Building A Road To The Playoffs:

Simulating the 2015-16 NHL Season

Benjamin Tanen
MATH-0087 Final Project
May 5, 2016

Abstract

Sports fans, betters, and analysts have all dreamt of predicting the future outcome of the big game, or even better, the upcoming season. So far, no model has been 100% accurate but many have come close. However, many of these models simply describe odds of a particular outcome instead of a team's journey to said outcome. The model described here attempts to fill this void by producing season-long simulations for the 30 NHL teams in an attempt to show how these teams progress towards playoff spots. By using past team performance (broken down by goal probability distributions) and the Monte Carlo method, the model generates game-by-game (and from this season-long) simulations, which can be used to analyze a team's expected performance and see how this might differ as the season progresses. The results of this model can also be compared against the actual performance of a team to analyze and pinpoint critical turning points in the season that would cause difficulties for this and other predictive simulations.

1 Introduction

April 10, 2016 marked the end of the regular season portion of the 2015-16 NHL season. At this point, the 16 teams that were to play in the playoffs had been set in stone. Some teams barely clinched playoff berths in the last days of the regular season while other teams earned spots weeks prior. A notable example is the Washington Capitals who ended the regular season with 120-points and who clinched a playoff spot on March 15 (a fairly remarkable feat). It was clear only a few weeks into the regular season that the Capitals would do very well and a betting man likely would have put money on them making the playoffs.

All betting aside, it is interesting to consider if it is possible to take the Capitals' (or any NHL team's) early season performance and use it to project their potential future performance. In the context of a whole season, we can use this to map out each team's "path to the playoffs" (or perhaps the impossibility of them making the playoffs). In order to accurately answer this question, given the complexity of a 30-team league, mapping one particular team's path requires the modeling of the entire league through the entirety of the season, which is where mathematical modeling and computation comes into play. This is the question that is addressed in this project.

2 Problem

The intention of this project is to construct a simulation for the 2015-16 NHL season in order to analyze the successes (and failures) of such a simulation in projecting any particular team's path to the playoffs.

2.1 Making the Stanley Cup Playoffs: A Brief Background

Before tackling a model that would simulate the performance of a particular season, it is important to understand how any particular team might succeed (or fail) in placing in the Stanley Cup Playoffs.

First, consider any regular season game with two teams, team A and team B. If team A wins in the game in regular time (the first 60-minutes of play), they receive 2-points towards their standings and team B receives 0-points. However, if the game is tied after the first 60-minutes, the game goes to sudden-death overtime (and eventually a shootout). As with the regulation time, if team A then goes on to win in the overtime

period or shootout, they still receive 2-points. However, team B also receives 1-point towards their standings because of the loss in OT.

At the end of the 1230 games played throughout the regular season, the teams to make the playoffs are determined as follows:

- A playoff spot is given to the top 3 teams from each of the four divisions in the NHL (Metropolitan, Atlantic, Central, and Pacific)
- Of the remaining 18 teams, the top two teams (not already selected) from the Eastern and Western Conferences are also given wildcard playoff spots

This results in 16 total teams in the playoffs. A team is said to clinch a playoff spot when it is determined that even if they lose every game for the rest of the season (and their competitors win all the games that matter), they will still rank in a position to gain a spot in the playoffs.

2.2 Building the Model

The primary premise of this project is to build a model that can simulate the behavior of the NHL season (the remaining games for the 30-teams) given a small sample of the league's performance. In order to do this, the model will need to take in some "known" game date (our sample data) and use it to construct metrics for the performance of each team. In this model, this metric will be the probability of scoring a certain number of goals[1]. Using this metric, each game will have to be simulated to return a particular result (ex: NYR win over WAS in OT) in order to award the correct number of points after each game. Using these points and how they change throughout the season, the model will be able to show how accurate the simulation is. If it is not accurate, analyzing the real vs. simulated points should indicate inaccuracies in the model that can be further analyzed for understanding.

In order to improve upon the accuracy of the predictive model, the simulation will be run many times, using the Monte Carlo method to converge on the expected behavior of the league (based on the metric used). By iterating over each season many times, we can normalize the results of the games / season to compensate for potential outliers in our results (i.e massive upsets, landslide victories).

Another important question to be answered is how early the simulation can start and still produce an accurate prediction. In order to do this, the model will include a variable simulation start date. By changing this simulation date, we can attempt to find where enough information is consumed to produce an accurate prediction of the season / outcome.

3 Literature Review

The concept of projecting the performance of a sports team is not a novel one and has been done by many people before. The purpose of this project is to take and combine the successful components of various different projections and use them to accurately model each team's path to the Stanley Cup Playoffs.

3.1 The NY Times' Road to the White House

The first article / projection of note comes from the NY Times' "The Cold Hard Math of How Trump Can Win." In this article, the authors attempt to map out the paths of each Republican candidate through the 2016 primaries given certain winning percentages for certain time periods. For example, if Donald Trump were to on average win 44% of the votes up until March 6 (as he did) and he were to on average win 32% of the votes in primaries after March 6, he would clinch the Republican nomination in late May. However, change these winning percentages and the road to the nomination changes very much for the candidates.

[1]This metric can be changed / improved upon. See **Improving the Model** for more.

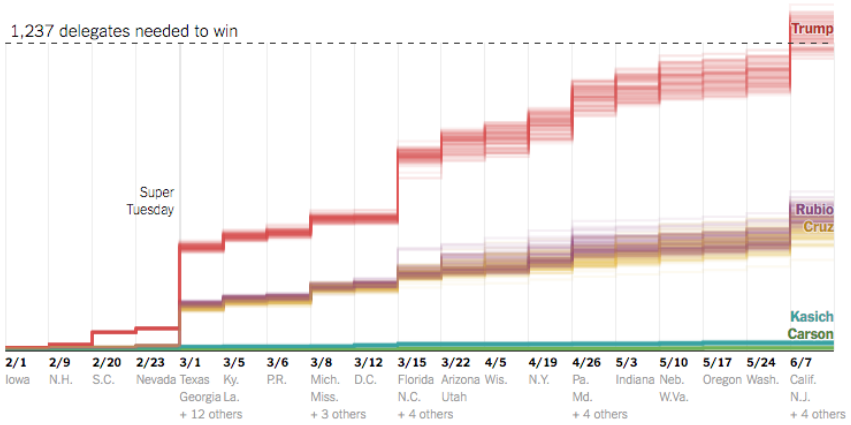
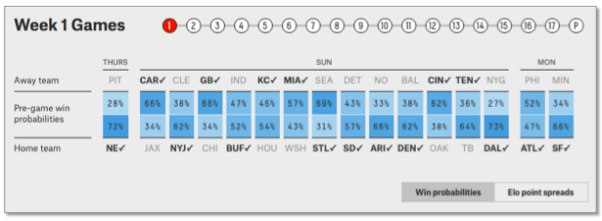


Figure A: Simulations of GOP Candidates Winning the Nomination

This style of projection is notable because of the paths it gives to the individual candidates, mapping their progression towards the nomination. This differs from other projections (see below) because many other articles, particularly those referring to sports projections, simply assign odds or probabilities of one team beating another. By using this style of projection, it makes it possible to analyze how much a projection veers away from reality beyond simply saying "was it right?" For this reason, this is the style of projection used here, first to provide more detailed analysis of the "road to the playoffs" and also for the opportunity to analyze the effectiveness and accuracy of the projections with greater detail.

3.2 FiveThirtyEight's ELO Rating System

For another example of projections with more application to sports, we can look at FiveThirtyEight's use of the ELO Ratings for the 2014-15 and 2015-16 NFL seasons. In this series of articles, the writers at FiveThirtyEight used the ELO rating system to make projected odds for the outcomes of certain games and the odds of a particular team to make the playoffs. In the first image below, we see the odds FiveThirtyEight gave for each game during Week 1 of the season. We can also see a table referencing the odds of a particular team of making the playoffs, winning the Super Bowl, etc.



ELO RATING	1-WEEK CHANGE	TEAM	DIVISION	AVG. SIMULATED SEASON WINS	LOSSES	POINT DIFF.	PLAYOFF ODDS MAKE PLAYOFFS	WIN DIVISION	FIRST ROUND BYE	WIN SUPER BOWL
1763	+8	New England 10-0	AFC East	14.8	1.2	+206.6	>99%	>99%	97%	33%
1669	+8	Carolina 10-0	NFC South	14.3	1.7	+153.5	>99%	98%	96%	18%
1655	+11	Arizona 8-2	NFC West	12.0	4.0	+154.7	97%	90%	60%	11%
1651	+32	Green Bay 7-3	NFC North	11.2	4.8	+92.0	95%	75%	33%	9%
1637	+7	Denver 8-2	AFC West	11.9	4.1	+72.1	98%	89%	50%	8%
1630	-11	Cincinnati 8-2	AFC North	12.1	3.9	+118.3	99%	89%	48%	8%

Figure B: FiveThirtyEight NFL Projections Using the ELO System

This projection style is of note because it uses a lot of data to compute the odds for a particular team. The ELO rating system gives an ELO rating to each team and updates this score after every game they play. For example, if team A (the #1 ranked team) is playing team B (the last place team) and team A wins, team A's ELO rating will go up by a small amount since this result is expected. However, if team B were to beat team A, team B's ELO rating would go up by a large amount in order to reflect the upset. By keeping track of these ELO ratings at a continuous basis, it allows projections to include much more information (recent and historical wins, upsets, etc.) in their projected odds.

I personally believe the ELO rating system is one of the best options to project the outcome of a single game. However, in order to accurately assign a rating for any particular team, FiveThirtyEight (and comparable outlets) incorporates the results from all games played by a team ever. So this means in order to compute the ELO rating of the Washington Capitals today, we would need to go back and compute the ELO rating

for every day between October 9, 1974 (their first game ever) to today. This requires a decent amount of data parsing and computation, especially since the NHL started back in 1917. For this reason, generating ELO ratings for the 30 NHL teams is out of the scope of this project, at least for now (see **Improving the Model** for more).

4 Methods

Given the methods used by other outlets tackling similar problems, this particular model primarily utilizes the Monte Carlo method for its simulation. Using the goal probability distributions (which describes the probability of a particular team scoring x goals) for each team prior to a set date, the outcome of each game after this set date is determined using random sampling of these distributions. This is repeated for each game of the season many times to obtain the expected results of the simulated season. For more information, see below.

4.1 Obtaining Data

Before any simulation can occur, it is necessary to obtain the data used in the model. More specifically, the model works off of existing and future season schedules of the year that is being simulated (in this paper, the 2015-2016 season is simulated but any other season should work). All of this data can be found on the website http://www.hockey-reference.com/ under their **Season Summary** tab[2].

At this point in the process, the data from Hockey-Reference.com must be converted to a MATLAB friendly format. Prior to this transition, the game schedule data comes in the following format:

Table A: Sample Game Data from Hockey-Reference.com

Date	Visitor	G	Home	G	Att.	LOG	Notes
2015-10-07	Vancouver Canucks	5	Calgary Flames	1	19289	152	

This is then translated into a number-based format. Examples of this format can be seen in **Appendix A**. In this format, teams are referred to as identifying numbers. A reference for this information (as well as team division and conference) can be seen in **Appendix B**.

4.2 Initializing Model

With the properly parsed data, the next step is to initialize the model and its important variables. These include the simulation start data `sim_date`, all of the game data `games`, and the unplayed and played ranges (based on what `sim_date` is set to). `sim_date` specifies what existing real data the model can / will use in order to make its simulations. For example, if `sim_date` is set to 10/15/2015 (only a few days into the 2015-16 season) the model will only be able to use 57 games in order to simulate the remaining 1173 games. Thus, we can see that the value of `sim_date` will greatly affect the accuracy of the model (for a further analysis of this effect, see the **Results** section). For games that occur after `sim_date`, the `games` matrix will be cleared and these games will be part of the `unplayed_rng`.

Another important part at the start of this model is the pulling and generation of the actual point accumulations over the course of the season being simulated. This matrix of real data `real_points` is used as a comparison between the simulated points matrix to measure the accuracy of the simulation (see below for more).

The code associated with initialization can be see the lines 1-24 of `simulate_nhl_season.m` (**Appendix F**).

[2] http://www.hockey-reference.com/leagues/NHL_2016_games.html

4.3 Generating Probability Distributions

The next crucial part of the model is the calculation / generation of the probability distributions used to determine the outcome of each game. This model is based on the goals scored by each team of a particular matchup. In order to accurately simulate this, the probabilities of a team scoring x number of goals is calculated based on the games in the played range (games before `sim_date`).

In order to do this, the function `calculate_goal_pcts.m` is used for each of the 30 teams (see **Appendix F** for the code). This function returns two components used later in the simulation: an array of probabilities for scoring goals in regular time and the probability of scoring goals in overtime[3]. These probabilities are generated by simply calculating the number of times i ($0 \leq i \leq 14$) goals were scored by a particular team in the games that have been played (all games before `sim_date`) and then dividing by the total number of games played.

The same methodology is used for OT goals, except that only one goal can be scored in overtime due to sudden death rules[4]. It is also important to note that for particular ranges of games, some teams might not have played any overtime games. In this case, the function `calculate_goal_pcts.m` returns `OT_goal_pcts` set to 0. This slightly disadvantages these teams because it might unfairly result in a loss for a team that is particularly strong in OT that has simply yet to play in OT. This should be noted when analyzing simulations with early start dates.

The goal probability distributions are calculated for each of the 30 teams are calculated and stored the matrix `goal_pcts`, which is used below.

4.4 Game and Season Simulations

Using the goal probability distributions for two teams, we can then use random variables to simulate the results of a single game. The code used for a particular game comes from `simulate_nhl_season.m` which also uses the function `calculate_game_score.m` (see **Appendix F**).

In order to simulate the results of a particular game, two random variables are generated with MATLAB's `rand()` and used to randomly sample the goals scored by team A and team B using their respective goal probability distributions. For example, consider a matchup between the New York Rangers and the Washington Capitals. We can use $f_{NYR}(x)$ and f_{WAS}, the goal probability distributions of these two teams, to obtain expected goals scored by each team in this matchup. By sampling these two distributions repeatedly, we will obtain $E_{NYR}(x)$ and $E_{WAS}(x)$, which can be used to project the outcome of this matchup (if $E_{NYR}(x) > E_{WAS}(x)$, we expect the Rangers will win). This is the process used to simulate the regular time of a particular matchup. However, if the game is tied after this, an overtime period is simulated by awarding an additional goal to the team with the better overtime probability. Using these results, the `games` matrix is then updated accordingly.

In order to simulate an entire season, this operation is repeated for each of the games that occur after `sim_date`. This model works primarily using the Monte Carlo method so the entire season is simulated many times for more accurate results. For the results discussed here, 100 iterations were run for each season but more or less iterations could be used (depending on the desire for accuracy vs. speed). For the code used to simulate many seasons, see lines 35-58 of `simulate_nhl_season.m` in **Appendix F**.

[3] In this particular implementation, the probabilities include those for 0 to 14 goals scored. This range allows for the maximum number of goals scored for the past 5 seasons. However, the maximum number of goals ever scored in a game was 16 by the Montreal Canadiens in 1920. This should be noted for any seasons being simulated to ensure accuracy.

[4] A game ending in sudden death OT or a shootout are considered to be the same. This is because they reward the same number of points to each team which is all that is necessary for the season simulation.

4.5 Generate Point Matrices

Each team's standings (and thus their place in the playoffs) is determined based on their respective points accumulated throughout the season. Therefore, the points accumulated by each of the 30-teams is calculated for each season based on the results of the games played.

In order to calculate this, the function `generate_points_matrix.m` is used (see **Appendix F**). This function simply iterates over all of the games played and awards 2 points to a team for winning and 0 points to a team for losing (or 1 point if they lost in overtime). The result is a matrix of each day and the points each team had on the date.

This points matrix is calculated after each season iteration and these are then averaged to obtain `avg_points`, the primary result of the model. This information is then compared against the actual points for the season (found in `real_points`) for analysis of accuracy of the model (see below).

5 Results

After running the simulation, the two primary blocks of information come from our two matrices: `avg_points` and `real_points`. Using these two, it's possible to compare the two and see how much the model (given the start date `sim_date`) veers away from the actual season results. We can also plot these two data series to obtain graphic representation of the model vs. actuality.

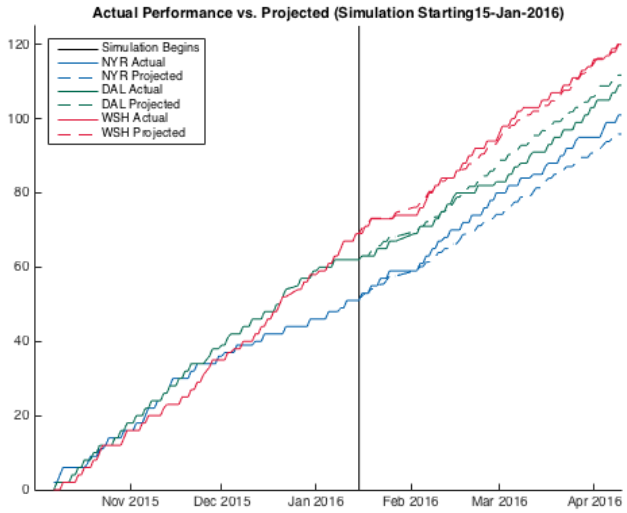


Figure C: Example Simulation vs. Actual Output

5.1 Error Analysis

To measure how accurate the modeled results were, we can analyze the differentials between the `avg_points` and `real_points` at the end of the season and throughout the course of the season in order to draw some conclusions on what worked and what did not. We can do this sort of analysis on individual teams or the league as a whole.

Table B: Example Point Differential (Simulation started 12/15/16)

	Max Point Differential	Mean Point Differential	End Point Differential
League	-	3.52	10.55
MTL	-26.17	-9.347	-24.85
WSH	11.29	4.96	8.39

For example, consider a simulation that started on 12/15/2016 (results above). In this simulation, the Washington Capitals seemed to perform fairly accurately, differing on average by only 4.96 points (or about 2.5 games) and by 8.39 points at the end of the simulation. On the other hand, in the same simulation, we can see that the Montreal Canadiens were projected quite poorly, differing on average by more than 9 points and ending with a differential of almost -25 points. This analysis allows us to pinpoint which teams were simulated well (relative to their actual performance) and it enables us to look deeper at the teams that were not.

5.2 Does More Data Make a Better Simulation?

Before we go into analysis of specific teams, we should first consider how the model may or may not improve with the inclusion of more data. We expect, as with all models, that as we work with more data (more game results), the accuracy of the model would improve. However, we should consider how much improvement there actually is in the model with the inclusion of more and more data.

Table C: Model Accuracy Improvements With Increasing Simulation Start

Simulation Start	Mean Overall Point Differential (Abs)	Mean Final Point Differential (Abs)	Largest Underestimated Point Differential	Largest Overestimated Point Differential
10/15/15	12.2912	24.9853	83.55 (ANA)	-44.62 (MIN)
11/15/15	5.3025	12.8393	42.26 (ANA)	-31.24 (MTL)
12/15/15	3.5442	10.6043	36.17 (ANA)	-24.32 (MTL)
01/15/16	1.9456	7.3101	28.12 (ANA)	-15.13 (ARI)
02/15/16	0.9983	4.9753	14.49 (ANA)	-8.35 (CGY)
03/15/16	0.4667	0.2646	8.23 (PIT)	-7.02 (BOS)

We can see that as time goes on, our metrics of accuracy converge towards 0 indicating an overall improvement in the model's accuracy. In fact, it appears that, as the simulation date is pushed back, the model continues to improve fairly uniformly (decreasing in error by about 45% on average each time), right up until the last simulation. While fairly expected, it is interesting to note that with each later simulation, relatively less new information is provided each time (one month of new data is 100% more information in October and roughly 20% more information in February). If new data meant better simulation, we would expect a comparable improvement in the model for the amount of data provided. However, the improvement seems to be consistent, and thus less directly tied to how much new data is provided.

This can likely be explained by what is happening when more data is provided. While a later simulation date will inherently produce more accurate and encompassing goal distributions (and thus performance expectations), a later simulation date also leaves less time for a teams performance to drastically differ, thus ruining the chance of a good predictive model. The later the simulation date is, the more we can ensure that the performance of the team we are attempting to simulate is inherently consistent, which provides a major benefit for this model (see below for more discussion).

5.3 The Good Teams

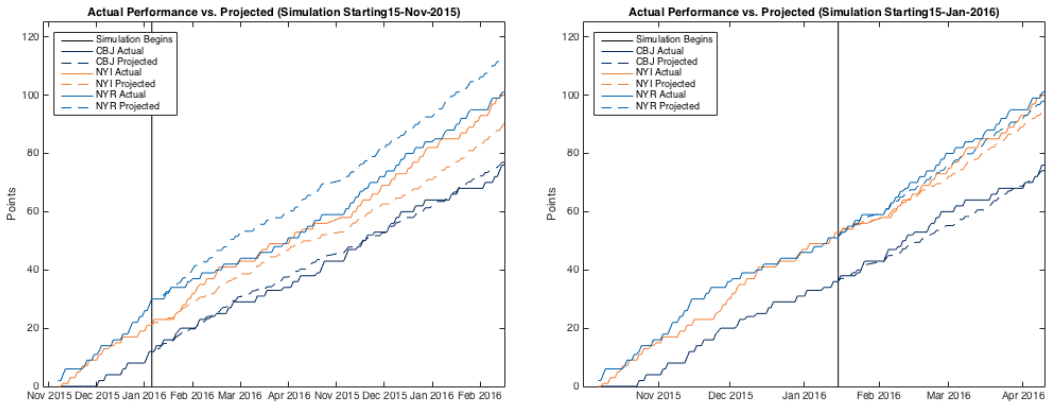
Given this analysis, it is interesting to see for which teams the simulation worked the best. In order to see this, we can consider multiple simulations with different starting dates and see which teams were consistently accurate. To find these, we can look at the teams with the lowest average final point differential over multiple different simulation start dates. This information can be seen in **Appendix E**. From this, we can see the top three most accurate simulations come for the Columbus Blue Jackets, the New York Islanders, and the New York Rangers (in order).

From **Table C** (below), we can see that these three teams all started with fairly accurate simulations and simply converged towards the real results. The question is, why do these teams in particular have better performing simulations?

Table C: Simulation Accuracy for CBJ, NYI, NYR (Various Simulation Dates)

	CBJ Mean Diff.	CBJ Final Diff.	NYI Mean Diff.	NYI Final Diff.	NYR Mean Diff.	NYR Final Diff.
10/15/15	-0.704213483	-2.58	1.05247191	1.87	6.702191011	9.19
11/15/15	-0.308089888	-1.26	3.693258427	9.08	-6.737359551	-12.57
12/15/15	2.049325843	5.41	-0.337191011	1.94	-1.182808989	-0.51
1/15/16	0.944157303	0.94	0.782078652	2.94	1.393764045	4.38
2/15/16	-0.346573034	-2.57	0.751853933	2.86	0.090786517	0.09
3/15/16	-0.356235955	-1.83	-0.21011236	-0.47	0.164325843	1.12

By looking at plots of the simulated seasons vs. actual season (below), we can see that the simulated season generally have much more linear behavior when compared to actual season performance. This stems from the fact of how the performance of a team is determined in either situation. In the model, the performance of a particular team one day will be very similar to that of another day because their "performance" is very closely related to their expected goals scored. Thus, if the New York Rangers were to play the Washington Capitals one day into the simulation and 100 days into the simulation, we would expect fairly similar results. However, actual hockey teams will go through cycles of good and bad performance. This makes it so that the first NYR-WAS matchup might end in a 1-4 score while the next might result in a 5-2 score[5].



For this reason, we would expect the best simulations to occur for teams that are fairly consistent in their performance. If a particular team has a consistent performance, we would therefore expect them to have a fairly linear path throughout the season, since they are not going through cyclical hot and / or cold streaks. To confirm this theory, we can compare the actual season results for our three top teams to linear regressions for their results. By using MATLAB's `fitlm()`, we get the following:

Table D: Testing for Linearity of CBJ, NYI, NYR

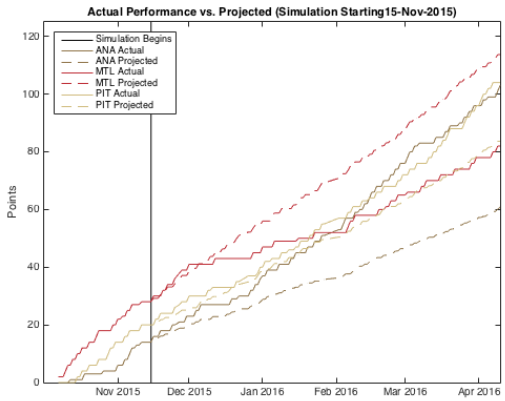
	CBJ	NYI	NYR
R^2	0.997	0.996	0.993

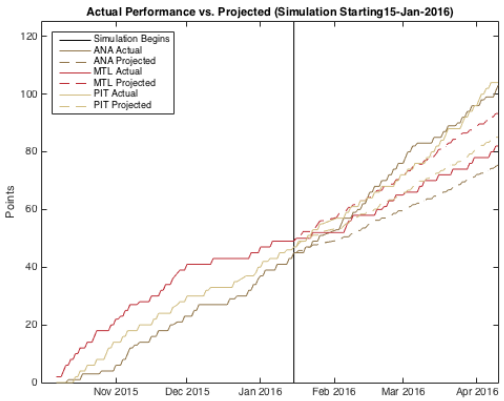
From this, we can see that, while not completely linear, these teams do have almost completely linear (and therefore fairly consistent) performances throughout the season. This would therefore confirm why these teams are able to be simulated with such consistent accuracy, even after only a few weeks of games, given their fairly linear (consistent) performances.

5.4 The Bad Teams

To further test the theory of linearity, we can look at the other end of the spectrum and analyze the poorly simulated teams. Again using **Appendix E**, we can see that the consistently poorest simulations come from the Anaheim Ducks, the Pittsburg Penguins, and the Montreal Canadiens.

[5]These are actual results from the 2015-16 season





We can again check the linearity of these teams' performance. We then get the following:

Table E: Testing for Linearity of ANA, PIT, MTL

	ANA	PIT	MTL
R^2	0.960	0.969	0.894

From this, we actually can see that the Montreal Canadiens appear to have a fair less linear performance than either Anaheim or Pittsburg, but they (on average) had a better simulation. By looking at the plot of real points scored by Montreal, we can see that there was a decent change in performance from before December 2015 to after. In fact, this transition from good to bad was a very note phenomena in the Canadiens' 2015-16 season. The Canadiens went on a very hot streak early on in the season, winning 19 of their first 26 games, placing them at the top of the league. After that, from their 27^{th} game (which took place on December 3, 2015) and on, they went 14-28-3 and became one of the worst teams in the league. No other team had such a sudden transition from good to bad and we can see that this transition would very much throw off a linear regression, resulting in a low R^2 value.

To compensate for this, we can look at two linear regressions for our worst three teams, one from the start of the season to December 1 and one from December 3 to the end of the season. If we do this, we get the following:

Table F: Testing for Linearity of ANA, PIT, MTL (Extended)

	ANA (Before Dec 1)	ANA (After Dec 3)	PIT (Before Dec 1)	PIT (After Dec 3)	MTL (Before Dec 1)	MTL (After Dec 3)
R^2	0.962	0.965	0.971	0.973	0.972	0.975

First, we can see that our regression for Montreal drastically improves when we factor in their sudden performance change. We can also see that, when compensating for Montreal's performance change, our R^2 values now indicate the decreased linearity we would expected from the three poorest simulated teams (in order) when comparing it to the significantly higher R^2 values of CBJ, NYI, and NYR.

From all of this (the good and the bad), the accuracy of this model (regardless of simulation date) appears to be very closely tied to the overall consistency (linearity) of the team's performance throughout the season. Thus, when using this model, we should take this into consideration and perhaps, in the future, consider adding some metric to compensate for performance changes (see below).

6 Conclusion

Based on the results above, we can see that this model is fairly successful at tackling the prediction problem. After only two months of play (with roughly 60% of the season still unplayed), this model is able to simulate the performance of the 30-team NHL within an average error of roughly 5-games, a fairly remarkable feat.

However, as we've seen, it is not a perfect model, particularly when we account for the inevitable inclusion of cyclical team performance. Thus, we can view this model as a stepping stone towards successfully predicting such a complex problem. With a solid base model, the next steps come down to how the model is improved and then how it is applied.

6.1 Improving the Model

As with any simulation, this model certainly has room for improvement in a number of ways.

The first, and perhaps the most obvious improvement to any model, is the inclusion of more information. This can be done in a number of ways, beyond simply adding more pre-simulation game data (see discussion above). As discussed in **Literature Review**, the ELO Rating System (when implemented correctly) can add a significant amount of insight to the future performance of a team since it can be built to encapsulate many different features of any particular game. In the model that FiveThirtyEight uses for their NFL model, projections utilize information like location of game, who is favored, current streaks, and much more to make a more robust simulation for the game. Implementing a similar system into this model would likely have the greatest improvement on the accuracy of the simulation, simply by adding more features to analyze in the data. For example, possible factors to include in a projection (beyond those already mentioned) could include: goal against averages (for the team as a whole or for the goalie in net), if the game is an in-division or out-of-division game, and key player injuries. By including and utilizing this information, this model might improve in its weaknesses by compensating for faults (like cyclical team performance).

Another major improvement that could be used in this model is the inclusion of rolling goal probability distributions. Currently, goal probability distributions are calculated for each team at the start of the simulation. This results in simulating fairly linear performance since their is no sense of hot or cold streaks. However, if a rolling window of goal probability distributions (of perhaps 15 - 30 games) were added, a teams performance would continuously compensate based on prior good performance (via the model). However, one thing to be careful of when including rolling goal distributions in the possibility of positive feedback. If implemented incorrectly, there is a very real possibility that a streak that should be only two or three games could result in a season-long winning streak, simply because the positive performance continuously feeds back into itself. Thus, if / when implementing this, we must make sure to watch for any sort of inappropriate feedback.

6.2 Applying the Model

The most obvious application of this project is in predicting the outcome of future NHL seasons. Given the results previously discussed, this model has its strengths and weaknesses, and as with any predictive model, its results should be taken with a grain of salt. However, this model has the potential to serve as a useful tool for anyone who benefits from understanding the outcome of the NHL. For example, consider one of the NHL organizations who is making trade decisions. By using this model, a team can understand where their weaknesses lie and how they can fill these gaps by potentially acquiring new talent. This is a growing field in sports and by including a model like this, teams might greatly benefit.

However, this model is not simply limited to uses in hockey. While this model was built with the intention of modeling the NHL specifically, there are many other dynamic multi-threaded systems that operate in a very similar ways. Thus, it is quite feasible that a model like the one described here can be restructured to meet the needs of countless other dynamic systems. By doing this, mathematicians, scientists, and data analysts can potentially have vast insight into how these systems interact internally and, at a greater level, how these systems interact with each other.

7 References

(1) "The Cold Hard Math of How Trump Can Win, and How Rubio Can Stop Him?, The New York Times. http://www.nytimes.com/interactive/2016/02/27/upshot/republican-delegate-calculator-how-trump-can-win.html

(2) "Introducing the Elo Ratings", FiveThirtyEight. http://fivethirtyeight.com/datalab/introducing-nfl-elo-ratings/

8 Appendix

1. **Appendix A**: Sample Game Data
2. **Appendix B**: Team References and Point Differentials
3. **Appendix C**: Average Point Projections for Select Teams (Simulation Starting 12/15)
4. **Appendix D**: Simulation Improvement Progression (10/15 to 03/15)
5. **Appendix E**: End Point Differential with Changing Simulation Start Date
6. **Appendix F**: MATLAB Code

Appendix A: Sample Game Data

Game Number	Date Year	Date Month	Date Day	Team A ID	Team A Goals	Team A Win	Team B ID	Team B Goals	Team B Win	Overtime?
1	2015	10	7	28	5	1	5	1	0	0
2	2015	10	7	20	3	1	7	2	0	0
3	2015	10	7	24	5	1	14	1	0	0
4	2015	10	7	16	3	1	27	1	0	0
5	2015	10	8	30	6	1	3	2	0	0
6	2015	10	8	21	3	1	4	1	0	0
7	2015	10	8	15	5	1	8	4	0	0
8	2015	10	8	23	0	0	10	3	1	0
9	2015	10	8	6	1	0	17	2	1	0
10	2015	10	8	12	1	0	25	3	1	0
11	2015	10	8	22	2	0	26	3	1	1
12	2015	10	9	20	4	1	9	2	0	0
13	2015	10	9	27	0	0	11	4	1	0
14	2015	10	9	2	4	1	14	1	0	0
15	2015	10	9	30	3	1	18	1	0	0
16	2015	10	9	7	3	1	19	2	0	1
17	2015	10	10	23	1	0	2	2	1	0
18	2015	10	10	16	4	1	3	2	0	0
19	2015	10	10	26	4	1	4	1	0	0
20	2015	10	10	11	4	1	6	3	0	0
21	2015	10	10	19	1	0	7	4	1	0
22	2015	10	10	10	3	0	8	6	1	0
23	2015	10	10	22	1	0	13	7	1	0
24	2015	10	10	25	2	0	15	3	1	0
25	2015	10	10	12	0	0	17	2	1	0
26	2015	10	10	9	2	0	20	5	1	0
27	2015	10	10	1	0	0	24	2	1	0
28	2015	10	10	21	5	1	27	4	0	1
29	2015	10	10	5	3	1	28	2	0	1
30	2015	10	10	18	3	0	29	5	1	0
31	2015	10	11	16	3	1	21	1	0	0
32	2015	10	12	28	2	1	1	1	0	1
33	2015	10	12	26	6	1	3	3	0	0
34	2015	10	12	9	2	0	4	4	1	0
35	2015	10	12	30	2	0	19	4	1	0
36	2015	10	12	13	0	0	22	1	1	0
37	2015	10	13	13	4	1	6	1	0	0
38	2015	10	13	25	4	1	5	3	0	0
39	2015	10	13	12	2	0	10	4	1	0
40	2015	10	13	26	1	0	11	3	1	0
41	2015	10	13	28	3	1	14	0	0	0
42	2015	10	13	17	3	1	18	1	0	0
43	2015	10	13	30	4	1	20	1	0	0
44	2015	10	13	16	3	1	23	2	0	0
45	2015	10	13	24	5	1	29	0	0	0
46	2015	10	14	2	4	1	1	0	0	0
47	2015	10	14	21	7	1	9	3	0	0
48	2015	10	14	3	6	1	8	2	0	0
49	2015	10	14	7	0	0	22	3	1	0
50	2015	10	15	15	4	1	2	3	0	0
51	2015	10	15	25	4	1	12	2	0	0
52	2015	10	15	4	2	0	13	3	1	0
53	2015	10	15	20	0	0	16	3	1	0
54	2015	10	15	17	3	0	19	4	1	0
55	2015	10	15	21	0	0	23	2	1	0

Appendix B: Team References and Point Differentials

NOTE: This results below come from a simulation starting on 12/15/2015

Team	Team ID	Conference	Division	Final Points Actual	Final Points Projected (Avg.)	Differential
ANA	1	Western	Pacific	103	67.35	35.65
ARI	2	Western	Pacific	78	91.41	-13.41
BOS	3	Eastern	Atlantic	93	105.01	-12.01
BUF	4	Eastern	Atlantic	81	77.21	3.79
CGY	5	Western	Pacific	77	89.9	-12.9
CAR	6	Eastern	Metropolitan	86	80.98	5.02
CHI	7	Western	Central	103	93.27	9.73
COL	8	Western	Central	82	87.63	-5.63
CBJ	9	Eastern	Metropolitan	76	70.51	5.49
DAL	10	Western	Central	109	120.92	-11.92
DET	11	Eastern	Atlantic	93	92.88	0.12
EDM	12	Western	Pacific	70	88.83	-18.83
FLA	13	Eastern	Atlantic	103	87.26	15.74
LAK	14	Western	Pacific	102	97.8	4.2
MIN	15	Western	Central	87	93.76	-6.76
MTL	16	Eastern	Atlantic	82	106.76	-24.76
NSH	17	Western	Central	96	84.45	11.55
NJD	18	Eastern	Metropolitan	84	90.1	-6.1
NYI	19	Eastern	Metropolitan	100	99.74	0.26
NYR	20	Eastern	Metropolitan	101	100.7	0.3
OTT	21	Eastern	Atlantic	85	101.32	-16.32
PHI	22	Eastern	Metropolitan	96	77.71	18.29
PIT	23	Eastern	Metropolitan	104	89.49	14.51
SJS	24	Western	Pacific	98	88.19	9.81
STL	25	Western	Central	107	91.91	15.09
TBL	26	Eastern	Atlantic	97	86.85	10.15
TOR	27	Eastern	Atlantic	69	76.28	-7.28
VAN	28	Western	Pacific	75	80.3	-5.3
WSH	29	Eastern	Metropolitan	120	110.27	9.73
WPG	30	Western	Central	78	86.19	-8.19

Appendix C: Average Point Projections for Select Teams (Simulation Starting 12/15)

NOTE: The horizontal-line at 12/15 indicates the start of the simulation.

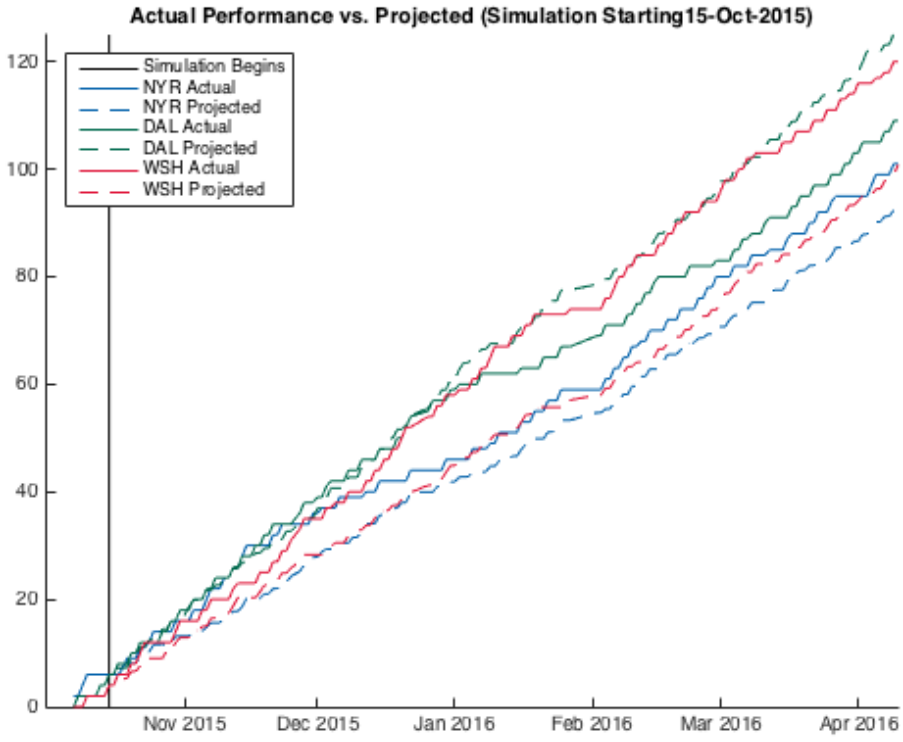
Dates	ANA Points (Proj.)	ANA Points (Real)	DAL Points (Proj.)	DAL Points (Real)	MTL Points (Proj.)	MTL Points (Real)	NYR Points (Proj.)	NYR Points (Real)	WAS Points (Proj.)	WAS Points (Real)
10/07/15	0	0	0	0	2	2	2	2	0	0
10/08/15	0	0	2	2	2	2	2	2	0	0
10/09/15	0	0	2	2	2	2	4	4	0	0
10/10/15	0	0	2	2	4	4	6	6	2	2
10/11/15	0	0	2	2	6	6	6	6	2	2
10/12/15	1	1	2	2	6	6	6	6	2	2
10/13/15	1	1	4	4	8	8	6	6	2	2
10/14/15	1	1	4	4	8	8	6	6	2	2
10/15/15	1	1	6	6	10	10	6	6	4	4
10/16/15	1	1	6	6	10	10	6	6	4	4
10/17/15	1	1	8	8	12	12	6	6	6	6
10/18/15	3	3	8	8	12	12	7	7	6	6
10/19/15	3	3	8	8	12	12	9	9	6	6
10/20/15	3	3	10	10	14	14	9	9	8	8
10/21/15	3	3	10	10	14	14	9	9	8	8
10/22/15	3	3	12	12	14	14	11	11	10	10
10/23/15	3	3	12	12	16	16	11	11	12	12
10/24/15	3	3	12	12	18	18	12	12	12	12
10/25/15	3	3	12	12	18	18	14	14	12	12
10/26/15	4	4	12	12	18	18	14	14	12	12
10/27/15	4	4	14	14	18	18	14	14	12	12
10/28/15	4	4	14	14	18	18	14	14	12	12
10/29/15	4	4	16	16	18	18	14	14	12	12
10/30/15	4	4	16	16	20	20	16	16	14	14
10/31/15	4	4	18	18	20	20	16	16	16	16
11/01/15	6	6	18	18	22	22	16	16	16	16
11/02/15	6	6	18	18	22	22	16	16	16	16
11/03/15	6	6	20	20	23	23	18	18	16	16
11/04/15	8	8	20	20	23	23	18	18	16	16
11/05/15	8	8	20	20	25	25	18	18	18	18
11/06/15	10	10	22	22	25	25	20	20	18	18
11/07/15	12	12	22	22	27	27	22	22	20	20
11/08/15	12	12	24	24	27	27	22	22	20	20
11/09/15	13	13	24	24	27	27	22	22	20	20
11/10/15	13	13	24	24	27	27	24	24	20	20
11/11/15	14	14	24	24	28	28	24	24	20	20
11/12/15	14	14	26	26	28	28	26	26	22	22
11/13/15	14	14	26	26	28	28	26	26	23	23
11/14/15	14	14	28	28	28	28	28	28	23	23
11/15/15	14	14	28	28	28	28	30	30	23	23
11/16/15	16	16	28	28	30	30	30	30	23	23
11/17/15	16	16	30	30	30	30	30	30	23	23
11/18/15	16	16	30	30	30	30	30	30	25	25
11/19/15	18	18	32	32	30	30	30	30	25	25
11/20/15	18	18	32	32	32	32	30	30	25	25
11/21/15	18	18	34	34	32	32	32	32	27	27
11/22/15	18	18	34	34	34	34	32	32	27	27
11/23/15	18	18	34	34	34	34	34	34	29	29
11/24/15	20	20	34	34	34	34	34	34	29	29
11/25/15	20	20	34	34	36	36	34	34	31	31
11/27/15	21	21	36	36	38	38	34	34	33	33
11/28/15	21	21	38	38	39	39	34	34	35	35
11/29/15	21	21	38	38	39	39	34	34	35	35

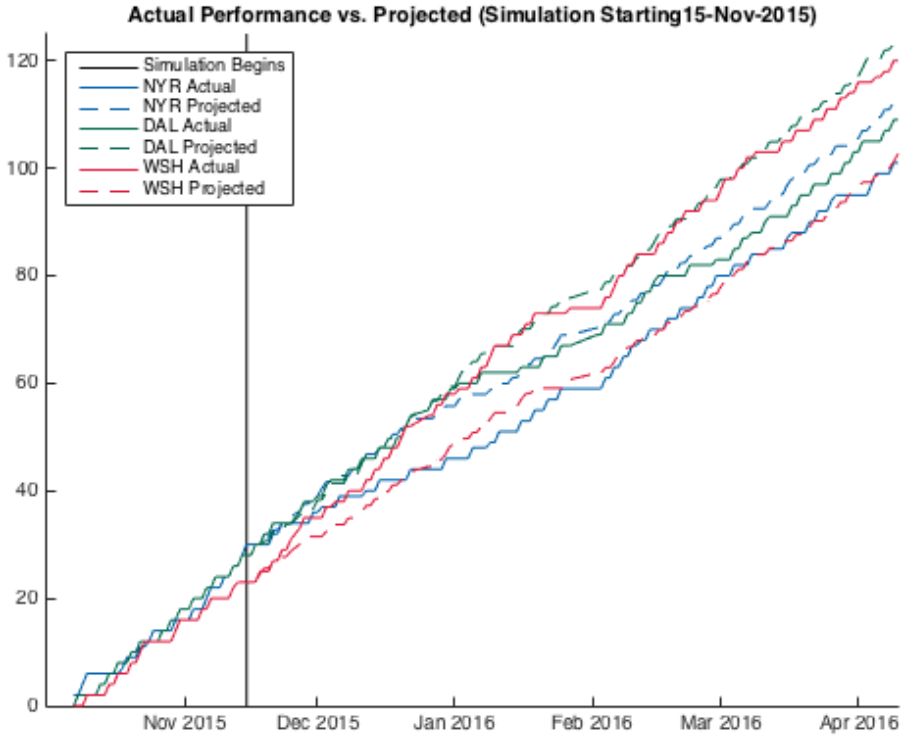
11/30/15	23	23	38	38	39	39	36	36	35	35
12/01/15	23	23	39	39	41	41	36	36	35	35
12/02/15	23	23	39	39	41	41	37	37	35	35
12/03/15	23	23	41	41	41	41	37	37	37	37
12/04/15	25	25	42	42	41	41	37	37	37	37
12/05/15	25	25	42	42	41	41	37	37	38	38
12/06/15	27	27	42	42	41	41	39	39	38	38
12/07/15	27	27	42	42	41	41	39	39	38	38
12/08/15	27	27	44	44	41	41	39	39	40	40
12/09/15	27	27	44	44	41	41	39	39	40	40
12/10/15	27	27	44	44	41	41	39	39	40	40
12/11/15	27	27	46	46	41	41	39	39	40	40
12/12/15	27	27	46	46	43	43	40	40	42	42
12/13/15	27	27	46	46	43	43	40	40	42	42
12/14/15	27	27	46	46	43	43	40	40	44	44
12/15/15	27	27	48	48	43	43	42	42	44	44
12/16/15	27	27	48	48	43	43	42	42	45.14	46
12/17/15	27.79	27	49.4	48	44.48	43	43.16	42	45.14	46
12/18/15	27.79	27	49.4	48	44.48	43	44.38	42	46.5	48
12/19/15	28.71	29	50.6	50	45.39	43	44.38	42	46.5	48
12/20/15	28.71	29	50.6	50	45.39	43	45.35	42	47.72	50
12/21/15	29.48	29	52.04	52	46.61	43	45.35	42	49.16	52
12/22/15	30.14	30	53.5	54	48.07	43	46.83	44	49.16	52
12/26/15	30.14	30	55.2	55	49.16	43	46.83	44	50.22	54
12/27/15	31.17	32	56.74	57	49.16	43	46.83	44	50.22	54
12/28/15	31.17	32	56.74	57	50.4	45	48.11	44	51.6	56
12/29/15	32.03	34	58.4	57	51.51	45	48.11	44	51.6	56
12/30/15	32.03	34	58.4	57	51.51	45	49.32	46	53.14	58
12/31/15	32.79	36	59.8	59	51.51	45	49.32	46	54.4	58
01/01/16	33.77	37	59.8	59	52.65	47	49.32	46	54.4	58
01/02/16	33.77	37	61.4	60	52.65	47	50.5	46	55.74	59
01/03/16	34.55	39	62.9	60	52.65	47	50.5	46	55.74	59
01/04/16	34.55	39	62.9	60	52.65	47	50.5	46	55.74	59
01/05/16	34.55	39	64.26	60	54.03	47	51.34	48	56.74	61
01/06/16	35.16	39	64.26	60	55.34	49	51.34	48	56.74	61
01/07/16	35.16	39	65.58	62	55.34	49	51.34	48	58.14	63
01/08/16	35.88	41	65.58	62	55.34	49	51.34	48	58.14	63
01/09/16	35.88	41	67	62	56.58	49	52.43	49	59.22	65
01/10/16	36.76	41	67	62	56.58	49	52.43	49	60.44	67
01/11/16	36.76	41	67	62	56.58	49	53.21	51	60.44	67
01/12/16	36.76	41	67	62	56.58	49	53.21	51	60.44	67
01/13/16	37.45	43	67	62	56.58	49	53.21	51	60.44	67
01/14/16	37.45	43	67	62	57.67	49	54.19	51	61.56	69
01/15/16	38	45	68.64	62	57.67	49	54.19	51	61.56	69
01/16/16	38	45	70.07	63	58.89	50	55.42	53	62.76	69
01/17/16	38.81	45	70.07	63	60.02	50	56.47	53	63.84	71
01/18/16	38.81	45	70.07	63	60.02	50	56.47	53	63.84	71
01/19/16	38.81	45	71.56	63	61.24	50	57.77	55	65.2	73
01/20/16	39.61	47	71.56	63	61.24	50	57.77	55	65.2	73
01/21/16	39.61	47	72.87	65	61.24	50	57.77	55	65.2	73
01/22/16	39.61	47	72.87	65	61.24	50	58.89	57	65.2	73
01/23/16	40.48	49	74.27	65	62.66	52	58.89	57	65.2	73
01/24/16	40.48	49	74.27	65	62.66	52	59.89	57	65.2	73
01/25/16	40.48	49	75.78	67	64.02	52	61.2	59	65.2	73
01/26/16	40.97	51	75.78	67	65.44	52	61.2	59	65.2	73
01/27/16	40.97	51	75.78	67	65.44	52	61.2	59	66.8	74
02/02/16	41.84	53	77.36	69	66.85	52	62.41	59	68.3	74
02/03/16	41.84	53	77.36	69	68.27	52	62.41	59	68.3	74
02/04/16	42.62	55	78.76	71	68.27	52	63.49	61	69.54	76

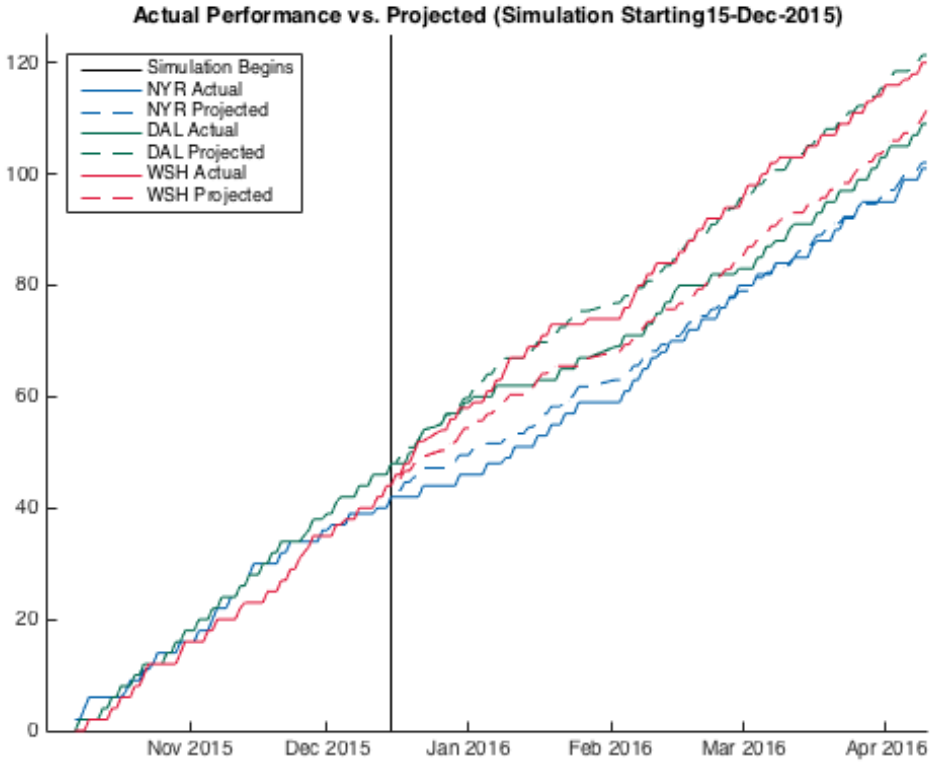
02/05/16	43.25	57	78.76	71	68.27	52	63.49	61	69.54	76
02/06/16	43.25	57	80.26	71	69.44	54	64.81	63	70.74	78
02/07/16	43.25	57	80.26	71	70.6	56	64.81	63	72.24	80
02/08/16	44.08	57	80.26	71	70.6	56	65.89	65	72.24	80
02/09/16	44.9	59	81.76	73	71.78	58	65.89	65	73.48	82
02/10/16	44.9	59	81.76	73	71.78	58	67.09	67	73.48	82
02/11/16	45.74	60	83.4	75	71.78	58	67.09	67	74.6	84
02/12/16	45.74	60	83.4	75	72.85	58	68.26	68	74.6	84
02/13/16	46.48	62	84.74	77	72.85	58	68.26	68	75.42	84
02/14/16	46.48	62	84.74	77	72.85	58	69.52	70	75.42	84
02/15/16	47.37	64	86.16	79	73.96	58	69.52	70	75.42	84
02/16/16	48.27	66	87.62	80	73.96	58	69.52	70	76.45	86
02/17/16	48.27	66	87.62	80	75.08	58	70.63	70	76.45	86
02/18/16	49.13	68	89.22	80	75.08	58	72.15	72	77.65	88
02/19/16	49.13	68	89.22	80	76.55	60	72.15	72	77.65	88
02/20/16	49.13	68	90.6	80	76.55	60	72.15	72	79.13	90
02/21/16	49.9	70	90.6	80	76.55	60	73.47	74	79.13	90
02/22/16	49.9	70	90.6	80	77.77	61	73.47	74	80.45	92
02/23/16	49.9	70	91.96	82	77.77	61	74.75	74	80.45	92
02/24/16	50.59	72	91.96	82	78.93	63	74.75	74	81.43	92
02/25/16	50.59	72	93.4	82	78.93	63	75.89	76	81.43	92
02/26/16	51.28	74	93.4	82	78.93	63	75.89	76	82.71	94
02/27/16	51.28	74	94.68	82	80.31	65	76.82	78	82.71	94
02/28/16	52.08	76	94.68	82	80.31	65	76.82	78	83.93	94
02/29/16	52.08	76	96.14	83	81.54	65	78.24	80	83.93	94
03/01/16	52.08	76	97.6	83	81.54	65	78.24	80	85.23	96
03/02/16	52.64	78	97.6	83	83.12	66	78.24	80	86.77	98
03/03/16	53.31	80	97.6	83	84.37	66	79.49	80	86.77	98
03/04/16	53.31	80	99.24	85	84.37	66	80.55	82	87.91	98
03/05/16	54.03	82	99.24	85	85.63	66	80.55	82	88.83	100
03/06/16	54.03	82	100.6	87	85.63	66	81.65	82	88.83	100
03/07/16	54.97	83	100.6	87	85.63	66	81.65	82	90.05	102
03/08/16	54.97	83	101.96	88	86.48	68	83.03	84	90.05	102
03/09/16	55.71	83	101.96	88	86.48	68	83.03	84	91.24	103
03/10/16	55.71	83	101.96	88	87.83	70	83.03	84	91.24	103
03/11/16	56.31	83	103.38	90	87.83	70	83.03	84	91.24	103
03/12/16	56.31	83	104.8	91	89.09	70	84.24	85	92.28	103
03/13/16	56.31	83	104.8	91	89.09	70	85.41	85	92.28	103
03/14/16	57.09	85	104.8	91	89.09	70	85.41	85	92.28	103
03/15/16	57.09	85	106.06	91	90.22	70	85.41	85	93.54	105
03/16/16	57.76	85	106.06	91	91.59	72	86.85	87	93.54	105
03/17/16	57.76	85	107.62	93	91.59	72	87.99	88	93.54	105
03/18/16	58.3	87	107.62	93	91.59	72	87.99	88	94.6	107
03/19/16	58.3	87	109.1	95	92.68	72	89.22	88	94.6	107
03/20/16	59.18	89	109.1	95	93.84	72	89.22	88	96.14	107
03/21/16	59.18	89	109.1	95	93.84	72	90.43	90	96.14	107
03/22/16	59.71	89	110.66	97	95.46	74	90.43	90	97.12	109
03/23/16	59.71	89	110.66	97	95.46	74	91.45	92	97.12	109
03/24/16	60.55	90	112.1	97	96.84	74	91.45	92	97.12	109
03/25/16	60.55	90	112.1	97	96.84	74	91.45	92	98.68	111
03/26/16	61.27	92	113.18	99	97.94	74	92.52	94	99.98	111
03/27/16	61.27	92	113.18	99	97.94	74	93.83	95	99.98	111
03/28/16	62.13	94	113.18	99	97.94	74	93.83	95	101.36	113
03/29/16	62.13	94	114.48	101	99.15	76	93.83	95	101.36	113
03/30/16	62.83	96	114.48	101	99.15	76	93.83	95	102.78	114
03/31/16	62.83	96	116.04	103	100.42	78	94.96	95	102.78	114
04/01/16	63.81	96	116.04	103	100.42	78	94.96	95	103.92	116
04/02/16	63.81	96	117.5	105	101.73	78	96.24	95	105.1	116
04/03/16	64.44	98	119.04	105	101.73	78	96.24	95	105.1	116

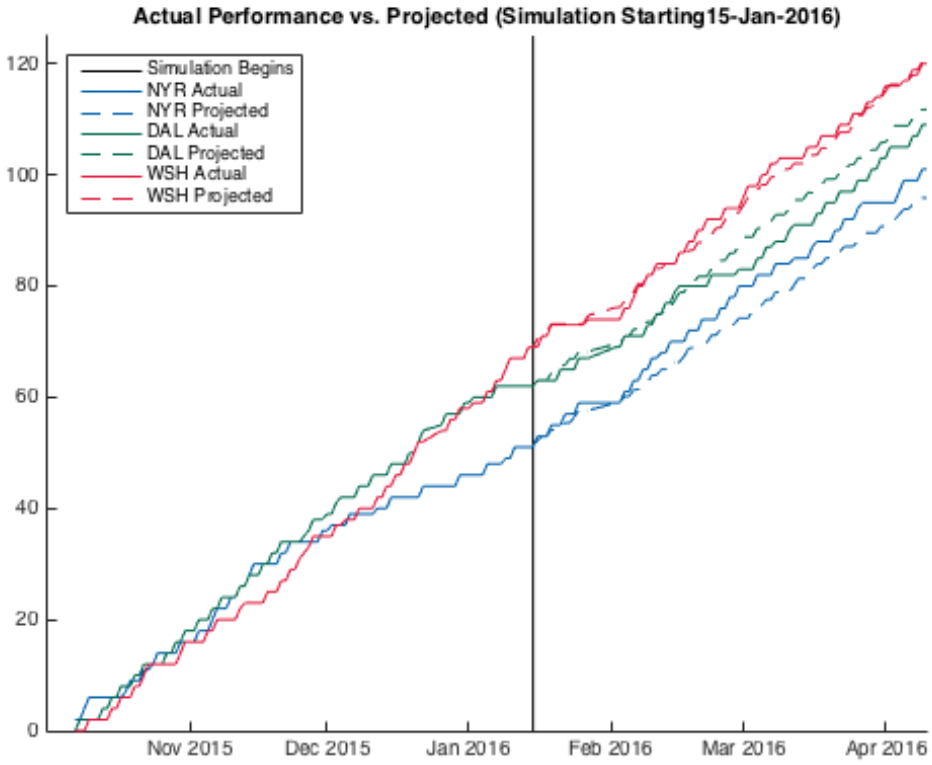
04/04/16	64.44	98	119.04	105	101.73	78	97.52	97	105.1	116
04/05/16	65.08	99	119.04	105	103	78	98.85	99	106.42	117
04/06/16	65.08	99	119.04	105	103	78	98.85	99	106.42	117
04/07/16	65.9	99	120.54	107	104.12	80	99.89	99	107.84	118
04/08/16	65.9	99	120.54	107	104.12	80	99.89	99	107.84	118
04/09/16	66.66	101	122	109	105.49	82	101.02	101	109.18	120
04/10/16	67.44	103	122	109	105.49	82	101.02	101	110.52	120

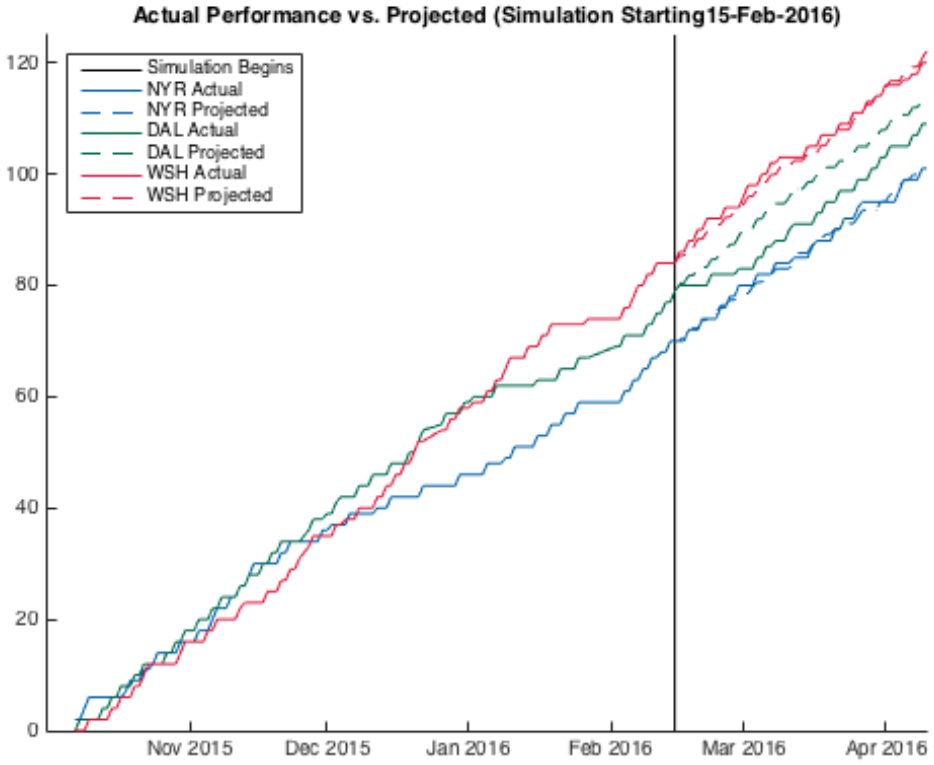
Appendix D: Simulation Improvement Progression (10/15 to 03/16)

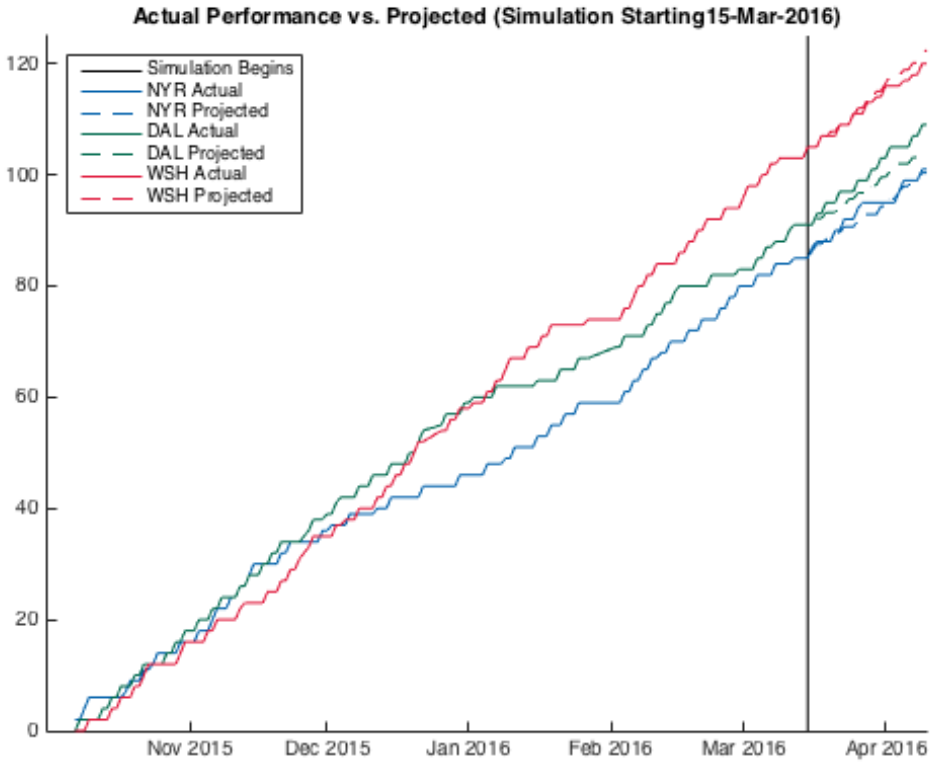





Actual Performance vs. Projected (Simulation Starting15-Dec-2015)
Simulation Begins
NYR Actual
NYR Projected
DAL Actual
DAL Projected
WSH Actual
WSH Projected
120
100
80
60
40
20
0
Nov 2015
Dec 2015
Jan 2016
Feb 2016
Mar 2016
Apr 2016


Actual Performance vs. Projected (Simulation Starting15-Jan-2016)
Simulation Begins
NYR Actual
NYR Projected
DAL Actual
DAL Projected
WSH Actual
WSH Projected
120
100
80
60
40
20
0
Nov 2015
Dec 2015
Jan 2016
Feb 2016
Mar 2016
Apr 2016


Actual Performance vs. Projected (Simulation Starting15-Feb-2016)
Simulation Begins
NYR Actual
NYR Projected
DAL Actual
DAL Projected
WSH Actual
WSH Projected
120
100
80
60
40
20
0
Nov 2015
Dec 2015
Jan 2016
Feb 2016
Mar 2016
Apr 2016


Actual Performance vs. Projected (Simulation Starting15-Mar-2016)
Simulation Begins
NYR Actual
NYR Projected
DAL Actual
DAL Projected
WSH Actual
WSH Projected
120
100
80
60
40
20
0
Nov 2015
Dec 2015
Jan 2016
Feb 2016
Mar 2016
Apr 2016

Appendix E: End Point Differential with Changing Simulation Start Date

Team	10/15/15	11/15/15	12/15/15	01/15/16	02/15/16	03/15/16	Average (Abs.)	Rank
ANA	83.82	44.2	35.5	27.02	14.16	4.7	34.9	30
ARI	-35.77	-18.94	-13.76	-15.32	-7.82	-1.84	15.57	26
BOS	-6.85	-7.73	-11.65	-2.46	-5.81	-6.5	6.83	7
BUF	12.37	-1.66	3.15	5.9	5.65	4.63	5.56	4
CGY	-5.03	3.98	-14.74	-12.48	-8.3	-1.71	7.71	10
CAR	25.39	16.63	5.44	2.3	1.85	-0.09	8.62	14
CHI	35.6	9.97	10.51	-3.66	-1.17	1.36	10.37	20
COL	-38.29	-3.72	-5.95	-8.88	-5.99	-5.51	11.39	22
CBJ	-2.58	-1.26	5.41	0.94	-2.57	-1.83	2.43	1
DAL	-15.06	-14.47	-13.64	-3.04	-3.19	4.52	8.99	16
DET	-35.52	16.9	1.13	-1.1	0.57	1.33	9.43	17
EDM	22.72	-17.21	-17.14	-8.61	-7.01	-1.27	12.32	24
FLA	-1.72	15.46	16.72	2.01	1.91	2.49	6.72	6
LAK	75.42	5.37	3.48	-1.65	1.47	-1.88	14.87	25
MIN	-44.66	-13.07	-6.78	-2.77	2.85	-0.23	11.72	23
MTL	-32.11	-32.32	-24.92	-11.92	-4.72	-2.01	18	28
NSH	12.33	-1.9	11.64	14.4	9.31	0.79	8.36	13
NJD	25.11	-8.79	-5.7	1.04	-4.53	-0.3	7.58	9
NYI	1.87	9.08	1.94	2.94	2.86	-0.47	3.19	2
NYR	9.19	-12.57	-0.51	4.38	0.09	1.12	4.64	3
OTT	-11.69	-9.56	-16.95	-5.37	-0.82	-2.12	7.75	11
PHI	30.22	28.37	18.06	9.41	11.3	2.67	16.67	27
PIT	54.64	19.45	14.85	19.39	13.76	8.03	21.68	29
SJS	-29.46	8.76	8.66	3.5	1.27	-1.66	8.89	15
STL	-1.98	9.04	15.64	15.6	12.04	4.06	9.73	18
TBL	-12.21	15.71	9.33	2.3	1.52	-2.82	7.32	8
TOR	8.76	-4.4	-6.35	-8.25	-5.52	-2.26	5.92	5
VAN	-27.18	-16.86	-4.46	-6.25	-5.21	-4.25	10.70	21
WSH	17.42	16.05	8.73	-1.87	-2.43	-2.06	8.09	12
WPG	-38.45	-5.3	-6.95	-5.47	-5.3	0.38	10.30	19

Appendix F: MATLAB Code

File: `simulate_nhl_season.m`

```
%%% PARSE DATA + SET VARS %%%
% define teams and conferences
teams       = ['ANA', 'ARI', 'BOS', 'BUF', 'CGY', 'CAR', 'CHI', 'COL', ...
               'CBJ', 'DAL', 'DET', 'EDM', 'FLA', 'LAK', 'MIN', 'MTL', ...
               'NSH', 'NJD', 'NYI', 'NYR', 'OTT', 'PHI', 'PIT', 'SJS', ...
               'STL', 'TBL', 'TOR', 'VAN', 'WSH', 'WPG'];
metropolitan_rng = [6 9 18 19 20 22 23 29];
atlantic_rng     = [3 4 11 13 16 21 26 27];
central_rng      = [7 8 10 15 17 25 30];
pacific_rng      = [1 2  5 12 14 24 28];

% get game information and set start date for simultation
sim_date    = [2015, 12, 15];
games       = csvread('all_games_201516.csv', 2, 0);
dates       = unique(datenum(games(:,2),games(:,3),games(:,4)));
real_points = generate_points_matrix(dates, games);

% get games played (given start date)
played_rng    = find(games(:,2) < sim_date(1) | ... % year is less
                     games(:,2) == sim_date(1) ...
                   & games(:,3) < sim_date(2) | ... % month is less
                     games(:,2) == sim_date(1) & games(:,3) == sim_date(2) ...
                   & games(:,4) <= sim_date(3)); % day is less (or equal)
unplayed_rng  = setdiff(games(:,1),played_rng);

% stores pcts of particular team to score certain number of goals
% goals_pcts(i,j) = prob of team i scoring j goals in reg. time (1<j<15)
% goals_pcts(i,16) = prob of team i scoring in OT
goal_pcts = zeros(30,16);
for i = 1:30,
    [reg_goals, OT_goals] = calculate_goal_pcts(i, games, played_rng);
    goal_pcts(i,:) = [reg_goals, OT_goals];
end

%%% SEASON SIMULATION %%%
num_iter   = 100;
avg_points = [dates zeros(size(dates,1),30)];

% monte carlo simulation
for n = 1:num_iter
    % determine outcome of remaining games
    games(unplayed_rng,[6 7 9 10 11]) = 0; % reset unplayed games
    for i = unplayed_rng'
        % for each game, calculate score and adjust points
        teamA_i = games(i,5);
        teamB_i = games(i,8);
        [teamA_goals, teamB_goals, inOT] = calculate_game_score(goal_pcts, teamA_i, teamB_i);
        games(i,[6 7])  = [teamA_goals (teamA_goals > teamB_goals)];
        games(i,[9 10]) = [teamB_goals (teamA_goals < teamB_goals)];
        games(i,11)     = inOT;
    end

    % convert game results into points per team
    points = generate_points_matrix(dates, games);
    avg_points(:,2:end) = avg_points(:,2:end) + points(:,2:end);
end

avg_points(:,2:end) = avg_points(:,2:end) / num_iter;

%%% DETERMINE RANKS %%%
proj_div_rank  = zeros(1,30);
proj_conf_rank = zeros(1,30);
real_div_rank  = zeros(1,30);
real_conf_rank = zeros(1,30);

```

```
proj_sorted_a = sort(avg_points(end,atlantic_rng+1),'descend');
proj_sorted_m = sort(avg_points(end,metropolitan_rng+1),'descend');
proj_sorted_e = sort(avg_points(end,union(atlantic_rng,metropolitan_rng)+1),'descend');
proj_sorted_p = sort(avg_points(end,pacific_rng+1),'descend');
proj_sorted_c = sort(avg_points(end,central_rng+1),'descend');
proj_sorted_w = sort(avg_points(end,union(central_rng,pacific_rng)+1),'descend');

real_sorted_a = sort(real_points(end,atlantic_rng+1),'descend');
real_sorted_m = sort(real_points(end,metropolitan_rng+1),'descend');
real_sorted_e = sort(real_points(end,union(atlantic_rng,metropolitan_rng)+1),'descend');
real_sorted_p = sort(real_points(end,pacific_rng+1),'descend');
real_sorted_c = sort(real_points(end,central_rng+1),'descend');
real_sorted_w = sort(real_points(end,union(central_rng,pacific_rng)+1),'descend');

for team_i = 1:30
    % determine division and conference
    if (sum(ismember(pacific_rng, team_i)) == 1)
        proj_div_rank(team_i)  = find(avg_points(end,team_i+1) == proj_sorted_p,1);
        proj_conf_rank(team_i) = find(avg_points(end,team_i+1) == proj_sorted_w,1);
        real_div_rank(team_i)  = find(real_points(end,team_i+1) == real_sorted_p,1);
        real_conf_rank(team_i) = find(real_points(end,team_i+1) == real_sorted_w,1);

    elseif (sum(ismember(central_rng, team_i)) == 1)
        proj_div_rank(team_i)  = find(avg_points(end,team_i+1) == proj_sorted_c,1);
        proj_conf_rank(team_i) = find(avg_points(end,team_i+1) == proj_sorted_w,1);
        real_div_rank(team_i)  = find(real_points(end,team_i+1) == real_sorted_c,1);
        real_conf_rank(team_i) = find(real_points(end,team_i+1) == real_sorted_w,1);

    elseif (sum(ismember(atlantic_rng, team_i)) == 1)
        proj_div_rank(team_i)  = find(avg_points(end,team_i+1) == proj_sorted_a,1);
        proj_conf_rank(team_i) = find(avg_points(end,team_i+1) == proj_sorted_e,1);
        real_div_rank(team_i)  = find(real_points(end,team_i+1) == real_sorted_a,1);
        real_conf_rank(team_i) = find(real_points(end,team_i+1) == real_sorted_e,1);

    elseif (sum(ismember(metropolitan_rng, team_i)) == 1)
        proj_div_rank(team_i)  = find(avg_points(end,team_i+1) == proj_sorted_m,1);
        proj_conf_rank(team_i) = find(avg_points(end,team_i+1) == proj_sorted_e,1);
        real_div_rank(team_i)  = find(real_points(end,team_i+1) == real_sorted_m,1);
        real_conf_rank(team_i) = find(real_points(end,team_i+1) == real_sorted_e,1);
    end
end
```

File: plot_team_points.m

```
%% plotting point data (projected vs. actual)
% team colors
team_colors = zeros(30,3);
team_colors(1,:)  = [145, 118,  75] / 255; % ANA gold
team_colors(2,:)  = [132,  31,  39] / 255; % ARI red
team_colors(3,:)  = [255, 196,  34] / 255; % BOS yellow
team_colors(4,:)  = [  0,  46,  98] / 255; % BUF blue
team_colors(5,:)  = [224,  58,  62] / 255; % CGY red
team_colors(6,:)  = [224,  58,  62] / 255; % CAR red
team_colors(7,:)  = [227,  38,  58] / 255; % CHI red
team_colors(8,:)  = [139,  41,  66] / 255; % COL maroon
team_colors(9,:)  = [  0,  40,  92] / 255; % CBJ blue
team_colors(10,:) = [  0, 106,  78] / 255; % DAL green
team_colors(11,:) = [236,  31,  38] / 255; % DET red
team_colors(12,:) = [230, 106,  32] / 255; % EDM orange
team_colors(13,:) = [213, 156,   5] / 255; % FLA yellow
team_colors(14,:) = [  0,   0,   0] / 255; % LAK black
team_colors(15,:) = [  2,  87,  54] / 255; % MIN green
team_colors(16,:) = [191,  47,  56] / 255; % MTL red
team_colors(17,:) = [253, 187,  47] / 255; % NSH yellow
team_colors(18,:) = [224,  58,  62] / 255; % NJD red
team_colors(19,:) = [245, 125,  49] / 255; % NYI orange
team_colors(20,:) = [  1,  97, 171] / 255; % NYR blue
team_colors(21,:) = [228,  23,  62] / 255; % OTT red
team_colors(22,:) = [244, 121,  64] / 255; % PHL orange
team_colors(23,:) = [209, 189, 128] / 255; % PIT gold
team_colors(24,:) = [  5,  83,  93] / 255; % SJS blue
team_colors(25,:) = [  5,  70, 160] / 255; % STL blue
team_colors(26,:) = [  1,  62, 125] / 255; % TBL blue
team_colors(27,:) = [  0,  55, 119] / 255; % TOR blue
team_colors(28,:) = [  4, 122,  74] / 255; % VAN green
team_colors(29,:) = [224,  23,  59] / 255; % WSH red
team_colors(30,:) = [168, 169, 173] / 255; % WPG silver

teams_to_plot     = [10, 20, 29];
clf

% plot simultation starting date
d = datetime(sim_date(1),sim_date(2),sim_date(3));
plot([d d], [0, 140], '-', 'Color', [0 0 0], 'DisplayName', 'Simulation Begins');

% plot team data
hold on
for t = teams_to_plot
    plot(dates, real_points(:,t + 1), '-', 'Color', team_colors(t,:), 'DisplayName', ...
                                                    strcat(teams(t*3-2:t*3), ' Actual'));
    plot(dates, avg_points(:,t + 1), '--', 'Color', team_colors(t,:), 'DisplayName', ...
                                                 strcat(teams(t*3-2:t*3), ' Projected'));
end

axis([datenum(2015,10,1) datenum(2016,4,10) 0 125])
title(strcat('Actual Performance vs. Projected (Simulation Starting   ',datestr(d), ')'))
legend('Location','northwest')
ylabel('Points')
hold off
```

File: calculate_goal_pcts.m

```
function [reg_goal_pcts, OT_goal_pcts] = calculate_goal_pcts(team_i, games, played_rng)
% arguments:
% - team_i:     the index of the team to calculate
% - games:      matrix of game data (taken from CSV file)
% - played_rng: a range of games already played (given end date)

% returns:
% - reg_goal_pcts: array containing probabilities of scoring particular
%                  number of goals.
%                  ex: reg_goals_pcts(4) = probability of scoring 4 goals
% - OT_goals_pcts: probability of a team scoring in OT

    team_home_rng = intersect(played_rng, find(games(:,5) == team_i));
    team_away_rng = intersect(played_rng, find(games(:,8) == team_i));
    team_reg_rng  = intersect(find(games(played_rng,11) == 0), union(team_away_rng,team_home_rng));
    team_OT_rng   = setdiff(union(team_away_rng,team_home_rng), team_reg_rng);
    team_win_rng  = union(intersect(find(games(:,7) == 1), team_home_rng), ...
                          intersect(find(games(:,10) == 1),team_away_rng));
    team_loss_rng = setdiff(union(team_away_rng,team_home_rng), team_win_rng);

    % determine the spread of goals scored in regular season
    reg_goal_pcts = zeros(1,15);
    for i = 1:15
        % number of times i goals scored in regular game
        reg_occurence  = sum(games(intersect(team_home_rng, team_reg_rng), 6) == i - 1) + ...
                         sum(games(intersect(team_away_rng, team_reg_rng), 9) == i - 1);

        % number of times i goals scored in OT game and team lost
        OT_L_occurence = sum(games(intersect(intersect(team_home_rng,team_OT_rng), ...
                                             team_loss_rng),6) == i - 1) + ...
                         sum(games(intersect(intersect(team_away_rng,team_OT_rng), ...
                                                 team_loss_rng),9) == i - 1);

        % number of times i + 1 goals scored in OT game and team won
        OT_W_occurence = sum(games(intersect(intersect(team_home_rng,team_OT_rng), ...
                                                 team_win_rng),6) == i) + ...
                         sum(games(intersect(intersect(team_away_rng,team_OT_rng), ...
                                                      team_win_rng),9) == i);

        reg_goal_pcts(i) = (reg_occurence + OT_L_occurence + OT_W_occurence) / ...
                           size(union(team_home_rng,team_away_rng),1);
    end

    % determine OT goals
    OT_goal_pcts = (sum(games(intersect(team_away_rng,team_OT_rng),10)) +  ...
                    sum(games(intersect(team_home_rng,team_OT_rng), 7))) / ...
                    max(size(team_OT_rng,1),1);
end
```

File: generate_points_matrix.m

```
function [points] = generate_points_matrix(dates, games)
% arguments:
% - dates: array containing distinct days games are played on
% - games: matrix of game data (taken from CSV file)
%
% returns:
% - points: matrix containing points earned throughout season for each team

    points = [dates zeros(size(dates,1),30)];
    for i = games(:,1)'
        date_i  = find(dates == datenum(games(i,2),games(i,3),games(i,4)));
        teamA_i = games(i,5);
        teamB_i = games(i,8);

        if (sum(points(date_i,2:end)) == 0 && date_i > 1)
            points(date_i,2:end) = points(date_i - 1,2:end);
        end

        if (games(i,7) == 1) % if teamA won
            points(date_i, teamA_i + 1) = points(date_i, teamA_i + 1) + 2;
            points(date_i, teamB_i + 1) = points(date_i, teamB_i + 1) + games(i,11);
        else                 % if teamB won
            points(date_i, teamA_i + 1) = points(date_i, teamA_i + 1) + games(i,11);
            points(date_i, teamB_i + 1) = points(date_i, teamB_i + 1) + 2;
        end
    end
end
```