Optimisation of Resource Allocation Algorithm
Linear Programming (The Money Ball Problem)

Nikhil Sasi

School of Mathematical Sciences
University College Cork
Ireland
November 2025

This report is wholly the work of the author, except where explicitly stated otherwise. The source of any material which was not created by the author has been clearly cited.

Date: 28/11/2025

Signature: Nikhil Sasi

Contents

Chapter 1

Applied Linear Programming at Scale

1.1 Introduction

Linear Programming is widely used in real world applications at scale, often involving thousands of decision variables and dozens of constraints.

At large scales, some Simplex assumptions become much more noticeable, and their limitations start to matter which can lead to numerical issues or unsolvable formulations, So it becomes important to revisit these assumptions.

We will discuss how these assumptions affect interpretation of LP outputs, the limitations that effect them and why the Simplex method becomes unsuitable at scale. And when other algorithms need to be considered.

1.2 Case Study : Football League 2025 - Moneyball

To demonstrate LP at scale, we implemented the Simplex method on the 2024 League of Ireland Premier Division dataset.[1] Building upon the concepts discussed in the previous chapter, we formulated two key performance metrics a weighted Defence Metric (Defence power or DP) based on goals saved and an Attack Metric (Attack power or AP) based on Goals scored per match, representing team strength across offensive and defensive dimensions. Each player has a cost value associated with them which is representative of the money required to acquire them for our "Moneyball team".

Ideally, we identify our threshold requirement based on data from a set of winning team's data and compute their overall Attack and Defence metric composition, but since we lack this data, we analysed the distribution of stats for all teams (assuming $4\times$ top 2% defenders and $4\times$ top 2% attackers) as constituent to the "Moneyball" team. So our performance constraint will require a team to have

$$\text{Defence threshold} = 4 \times \text{Top 2\% player Defence metric}$$

$$\text{Attack threshold} = 4 \times \text{Top 2\% player Attack metric}$$

1.2.1 Structure

The primary objective is to construct a "Moneyball Team". Our data driven selection of players requires the players to:

- Meet or exceed predefined percentile thresholds in both metrics.
- Minimizes the total acquisition cost, achieving efficiency in resource allocation.
- Team size being capped at 10 players.

$$\text{Team acquisition cost} = Min(C^T X)$$

Such that,

$$\begin{pmatrix} \alpha_1 & \alpha_2 & \dots & \alpha_N & s_1 & 0 & 0 \\ \beta_1 & \beta_2 & \dots & \beta_N & 0 & s_2 & 0 \\ 1 & 1 & \dots & 1 & 0 & 0 & s_3 \end{pmatrix} . \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}$$

Here,

- x_i is the decision variable representing selection weight of player i
  - $x \in [0, 1]$
- α_i is the attack metric for the player i
- β_i is the defence metric for that player i
- s_1, s_2 and s_3 are the slack variables
- b_1, b_2 and b_3 are the threshold values for the 3 constraints
- C^T is the cost vector with γ_i being player acquisition cost for player i

1.2.2 Assumptions and Limitations

1.2.2.1 Linearity and Independence

1. One of the key assumptions made during this exercise and a core aspect of Linear programming is the linearity assumption. This dictates that the decision variables interact additively. This also enforces the lack of interaction terms between variables since that can introduce non linearity into the system. Real world systems often display non-linear interactions that violate this assumption

   For example, a strong defensive player can indirectly boost attacking capability by freeing midfielders to act more aggressively, an interaction LP cannot capture.

Interaction terms $= \alpha_i.\beta_i$	Not Considered
Non linear terms $= \alpha_i^2$ or $1/\beta_i$	Not Considered

2. The independence requirement dictates that each data points value should be independent of the values of other players. This assumption ignores synergy

effects, such as players who amplify each other's performance. Although unrealistic, we proceed with this simplification.

Presence of player 1 boosts Attack Metric of player 2	Not considered

1.2.2.2 Continuous decision variables

Simplex handles only continuous decision variables. Ideally the decision variables $x_1, x_2, \ldots, x_N$ would be binary. But the limitations of simplex allows only bounded continuous variables, so we settle for

$$\text{Simplex :} \qquad x \in [0, 1]$$

Limitation

We choose the 10 decision variables with the largest non zero values, But since Simplex optimises the objective using fractional decision variables, which lacks practical meaning in this context. This limitation distorts the optimisation outcome,

For example, even when we build a team whose Attack and Defence scores exceed the 99% percentile, the solver may return a team size of 11 because the fractional selections still sum to 10. Limiting the player to top 10 players in this case would violate the performance constraints

Workaround

Solving optimisation problems where decision variables can take discrete values as decision variables require different methodologies like Mixed Integer linear programming (MILP) or Integer linear programming (ILP). These are more conceptually and computationally harder problems. With MILP we can have

$$\text{MILP / ILP :} \qquad x \in \{0, 1\}$$

1.2.2.3 Bounded Constraints

All linear programming problems need to be bounded to apply the simplex algorithm. This means that the Feasible region needs to be finite and limited. So the performance constrains,

$$\alpha_1 x_1 + \alpha_2 x_2 + \cdots + \alpha_N x_N \geq \text{Attack Metric}$$
$$\beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_N x_N \geq \text{Defensive Metrics}$$

Need to be converted into bounded $(\leq)$ form.

Workaround

A very simple workaround would be to add a $-$ve sign to both sides of the inequality. This flips the inequality form along the constraint line and turns an unbounded *feasible region* into a bounded one.

$$-(\alpha_1 x_1 + \alpha_2 x_2 + \cdots + \alpha_N x_N) \leq -\text{Attack Metric}$$
$$-(\beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_N x_N) \leq -\text{Defensive Metric}$$

1.2.3 Outcome

1.2.3.1 The Moneyball team

The final selected team, which exceeds 98% of random teams, consists of:

	player	squad	value	defence_metric	attack_metric
202	Maurice Nugent	Galway United	€20k	1.241379	0.310000
206	Michael Duffy	Derry City	€10k	0.580645	0.190000
36	Bridel Bosakani	Drogheda United	€72k	0.155172	2.650000
11	Al Amin Kazeem	St Patrick's	€25k	0.972973	0.430000
14	Alex Nolan	St Patrick's	€19k	0.486486	0.250000
227	Rayhaan Tulloch	Shelbourne FC	€19k	0.666667	0.310000
198	Mason Melia	St Patrick's	€25k	0.486486	0.370000
235	Rob Slevin	Galway United	€19k	1.241379	0.090000

1.2.3.2 The Simplex algorithm outcome

The simplex solver stats and outcome is :

Table 1.1: Optimization Results Summary

Metric	**Value**
minimum required Attack Power (AP)	4.4848
minimum required Defence Power (DP)	5.3333
Solver Status	Optimization terminated successfully
Slack Values	$[-4.44 \times 10^{-15},\ 0.0,\ 2.625]$
Selected Team AP	4.6000
Selected Team DP	5.8312
Selected Team Cost	€209,000.00

1.2.4 Interpretation

- The constrains are all valid since the team minimum required team Attack Power and team Defence power threshold is met and exceeded.
- The slack variables are all non negative and 2 of them are 0 (The AP and DP limits are maximized) meanwhile the team size constraint is still positive non zero showing leeway for this constraint.
- The selected team cost is €209,000, the optimised objective cost to acquire this team.

1.3 Linear Programming in application

1.3.1 Conversion to standard form[2]

The standard form of Linear Programming requires bounded constraints, where each constraint is expressed as a minimizing inequality ($\leq$).

All equality and greater-than-or-equal-to constraints must be converted into bounded inequalities ($\leq$) to fit the standard form.

For applications where negative decision variables are allowed ($x < 0$) we need to add another variable on both sides of the inequality to turn it into a non-negativity inequality $x + k \geq 0$

The standard form also requires the *objective function* to be a maximization function.

However, Python's `scipy` module performs minimization by default, so functions that need to be maximized must have their signs flipped before solving.

The ultimate goal of the standard form here is to simplify the computation for LP applications. The true mathematical core is based on the Karush–Kuhn–Tucker or (KKT) conditions which itself are derived from the Lagrangian.

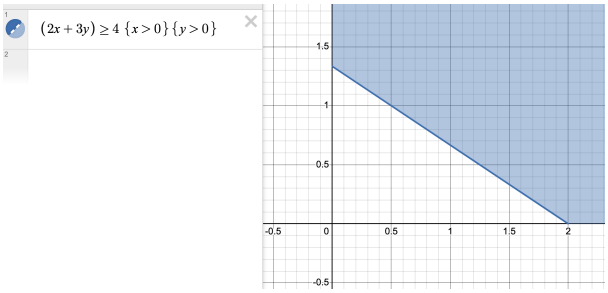

Figure 1.1: Unbounded constraint inequality

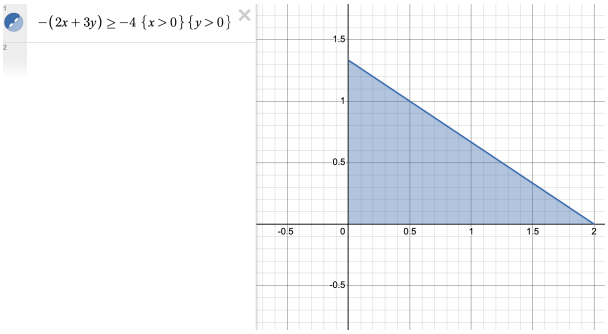

Figure 1.2: Bounded constraint inequality (Signs flipped both sides)

1.3.2 Intrinsic LP issues

1.3.2.1 Constraint redundancy

A *Redundant Constraint* is one that lie outside the *feasible region* and do not contribute new *BFS* points, these constraints are unnecessary and add complexity to the optima search and need to be removed.

1.3.2.2 Feasibility and Unfeasibility

When too few constraints exist, usually accompanied by a large amount of decision variables, we can have *Unfeasible* solutions that have no intersection points that can

act as *BFS*. This leads to the solver failing on the first step itself since there exists no solution for the optimisation problem.

Furthermore some solvers use advanced applications of *facial reduction* to reduce high dimensional decision variable space to smaller hyperplanes where the feasible region subspace lies. drastically reducing complexity. [4](Page 24, Image 4.1 and Page 25, table 4.1 from the citation shows much greater efficiency with *facial reduction*)

1.3.2.3 Degeneracy

Degeneracy is a dangerous condition that can exist for LP problems with a *Feasible Region* and valid BFSs and is usually harder to detect. It causes the algorithm to *stall* or even *cycle* during its pivots.Which can prevent Simplex from reaching optimum unless *anti-cycling rules* are used.

Degeneracy occurs when the *BFS* has more than needed constraints passing through it. This causes the basis variable which should be positive to become zero and which causes the pivot to fail to improve the objective value and hence optima may never be reached or take more time to reach.[4] (Section 2.1.1 has a brief mathematical description)

Most solvers have anti-cycling strategies like *blands rule*[3] and facial reduction [4] to prevent degeneracy from slowing convergence to optima.

1.3.3 Parameters affecting Linear Programming

1.3.3.1 Numerical Scaling

Scaling is the numerical operation where the coefficients and RHS of the inequalities - constraints and objective functions are rescaled to similar bounded ranges. This improves numerical stability and can reduce pivot count and iteration time. Depending on the scaling methodology and problem type, Improvements can range from minor to drastic.

Studies show that scaling can often have subjective outcomes and need to be considered case by case.[5]

Scaling of objective function showed no improvements in the 2024 League data set so it was not implemented.

1.3.4 Time complexity and comparisons of LP algorithms

Time complexity is another major cornerstone for LP algorithms. Continuous LP problems are polynomial time solvable in theory, but methods like Interior-point and simplex perform well on real world data (Simplex - despite its exponential worst case scenario). Furthermore presence of presolvers hasten the process even more.

We now examine how these presolvers influence time complexity through operations such as scaling, redundancy elimination, and anti-cycling detection.

The "HiGHS" Linear Programming algorithm [6] is particularly suited for this, as it incorporates advanced presolver routines that automatically reduce problem size and

improve numerical stability. These routines dynamically rescale coefficients, detect and remove redundant constraints, and eliminate non-essential decision variables, thus effectively simplifying the optimisation problem before the main Simplex iterations begin. It also has implementations of "Revised simplex", "Interior Point methods" and other advanced algorithms which are dynamically chosen for the problem.

A breakdown of LP solver performances show HiGHS to perform exceptionally good, having a respectable standing along other commercial algorithms.[7]

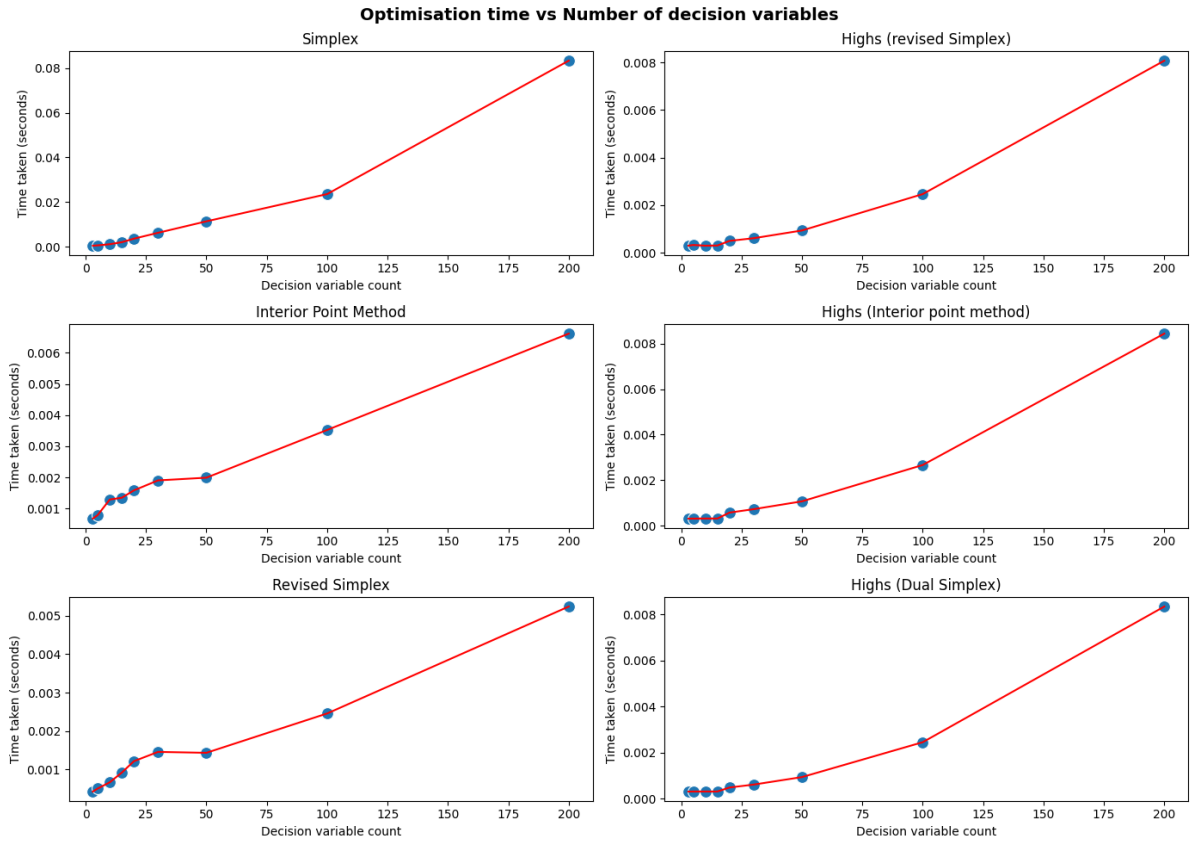

Figure 1.3: Time complexity (simplex, ipm and revised simplex) vs highs implementations

HiGHS performs on average almost 10× better than Simplex as can be confirmed by the performance comparison on the 2024 League of Ireland data (y-axis differs for both graph) with various player count considerations, and comparatively has a smoother increase in time complexity as compared to Interior Point Methods and Revised simplex for smaller decision variables.(Code in Jupyter Notebook)

There is still some overhead in HiGHS due to the numerous presolver operations. And IPM / Revised simplex does better than HiGHS implementation for smaller decision variables. Studies in HiGHS demonstrate much relatively better performance at very large scales[6].

Chapter 2

Conclusions

The objective of the project was to study and implement linear programming, we looked at one particular algorithm in detail - Simplex optimisation due to its prominence. We explored the algorithm through demonstration of the underlying theory, then further development of automated models concluded by a broader investigation and implementation of linear programming. The final outcome was a model capable of selecting a squad of players whom in theory could win the Irish premier league on the lowest budget.

Difficulties/ Vulnerabilities

Our league model features over 300 player entries across 10 teams with a large variety of statistics recorded for different in game actions. As with any data reflective of the real world it is difficult to quantify the reliability of such data. However, through logical decision making at the points of model instability we believe that our model provides a robust output depending on our user inputs.

The model required a combination of acute feature engineering of available statistics alongside evaluation of past team efforts to build realistic constraints that would act as guardrails to evaluate the teams on the right descriptors of match performance. Constraint building is a core part of LP problems and effort was taken to make these constraints still interpretable.

League implications

The motivation for this project came directly from Michael Lewis's 2003 book Moneyball:The Art of Winning an Unfair Game.[9] The book details how effective the use of league specific data analysis can be on picking a group of players that will win games. Although it can be assumed this report hasn't had the kind of wider implications on data collection and analysis for soccer that coach Billy Beane had for baseball, our model still highlights a unique way optimising a team capable of winning the Irish premiership.

Further research

What set Billy Bean and the Oakland Athletics team apart from existing data analysis methods was the importance placed on players 'on base percentage' this at the time was a novel statistic that few teams collected.[9] Our existing analysis focuses on two metrics that are harshly scrutinized goals scored and goals conceded. With further research a model that affiliates less common statistic such as yellow cards and player height could provide a valuable insight into team performance, however, this is speculative and should be explored.

Bibliography

[1] FBref 2024 League of Ireland Premier Division Stats. https://fbref.com/en/comps/80/2024/stats/2024-League-of-Ireland-Premier-Division-Stats

[2] Massachusetts Institute of Technology, *Lecture 1: Linear Programming (Spring 2005)*, MIT OpenCourseWare, YouTube, 2020. Available at: https://www.youtube.com/watch?v=WwMz2fJwUCg&t=1970s

[3] R. G. Bland, *New finite pivoting rules for the simplex method*, May 1977. Available at: https://www.cs.bu.edu/faculty/gacs/courses/cs530/lectures/BlandRule.pdf

[4] Haesol Im and Henry Wilkowicz, *Revisiting Degeneracy, Strict Feasibility, Stability, in Linear Programming* 2023. Available at: https://arxiv.org/abs/2203.02795

[5] J. A. Tomlin, *On Scaling Linear Programming Problems*, Mathematical Programming Studies, vol. 4, pp. 146–166, 1975. Available at: https://link.springer.com/content/pdf/10.1007/BFb0120718.pdf

[6] HiGHS: High Performance Linear Optimization Software. *HiGHS Official Documentation*, ERGO Code Project. Available at: https://ergo-code.github.io/HiGHS/dev/

[7] Mittelmann, H. (2025). LPopt Benchmark: Find Optimal Basic Solution. Arizona State University. Available at: https://plato.asu.edu/ftp/lpopt.html

[8] Transfermarkt, *League of Ireland Premier Division – Market Values*. Available at: https://www.transfermarkt.co.uk/premier-division/marktwerte/wettbewerb/IR1

[9] M. Lewis, *Moneyball: The Art of Winning an Unfair Game*. New York, NY, USA: W. W. Norton, 2003.

[10] "5-4-1 Formation," Footballizer. [Online]. Available: https://www.footballizer.com/academy/5-4-1-formation/.