why some 100-mile-per-hour fastballs are easy to hit while others are near unhittable. Pitchers do this by trying to create greater friction between their fingers and the ball upon releasing a pitch. Typically, pitchers will want to produce more spin on the baseball because it will create more movement (greater deviation from the expected path) on the pitch by exaggerating the Magnus Effect. The spin creates an asymmetric wake of air behind the ball that pushes the ball in the direction of the spin. [1] On some pitches such as fastballs, this is mostly backspin that moves the ball upwards, fighting against the pull of gravity. For other pitch types, this can be topspin that causes the ball to move downwards or sidespin that creates sideways movement.

The spin rate of every Major League pitch is found by Major League Baseball using a combination of Hawk-Eye and Statcast. Hawk-Eye is a system of high-speed framerate cameras used by many different sports leagues around the world for use in optical analysis. Statcast, developed by MLB, captures and analyzes data from systems like Hawk-Eye into many different statistics and metrics. [5]

Pitchers may use illegal foreign substances to often improve their grip on the baseball and to increase the amount of spin on their pitches to be able to get batters out more effectively. Every pitch's spin has an axis of rotation, which will differ depending on how the pitcher grips and releases the ball. Some such foreign substances are pine tar (a substance batters

use to get a better grip on the bat), spider tack (a substance typically used by strongmen in weightlifting competitions) [1], and a combination of two perfectly legal substances: sunscreen and sweat. These substances increase spin by the stickiness of the foreign substances increasing connection and friction between a pitcher's fingers and the ball upon release.

The widespread use of foreign substances by pitchers has always been an open secret in Major League Baseball. However, not all "sticky stuff" is illegal, and the longtime rules on such substances are confusing, rarely enforced, and never enforced on a league-wide scale leading up to June 21, 2021. [2] Typically, a few pitchers will be ejected and suspended from games every year for using illegal foreign substances. However, the accusation only gets given out when it is either incredibly obvious (like a visible shiny streak of some substance on a pitcher's skin) or if there is greater context between players and teams causing petty arguments over league rules. [3]

Figure 1: Davis, J. (2021) [4] Enforcement of MLB's foreign substance checks began on June 21, 2021.

Over many years, Major League Baseball had seen a decline in offensive performance as well as an increase in what is to be considered relatively 'boring' outcomes: strikeouts and walks (results where the ball is not hit into play). This is the motivation Major League Baseball has for instituting this crackdown on "sticky stuff". This trend continued into the first couple of months of the 2021 MLB season, which led to Major League Baseball reasoning that they should do something to counteract it. [1] [3] By June 3, 2021, rumors had begun around the league that enforcement against foreign substances was imminent. [3] On June 15, Major League Baseball announced the procedures for the new way the league is to apply its rules against pitchers using illegal substances, with regular checks of pitchers' hands, fingers, and gloves by umpires at the end of innings and when a pitcher is taken out of the game. The enforcement of these rules began on June 21, 2021.

The dataset is league-wide average spin rate, in rotations per minute (RPM), grouped for a data point every two days between June 01, 2021 and July 10, 2021. The data is grouped as so so that we are working with fewer data points in total (20 instead of 40 for computing time) and to protect against a small sample size on any one day meaningfully impacting the data.

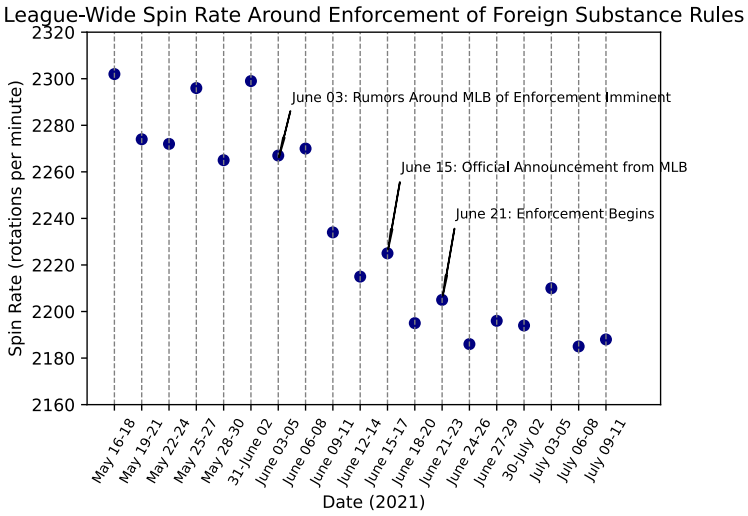

Figure 2: (a) The decline in spin rate (in rotations per minute) around the beginning of MLB's enforcement of foreign substance rules.

Date	Spin Rate (RPM)
May 16-18	2302
May 19-21	2274
May 22-24	2272
May 25-27	2296
May 28-30	2265
May 31-June 02	2299
June 03-05	2267
June 06-08	2270
June 09-11	2234
June 12-14	2215

Date	Spin Rate (RPM)
June 15-17	2225
June 18-20	2195
June 21-23	2205
June 24-26	2186
June 27-29	2196
June 30-July 02	2194
July 03-05	2210
July 06-08	2185
July 09-11	2188

Table 1: The dataset is the league-wide average spin rate, in rotations per minute (RPM), of every three days from May 16 to July 11, 2021.

2 Methods

This paper looks to analyze the average spin rate of pitches directly leading up to and following Major League Baseball's crackdown on foreign substances on June 21, 2021. This will be done by creating an interpolating polynomial for the entirety of the dataset using the divided differences method as well as a cubic spline interpolating the data points. The numerical integration techniques of composite trapezoidal sums and Romberg's method will be used to compare the value of the area under the curve of the interpolating functions of the league-wide spin rate data to analyze the effectiveness of MLB's foreign substance crackdown.

Following this, we will compare the areas under our interpolating functions between the first half and second half of our dataset to see if there truly is a discernible drop in spin rate during the transitional period between when rumors began about enforcement and when enforcement began. This will be done over the whole of the interval with the plot split directly into

$$\int_0^9 f(x)\,dx \text{ and } \int_9^{18} f(x)\,dx \tag{1}$$

The midpoint of the data was chosen to be at the specific data point of June 12-14 as it is halfway between when rumors began of enforcement on June 3 and when enforcement began on June 21. To be able to decide how significant our results are, we must introduce a few new topics and results at the beginning of the Conclusion section that are specific to baseball and will assist in our concluding analysis.

In addition, a logistic differential equation will be used to model and fit the data. The differential equation will be solved using the Runge-Kutta Method of Order Four. Why are we using a logistic differential model when that is typically used to model population growth or decay? Despite lacking a 'population' in this context, we are trying to model time series data moving from one relatively stable level to another. This is something that we can reasonably model using a logistic differential equation.

3 Numerical Analysis

3.1 Numerical Interpolation

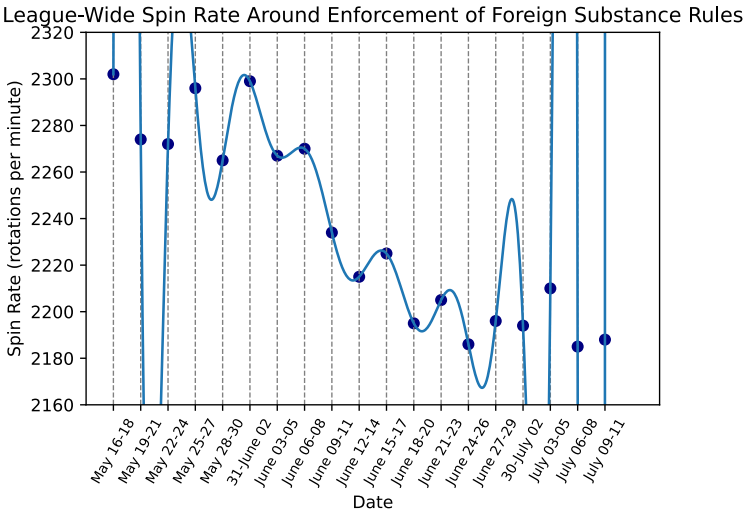

Figure 3: The divided differences method polynomial curve imposed on the scatter plot of the data.

Superimposing the polynomial resulting from the divided differences method as well as the cubic spline separately onto the scatter plot of our data gives us two different curves to

express the interpolation of the data. As we can see, the polynomial interpolation is less reliable than the cubic spline interpolation due to the wild fluctuations in the polynomial on the ends. The curve goes as low as $-9,000$ and as high as $100,000$ in these highly oscillatory parts of the function.

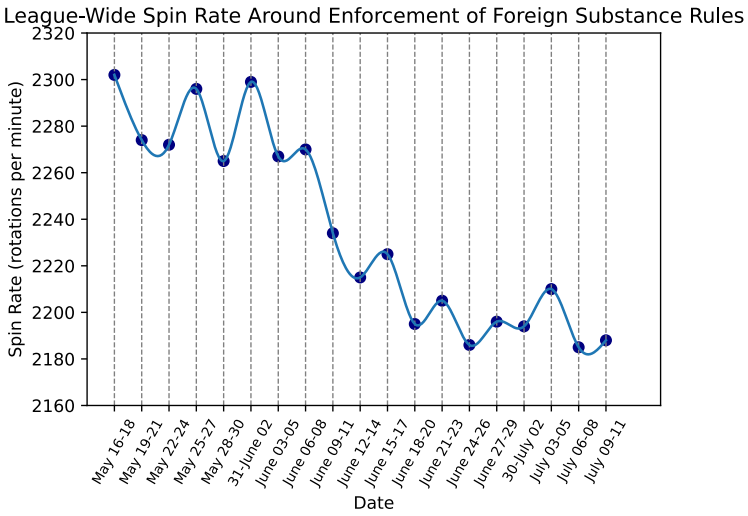

Figure 4: The cubic spline curve is imposed on the scatter plot of the data.

3.2 Numerical Integration

Using Romberg's method on the divided differences interpolating polynomial, we get an area under the curve of ≈ 491415.078. The massive fluctuations we receive from the polynomial significantly affect the resulting areas. Dividing by 19 to get another weighted average shows just how unreasonable this interpolation is, with this weighted area being ≈ 25863.951.

Taking the area of the Cubic Spine, the far more reliable function for describing the area that we have, using the composite trapezoidal rule gives us ≈ 42421.473. Dividing by the amount of data points we are interpolating over, 19, we get a good value for a weighted average of the data: ≈ 2232.709. Comparing visually to what we would expect an average to be from the scatter plot, the cubic spline is effective at interpolating this dataset.

The integrals for our cubic spline function will be the ones noted above, where $f(x)$ is our cubic spline function. Therefore, we must normalize our final integrals by the size of the intervals, dividing by 9 and 9, respectively. Utilizing the composite trapezoidal rule to numerically evaluate the integrals, the normalized area per unit before enforcement began is ≈ 2267.529, whereas the normalized area per unit after enforcement began is ≈ 2197.886. This results in $\approx 3.071\%$ decrease in spin rate after MLB began enforcing its foreign substance rules.

For the divided differences polynomial, the wild fluctuations in the function before the May 25-27 data point and after the June 30-July 02 data point lead to any numerical integration of the curve at face value being unreliable in actually describing the relationship between the spin rate before and after enforcement. However, consider discounting the area under the highly oscillatory portions of the curve, focusing between May 25 and July 02.

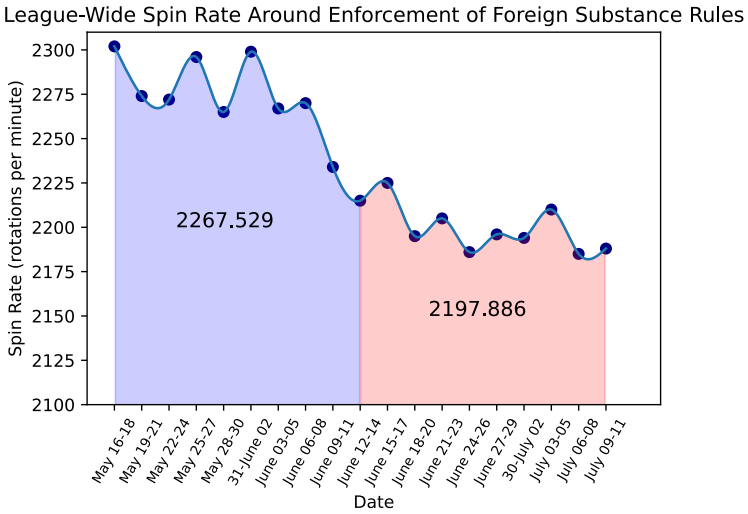

Figure 5: The difference of the weighted area under the curve before the midpoint of the transition to the new enforcement rules and after.

The integrals to calculate using Romberg's method will then be:

$$\int_3^9 p(x)\ dx \text{ and } \int_9^{15} p(x)\ dx \tag{2}$$

where $p(x)$ is the interpolating polynomial found using the divided differences method. We must normalize by the size of the intervals again, dividing by 6 this time. The normalized area per unit before enforcement began on June 21 was found to be ≈ 2262.570 while the normalized area per unit after enforcement began was found to be ≈ 2206.086, giving us an $\approx 2.496\%$ decrease after enforcement on foreign substances began. This is less extreme than what was found from the Cubic Spline and Romberg's Method, but it is still a noticeable decrease in spin rate over this time.

3.3 Differential Equation Model

Now, let us attempt to model this dataset with a logistic differential equation. As noted above, despite the lack of a physical 'population', to model this environment we must use a logistic differential equation with a carrying capacity and threshold population, levels where the curve of the solution will asymptotically converge. This differential equation model is:

$$\frac{dX}{dt} = -r \cdot X \cdot \left(1 - \frac{X}{C}\right) \cdot \left(1 - \frac{X}{T}\right) \tag{3}$$

where X is our dependent variable, spin rate, r is a parameter establishing the rate at which the solution asymptotically converges, C is the carrying capacity (the upper bound of our system), and T is the threshold population (the lower bound of our system). We will take C as the maximum of our data plus ten, 2312, while T is the minimum of our data minus ten, 2175. These extensions of the interval above and below the maximum and minimum of our dataset allow for margins for where our solution can account for and model the cases where

the average spin rate may extend slightly beyond the limits of what was originally found in the data.

To utilize the Runge-Kutta Method of Order Four, we must choose a staring point and an r, the growth rate. After implementing a least squares algorithm to minimize the error of our logistic model, the optimal growth rate was found to be -5.2. The initial condition will be our first date point, at $(0, 2302)$. Our differential equation for the dataset, thus, is

$$\frac{dX}{dt} = 5.2 \cdot X \cdot \left(1 - \frac{X}{2312}\right) \cdot \left(1 - \frac{X}{2175}\right) \tag{4}$$

Utilizing the Runge-Kutta Method of Order Four to numerically solve the above differential equation, we get the following solution curve:

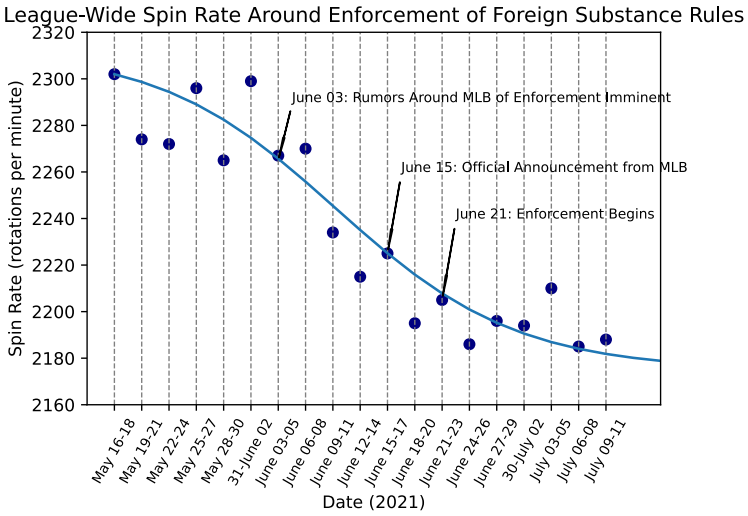

Figure 6: Plot of the solution to the logistic differential equation on top of the scatter plot of data.

We see here that the solution curve for our differential equation model reasonably fits our data with our curve proceeding to converge toward our lower limit, which we can view as the spin rate as a function of time converging toward a lower value after the enforcement of the foreign substance rules.

4 Conclusion

Can it be said there was a significant drop in average spin rate because of the new enforcement procedures beginning? It is difficult to call even a 3.071% decrease significant in the more reliable cubic spline. To assist in determining the significance of this result, let us introduce the statistics OPS, OPS+, OBP, and SLG which each describe the effectiveness of a batter's hitting in different ways.

OBP stands for On-base Percentage and is the fraction of all times a batter goes to hit that the batter gets on base. SLG stands for Slugging Percentage and is a statistic similar to On-base Percentage, except different hitting outcomes are weighted by the number of bases

the batter achieves in their attempt at-bat; i.e. a single (one-base hit) has a weight worth 1, a double (two-base hit) has a weight worth 2, a triple (three-base hit) has a weight of 3, and a home run (batter goes around all four bases) has a weight worth 4. The convention for how these statistics are displayed is as decimals with 3 decimal places without a preceding 0 if the value is less than 1. For example, an OBP or SLG may be displayed as .343 or 1.000.

OPS stands for On-base Plus Slugging and is the sum of a batter's On-base Percentage and their Slugging Percentage. OPS is typically considered one of the more reliable and well-rounded statistics in evaluating a hitter's effectiveness out of the more popular hitting ones like OBP, SLG, or AVG (Batting Average). These other statistics only evaluate one dimension of hitting; either getting on base in the case of OBP or AVG, or hitting for power and scoring other players already on base in the case of SLG. However, OPS allows one to evaluate both dimensions of hitting by combining them into one statistic. OPS+ is a modification to the OPS statistic that centers the league average OPS to be 100. The formula for OPS+ from a player's OPS (recall: OPS = OBP + SLG) is:

$$\text{OPS+} = 100 \cdot \left(\frac{\text{OBP}}{{}_L\text{OBP}} + \frac{\text{SLG}}{{}_L\text{SLG}} - 1 \right) \tag{5}$$

where ${}_L$OBP and ${}_L$SLG are the league averages for On-base Percentage and Slugging Percentage, respectively. [6] This formula was designed in such a way that, for a player's OPS+ of $(100 + r)$, then that player's OPS is $r\%$ above (or below in the case of $r < 0$) league average.

There are a few other considerations when calculating OPS+ like Park Factors (every baseball park has different physical dimensions, which will impact various statistics. Park Factors are different coefficients on parts of the calculation for each ballpark that control for the different environments), but for simplification, we will not be incorporating them. As we are considering a hypothetical neutral player in a hypothetical neutral environment, controlling only for spin rate, the park factors should not matter. We will be using this formula to analyze and compare the hitting statistics, specifically OPS+, against the two different calculated weighted averages from the Cubic Spline to see if there is a significant difference in the level of offensive performance.

As noted above, 100 is always the league average OPS+. An OPS+ upwards of 120 is considered to be very good to great while below 80 is considered to be mediocre at best. An OPS+ above 150 is considered to be excellent, with only a few hitters having such an OPS+ in any given season. [6]

We will use the found areas from the Cubic Spline, truncated to integers 2267 and 2197. This was done as all spin rate data captured by Major League Baseball is found as an integer. Using each of these spin rates, we used the same query database used to capture our original data to find the OBP, SLG, OPS, OPS+, and Whiff% (a statistic that calculates the percentage of pitches that are swung at and missed) of all batters combined against pitches at each of our found spin rate levels in 2021.

As we can see from the table below, when the spin rate decreases from 2267 RPM to 2197 RPM the OBP, SLG, OPS, and OPS+ increase dramatically while the Whiff% decreases by 3%. Most importantly, let us look at the difference in OPS+. The hypothetical batter against all pitches at a spin rate of 2267 RPM in 2021 had an OPS+ of 102.26, just slightly above average, while the hypothetical batter against all pitches at a spin rate of 2197 RPM

Spin Rate (RPM)	OBP	SLG	OPS	Whiff%	OPS+
League Average	.314	.411	.725	25.9	100
2267	.315	.419	.734	26%	102.26
2197	.346	.500	.846	23%	131.85

Table 2: Comparison of OBP, SLG, OPS, and Whiff Rate of the league average, hitters against 2267 RPM, and hitters against 2197 RPM in 2021.

in 2021 had an OPS+ of 131.85, well above average. When referring back to the earlier statements of how good a value of OPS+ is, we can say that throwing a 2267 RPM pitch is similar to pitching to about a league-average hitter while throwing a 2197 RPM pitch is similar to pitching to a well above average hitter. This is a significant difference in the quality of the pitch and hitting outcomes based on the difference in spin rates found using Numerical Interpolation and Integration.

Going back to the original question we are looking to answer, the about 3% decrease in spin rate found around the time of Major League Baseball's crackdown on foreign substances was significant as the difference in the effectiveness of pitches at these two spin rates is significant.

Despite the results found, there are limitations to this analysis. Different pitchers will have their spin rates hover around different values, with each pitch type having its different typical spin rate as well. Another analysis to consider would be to go more in-depth controlling for the typical average spin rates of each of the players that pitched on any given day in the dataset, analyzing each pitcher's deviation from their typical average spin rate. Such an analysis would likely give a much better insight into whether Major League Baseball's enforcement of its foreign substance rules significantly impacted the average spin rate.

References

[1] Goff J.E. (The Conversation, 2021). *Sticky baseballs: Explaining the physics of the latest scandal in Major League Baseball*, Available at: `https://theconversation.com/sticky-baseballs-explaining-the-physics-of-the-latest-scandal-in-major-league-baseball-162415` [Accessed March 23, 2024]

[2] Katz J., Quealy K. and Kepner T. (The New York Times, 2021). *The Pitchers Whose Spin Rates Fell Most After a Crackdown on Sticky Substances*, Available at: `https://www.nytimes.com/interactive/2021/07/19/upshot/major-league-baseball-spin-rate-shift.html` [Accessed February 21, 2024]

[3] Janes C., Tran A.B., Samuels E., Bennett D., Cahlan S. and Taylor D. (The Washington Post, 2021). *How baseball's War on Sticky Stuff is Already Changing the Game*, Available at: `https://www.washingtonpost.com/sports/2021/07/02/sticky-stuff-baseball-data/` [Accessed March 6, 2024]

[4] Davis J. (2021). *Since the June 21 crackdown, pitchers like the Red Sox' Garrett Richards must be checked each inning for foreign substances on their hands, gloves, hats, and belts.* Available at `https://apps.bostonglobe.com/sports/graphics/2021/07/mlb-sticky-stuff/assets/mlb-foreign-substances.jpg` [Accessed April 1, 2024]

[5] Major League Baseball (2024) *Statcast*. Available at: `https://www.mlb.com/glossary/statcast` (Accessed April 3, 2024).

[6] StatCorner (2024) *What are OPS and OPS+?*. Available at: `https://statcorner.com/mlb/stats/ops-and-ops-plus` (Accessed June 12, 2024).

Appendix A Dataset

A.1. Data queried from Major League Baseball. (2024). *Statcast Search*, Available at: `https://baseballsavant.mlb.com/statcast_search` [Accessed March 6, 2024]

Appendix B Newton's Interpolating Polynomial

B.1.

$$\begin{aligned} p(x) = &- 8.075 \times 10^{-11}x^{18} - 1.243 \times 10^{-8}x^{17} + 8.772 \times 10^{-7}x^{16} - 3.763 \times 10^{-5}x^{15} + 0.001x^{14} \\ &- 0.023x^{13} + 0.355x^{12} - 4.150x^{11} + 36.724x^{10} - 245.719x^{9} + 1228.897x^{8} - 4487.919x^{7} \\ &+ 11465.778x^{6} - 18806.711x^{5} + 15605.017x^{4} + 1771.776x^{3} - 13711.272x^{2} + 7119.246x + 2302 \end{aligned} \tag{6}$$

Appendix C Cubic Spline Piecewise Polynomial

$$f(x) = \begin{cases} 3.599x^3 - 31.599x + 2302 & \text{, for } 0 \leq x \leq 1 \\ 8.007x^3 - 13.224x^2 - 18.375x + 2297.592 & \text{, for } 1 \leq x \leq 2 \\ -35.625x^3 + 248.565x^2 - 541.951x + 2646.643 & \text{, for } 2 \leq x \leq 3 \\ 53.493x^3 - 553.493x^2 + 1864.222x + 240.470 & \text{, for } 3 \leq x \leq 4 \\ -58.346x^3 + 788.572x^2 - 3504.037x + 7398.149 & \text{, for } 4 \leq x \leq 5 \\ 48.891x^3 - 819.989x^2 + 4538.768x - 6006.526 & \text{, for } 5 \leq x \leq 6 \\ -36.219x^3 + 712.006x^2 - 4653.206x + 12377.422 & \text{, for } 6 \leq x \leq 7 \\ 21.987x^3 - 510.321x^2 + 3903.088x - 7587.264 & \text{, for } 7 \leq x \leq 8 \\ 4.273x^3 - 85.199x^2 + 502.104x + 1482.024 & \text{, for } 8 \leq x \leq 9 \\ -27.079x^3 + 761.316x^2 - 7116.495x + 24337.824 & \text{, for } 9 \leq x \leq 10 \\ 35.0434x^3 - 1102.363x^2 + 11520.256x - 37784.681 & \text{, for } 10 \leq x \leq 11 \\ -33.094x^3 + 1146.183x^2 - 13213.750x + 52906.676 & \text{, for } 11 \leq x \leq 12 \\ 28.334x^3 - 1065.244x^2 + 13323.375x - 53241.8245 & \text{, for } 12 \leq x \leq 13 \\ -22.242x^3 + 907.235x^2 - 12318.841x + 57874.444 & \text{, for } 13 \leq x \leq 14 \\ 19.635x^3 - 851.596x^2 + 12304.788x - 57035.825 & \text{, for } 14 \leq x \leq 15 \\ -26.297x^3 + 1215.313x^2 - 18698.852x + 97982.373 & \text{, for } 15 \leq x \leq 16 \\ 26.551x^3 - 1321.408x^2 + 21888.682x - 118484.475 & \text{, for } 16 \leq x \leq 17 \\ -10.910x^3 + 589.159x^2 - 10590.948x + 65566.763 & \text{, for } 17 \leq x \leq 18 \end{cases}$$

Appendix D Python Code

```
#Project Stuff

import numpy as np
import sympy as sym
import matplotlib.pyplot as plt
x = sym.symbols('x')
# %%
####### Old Data
'''data = np.array([2282,2299,2275,2267,2279,2300,2265,2286,2245,
                    2217,2251,2201,2215,2225,2189,2210,2204,2177,
                    2193,2184,2210,2206,2200,2186,2182])

dates = np.array(['May 22-23','May 24-25','May 26-27','May 28-29','May 30-
                                        31','June 01-02',
                   'June 03-04','June 05-06','June 07-08','June 09-10',
                   'June 11-12','June 13-14','June 15-16','June 17-18','
                                                          June 19-20',
                   'June 21-22','June 23-24','June 25-26','June 27-28','
                                                          June 29-30',
                   'July 01-02','July 03-04','July 05-06','July 07-08','
                                                          July 09-10'])'''
####### New Data
data = np.array([2302,2274,2272,2296,2265,2299,2267,2270,2234,2215,
                 2225,2195,2205,2186,2196,2194,2210,2185,2188])
dates = np.array(['May 16-18','May 19-21','May 22-24','May 25-27','May 28-
                                        30',
```

```
                    '31-June 02','June 03-05','June 06-08','June 09-11',
                    'June 12-14','June 15-17','June 18-20','June 21-23',
                    'June 24-26','June 27-29','30-July 02','July 03-05',
                    'July 06-08','July 09-11'])
m = len(data)

# %%
plt.grid(True, ls = '--', color = 'gray', axis = 'x')
plt.scatter(np.arange(0,len(data),1),data,c = 'navy')
plt.title('League-Wide Spin Rate Around Enforcement of Foreign Substance
                                        Rules')
plt.annotate('June 03: Rumors Around MLB of Enforcement Imminent',xy = (6,
                                        2265),
             xytext=(6.5,2290),fontsize = 8, arrowprops=dict(facecolor='
                                                    black', width=0.1,
                                                    headwidth=1,shrink=0.05)
                                                    )
plt.annotate('June 15: Official Announcement from MLB', xy = (10,2225),
             xytext=(10.5,2260),fontsize = 8, arrowprops=dict(facecolor='
                                                    black', width=0.1,
                                                    headwidth=1,shrink=0.05)
                                                    )
plt.annotate('June 21: Enforcement Begins',xy=(12,2205),xytext=(12.5,2240)
                                        ,
             fontsize = 8, arrowprops=dict(facecolor='black', width=0.1,
                                                    headwidth=1,shrink=0.05)
                                                    )
plt.xticks(np.arange(0,len(data)),dates)
plt.xticks(fontsize=8, rotation=60)
plt.xlim(-1,m+1)
plt.ylim(2160,2320)
plt.ylabel('Spin Rate (rotations per minute)')
plt.xlabel('Date (2021)')
plt.savefig("scatter.svg", format = 'svg', dpi=300)
plt.show()

#%% Divided Differences
def myDD(xs,ys):
    x = sym.symbols('x')
    n = len(xs)
    C = np.zeros(shape=(len(xs),len(xs)))
    C[:,0]=ys
    for j in np.arange(1,np.size(xs)):
        for i in np.arange(0,np.size(xs)-j):
            C[i,j]=(C[i+1,j-1]-C[i,j-1])/(xs[i+j]-xs[i])

    p = 0*x
    for i in np.arange(0,len(xs)):
        prod = x**0
        for j in np.arange(0,i):
            prod = prod*(x-xs[j])

        p = p + C[0,i]*prod
```

```
    for i in np.arange(0,len(xs)):
        for j in np.arange(0,len(xs)):
            if i+j>=len(xs):
                C[i,j]=np.nan

    return C, p

poly = myDD(np.arange(0,len(data),1),data)[1]
X = np.linspace(0,m,500)
Y = [poly.subs(x,val) for val in X]

print('Min:',min(Y),'Max:',max(Y))

plt.plot(X,Y)
plt.scatter(np.arange(0,m,1),data,c = 'navy')
plt.grid(True, ls = '--', color = 'gray', axis = 'x')
plt.xlim(-1,m+1)
plt.ylim(2160,2320)
plt.title('League-Wide Spin Rate Around Enforcement of Foreign Substance
                                        Rules')
'''plt.annotate('June 03: Rumors Around MLB of Enforcement Imminent',xy =
                                        (1,2265),
             xytext=(1.5,2290),fontsize = 8, arrowprops=dict(facecolor='
                                                black', width=0.1,
                                                headwidth=1,shrink=0.05)
                                                )
plt.annotate('June 15: Official Announcement from MLB', xy = (7,2215),
             xytext=(7.5,2260),fontsize = 8, arrowprops=dict(facecolor='
                                                black', width=0.1,
                                                headwidth=1,shrink=0.05)
                                                )
plt.annotate('June 21: Enforcement Begins',xy=(10,2210),xytext=(10.5,2240)
                                        ,
             fontsize = 8, arrowprops=dict(facecolor='black', width=0.1,
                                                headwidth=1,shrink=0.05)
                                                )'''
plt.xticks(np.arange(0,m),dates)
plt.xticks(fontsize=8, rotation=60)
plt.ylabel('Spin Rate (rotations per minute)')
plt.xlabel('Date')
plt.savefig("poly.svg", format = 'svg', dpi=300)
plt.show()

# %% ############ Cubic Spline ########################

#input myCubicSpline(np.arange(0,len(xs),1),data)
def myCubicSpline(xs,ys):
    n = len(xs)

    x,a,b,c,d = sym.symbols('x a b c d')
    a = sym.symbols('a:%d'%(n-1)) #creates a0,a1,...,an-1
    b = sym.symbols('b:%d'%(n-1))
    c = sym.symbols('c:%d'%(n-1))
    d = sym.symbols('d:%d'%(n-1))
```

```
    equations = np.array([])
    sn_arr = np.array([])
    for i in np.arange(0,n-1,1):
        sn_arr = np.append(sn_arr,a[i]+b[i]*x+c[i]*x**2+d[i]*x**3)

    for i in np.arange(0,len(xs)-1):
        initl = (sn_arr[i]-ys[i]).subs(x,xs[i])
        conns = (sn_arr[i]-ys[i+1]).subs(x,xs[i+1])

        if i<(len(xs)-2):
            smooth = sym.diff(sn_arr[i],x,1).subs(x,xs[i+1])-sym.diff(
                                                    sn_arr[i+1],x,1).subs(x,
                                                    xs[i+1])
            doub_smooth = sym.diff(sn_arr[i],x,2).subs(x,xs[i+1])-sym.diff
                                                    (sn_arr[i+1],x,2).subs(x,
                                                    xs[i+1])

        equations = np.append(equations,[initl,conns,smooth,doub_smooth])

    #Normal equations
    equations = np.append(equations, sym.diff(sn_arr[0],x,2).subs(x,xs[0])
                                            )
    equations = np.append(equations, sym.diff(sn_arr[-1],x,2).subs(x,xs[-1
                                            ]))

    Sols = sym.solve(equations,dict = True)

    sol_arr = np.array([])
    for i in np.arange(0,n-1,1):
        sol_arr = np.append(sol_arr,Sols[0][a[i]]+Sols[0][b[i]]*x+Sols[0][
                                            c[i]]*x**2+Sols[0][d[i]]*x**3
                                            )

    return sol_arr

spline_funcs = myCubicSpline(np.arange(0,m,1),data)

# %% Plotting spline functions
x = sym.symbols('x')
xs = np.arange(0,len(data))

func = 0*x

func = sym.Piecewise((func,False),(spline_funcs[0],x<=xs[1]))

for i in np.arange(1,len(xs)-2,1):
    func = sym.Piecewise((func,x<=xs[i]),(spline_funcs[i],x<=xs[i]+1))

func = sym.Piecewise((func,x<=xs[-1]-1),(spline_funcs[-1],x>xs[-1]-1))

X = np.linspace(0,m-1,500)
Y = [func.subs(x,val) for val in X]
```

```
plt.plot(X,Y)
plt.xlim(-1,m+1)
plt.ylim(2160,2320)
plt.grid(True, ls = '--', color = 'gray', axis = 'x')
plt.title('League-Wide Spin Rate Around Enforcement of Foreign Substance
                                        Rules')
'''plt.annotate('June 03: Rumors Around MLB of Enforcement Imminent',xy =
                                        (1,2265),
             xytext=(1.5,2290),fontsize = 8, arrowprops=dict(facecolor='
                                                black', width=0.1,
                                                headwidth=1,shrink=0.05)
                                                )
plt.annotate('June 15: Official Announcement from MLB', xy = (7,2215),
             xytext=(7.5,2260),fontsize = 8, arrowprops=dict(facecolor='
                                                black', width=0.1,
                                                headwidth=1,shrink=0.05)
                                                )
plt.annotate('June 21: Enforcement Begins',xy=(10,2210),xytext=(10.5,2240)
                                        ,
             fontsize = 8, arrowprops=dict(facecolor='black', width=0.1,
                                                headwidth=1,shrink=0.05)
                                                )'''

plt.xticks(np.arange(0,m),dates)
plt.xticks(fontsize=8, rotation=60)
plt.ylabel('Spin Rate (rotations per minute)')
plt.xlabel('Date')
plt.scatter(np.arange(0,len(data),1),data,c = 'navy')
plt.savefig("spline.svg", format = 'svg', dpi=300)
plt.show()

# %% Close-up
plt.plot(X,Y)
plt.xlim(9.9,10.1)
plt.ylim(2210,2235)
plt.grid(True, ls = '--', color = 'gray', axis = 'x')
plt.title('League-Wide Spin Rate Around Enforcement of Foreign Substance
                                        Rules')

plt.xticks(np.arange(10,11),dates[10:11])
#plt.xticks(fontsize=8, rotation=60)
plt.ylabel('Spin Rate (rotations per minute)')
plt.xlabel('Date')
plt.scatter(np.arange(0,len(data),1),data,c = 'navy')
plt.savefig("smoothness.svg", format = 'svg', dpi=300)
plt.show()
# %% Composite Trapezoidal Rule

def CTR(f,I,n):
    x = sym.symbols('x')

    a = I[0]
    b = I[1]
```

```
    h = (b-a)/n

    xs = np.linspace(a,b,n+1)

    ys = np.array([])
    for j in np.arange(0,len(xs)):
        ys = np.append(ys,f.subs(x,xs[j]))

    foo = 0
    goo = 0

    for k in np.arange(1,len(xs)-1):
        goo = goo+ys[k]

    foo = (ys[0]+ys[-1]+2*goo)*h/2

    return foo

n_s = 100

area_spline = CTR(func,np.array([0,m]),n_s)
print(area_spline)
print(area_spline/len(data))

area_1_spline = CTR(func,np.array([0,m/2]),n_s)
#print(area_1_spline/10)
area_2_spline = CTR(func,np.array([m/2,m]),n_s)
#print(area_2_spline/9)

area_1_spline_norm = area_1_spline/(m/2)
area_2_spline_norm = area_2_spline/(m/2)

perc_dec = 1-(area_2_spline_norm/area_1_spline_norm)
print(perc_dec)


# %% Romberg Integration

def myRomberg(f,I,M):
    x = sym.symbols('x')

    a = I[0]
    b = I[1]

    R0 = np.array([])

    for i in np.arange(0,M+1,1):
        xs = np.linspace(a,b,(2**i-1)+2)

        h = (b-a)/(2**i)

        ys = np.array([])
        for j in np.arange(0,len(xs)):
            if f.subs(x,xs[j])==sym.nan:
```

```
                ys = np.append(ys,sym.limit(f,x,xs[j]))
            if f.subs(x,xs[j])!=sym.nan:
                ys = np.append(ys,f.subs(x,xs[j]))

        foo = 0
        goo = 0

        for k in np.arange(1,len(xs)-1):
            goo = goo+ys[k]

        foo = (ys[0]+ys[-1]+2*goo)*h/2

        R0 = np.append(R0,foo)

    R = np.zeros(shape = (M+1,M+1))
    R[:,0]=R0

    for j in np.arange(1,M+1,1):
        for i in np.arange(j,M+1,1):
            R[i,j]=(4**j)/(4**j-1)*R[i,j-1]-(R[i-1,j-1])/(4**j-1)

    for i in np.arange(0,M+1,1):
        for j in np.arange(0,M+1,1):
            if j > i:
                R[i,j]=np.nan

    return R

area_romberg = myRomberg(poly,np.array([0,m]),7)[-1,-1]
print(area_romberg)
print(area_romberg/len(data))

area_1r = myRomberg(poly,np.array([3,(m-1)/2]),7)[-1,-1]
area_2r = myRomberg(poly,np.array([(m-1)/2,(m-1)-3]),7)[-1,-1]

print(area_1r/((m-1)/2-3))
print(area_2r/((m-1)/2-3))

perc_diff_rom = 1-(area_2r/((m-1)/2-3))/(area_1r/((m-1)/2-3))
print(perc_diff_rom)

#%% Analysis Plots

plt.plot(X,Y)
plt.xlim(-1,m+1)
plt.ylim(2100,2310)
plt.title('League-Wide Spin Rate Around Enforcement of Foreign Substance
                                        Rules')
'''plt.annotate('June 03: Rumors Around MLB of Enforcement Imminent',xy =
                                        (1,2265),
             xytext=(1.5,2290),fontsize = 8, arrowprops=dict(facecolor='
                                                black', width=0.1,
                                                headwidth=1,shrink=0.05)
                                                )
```

```
plt.annotate('June 15: Official Announcement from MLB', xy = (7,2215),
             xytext=(7.5,2260),fontsize = 8, arrowprops=dict(facecolor='
                                                 black', width=0.1,
                                                 headwidth=1,shrink=0.05)
                                                 )
plt.annotate('June 21: Enforcement Begins',xy=(10,2210),xytext=(10.5,2240)
                                              ,
             fontsize = 8, arrowprops=dict(facecolor='black', width=0.1,
                                                 headwidth=1,shrink=0.05)
                                                 )'''

plt.xticks(np.arange(0,len(data)),dates)
plt.xticks(fontsize=8, rotation=60)
plt.ylabel('Spin Rate (rotations per minute)')
plt.xlabel('Date')
plt.scatter(np.arange(0,len(data),1),data,c = 'navy')

plt.fill_between(
        x= X.astype(np.float64),
        y1= np.asarray(Y).astype(np.float64),
        where= (0 < X)&(X < 9),
        color= "b",
        alpha= 0.2)

plt.fill_between(
        x= X.astype(np.float64),
        y1= np.asarray(Y).astype(np.float64),
        where= (9 < X)&(X < 18),
        color= "r",
        alpha= 0.2)

plt.savefig("area_pre.svg", format = 'svg', dpi=300)

plt.annotate('2267.529',xy=(2.25,2200),xytext=(2.25,2200),fontsize=12)
plt.annotate('2197.886',xy=(11.5,2150),xytext=(11.5,2150),fontsize=12)

plt.savefig("area_post.svg", format = 'svg', dpi=300)

plt.show()

#%% Differential Equation

r = -50

x = sym.symbols('x')
t = sym.symbols('t')

pp = lambda t,p: -r*p*(1-(p/(max(data)+10)))*(1-(p/(min(data)-10)))
Ivl = np.array([0,len(data)])
h = 0.1

#RKMO4

def myRKMO4(x_prime,t0,x0,I,h):
```

```
    a, b = I[0], I[1]
    N = int(abs((b-a)/h))
    #print(N)

    #need to make sure x0 corresponds with correct t
    if t0 == a:
        ts = np.linspace(a,b,N+1)

    if t0 == b:
        ts = np.linspace(b,a,N+1)

    #print(ts)
    ws = np.array([])
    ws = np.append(ws,x0)

    for i in np.arange(1,N+1,1):
        k1 = h*x_prime(ts[i-1],ws[i-1])
        k2 = h*x_prime(ts[i-1]+h/2 , ws[i-1]+k1/2)
        k3 = h*x_prime(ts[i-1]+h/2 , ws[i-1]+k2/2)
        k4 = h*x_prime(ts[i-1]+h , ws[i-1]+k3)
        new_w = ws[i-1]+(1/6)*(k1+2*k2+2*k3+k4)
        ws = np.append(ws,new_w)

    if t0 == a:
        return ws

    if t0 == b:
        return np.flip(ws) #reverse order


ans = myRKM04(pp,0,data[0],Ivl,h)

plt.grid(True, ls = '--', color = 'gray', axis = 'x')
plt.scatter(np.arange(0,len(data),1),data,c = 'navy')
plt.title('League-Wide Spin Rate Around Enforcement of Foreign Substance
                                    Rules')
plt.annotate('June 03: Rumors Around MLB of Enforcement Imminent',xy = (6,
                                    2265),
             xytext=(6.5,2290),fontsize = 8, arrowprops=dict(facecolor='
                                                    black', width=0.1,
                                                    headwidth=1,shrink=0.05)
                                                    )
plt.annotate('June 15: Official Announcement from MLB', xy = (10,2225),
             xytext=(10.5,2260),fontsize = 8, arrowprops=dict(facecolor='
                                                    black', width=0.1,
                                                    headwidth=1,shrink=0.05)
                                                    )
plt.annotate('June 21: Enforcement Begins',xy=(12,2205),xytext=(12.5,2240)
                                    ,
             fontsize = 8, arrowprops=dict(facecolor='black', width=0.1,
                                                    headwidth=1,shrink=0.05)
                                                    )
plt.xticks(np.arange(0,len(data)),dates)
plt.xticks(fontsize=8, rotation=60)
```

```
plt.xlim(-1,m+1)
plt.ylim(2160,2320)
plt.ylabel('Spin Rate (rotations per minute)')
plt.xlabel('Date (2021)')
plt.plot(ans)
plt.savefig("logistic.svg", format = 'svg', dpi=300)
plt.show()
```