# Optimizing Goalie Pulls in Hockey: Take Me Out, Coach!

Laura Roettges (roettges@wisc.edu) — Ryan Moreno (rrmoreno@wisc.edu)

Aug 2024

## 1 Introduction

In hockey, it is common practice for a team to "pull the goalie" in the last few minutes of a game if they are losing. Pulling the goalie entails subbing out the goalie for an additional offensive player. This way, the losing team has 6 skaters on the ice (and no goalie) instead of the usual 5 skaters and 1 goalie. The key idea behind this is the following: winning/losing the game is solely decided by which team ends with more points, not *how many* points they win/lose by.

For clarity throughout this paper, we will refer to the "underdog" team which is currently losing but we are rooting for as *the Lightning* and the opposing team as *the Avalanche*.

If the Lightning are down by 2 points with 1 minute left in the game, there is a very small chance of them catching up by the end of the game if the play remains 5v5. Although pulling the goalie makes it more likely that the Lightning will be scored on (called an "empty net" goal), there is minimal cost associated with getting scored on at this point in the game, because team the Lightning were almost certainly going to lose anyways. Pulling the goalie also makes it more likely that the Lightning will score a goal, since they have 6 offensive skaters. Even though there is higher risk of getting scored on, the reward associated with potentially scoring outweighs the risk since scoring could bring the Lightning to a tie or win. In essence, after a goalie is pulled, there is higher variance in the number of goals scored (on both sides). Since the Lightning is about to lose the game, this higher variance is to their benefit.

Our question is, given the score of a hockey game and the number of minutes left, when (if ever) is the optimal time for the Lightning to pull their goalie?

### 1.1 Data

Our model depends on four key probabilities, related to the rates at which team the Lightning and the Avalanche are expected to score during 5v5 play, the rate at which the Lightning are expected to score during 6v5 play (after their goalie has been pulled), and the rate at which the Avalanche are expected to score during 5v6 play (after the Lightning's goalie has been pulled).

To define the probabilities of scoring during 5v5 play, we used statistics from NHL games between 2016 and 2024 [League, 2024b] [League, 2024a]. Because the vast majority of a game occurs as 5v5 play, we estimated the rates of scoring based on the number of goals scored throughout each game.

To define the probabilities of scoring during 5v6 and 6v5 play, we used a set of pre-processed data in which the author has scraped NHL data from 2003-2019 to determine goalie pull times as well as the time of the first goal following a goalie pull [Agalea, 2023].

The inputs to our model consist of the current score of the game and the number of minutes left. We solved our model for a range of inputs, resulting in recommendations for when to pull the goalie if the Lightning is down by 1, 2, or 3 points (a reasonable set of score differentials for the end of a hockey game).

# 2 Mathematical Model

## 2.1 Definitions and Assumptions

- Inputs
  - $A_0$: the Lightning's current score.
  - $B_0$: the Avalanche's current score.
  - $T$: number of seconds remaining in the game
- Decision
  - $s \in [0, 1, \ldots T]$: how many seconds to wait before pulling the goalie ($s = T$ would indicate never pulling the goalie)
  - we will refer to the period from time 0 (now) until $s$ as "epoch 1" and the period from $s$ until the end of the game as "epoch 2"
- Objective
  - maximize the probability that the Lightning ties or wins the game.
- Data
  - $p_{a1}$: the probability that the Lightning scores during a given 10 second stretch during 5v5 play
  - $p_{b1}$: the probability that the Avalanche scores during a given 10 second stretch during 5v5 play
  - $p_{a2}$: the probability that the Lightning scores during a given 10 second stretch during 6v5 play
  - $p_{b2}$: the probability that the Avalanche scores during a given 10 second stretch during 5v6 play
- Simplifications/assumptions

- The Lightning is currently losing
- It is the third period ($T \leq 1200$)
- The Avalanche will not pull their goalie
- Once the Lightning pulls their goalie, the goalie will not be put back in for the rest of the game
- The Lightning and the Avalanche are equal in skill, *i.e.*, $p_{a1} = p_{b1}$ (possibly our most egregious assumption since the Lightning are clearly superior...)
- For each 10 second stretch of time, a team can score either 0 or 1 goals, according to a Bernoulli random variable

## 2.2 Stochastic Programming Model

Because our situation involves maximizing an expected value which depends on randomness, we used a stochastic programming model, described below. Because we include a set of indicator variables, this is a mixed integer program. Further, this is a nonlinear, non-convex program because our constraints involve a nonlinear and non-convex equation. The formal model is given below, preceded by its derivation.

- **Stochastic programming formulation**:
  - Stage 1 decision: $s$ - how many seconds until the goalie is pulled
  - Randomness:
    - $G_{a1}$: Number of goals scored by the Lightning during epoch 1 (Binomial RV)
    - $G_{a2}$: Number of goals scored by the Lightning during epoch 2 (Binomial RV)
    - $G_{b1}$: Number of goals scored by the Avalanche during epoch 1 (Binomial RV)
    - $G_{b2}$: Number of goals scored by the Avalanche during epoch 2 (Binomial RV)
    - $\mathscr{E}_i = [g_{a1} \in G_{a1}, g_{a2} \in G_{a2}, g_{b1} \in G_{b1}, g_{b2} \in G_{b2}]$
      - $\mathbf{E} = \cup \mathscr{E}_i$ is the possible event space (all combinations of possibilities for how many goals were scored by each team within each epoch)
      - $\mathscr{E}_i$ is defined such that $g_{a1}, g_{a2}, g_{b1}, g_{b2} \in [0, 1, \ldots, \lfloor T/10 \rfloor]$
    - $\sim 2{\times}10^{12}$ situations exist for $g_{a1}, g_{a2}, g_{b1}, g_{b2} \in [0, 1, \ldots, \lfloor T/10 \rfloor]$
  - No Stage 2 recourse decision
  - Decision-dependent probability (endogenous uncertainty)
- **Objective:**

$$\begin{aligned} \max(\mathbb{P}[\text{success}|s]) &= \max(\mathbb{P}[\text{our team wins or ties}|s]) \\ &= \max(\mathbb{P}[a_0 + a_1 + a_2 \geq b_0 + b_1 + b_2|s]) \end{aligned}$$

- $\mathbb{P}[\text{success}|s] = \sum_{\mathscr{E}_i} \mathbb{P}[\mathscr{E}_i|s]\mathbb{1}_{\text{success}(\mathscr{E}_i)}$
- We define $\mathbb{P}[\text{success}|s]$ such that if $\mathscr{E}_i$ is invalid for $s$ then $\mathbb{P}[\mathscr{E}_i|s] = 0$. $\mathscr{E}_i$ is invalid for $s$ if $a_{i1} > s, b_{i1} > s, a_{i2} > T - s$, or $b_{i2} > T - s$.

$$\begin{aligned}\mathbb{P}[\mathscr{E}_i|s] &= \mathbb{P}[(G_{a1} == g_{a1})\&(G_{a2} == g_{a2})\&(G_{b1} == g_{b1})\&(G_{b2} = g_{b2})|s] \\ &= \prod_{x\in\{a,b\}} \binom{s}{g_{x1}}(p_{x1})^{g_{x1}}(1-p_{x1})^{s-g_{x1}} \cdot \binom{T-s}{g_{x2}}(p_{x2})^{g_{x2}}(1-p_{x2})^{T-s-g_{x2}}\end{aligned}$$

- This probability formula comes from the definition of Binomial RVs.

- **Objective (simplified)**: $\max(\sum_{\mathscr{E}_i} p_{\mathscr{E}i} * y_{\mathscr{E}i})$
- **Constraints:**
  - $p_{\mathscr{E}i} \leq \mathbb{P}[\mathscr{E}_i|s]$ (amount incorporated into the sum can be at most the probability of the event, which is allowed when $\mathbb{1}_{\text{success}(\mathscr{E}_i)} = 1$; note that $\mathbb{P}[\mathscr{E}_i|s]$ is non-convex, so this is a non-convex constraint)
  - $\mathbb{1}_{\text{success}(\mathscr{E}_i)} = 0 \rightarrow y_{\mathscr{E}_i} = 0$ (nothing gets incorporated into the sum if the event is not a success; note that we do not need to enforce the other direction of this implication because we are maximizing)
    - Consider the values of $a_j$ and $b_j$ specific to event $\mathscr{E}_i$:
    - let $a = a_0 + a_1 + a_2$ be our final score
    - let $b = b_0 + b_1 + b_2$ be team B's final score
    - let $d = a + 1 - b$. Note that $d \geq 1$ if we tie or win and $d \leq 0$ if we lose
    - This yields the equivalent logical constraint $d_{\mathscr{E}_i} \leq 0 \rightarrow y_{\mathscr{E}_i} = 0$
    - we can reformulate this logical constraint using the so-called "slide of trix" as $d_{\mathscr{E}_i} + m \cdot y_{\mathscr{E}_i} \geq m + 1$
    - Finding a value for $m$: $d = a + 1 - b \geq 0 + 1 - 400 = -359 = m$ ($b = 400$ is the worst case scenario in which the Avalanche score every 10 seconds of the game and the lightning never score)
  - Final reformulation of the constraints on $y_{\mathscr{E}_i}$:
    - $d_{\mathscr{E}_i} - 359 \cdot y_{\mathscr{E}_i} \geq -358$
    - $y_{\mathscr{E}_i} \in \{0, 1\}$
  - $s \in \{10k|k \in \mathbb{Z}^+\}$ (we can only pull the goalie once every 10 seconds)
  - $0 < s \leq 1200$ (we are in the final period of the game)

#### 2.2.1 Formal model:

Our model is a non-convex, mixed integer, stochastic programming model, with $p, y \in \mathbb{R}^{|\mathbf{E}|}$, $s \in \mathbb{R}$, and $\mathbf{E}$ defined above:

$$\max_{s,p,y}[\sum_{\mathscr{E}_i \in \mathbf{E}} p_{\mathscr{E}i} \cdot y_{\mathscr{E}i}] \text{ s.t.}$$

$$p_{\mathscr{E}i} \leq \mathbb{P}[\mathscr{E}_i|s] \quad \forall \mathscr{E}_i \in \mathbf{E}$$
$$d_{\mathscr{E}_i} - 359 \cdot y_{\mathscr{E}_i} \geq -358 \quad \forall \mathscr{E}_i \in \mathbf{E}$$
$$s \in \{10k | k \in \mathbb{Z}^+\}$$
$$0 < s \leq 1200$$
$$y_{\mathscr{E}_i} \in \{0, 1\} \quad \forall \mathscr{E}_i \in \mathbf{E}$$

# 3 Solution

## 3.1 Computing probabilities from NHL stats

### 3.1.1 Computing $p_{a1}$ and $p_{b1}$

To determine the values of $p_{b1}$ and $p_{a1}$, we reviewed goals scored and conceded during playoff games between 2016 and 2024 [League, 2024a,b]. This dataset was not accompanied by annotations of time ranges for 5v5 play, so for simplicity we assumed 5v5 play for the full 3 periods. Please see data_explore.py for the implementation. This resulted in two probabilities:

- Probability of scoring a goal in a given 10 second stretch during 5v5 play: $4.98 * 10^{-3}$
- Probability of conceding a goal in a given 10 second stretch during 5v5 play: $5.37 * 10^{-3}$

Since we are assuming $p_{b1} = p_{a1}$, we averaged these two values to get $5.18 * 10^{-3}$.

### 3.1.2 Pre-computing $p_{a2}$ and $p_{b2}$

To define the probabilities of $p_{a2}$ (scoring during 6v5) and $p_{b2}$ (scoring during 5v6 play), we used a set of pre-processed data in which the author has scraped NHL data from 2003-2019 to determine goalie pull times as well as the time of the first goal following a goalie pull [Agalea, 2023]. To compute $p_{a2}$, we analyzed the average amount of time between a team pulling their goalie and that team scoring. To compute $p_{b2}$, we analyzed the average amount of time between a team pulling their goalie and their opponents scoring.

Because goalie pulls occur near the end of a game, there are many data points in which no goals are scored after a goalie is pulled. In order to avoid biasing our data by removing these data points, we used the Kaplan-Meier estimator to

determine $p_{a2}$ and $p_{b2}$ from the data [Kaplan and Meier, 1958]. The Kaplan-Meier estimator is an exponential survival model used in clinical data settings, where researchers often lose contact with patients prior to the patient's death. In the clinical case, a data point (the time from diagnoses to death, for example) is considered censored if the patient is lost to follow-up. Although the time of death is not observed, the fact that the patient was alive when they were censored can be utilized while calculating average time of survival. In our case, a data point (the time from the goalie pull until the team of interest scores) is considered censored if the game ends prior to the team of interest scoring. We also consider a data point censored if the team of interest concedes a goal prior to scoring (because the dataset we used only recorded the timestamp of the first goal scored). To use the Kaplan-Meier estimator, we identified data points that were censored as "failures" and those that were not censored as "successes".

- Successes are instances in which a goalie is pulled and the team of interest subsequently scores. In this case, we store the time difference between the goalie being pulled and the team of interest scoring.
- Failures include:
  - Instances in which a goalie is pulled and the team of interest subsequently conceded a goal. Here we store the time from when the goalie was pulled until the team of interest conceded a goal.
  - Instances in which a goalie is pulled no goal is scored by the end of the game. Here we store the time from when the goalie was pulled to when the game ended.

Since there was a significant amount of censored data, our estimator does not actually achieve values at or below a survival probability of 0.5 when calculating $p_{a2}$ (see figure 1), so we cannot use the point at which our probability of "survival" is 0.5 to indicate our expected length of "survival". To account for this, we linearly extrapolated the data to estimate the amount of time until a survival probability of 0.5, resulting in an estimate of 322.5 seconds. In other words, we would expect the Lightning to score 322.5 seconds after pulling their goalie. This yields an approximate probability, $p_{a2} = 10/322.5$ or $p_{a2} = 3.10 * 10^{-2}$.

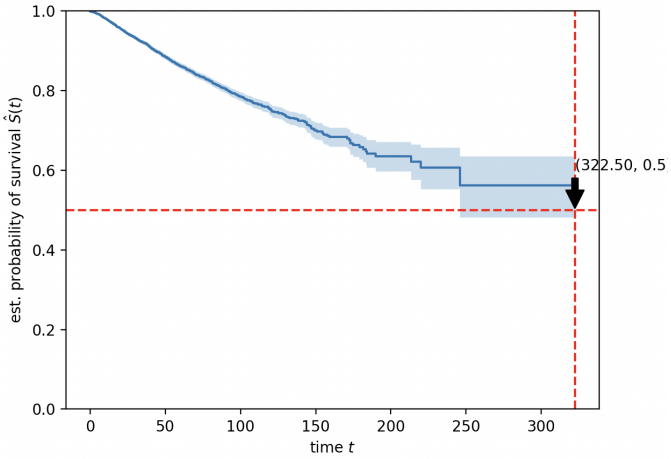



Figure 1: Kaplan Meier Survival Estimation for Team Successfully Scoring After Pulling their Goalie

When calculating $p_{b2}$, our estimator resulted in a survival probability of 0.5 at 117.5 seconds (see figure 2). This yields $p_{b2} = 10/117.5$ or $p_{b2} = 8.51 * 10^{-2}$.

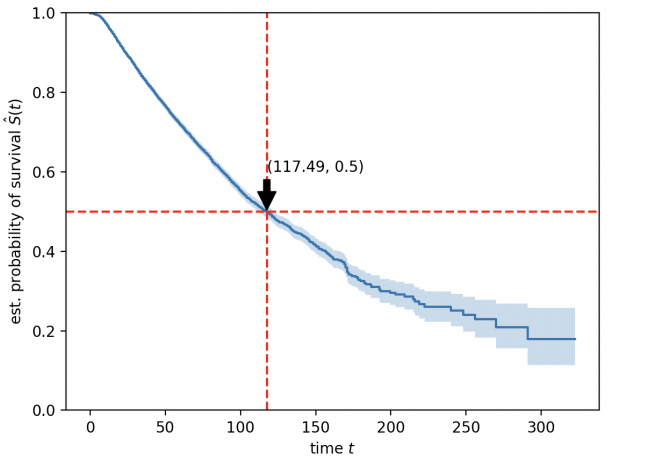



Figure 2: Kaplan Meier Survival Estimation for Oponent Scoring After Goalie Pull

### 3.1.3 Pre-computing combinatorics

We precomputed combinatorics $\binom{n}{s}$ for all values $s \leq 1200, n \leq s$ in order to save time during our model's execution since these values are reused frequently

when computing the probability of a given event. Please see data_explore.py for the implementation.

## 3.2 Optimization implementation

### 3.2.1 Decision-dependent uncertainty

Our model is made complicated by the fact that the probability that a given event $\mathscr{E}_i = [g_{a1}, g_{a2}, g_{b1}, g_{b2}]$ occurs is necessarily based on our decision $s$. In other words, we have *decision-dependent uncertainty* [Hellemo et al., 2018].

One way to handle this decision-dependent uncertainty is to model our choice as selecting between probability distributions. Because we have the constraints $0 < s \leq 1200$ and $s \in \{10k | k \in \mathbb{Z}\}$, we have a finite number of choices, which means that we are selecting from a finite number of probability distributions; each choice $s$ can be mapped to a discrete probability distribution over our set of possible events $E$. These types of "decision-dependent distribution selection" problems are easier to solve than problems with "decision-dependent uncertainty" more generally.

### 3.2.2 Modeling approach

Our problem is complicated by its non-convex structure. We attempted to utilize a solver such as HiGHS or Ipopt, but none of the solvers that we have explored in this class are appropriate for this problem. Our problem requires a solver that can handle mixed integer programs (which HiGHS can, but Ipopt cannot), non-linear constraints (which HiGHS cannot), and stochastic events (which solvers like GLPK can, but GLPK can only handle linear cases). Our initial attempt can be found in model_using_solver.ipynb notebook, but please do not attempt to run this, as it will not yield a solution.

Because of our problem's non-convex structure and the inefficiencies of the open source solvers we are familiar with, we took a 'brute-force' approach to this problem in which we compute the probability of the Lightning tying or winning the game for every possible value of $s$ at which they could choose to pull their goalie. We enforce the constraints on $\mathscr{E}_i$ in the formulation of our loops over $a_1, b_1, a_2$, and $b_2$, excluding situations that are invalid for a particular value of $s$. We implement the logical constraints on $y_{\mathscr{E}_i}$ in our definition of the variable `success = (a >= b)`. The constraint that $p_{\mathscr{E}_i} \leq \mathbb{P}[\mathscr{E}_i | s]$ is enforced by our definition of the variable `prob`, which follows the mathematical formula given above. Further, since instances of $\mathscr{E}_i$ which are invalid for a given $s$ are skipped in the iterations, this implicitly sets $p_{\mathscr{E}_i} = 0$ in those cases.

The code for our brute force model can be viewed and run using the notebook brute_force_model.ipynb. The complexity implications of taking a brute-force approach are discussed in Sec 5.1.

# 4 Results

The optimal time for the Lightning to pull their goalie is dependent on how much time is left in the game as well as the current score difference. Fig 3 and Fig 4 explore these relationships in detail.

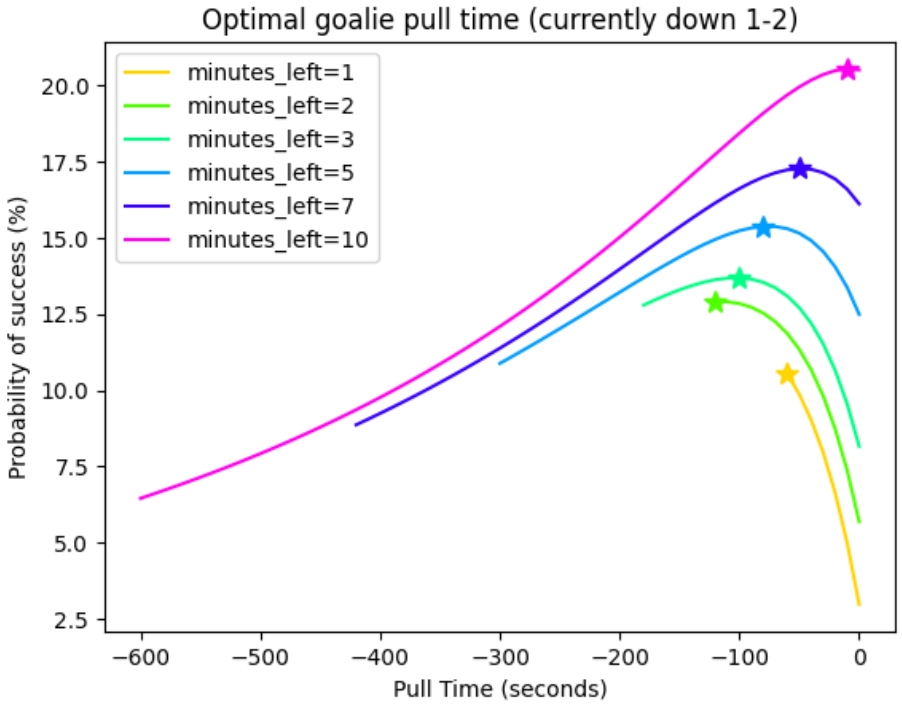



Figure 3: Here we detail how the optimal time to pull the goalie, indicated by the stars, varies with the number of minutes left in the game. If the Lightning are down by 1 with 10 minutes left in the game, there is still a large enough probability of the Lightning coming back for a tie/win without pulling their goalie that they should choose to not pull their goalie for the rest of the game, if they had to decide at that moment. However, given the same score with 7 minutes left, the Lightning should pull their goalie with ~1 minute left in the game. As the amount of time left in the game decreases, the Lightning should choose to pull their goalie earlier. Eventually, given the same score, when there are 2 minutes or less left in the game, the optimal decision is always to pull the goalie immediately.

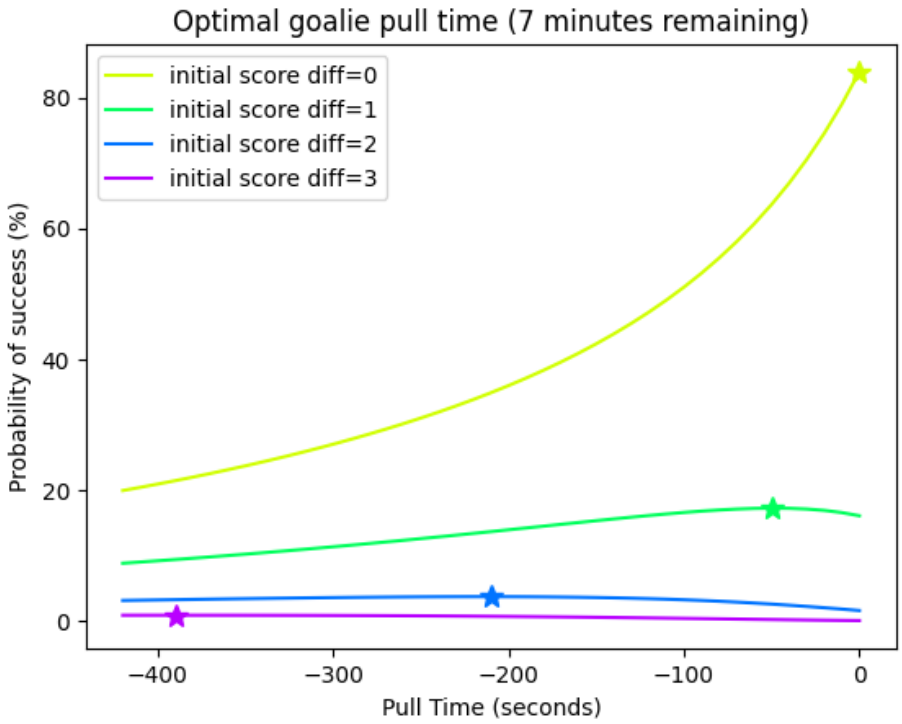


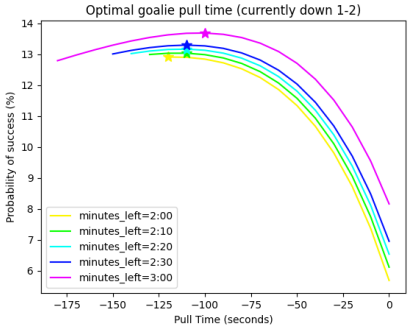

Figure 4: Here we detail how the optimal time to pull the goalie, indicated by the stars, varies with the current score of the game. In the scenarios plotted, there are 7 minutes left in the game and the Lightning are currently down by 0, 1, 2, or 3 points. Note that because we define a tie as a success, if the lightning are currently tied, their optimal choice is to never pull their goalie, resulting in a probability of success ~80%. As expected, the more that the Lightning are losing by, the earlier they should choose to pull their goalie.

It is important to realize that our model answers the question of the optimal goalie pull time if the Lightning had to choose the goalie pull time *at that exact moment* in the game. We'll call this the "choose now" setting. If, for example, there are 3 minutes left in the game, the Lightning are down by 1, and they have to choose when to pull their goalie right now, the optimal decision would be to pull the goalie with 1 minute left in the game. However, if another 30 seconds passes and there are now 2:30 minutes left in the game, and the Lightning are still down by 1, the optimal decision now would be to pull the goalie with 1 minute and 10 seconds left in the game.

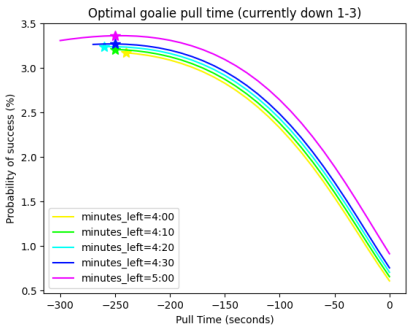
A more realistic question is when the Lightning should pull their goalie, assuming that they can make that decision at any moment in the game (or more precisely, that they can update their decision every 10 seconds). We will call this the "ongoing choice" setting. Note that the "ongoing choice" setting can be formulated as a problem in which the only input is the current score of

the game. The "ongoing choice" problem asks the time point during the third period for which the optimal choice switches from the Lightning not pulling their goalie to the Lightning pulling their goalie, given that they are down by $x$. Note that only the score difference (down by $x$) matters, not the actual score.

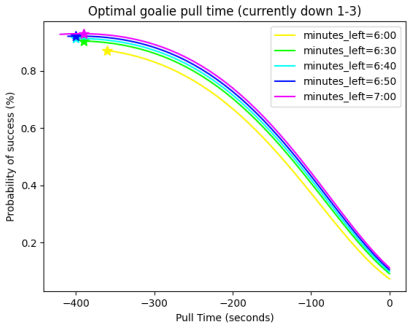
To translate between the "ongoing choice" problem and our model, we see that the optimal decision in the "ongoing choice" setting is the earliest time at which the Lightning should pull their goalie immediately in the "choose now" setting. As can be seen in Fig 5, we solved the "ongoing choice" problem for several reasonable settings: the Lightning being down by 1, 2, or 3 points. Our analysis results in the recommendation that the Lightning pull their goalie with 2:00 minutes left in the game if they are down by 1, should pull their goalie with 4:30 minutes left in the game if they are down by 2, and should pull their goalie with 6:40 minutes left in the game if they are down by 3.




(a) If the Lightning are down by 1, they should pull their goalie with 2:00 minutes left in the game.




(b) If the Lightning are down by 2, they should pull their goalie with 4:30 minutes left in the game.




(c) If the Lightning are down by 3, they should pull their goalie with 6:40 left in the game.

Figure 5: The optimal decision in the "ongoing choice" setting is the earliest time at which the Lightning should pull their goalie immediately in the "choose now" setting. In the "ongoing choice" setting, the Lightning can make the decision to pull their goalie at any point in the game. In the "choose now" setting, there is a certain amount of time left in the game, and the Lightning must choose right now when they will pull their goalie.

# 5 Discussion

## 5.1 Complexity

Mixed integer programming (MIP) is NP-hard, so there is no guarantee that we have an efficient method for solving MIP optimization problems. In fact, even mixed integer linear programs are NP-complete. In practice, however, models are able to leverage approximations and take advantage of specific problem's structures in order to solve these problems [Smith, 2024]. Unfortunately, in our case we have a non-convex function, so we do not have simple methods to solve this problem. In this case, without a solver particular to our problem structure,

a brute-force approach to the problem is not significantly more computationally expensive than a more nuanced approach.

In order to make solving this problem tractable, we have chosen to model the results of a hockey game such that each time has a binary result (scoring or not scoring) during each 10 seconds of the game. Initially, we had these "buckets" set as 1 second, but this resulted in excessive computational requirements. In essence, broadening our "bucket" size is similar to the technique of sampling discrete events from a probability distribution in order to have a smaller event space. Prior to broadening our "bucket" size to 10 seconds, a single instance of our problem took over 24 hours, even leveraging access to a super computer. After making this change, an instance of our problem runs in an order of minutes on an M2 Mac.

Finally, in order to take advantage of our problem's "distribution selection" flavor, we chose to precompute all values of $n$ choose $k$ necessary for computing the probability distributions. Although we did not precompute the probability distribution for each possible decision, this step saves significant compute time and avoids redundant calculations.

# 6 Conclusion

In this paper, we formulated the question of when a hockey team should pull their goalie as a mixed integer, non-convex, stochastic optimization problem. Because this problem involves Binomial RVs, it has a complex structure for which there is not a tailored open source solver, to our knowledge. We implemented the optimization using a brute-force method, and came up with concrete recommendations for when a hockey team should pull their goalie if they are losing in the third period.

## 6.1 Future Work

There are a number of interesting directions for further exploration. A primary next step would be using an existing solver to implement this problem. One solver that we considered is EAGO, which is open source and can handle mixed integers and non-linear constraints. However, we would need to find or implement a solver that can also handle stochastic problem formulations, which was out of the scope of our project.

It would also be interesting to investigate what the effect would be of varying probabilities $p_{a2}$ and $p_{b2}$. During our first attempt at solving this problem, we had a typo resulting in inaccurate probabilities for $p_{a2}$ and $p_{b2}$. Interestingly, those incorrect probabilities resulted in an optimal solution of either pulling the goalie immediately or never pulling the goalie for the vast majority of situations. To explore this effect, we tried varying $p_{a2}$ and found that as $p_{a2}$ (the Lightning's probability of scoring after they pull their goalie) increased, approaching $p_{b2}$, it became increasingly beneficial to pull their goalie immediately. As $p_{a2}$ decreased, there was a tipping point after which never pulling the goalie was

consistently the optimal solution (because it was so unlikely that the Lightning would score). It so happens that the actual probabilities we computed, based on NHL statistics, fall into the seemingly small range in which this problem is "interesting" in that there are optimal solutions other than immediately or never pulling the goalie. Future work could explore the boundaries of this "interesting" range of probabilities.

## 6.2 How to Replicate

To replicate our results, please review our Github repository here: https://github.com/ryan-moreno/CS524-project; the README.md file has concise information about our data processing and model.

# References


A. Agalea. Nhl goalie pull optimization: Processed data, 2023. URL https://github.com/agalea91/nhl-goalie-pull-optimization/tree/master/data/processed/csv. Accessed: 2024-07-27.

L. Hellemo, P. I. Barton, and A. Tomasgard. Decision-dependent probabilities in stochastic programs with recourse. *Computational Management Science*, 15(3):369–395, 2018. ISSN 1619-6988. doi: 10.1007/s10287-018-0330-0. URL https://doi.org/10.1007/s10287-018-0330-0.

E. L. Kaplan and P. Meier. Estimation from incomplete observations. *Journal of the American Statistical Association*, 53(282), 1958. doi: http://www.jstor.org/stable/2281868.

N. H. League. Team statistics: Goals against by strength, 2024a. URL https://www.nhl.com/stats/teams?aggregate=0&report=goalsagainstbystrength&reportType=game&seasonFrom=20162017&seasonTo=20232024&dateFromSeason&gameType=3&sort=a_teamFullName&page=14&pageSize=100. Accessed: 2024-07-27.

N. H. League. Team statistics: Goals for by strength, 2024b. URL https://www.nhl.com/stats/teams?aggregate=0&report=goalsforbystrength&reportType=game&seasonFrom=20162017&seasonTo=20232024&dateFromSeason&gameType=3&sort=a_teamFullName&page=14&pageSize=100. Accessed: 2024-07-27.

A. Smith. Lecture 1 - problem complexity or, why integer programs are so hard to solve. module 12 - integer programming basics, July 2024.