
Towards Mixed-Integer Programming for Strategy Formulation: A Study in Convex Optimization Under Quasiconvex Constraints for Pokémon

Hailey James
Department of Computer Science
University of California San Diego
La Jolla, CA 92037
hjames@ucsd.edu

Avni Kothari
Department of Computer Science
University of California San Diego
La Jolla, CA 92037
akothari@ucsd.edu

Hayden McTavish
Department of Computer Science
University of California San Diego
La Jolla, CA 92037
hmctavish@ucsd.edu

1 Introduction

Convex Optimization has become an increasingly important tool in solving complex optimization problems in a number of fields, from autonomous driving to green energy planning. In the course of pursuing real-world research objectives, machine learning methods are often applied to games, such as Go [12] and chess [3, 13]. We seek to advance the state of machine learning through application of another game: Pokémon.

In the context of Pokémon, convex optimization techniques can be used to optimize various strategies (e.g., battle planning [9]). The objective of this research paper is to explore the applications of convex optimization in the Pokémon world and its potential to improve performance in Pokémon battles. Overall, this research aims to demonstrate the potential of convex optimization in enhancing the performance and strategy of Pokémon trainers and players.

1.1 Motivation

Although much research focuses on agent decision making at competition time (i.e. for chess), in Pokémon players are allowed to make decisions about the Pokémon they bring to a tournament, before the tournament begins. A core piece of this decision making is how players customize their Pokémon's "defense" and "hit point" values under a budget. In Pokémon, players each bring a number of Pokémon, (digital creatures represented by integer stat values) which then battle the opponents' Pokémon. The key math arbitrating this exchange is the damage formula [4] - each Pokémon starts with a hit points value, which is decremented as Pokémon take turns dealing damage according to the formula, until one player's Pokémon all reach 0 hit points and that player loses. Players can customize their Pokémon - giving more defense points to reduce damage taken, or more hit points so the Pokémon can take more damage. There is a limit to how much a Player can increase a given Pokémon's hit points and defense - investing in one can require investing less in the other, and increasing a Pokémon's hit points or defense means that the Pokémon deals less damage as well. Therefore, we investigate ways to use convex optimization in a setting where a player wants to minimize how much they increase hit points and defense, but where a player still wants to satisfy certain constraints (such as making sure a specific attack does not do more damage than the Pokémon

has hit points). We start with simpler constraints and a much simplified version of the damage formula, with the potential to add complexity as we proceed.

The current primal formulation is a relaxed version of a more complex non-convex formulation. In our experiments, we plan on comparing results and feasible regions with the relaxed formulation and the non-convex formulation. As most problem settings are non-convex this useful comparison will be helpful in many different types of problem settings.

1.2 Related Work

The field of competitive gaming has seen a rise in popularity in recent years, with various games having established professional circuits and tournaments. One such game is Pokémon, where players compete with their teams of creatures called Pokémon. The strategic aspects of the game involve creating a team with complementary strengths and weaknesses, as well as predicting and countering opponents' moves. In this paper, we propose a defense strategy for Pokémon tournaments that leverages the principles of convex optimization.

Our work builds upon previous research in the areas of convex optimization and game strategy design. Such work includes optimization methods applied to games such as Go [12] and chess [3, 13].

Machine learning methods for strategy design have also been applied to Pokémon. One notable study [9] model-free reinforcement learning to find an optimal battle strategy. The authors found that using softmax exploration with Q-learning resulted a reasonably high win rate against a random agent. [8] similarly uses reinforcement learning to create a state-of-the-art game play strategy and investigate the transferability of the learned strategy.

Other research also considers characteristics of attackers and defenders in Pokémon duel scenarios [2], and [17] seeks to build a tool to leverage these characteristics to help novice players construct teams.

Solving optimization problems in order to solve a real world application can be found in many different domains [14]. One such domain is the field of recourse in machine learning. In this setting, a user may be denied by a machine learning model for a publicly accessible good such as a loan or government healthcare. A mixed integer program is used to solve the problem of how a user may change their features in order to be approved for this publicly accessible good if a linear model was used to classify them [16]. Our work contributes to the field of using optimization problems to solve problems, and we hope our work can be extended to real world applications such as recourse in machine learning.

To the best of our knowledge, this is the first work to use convex optimization to create a defense strategy specifically for Pokémon tournaments.

1.3 Contributions

Our contributions can be summarized as follows:

1. We formulate an optimization problem to find the optimal strategy in Pokémon.
2. We create a method to beat the baseline method in Pokémon.
3. We propose a closed form solution to our optimization problem that Pokémon players can use as a defense strategy.

For reproducibility all source code is available at: https://github.com/avni510/convex_wi23

1.4 Organization of the Paper

The rest of this paper is organized as follows. We begin by stating and describing the problem, including exposition of the primal formulation, the dual formulation, and the KKT conditions. We describe our intended approaches, including an exact closed form solution and an approach using Mixed Integer Quadratically Constrained Programming (MIQCP). We then present our experimental results, and conclude with a discussion and future work.

2 Statement of the Problem

Let $\mathbf{x} \in \mathbb{R}_{++}^3$ be a vector such that x_2 represents a Pokémon's defense stat, x_3 represents a Pokémon's special defense stat, and x_1 represents a given Pokémon's hit points stat. Our problem is for a Pokémon to survive an attack from an opposing Pokémon, using a heavily simplified version of the damage formula, while satisfying limitations on how high the Pokémon's hit points, defense, special defense, and the combination of all 3 can be, while minimizing the values of hit points and defense. We further constrain a minimum value for x_1, x_2 , and x_3 to match the game (each Pokémon has a lower bound on its stat values, which it is allowed to have without a cost). We formulate our problem as follows:

2.1 Primal Formulation

$$\begin{aligned}
 \min_{\mathbf{x}} \quad & \|\mathbf{x}\|_1 \\
 \text{s.t.} \quad & x_1 \leq b_1, \\
 & x_2 \leq b_2, \\
 & x_3 \leq b_3, \\
 & x_1 \geq b_1 - 32, \\
 & x_2 \geq b_2 - 32, \\
 & x_3 \geq b_3 - 32, \\
 & (b_4/x_2) - x_1 \leq \epsilon \\
 & (b_5/x_3) - x_1 \leq \epsilon
 \end{aligned} \tag{1}$$

Where $\mathbf{b} \in \mathbb{R}_{++}^5$, $b_1 > 33$, $b_2 > 33$, $b_3 > 33$ is a set of parameters that correspond to the attacking and defending Pokémon and ϵ can be specified as any small value just above 0 (we set it to .0001 for our MIQCP experiments). b_4 and b_5 corresponds to the offensive potential of benchmark attackers, and b_1, b_2, b_3 correspond to the limitations on how high the defender's hit points, defense, and special defense can be.

Note that because $b_1, b_2, b_3 > 33$, we do not need to add any constraints to ensure that x values remain strictly positive.

We can write the problem in standard notation as follows:

$$\begin{aligned}
 \min_{\mathbf{x}} \quad & \|\mathbf{x}\|_1 \\
 \text{s.t.} \quad & x_1 - b_1 \leq 0, \\
 & x_2 - b_2 \leq 0, \\
 & x_3 - b_3 \leq 0, \\
 & b_1 - x_1 - 32 \leq 0, \\
 & b_2 - x_2 - 32 \leq 0, \\
 & b_3 - x_3 - 32 \leq 0, \\
 & \frac{b_4}{x_2} - x_1 - \epsilon \leq 0 \\
 & \frac{b_5}{x_3} - x_1 - \epsilon \leq 0
 \end{aligned}$$

Multiplying the last two constraints through by the positive values x_2/x_1 and x_3/x_1 , respectively, and adjusting epsilon slightly (to be a different, but still small and near 0 value), we can have:

PRIMAL FORMULATION:

$$\begin{aligned}
\min_{\mathbf{x}} \quad & \|\mathbf{x}\|_1 \\
\text{s.t.} \quad & x_1 - b_1 \leq 0, \\
& x_2 - b_2 \leq 0, \\
& x_3 - b_3 \leq 0, \\
& b_1 - x_1 - 32 \leq 0, \\
& b_2 - x_2 - 32 \leq 0, \\
& b_3 - x_3 - 32 \leq 0, \\
& \frac{b_4}{x_1} - x_2 - \epsilon \leq 0 \\
& \frac{b_5}{x_1} - x_3 - \epsilon \leq 0
\end{aligned} \tag{2}$$

2.2 Dual Formulation

$$\begin{aligned}
\text{minimize} \quad & \|\mathbf{x}\|_1 \\
\text{s.t.} \quad & x_1 - b_1 \leq 0, \\
& x_2 - b_2 \leq 0, \\
& x_3 - b_3 \leq 0, \\
& b_1 - x_1 - 32 \leq 0, \\
& b_2 - x_2 - 32 \leq 0, \\
& b_3 - x_3 - 32 \leq 0, \\
& \frac{b_4}{x_1} - x_2 - \epsilon \leq 0 \\
& \frac{b_5}{x_1} - x_3 - \epsilon \leq 0
\end{aligned}$$

Because of the constraints on b which ensure x is positive, we know that $\|\mathbf{x}\|_1 = \sum_{i=1}^3 x_i$.

$$\begin{aligned}
\text{minimize} \quad & x_1 + x_2 + x_3 \\
\text{s.t.} \quad & x_1 - b_1 \leq 0, \\
& x_2 - b_2 \leq 0, \\
& x_3 - b_3 \leq 0, \\
& b_1 - x_1 - 32 \leq 0, \\
& b_2 - x_2 - 32 \leq 0, \\
& b_3 - x_3 - 32 \leq 0, \\
& \frac{b_4}{x_1} - x_2 - \epsilon \leq 0 \\
& \frac{b_5}{x_1} - x_3 - \epsilon \leq 0
\end{aligned}$$

We can write the Lagrangian as

$$\begin{aligned}
L(x, \lambda) = & x_1 + x_2 + x_3 \\
& + \lambda_1(x_1 - b_1) \\
& + \lambda_2(x_2 - b_2) \\
& + \lambda_3(x_3 - b_3) \\
& + \lambda_4(b_1 - x_1 - 32) \\
& + \lambda_5(b_2 - x_2 - 32) \\
& + \lambda_6(b_3 - x_3 - 32) \\
& + \lambda_7\left(\frac{b_4}{x_1} - x_2 - \epsilon\right) \\
& + \lambda_8\left(\frac{b_5}{x_1} - x_3 - \epsilon\right)
\end{aligned}$$

$$\begin{aligned}
L(x, \lambda) &= (1 + \lambda_1 - \lambda_4)x_1 \\
&+ \frac{b_4\lambda_7}{x_1} \\
&+ \frac{b_5\lambda_8}{x_1} \\
&+ (1 + \lambda_2 - \lambda_5 - \lambda_7)x_2 \\
&+ (1 + \lambda_3 - \lambda_6 - \lambda_8)x_3 \\
&- \lambda_1 b_1 - \lambda_2 b_2 \lambda_3 b_3 + \lambda_4(b_1 - 32) + \lambda_5(b_2 - 32) + \lambda_6(b_3 - 32) \\
&+ (\lambda_7 + \lambda_8)(\epsilon)
\end{aligned}$$

and the Lagrangian dual function as

$$g(\lambda) = \inf_{x \in \mathcal{D}} L(x, \lambda)$$

and the Lagrangian dual problem as

$$\begin{aligned}
&\text{maximize} && g(\lambda) \\
&\text{subject to} && \lambda \succeq 0.
\end{aligned}$$

Without changing our result, we can constrain λ so that $g(\lambda) > -\infty$.

To find $\inf_{x \in \mathcal{D}} L(x, \lambda)$, we do:

$$\begin{aligned}
\frac{\partial}{\partial x_1} L(x, \lambda) &= (1 + \lambda_1 - \lambda_4) - \frac{b_4\lambda_7}{x_1^2} - \frac{b_5\lambda_8}{x_1^2} \\
\frac{\partial}{\partial x_2} L(x, \lambda) &= (1 + \lambda_2 - \lambda_5 - \lambda_7) \\
\frac{\partial}{\partial x_3} L(x, \lambda) &= (1 + \lambda_3 - \lambda_6 - \lambda_8)
\end{aligned}$$

Setting the gradient for $x_1 = 0$ yields:

$$\begin{aligned}
(1 + \lambda_1 - \lambda_4) &= \frac{b_4\lambda_7}{x_1^2} - \frac{b_5\lambda_8}{x_1^2} \\
x_1^2 &= \frac{b_4\lambda_7 + b_5\lambda_8}{(1 + \lambda_1 - \lambda_4)} \\
x_1 &= \pm \sqrt{\frac{(b_4\lambda_7 + b_5\lambda_8)}{(1 + \lambda_1 - \lambda_4)}}
\end{aligned}$$

Meanwhile for x_2 and x_3 , the derivative is a constant, which is nonzero for x_2 if and only if $(1 + \lambda_2 - \lambda_5 - \lambda_7) \neq 0$, and nonzero for x_3 if and only if $(1 + \lambda_3 - \lambda_6 - \lambda_8) \neq 0$.

so to minimize L , we have: $x_1 = \pm \sqrt{\frac{(b_4\lambda_7 + b_5\lambda_8)}{(1 + \lambda_1 - \lambda_4)}}$

$$x_2 = \begin{cases} 0, & (1 + \lambda_2 - \lambda_5 - \lambda_7) = 0 \\ \infty, & (1 + \lambda_2 - \lambda_5 - \lambda_7) < 0 \\ -\infty, & (1 + \lambda_2 - \lambda_5 - \lambda_7) > 0 \end{cases}$$

$$x_3 = \begin{cases} 0, & (1 + \lambda_3 - \lambda_6 - \lambda_8) = 0 \\ \infty, & (1 + \lambda_3 - \lambda_6 - \lambda_8) < 0 \\ -\infty, & (1 + \lambda_3 - \lambda_6 - \lambda_8) > 0 \end{cases}$$

Because we have some constraints that are quasiconvex rather than convex, it is necessary to check that setting the derivative to 0 and solving does give us the infimum.

Note that our hessian with respect to x is

$$\nabla_x^2 L(x, \lambda) = \begin{bmatrix} \frac{b_4 \lambda_7 + b_5 \lambda_8}{x_1^2} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

So the hessian is clearly positive semidefinite (since beta is non-negative and so is lambda) and setting the derivative with respect to x equal to 0 does yield the infimum.

Considering that the overall objective of $g(\lambda)$ will be negative infinity if $(1 + \lambda_3 - \lambda_6 - \lambda_8)$ or $(1 + \lambda_2 - \lambda_5 - \lambda_7)$ are nonzero, we can include these as constraints, and then disregard x_2 and x_3 . Further, we consider that the square root operation must be valid, and the numerator of the fraction inside the square root for x_1 is always positive, so we must have the denominator $(1 + \lambda_1 - \lambda_4)$ greater than 0.

Thus our dual is:

$$\begin{aligned} & (1 + \lambda_1 - \lambda_4) \left(\pm \sqrt{\frac{(b_4 \lambda_7 + b_5 \lambda_8)}{(1 + \lambda_1 - \lambda_4)}} \right) \\ \text{maximize} \quad & + \frac{b_4 \lambda_7}{\left(\pm \sqrt{\frac{(b_4 \lambda_7 + b_5 \lambda_8)}{(1 + \lambda_1 - \lambda_4)}} \right)} \\ & + \frac{b_5 \lambda_8}{\left(\pm \sqrt{\frac{(b_4 \lambda_7 + b_5 \lambda_8)}{(1 + \lambda_1 - \lambda_4)}} \right)} \\ & - \lambda_1 b_1 - \lambda_2 b_2 \lambda_3 b_3 + \lambda_4 (b_1 - 32) + \lambda_5 (b_2 - 32) + \lambda_6 (b_3 - 32) \\ & + (\lambda_7 + \lambda_8) (\epsilon) \\ \text{subject to} \quad & \lambda \geq 0 \\ & (1 + \lambda_3 - \lambda_6 - \lambda_8) = 0 \\ & (1 + \lambda_2 - \lambda_5 - \lambda_7) = 0 \\ & (1 + \lambda_1 - \lambda_4) > 0 \end{aligned}$$

For our finalized dual, we note that the above function is always minimized for $x_1 = -\sqrt{\frac{(b_4 \lambda_7 + b_5 \lambda_8)}{(1 + \lambda_1 - \lambda_4)}}$ rather than the positive version. This follows because every term multiplied by this number, or multiplied by the inverse of this number is constrained to be non-negative. $1 + \lambda_1 - \lambda_4$ as well as $b_4 \lambda_7$ and $b_5 \lambda_8$ must all be positive.

DUAL:

$$\begin{aligned} & (1 + \lambda_1 - \lambda_4) \left(-\sqrt{\frac{(b_4 \lambda_7 + b_5 \lambda_8)}{(1 + \lambda_1 - \lambda_4)}} \right) \\ \text{maximize} \quad & + \frac{b_4 \lambda_7}{\left(-\sqrt{\frac{(b_4 \lambda_7 + b_5 \lambda_8)}{(1 + \lambda_1 - \lambda_4)}} \right)} \\ & + \frac{b_5 \lambda_8}{\left(-\sqrt{\frac{(b_4 \lambda_7 + b_5 \lambda_8)}{(1 + \lambda_1 - \lambda_4)}} \right)} \\ & - \lambda_1 b_1 - \lambda_2 b_2 \lambda_3 b_3 + \lambda_4 (b_1 - 32) + \lambda_5 (b_2 - 32) + \lambda_6 (b_3 - 32) \\ & + (\lambda_7 + \lambda_8) (\epsilon) \\ \text{subject to} \quad & \lambda \geq 0 \\ & (1 + \lambda_3 - \lambda_6 - \lambda_8) = 0 \\ & (1 + \lambda_2 - \lambda_5 - \lambda_7) = 0 \\ & (1 + \lambda_1 - \lambda_4) > 0 \end{aligned}$$

2.3 KKT Conditions

Primal Feasibility:

$$\begin{aligned}
 x_1 - b_1 &\leq 0, \\
 x_2 - b_2 &\leq 0, \\
 x_3 - b_3 &\leq 0, \\
 b_1 - x_1 - 32 &\leq 0, \\
 b_2 - x_2 - 32 &\leq 0, \\
 b_3 - x_3 - 32 &\leq 0, \\
 \frac{b_4}{x_1} - x_2 - \epsilon &\leq 0 \\
 \frac{b_5}{x_1} - x_3 - \epsilon &\leq 0
 \end{aligned}$$

Dual Feasibility:

$$\begin{aligned}
 \lambda &\succeq 0 \\
 (1 + \lambda_3 - \lambda_6 - \lambda_8) &= 0 \\
 (1 + \lambda_2 - \lambda_5 - \lambda_7) &= 0 \\
 (1 + \lambda_1 - \lambda_4) &> 0
 \end{aligned}$$

Complementary Slackness:

$\lambda_i f_i(x) = 0$, $\forall i$ means we have:

$$\begin{aligned}
 \lambda_1(x_1 - b_1) &= 0, \\
 \lambda_2(x_2 - b_2) &= 0, \\
 \lambda_3(x_3 - b_3) &= 0, \\
 \lambda_4(b_1 - x_1 - 32) &= 0, \\
 \lambda_5(b_2 - x_2 - 32) &= 0, \\
 \lambda_6(b_3 - x_3 - 32) &= 0, \\
 \lambda_7\left(\frac{b_4}{x_1} - x_2 - \epsilon\right) &= 0 \\
 \lambda_8\left(\frac{b_5}{x_1} - x_3 - \epsilon\right) &= 0
 \end{aligned}$$

Stationarity: $\nabla_x L(x, \lambda) = 0$, or equivalently:

$$\begin{aligned}
 x_1 &= \pm \sqrt{\frac{(b_4 \lambda_7 + b_5 \lambda_8)}{(1 + \lambda_1 - \lambda_4)}} \\
 x_2 &= \begin{cases} 0, & (1 + \lambda_2 - \lambda_5 - \lambda_7) = 0 \\ \infty, & (1 + \lambda_2 - \lambda_5 - \lambda_7) < 0 \\ -\infty, & (1 + \lambda_2 - \lambda_5 - \lambda_7) > 0 \end{cases} \\
 x_3 &= \begin{cases} 0, & (1 + \lambda_3 - \lambda_6 - \lambda_8) = 0 \\ \infty, & (1 + \lambda_3 - \lambda_6 - \lambda_8) < 0 \\ -\infty, & (1 + \lambda_3 - \lambda_6 - \lambda_8) > 0 \end{cases}
 \end{aligned}$$

3 Approaches

We take two routes towards solving this problem. The first is to present an exact, closed form solution for the primal formulation above when epsilon is exactly 0. The second is to use a programmatic mixed integer quadratically constrained program solver (MIQCP) to handle two additional challenges - in the Pokémon game, x_1 , x_2 , and x_3 are constrained to be integers, and we also consider epsilon nonzero.

3.1 Computational Complexity

3.1.1 Exact

Solving the optimization problem as an MIQCP is NP hard. This led to the investigation of finding the closed form solution to have a faster runtime. The closed form solution is derived section 4.5 and

it is proved to have a runtime of $O(1)$ when ϵ is 0. The non-trivial nature of some of the quasi-convex constraints led to expressing some of the constraints as linear expressions in order to find the closed form. Linearizing quasi-convex constraints is frequently done in mixed integer programs [6].

3.1.2 Mixed Integer Quadratically Constrained Program (MIQCP)

Due to the quasi-convex nature of the constraints, the primal problem is not linear. One way to tackle this challenge is to use a MIQCP solver. A Mixed Integer Quadratically Constrained Program (MIQCP) is a type of optimization problem that involves finding the optimal values of a set of decision variables, subject to a set of linear and quadratic constraints, as well as the requirement that some of the variables must be integer. The objective function of an MIQCP is typically a quadratic function, which may be either maximized or minimized subject to the constraints. The constraints in an MIQCP may be linear or quadratic in nature, and may include both equality and inequality constraints. The variables in the problem may be either continuous or discrete (i.e., integer-valued).

Solving an MIQCP is a computationally challenging task, as it involves finding the optimal values of both continuous and discrete variables, subject to complex nonlinear constraints. However, there are a variety of algorithms and software packages available for solving MIQCPs, including branch-and-bound methods, interior-point methods, and global optimization techniques.

We decided to use IBM's DOcplex to code our problem in Python. Although MIQCP's are NP hard, for all the practical problems we explore, the runtime has been small. The flexibility of DOcplex allowed for x_1 , x_2 , and x_3 to be constrained to integers.

The quasi-convex and integer constraints on x_1 , x_2 , and x_3 lead to the MICQP to be non-convex. In order to solve a non-convex problem, the IBM DOcplex library solves the MIQCP with either the branch and bound search tree based method or an outer approximation method relying on the continuous LP solver. The branch and bound method breaks the optimization problem into smaller problems and eliminates smaller problems that cannot contain the optimal solution [7]. In contrast, the outer approximation method approximates the quadratic constraints with linear constraints. These linear constraints are updated and refined to find an optimal solution to the quadratic constraints [11]. DOcplex will choose the strategy that works best for a particular MICQP in order to solve the problem in the most optimal way.

4 Results

In this section we discuss our experimental results running a generated dataset representative of players in Pokémon against the baseline, MIQCP, and closed form solution.

4.1 Dataset

The dataset used is generated by gathering data from the Pokémon API [10]. The API is queried for the attack and defense statistics for various Pokémon. The type of defender affects the variables b_1 , b_2 , and b_3 and the type of attacker affects the variables b_4 , and b_5 . Once the base level stats are queried from the API for the attacker and defender \mathbf{b} can be calculated. b_1 is calculated by computing $base_1 + 75 + 32$ and b_2 and b_3 are calculated by $base_i + 20 + 32$. These correspond to the maximum values of hit points, defense, and special defense for a Pokémon under tournament conditions.¹

The attacker variables of b_4 and b_5 are calculated by using the damage formula [4] of $base_power_i * (base_offense_i + 20 + 32) * \frac{2}{5} * 1.5 * \frac{4}{3}$. This corresponds to an approximation of the damage dealt, before considering the defender's stats. Base power is determined on a per Pokémon basis - we search the list of all attacks a Pokémon knows from the API (attacks being the mechanism a Pokémon uses to do damage), and pick the maximum value possible for that Pokémon². Through this data generation, 145 samples were calculated with various attackers and defenders and used in the experiments.

¹Tournament conditions refer to a Pokémon being level 50

²The 1.5 and 4/3 factors, respectively, refer to the bonus for Pokémon attacking with a move of the same "type" as the Pokémon, while activating a special damage-bosting mechanic called "terastallization" - this corresponds to some of the strongest attack situations possible in the Pokémon game

4.2 MIQCP

The MIQCP was run for all the various samples in the dataset.

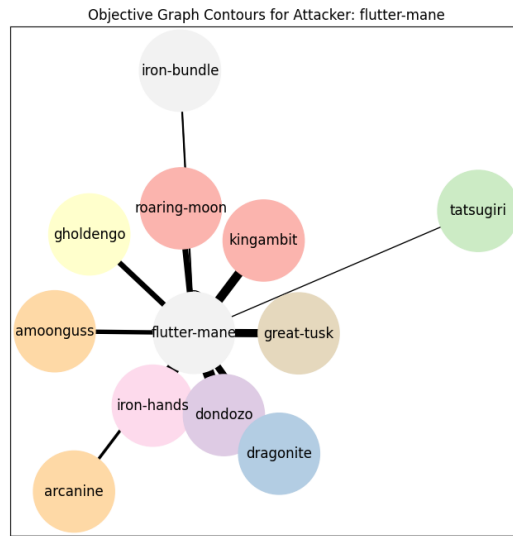


Figure 1: This spring graph is for the attacker flutter-mane. It shows which defenders flutter-mane has the lowest objective for. The further a node is the higher the objective value it has. It can be seen through this diagram that flutter-mane has the lowest objective with iron-hands. Therefore, flutter-mane would perform the best against iron-hands. Here, nodes that are closer together have a higher objective value (as represented by the thicker edges).

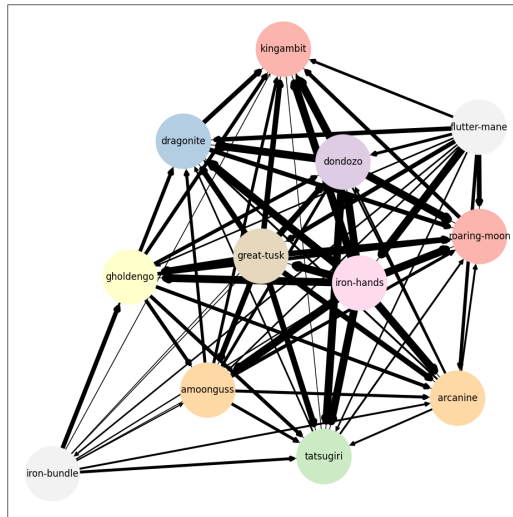


Figure 2: In spring graph above, the Pokémon are represented as nodes where a directed edge is from an attacker to a defender. If the solution is infeasible to the optimization problem then no edge is drawn. The higher the objective value is between attacker and defender, the further the nodes will be. Here it can be seen that when iron-bundle is the attacker it generally has a higher objective value compared to the other attackers. By contrast, the attacker iron-hands has an overall lower objective than the other attackers.

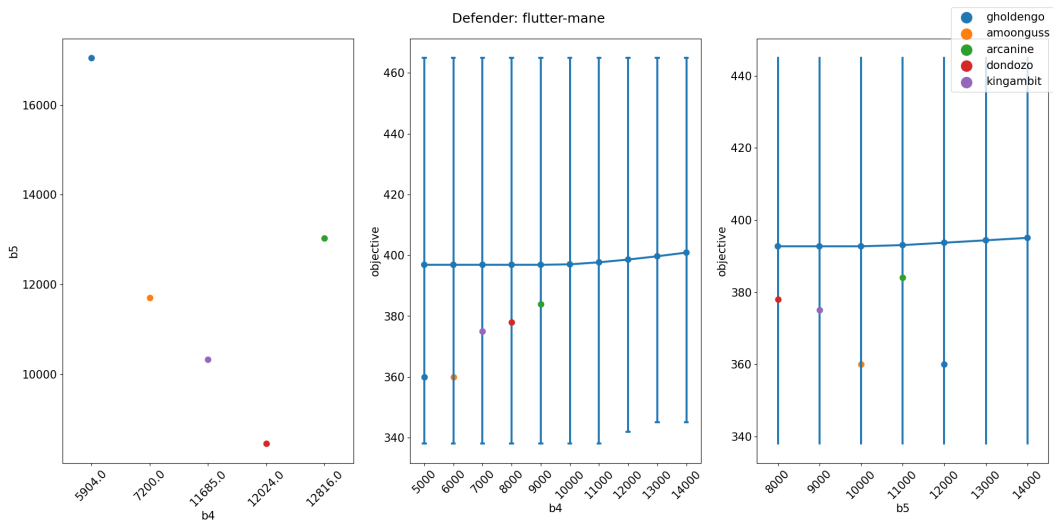


Figure 3: These scatter plots display 3 projections when the defender is flutter-mane. The leftmost graph plots the b_4 value against the b_5 value when the attackers are gholdengo, amoonguss, arcanine, dondozo, and kingambit. The last two graphs plot the b_4 or b_5 value against the *objective value* for the attackers. It also displays vertical bars that represent for each x-axis value what the maximum and minimum objective would be given b_5 or b_4 varies respectively. The horizontal line represents the average objective value when the x-axis varies. From these projections we can conclude that lower b_4 values and higher b_5 values lead to a lower objective value and the objective value is highly sensitive to all values of b_4 and b_5 .

Objective Value Contours for Defender: flutter-mane

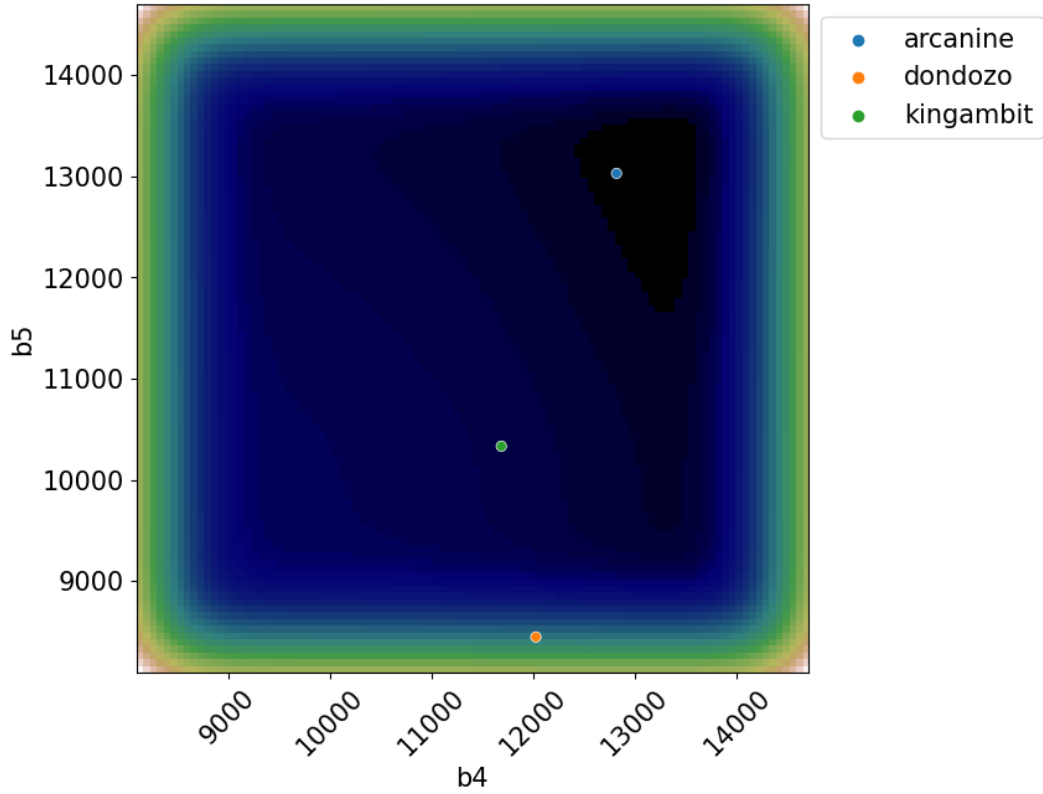


Figure 4: This graph represents when flutter-mane is the defender and plots the b_4 and b_5 values for 3 Pokémon. The contours on the graph display the objective value. The deeper blue illustrates a higher objective value. It can be shown that arcanine has the highest objective value when the defender is flutter-mane.

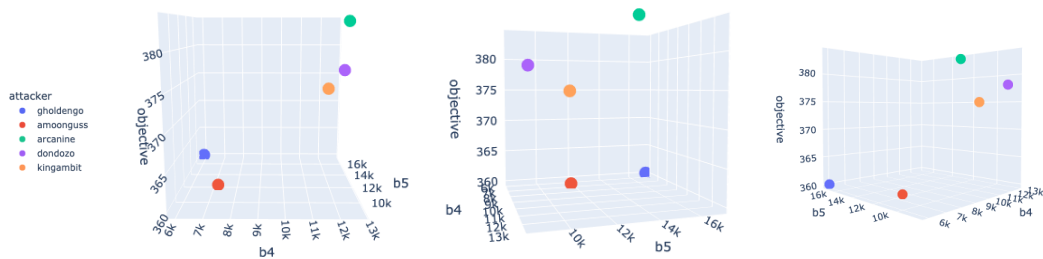


Figure 5: This is another 3D visualization when the attacker is flutter-mane. The different points display the various Pokémon defenders. This is a different way to visualize the lowest objective value for flutter-mane which is when the defender is gholdengo.

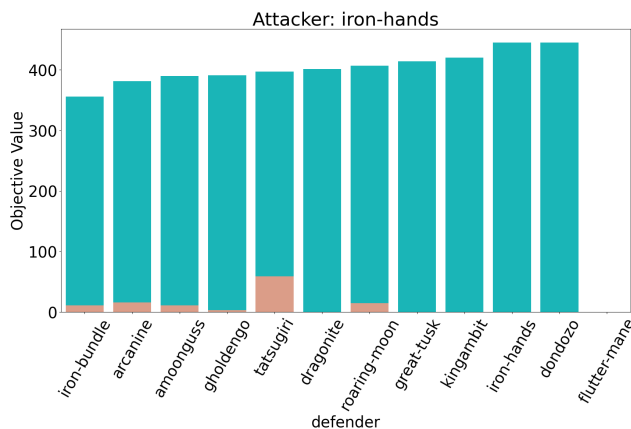


Figure 6: This plot shows the objective values realized for different choices of defending Pokémon (each one corresponds to a specific b_1, b_2, b_3 actually seen in practice) with a fixed attacking Pokémon, Iron Hands (which determines b_4, b_5). Although b_4 and b_5 are fixed, the choice of b_1, b_2, b_3 still plays a substantial role in the realized objective value. Since b_1, b_2 , and b_3 determine both a lower and an upper bound on each x , we show, in salmon, the portion of the objective that is in excess of the lower bound (that is, salmon shows our objective minus the lower bound of $b_1 + b_2 + b_3 - 96$, while teal+salmon together shows our realized objective). Many defenders in this dataset do not need to exceed their lowest possible objective at all, because their lowest possible objective is quite high (i.e. dondozo). Others need to exceed their lowest possible bound by only a small amount, reflecting that most Pokémon are able to handle strong attacks from this attacker with minimal cost. Some Pokémon, however, need to exceed their lowest bound from b_1, b_2 and b_3 by quite a lot, such as tatsugiri, even reaching total objective values that are much higher than some other Pokémon that did not exceed their lower bounds. This reflects a situation where b_1, b_2 and b_3 are far off from from what the optimal distribution would be based on b_4 and b_5 . Tatsugiri has a low b_2 value relative to b_1 and b_3 , but the b_4 value due to iron hands is much higher than b_5 . A similar factor is at play for flutter mane, for which there is no feasible solution.

4.3 MIQCP compared with baseline

We consider two state of the art approaches, detailed by top tier competitive Pokemon players. The first, automated approach which we use as a baseline is to consider a small set of candidate solutions, and pick the smallest feasible one in this set [15]. The set we use is $\left\{ \begin{bmatrix} b_1 - 32 \\ b_2 - 32 \\ b_3 - 32 \end{bmatrix}, \begin{bmatrix} b_1 \\ b_2 - 32 \\ b_3 - 32 \end{bmatrix}, \begin{bmatrix} b_1 \\ b_2 \\ b_3 - 32 \end{bmatrix}, \begin{bmatrix} b_1 - 32 \\ b_2 - 32 \\ b_3 \end{bmatrix}, \begin{bmatrix} b_1 - 32 \\ b_2 - 16 \\ b_3 - 16 \end{bmatrix}, \begin{bmatrix} b_1 - 32 \\ b_2 \\ b_3 \end{bmatrix} \right\}$, based on [15]. Again, we pick the smallest feasible solution in this set, based on b_4 and b_5 .

Another state of the art approach does exist [5]. This state of the art approach, written by a world champion, requires many steps of brute force searching through regions likely to be optimal, with many steps of backtracking. It is not feasible as an automated approach for a beginner to employ when helping them to learn the game, nor is it a fast approach. In support of the prior approach as a baseline, we note that World Champion Glick mentions only employing the elaborate approach when certain it is needed [15].

Our approach aims to beat the baseline [15] with higher quality solutions, while being much faster and more automated than the non-algorithmic, expert by-hand approach [5].

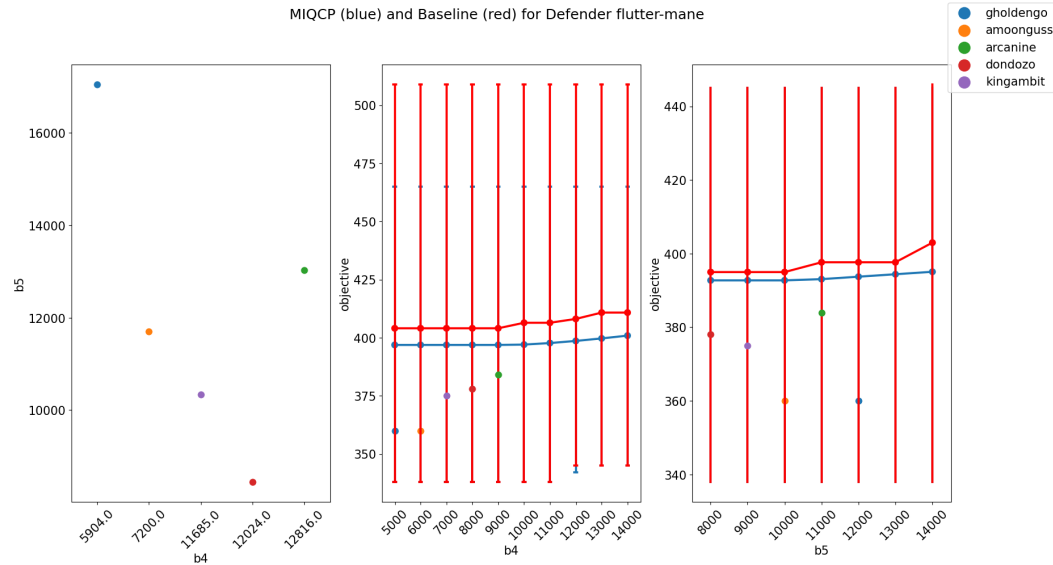


Figure 7: These scatter plots display 3 projections where the defender is flutter-mane. It is very similar to the figure above, but compares the MIQCP (in blue) with the baseline strategy (in red). The horizontal blue and red lines represent the average MIQCP and baseline objective values respectively. It is clear that the average MIQCP outperforms the average baseline across all values of b_4 and b_5 . The vertical bars represent the maximum and minimum objective value for varying values of b_5 and b_4 respectively. It can be seen that the MIQCP has tighter bounds than the baseline method. The bounds for b_4 range from 5000 to 16000 and the bounds for b_5 range from 8000 to 22000, roughly following the ranges seen in real-world game play.

4.4 MIQCP compared with closed form

A closed form solution was found to the MIQCP. It can be see from the below chart that performs very close to the MIQCP but with a much faster run time of $O(1)$.

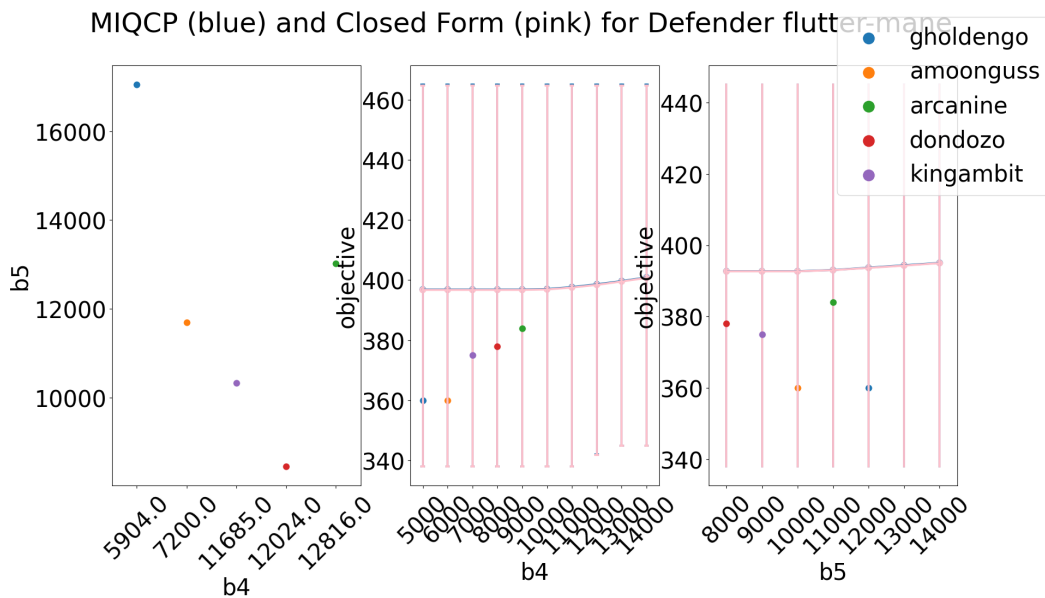


Figure 8: These scatter plots display 3 projections where the defender is flutter-mane. It is very similar to the figure above, but compares the MIQCP (in blue) with the closed form (in pink). The horizontal blue and pink lines represent the average MIQCP and closed form objective values respectively. It is clear that the average MIQCP and the closed form perform very similarly across all values of b_4 and b_5 . The vertical bars represent the maximum and minimum objective value for varying values of b_5 and b_4 respectively. It can be seen that the MIQCP and the closed form have very similar bounds. The MIQCP is sometimes slightly larger in objective because of the additional constraint in the MIQCP that all x values are integers.

4.5 Analytical Solution (closed form)

We solve the closed form for the case where ϵ is 0, although very slight modifications can be made for validity in all cases. Despite the relatively simple nature of our problem, this derivation is far from trivial - the quasiconvex nature of the constraints means many results do not follow directly from the material we discussed in class, focusing on convex constraints.

Rewrite formulation as:

Start from the formulation we used for finding the dual:

$$\begin{aligned} \text{minimize} \quad & x_1 + x_2 + x_3 \\ \text{s.t.} \quad & x_1 - b_1 \leq 0, \\ & x_2 - b_2 \leq 0, \\ & x_3 - b_3 \leq 0, \\ & b_1 - x_1 - 32 \leq 0, \\ & b_2 - x_2 - 32 \leq 0, \\ & b_3 - x_3 - 32 \leq 0, \\ & \frac{b_4}{x_1} - x_2 \leq 0 \\ & \frac{b_5}{x_1} - x_3 \leq 0 \end{aligned}$$

Simplify:

$$\begin{aligned} \text{minimize} \quad & x_1 + x_2 + x_3 \\ \text{s.t.} \quad & x_1 - b_1 \leq 0, \\ & x_2 - b_2 \leq 0, \\ & x_3 - b_3 \leq 0, \\ & b_1 - x_1 - 32 \leq 0, \\ & b_2 - x_2 - 32 \leq 0, \\ & b_3 - x_3 - 32 \leq 0, \\ & x_2 \geq \frac{b_4}{x_1} \\ & x_3 \geq \frac{b_5}{x_1} \end{aligned}$$

Note that the derivative of our convex objective with respect to x_2 and x_3 is positive and constant, so x_2 and x_3 should always be the smallest possible. Looking at all constraints on x_2 , we know either the solution is infeasible, or $x_2 = \max(\frac{b_4}{x_1}, b_2 - 32)$ for some x_1 . Similarly, $x_3 = \max(\frac{b_5}{x_1}, b_3 - 32)$. Substituting this in for the older constraints, we have:

$$\begin{aligned} \text{minimize} \quad & x_1 + x_2 + x_3 \\ \text{s.t.} \quad & x_1 - b_1 \leq 0, \\ & x_2 - b_2 \leq 0, \\ & x_3 - b_3 \leq 0, \\ & b_1 - x_1 - 32 \leq 0, \\ & b_2 - x_2 - 32 \leq 0, \\ & b_3 - x_3 - 32 \leq 0, \end{aligned}$$

$$\begin{aligned} \text{minimize} \quad & x_1 + \max\left(\frac{b_4}{x_1}, b_2 - 32\right) + \max\left(\frac{b_5}{x_1}, b_3 - 32\right) \\ \text{s.t.} \quad & x_1 - b_1 \leq 0, \\ & b_2 \geq \max\left(\frac{b_4}{x_1}, b_2 - 32\right), \\ & b_3 \geq \max\left(\frac{b_5}{x_1}, b_3 - 32\right), \\ & b_1 - x_1 - 32 \leq 0, \\ & x_2 = \max\left(\frac{b_4}{x_1}, b_2 - 32\right) \\ & x_3 = \max\left(\frac{b_5}{x_1}, b_3 - 32\right) \end{aligned}$$

$$\begin{aligned} \text{minimize} \quad & \left(x_1 + \max\left(\frac{b_4}{x_1}, b_2 - 32\right) + \max\left(\frac{b_5}{x_1}, b_3 - 32\right)\right) \\ \text{s.t.} \quad & x_1 - b_1 \leq 0, \\ & b_2 \geq \frac{b_4}{x_1}, \\ & b_3 \geq \frac{b_5}{x_1}, \\ & b_1 - x_1 - 32 \leq 0, \end{aligned}$$

$$\begin{aligned}
& \text{minimize} && \left(x_1 + \max\left(\frac{b_4}{x_1}, b_2 - 32\right) + \max\left(\frac{b_5}{x_1}, b_3 - 32\right) \right) \\
& \text{s.t.} && x_1 \leq b_1, \\
& && x_1 \geq \frac{b_4}{b_2}, \\
& && x_1 \geq \frac{b_5}{b_3}, \\
& && x_1 \geq b_1 - 32,
\end{aligned}$$

$$\begin{aligned}
& \text{minimize} && \left(x_1 + \max\left(\frac{b_4}{x_1}, b_2 - 32\right) + \max\left(\frac{b_5}{x_1}, b_3 - 32\right) \right) \\
& \text{s.t.} && b_1 \geq x_1 \geq \max\left(\frac{b_4}{b_2}, \frac{b_5}{b_3}, b_1 - 32\right)
\end{aligned}$$

Ideally, we would redefine the objective to be a piecewise function:

$$\text{define } f(x_1) = \begin{cases} x_1 + \frac{b_4}{x_1} + \frac{b_5}{x_1}, & x_1 \leq \min\left(\frac{b_4}{b_2-32}, \frac{b_5}{b_2-32}\right) \\ x_1 + \frac{b_4}{x_1} + b_3 - 32, & \frac{b_5}{b_3-32} \leq x_1 \leq \frac{b_4}{b_2-32} \\ x_1 + b_2 - 32 + \frac{b_5}{x_1}, & \frac{b_4}{b_2-32} \leq x_1 \leq \frac{b_5}{b_3-32} \\ x_1 + b_2 - 32 + b_3 - 32, & x_1 \geq \max\left(\frac{b_4}{b_2-32}, \frac{b_5}{b_2-32}\right) \end{cases}$$

Noting that for a given value of b_2, b_3, b_4, b_5 we only need one of the two middle cases (the objectives are equivalent when $\frac{b_4}{b_2-32} = x_1 = \frac{b_5}{b_3-32}$, and other than that only at most one of the conditions can be true), we redefine without loss of generality (b_2, b_4) such that $b_4/(b_2 - 32) \geq b_5/(b_3 - 32)$ (we do this by swapping the old (b_2, b_4) with (b_3, b_5) if needed, and noting that this does not affect the objective).

$$\text{now we have } f(x_1) = \begin{cases} x_1 + \frac{b_4}{x_1} + \frac{b_5}{x_1}, & x_1 \leq \min\left(\frac{b_4}{b_2-32}, \frac{b_5}{b_2-32}\right) \\ x_1 + \frac{b_4}{x_1} + b_3 - 32, & \frac{b_5}{b_3-32} \leq x_1 \leq \frac{b_4}{b_2-32} \\ x_1 + b_2 - 32 + b_3 - 32, & x_1 \geq \max\left(\frac{b_4}{b_2-32}, \frac{b_5}{b_2-32}\right) \end{cases}$$

This piecewise function satisfies some nice properties, namely that it corresponds exactly to the set of maximums

$$f(x_1) = \max\left(x_1 + \frac{b_4}{x_1} + \frac{b_5}{x_1}, x_1 + \frac{b_4}{x_1} + b_3 - 32, x_1 + b_2 - 32 + b_3 - 32\right)$$

And therefore, because for positive b and the domain $x_1 > 0$ all 3 of those component functions are convex, and a maximum of convex functions is still convex, this objective is still convex, despite the fact that we have now moved the quasiconvex constraints into the objective.

So we now have:

$$\begin{aligned}
& \text{minimize} && f(x_1) \\
& \text{s.t.} && b_1 \geq x_1 \geq \max\left(\frac{b_4}{b_2}, \frac{b_5}{b_3}, b_1 - 32\right)
\end{aligned}$$

Since we have a convex function with convex constraints now, we can solve as normal, taking the derivative and setting it to 0. At points where the boundary prevents us from reaching the derivative at 0, since we are in just one dimension we can simply pick the boundary point closest to the point where the derivative is 0. If no points satisfy the constraint then there is no solution.

Take the derivative:

$$f'(x_1) = \begin{cases} 1 - \frac{b_4}{x_1^2} - \frac{b_5}{x_1^2}, & x_1 \leq \min\left(\frac{b_4}{b_2-32}, \frac{b_5}{b_2-32}\right) \\ 1 - \frac{b_4}{x_1^2}, & \frac{b_5}{b_3-32} \leq x_1 \leq \frac{b_4}{b_2-32} \\ 1 & x_1 \geq \max\left(\frac{b_4}{b_2-32}, \frac{b_5}{b_2-32}\right) \end{cases}$$

Set to 0:

$$\text{We hit 0 if: } \begin{cases} 1 - \frac{b_4}{x_1^2} - b_5 x_1^2 = 0, & x_1 \leq \min\left(\frac{b_4}{b_2-32}, \frac{b_5}{b_2-32}\right) \\ 1 - \frac{b_4}{x_1^2} = 0, & \frac{b_5}{b_3-32} \leq x_1 \leq \frac{b_4}{b_2-32} \\ 1 & x_1 \geq \max\left(\frac{b_4}{b_2-32}, \frac{b_5}{b_2-32}\right) \end{cases}$$

$$\text{So our possible derivatives being 0 are: } \begin{cases} x_1 = \sqrt{(b_4 + b_5)}, & \sqrt{(b_4 + b_5)} \leq \min\left(\frac{b_4}{b_2-32}, \frac{b_5}{b_2-32}\right) \\ x_1 = \sqrt{b_4} & \frac{b_5}{b_3-32} \leq \sqrt{b_4} \leq \frac{b_4}{b_2-32} \end{cases}$$

So if one of these cases exists, our optimum is either at x_1 equal to this or x_1 as close as possible given the upper and lower bounds on x_1 .

The derivative is not defined at a few points in our function ($\frac{b_5}{b_3-32}$ and $\frac{b_4}{b_2-32}$) so we do have to check these explicitly as well (if they are feasible), and take the minimum across these and the 0 values for the derivatives.

If the derivative is not 0, then we know the solution lies at a boundary point (or at one of those points with undefined derivative). So our solution becomes:

List the $O(1)$ number of possible critical points:

$$C = \{b_1, \max(\frac{b_4}{b_2}, \frac{b_5}{b_3}, b_1 - 32), \sqrt{(b_4 + b_5)}, \sqrt{b_4}, \frac{b_5}{b_3 - 32}, \frac{b_4}{b_2 - 32}\}$$

Remove all elements in C that are infeasible

$$C' = \{c | b_1 \geq c \geq \max(\frac{b_4}{b_2}, \frac{b_5}{b_3}, b_1 - 32), c \in C\}$$

And we have solution:

$$\min_{x_1 \in C'} f(x_1)$$

All of these steps are constant time, so we can solve this in constant time.

Our plan is to solve this problem across a wide range of attacker-defender pairs based on a dataset from a popular Pokémon simulator [1].

We then want to compare the optima we find, with the convex relaxation, to the exact optima one would find (with a much less computationally tractable approach!) on the true, non-convex damage formula (which contains some intermediate flooring steps). We expect to see approximate alignment, but there may be a few cases of larger difference. We would like to compare not only the optima, but also the regions we are optimizing over, to see how they differ with the smoothed, convex versions of the regions vs the non-convex exact regions. We expect to see a larger feasible region with the exact formulation than the convex approximation of the damage formula, because the convex relaxation of the damage formula is an upper bound on the exact version. The exact way that it differs will be intriguing to visualize.

5 Conclusion and Future Work

In this paper, we have proposed a novel approach for optimizing Pokémon strategy using convex optimization techniques. Our approach involves formulating the problem of selecting the optimal strategy in a battle and moves as a convex optimization problem, and solving it using both an exact closed form solution and an MIQCP. We have demonstrated the effectiveness of our approach through experimentation with both real data gathered from the Pokémon API and from generated data using the range of attack and defense constants. Our results demonstrate that our method outperforms the baseline strategy method.

Our approach has several limitations that can be addressed in future work. First, our constraints and primal formulation could be adapted to more closely resemble true game play.

Second, our approach considers only one battle at a time, and it does not account for the overall strategy of the player. Therefore, another interesting direction for future work is to develop a multi-battle strategy that takes into account the player's overall goals and the opponents' tendencies.

In conclusion, our work demonstrates the potential of convex optimization techniques for optimizing Pokémon strategy. We believe that our approach can be extended and improved to address more complex and challenging scenarios in the future.

References

- [1] Smogon usage stats. <https://www.smogon.com/stats/>. Accessed: 2023-02-01.
- [2] Pablo Barros and Alessandra Sciutti. All by myself: Learning individualized competitive behavior with a contrastive reinforcement learning optimization. *Neural Networks*, 150:364–376, 2022.
- [3] Murray Campbell, A Joseph Hoane Jr, and Feng-hsiung Hsu. Deep blue. *Artificial intelligence*, 134(1-2):57–83, 2002.
- [4] III Craft, Leonard. Dawoblefet's damage dissertation. <https://www.trainertower.com/dawoblefets-damage-dissertation>. Accessed: 2023-01-31.
- [5] Wolfe Glick. Eving #3 - how to make a complex ev spread. <https://www.vgcguide.com/eving-3>. Accessed: 2023-03-22.
- [6] Mahdi Hamzeei and James Luedtke. Linearization-based algorithms for mixed-integer nonlinear programs with convex continuous relaxation. *Journal of Global Optimization*, 59(2-3):343–365, 2014.
- [7] He He, Hal Daume III, and Jason M Eisner. Learning to search in branch and bound algorithms. *Advances in neural information processing systems*, 27, 2014.
- [8] Dan Huang and Scott Lee. A self-play policy optimization approach to battling pokémon. In *2019 IEEE Conference on Games (CoG)*, pages 1–4. IEEE, 2019.
- [9] Akshay Kalose, Kris Kaya, and Alvin Kim. Optimal battle strategy in pokemon using reinforcement learning. Web: <https://web.stanford.edu/class/aa228/reports/2018/final151.pdf>, 2018.

- [10] Pokeapi. <https://pokeapi.co/>.
- [11] Andrea Qualizza, Pietro Belotti, and François Margot. Linear programming relaxations of quadratically constrained quadratic programs. In *Mixed Integer Nonlinear Programming*, pages 407–426. Springer, 2012.
- [12] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.
- [13] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dhharshan Kumaran, Thore Graepel, et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018.
- [14] J Cole Smith and Z Caner Taskin. A tutorial guide to mixed-integer programming models and solution techniques. *Optimization in Medicine and Biology*, pages 521–548, 2008.
- [15] Aaron Traylor. Eving #1 - how to make simple ev spreads. <https://www.vgcguide.com/eving-1-how-to-make-simple-ev-spreads>. Accessed: 2023-03-22.
- [16] Berk Ustun, Alexander Spangher, and Yang Liu. Actionable recourse in linear classification. In *Proceedings of the conference on fairness, accountability, and transparency*, pages 10–19, 2019.
- [17] Daniel Verdear and Ubbo Visser. Ontology-based knowledge system and team verification tool for competitive pokemon. In *The International FLAIRS Conference Proceedings*, volume 34, 2021.

A Appendix

B Task Assignments

All Tasks:

1. **Introduction**
2. **Motivation**
3. Previous Work
4. Intended Contributions
5. Organization of the Paper
6. **Statement of the Problem**
7. **Primal Formulation**
8. Dual Formulation
9. KKT Conditions
10. Intended Approaches
11. **Conjectured Results**
12. Conclusion & Possible Future Works
13. **References**

B.1 Hayden

1. Dual Formulation
2. Related Work
3. Motivation
4. KKT conditions
5. Closed Form Solution
6. Baseline & State-of-the-art formulation
7. KKT Conditions

B.2 Avni

1. Primal Formulation
2. Intended Approaches
3. Experiments
4. MIQCP infrastructure
5. Visualization

B.3 Hailey

1. Synthetic Data Generation
2. Experiment Results
3. Visualizations
4. Dual Formulation
5. Introduction
6. Previous Works
7. References