Pop Group Optimization

Group 2 - Michelle Deng, Mishka Jain, Tanaya Kaundal, James Zhong

March 17, 2025

1 Abstract

This paper explores the application of combinatorial optimization techniques to form optimal music artist collaborations under budget constraints. We evaluate four methods—brute force search, greedy algorithms, integer programming, and simulated annealing—to maximize group potential by analyzing artists' skills, collaboration compatibility, and cost considerations. Using data from Spotify's top 25 artists with the most monthly listeners, we demonstrate that simulated annealing achieves optimal results with significantly less computational expense than brute force methods, while greedy algorithms provide quick but suboptimal solutions. We compare two objective function formulations, L1 and L2 norms, to assess how they influence the composition and synergy of selected artist groups. Our findings show that the L2 formulation creates more synergistic groupings than the L1 formulation, providing better overall group selection by maximizing both individual skill contributions and collaborative potential, all while adhering to budget constraints.

2 Introduction

In the world of modern music, many of the biggest artists today have achieved success as solo acts. However, an intriguing question arises when considering the potential benefits of collaboration: how can we combine some of these top artists into a group that maximizes profits, while adhering to a set budget? The objective is to find the best possible combination of artists that optimizes revenue generation by analyzing each artist's skills, willingness to collaborate and the financial constraints they incur. By applying combinatorial optimization techniques, we can explore how strategic groupings of artists can lead to greater artistic and economic success.

3 Definitions and Equations

Given the variables:

- x_i: decision variable denoting whether artist i is chosen
- c_i: price of artist i; $\frac{\text{number of monthly listeners}}{1000}$
- s_{ik}: proficiency of artist i in skill k
- Z: collaboration matrix, with z_{ij}: collaboration value between artist i and artist j

- m: total number of artists
- o: total number of skills
- b: total budget available
- λ: cohesion penalty strength; constant multiplier for penalty incurred by more artists added to the group

Objective Function (group score) – L1 norm formulation:

$$\text{Maximize } \sum_{i=1}^{m} x_i \left(\sum_{k=1}^{o} |s_{ik}| + \sum_{j=i+1}^{m} x_j z_{ij} \right) - \lambda \sum_{i=1}^{m} x_i \tag{1}$$

Objective Function (group score) – L2 norm formulation:

$$\text{Maximize } \sum_{i=1}^{m} x_i \left(\sqrt{\sum_{k=1}^{o} s_{ik}^2} + \sum_{j=i+1}^{m} x_j z_{ij} \right) - \lambda \sum_{i=1}^{m} x_i^2 \tag{2}$$

Subject to:

$$\sum_{i=1}^{m} x_i \cdot c_i \leq b \tag{3}$$

$$z_{ij} \geq x_i + x_j - 1 \tag{4}$$

$$\forall i, j \in \{1, ..., m\}, i \neq j \tag{5}$$

The objective function balances three key components:

1. Skill contribution:
   (a) L1 norm formulation:
   $$\sum_{i=1}^{m} x_i \sum_{k=1}^{o} |s_{ik}|$$
   This represents the total skill value provided by selected artists.
   (b) L2 norm formulation:
   $$\sum_{i=1}^{m} x_i \sqrt{\sum_{k=1}^{o} s_{ik}^2}$$
   This approach places a moderate premium on specialization while still valuing versatility. For example, five artists with singing skills (7,7,7,7,7) would have a total singing score of 15.65 under our model, whereas a group that contains artists that are specialized in singing such as, (10,10,10,0,0) would have a total of 17.32. This controlled premium on specialization reflects the music industry's valuation of distinctive talents without excessively rewarding one-dimensional specialists. Additionally, the L2 norm provides mathematical benefits through diminishing returns when adding artists with similar skill profiles, encouraging diversity in the optimal solution.

2. Collaboration value:
$$\sum_{i=1}^{m}\sum_{j=i+1}^{m} x_i x_j z_{ij}$$
captures the synergy between selected artists

3. Cohesion penalty:
$$\lambda \sum_{i=1}^{m} x_i^2$$
penalizes overly large groups through a quadratic term, λ can be adjusted to penalize larger groups of artists

The first constraint ensures that the total cost of all selected contributors remains within a predetermined budget. The next few constraints enforce that collaboration values are only counted when both artists are selected. We also restrict the decision variables to binary values of 0 or 1, representing the selected group of artists. Lastly, the constraints state that every artist only appears once in the equation to avoid duplication.

4 Modeling Process

We decided to take the current 25 artists with the most monthly listeners on Spotify to model our problem. To quantify an artist's talents, we first defined 5 different types of skills, consisting of singing, dancing, rapping, composing, and producing. We assigned values to each artist on a scale of $1 - 10$ with 1 representing unfamiliarity of skill and 10 demonstrating a great specialty in the skill. These skill scores were generated using AI based on publicly available information about each artist's career and performances.

We also defined a collaboration matrix, evaluating the possibility of collaboration between each pairing of artists and used the upper triangle of the matrix in each of our methods to avoid double counting artist pairs. Similar to the skill value, the collaboration value is a value between $0-10$ with 0 meaning that the two artists were extremely unlikely to work together, and 10 meaning that the two artists were very likely to work together. These collaboration scores were also generated using AI to analyze factors such as musical style compatibility, past collaborations, and public statements about potential partnerships. While our data was AI-generated, this approach could easily be adjusted for real-world applications, where a company would likely have their own assessment of artists and more detailed data on collaboration potential.

In order to solve the problem of selecting the best combination of artists while considering budget constraints, collaboration likelihood, and skill distribution, we explored multiple combinatorial optimization techniques. The applied methodologies include brute force search, greedy algorithms, integer programming, and simulated annealing.

We developed two mathematical formulations of the objective function using different norm penalties: L1 and L2. The fundamental difference between these formulations lies in how penalties scale with group size. The L1 norm applies a linear penalty proportional to the number of artists, while the L2 norm (our primary approach) introduces a quadratic penalty that increases more rapidly as group size grows. Mathematically, this means the L2 formulation exhibits diminishing returns for each additional artist added, creating a natural balance point where the marginal benefit of adding another artist is exactly offset by the increased cohesion penalty. This property of

L2 regularization encourages more balanced, synergistic groups by preventing the algorithm from simply selecting as many high-skill artists as the budget allows.

Implementation constraints with integer programming necessitated using only the L1 formulation for that method, which evolved into a valuable research opportunity. We implemented both formulations across our other optimization methods to provide comprehensive analysis of how mathematical modeling choices affect artist group composition and performance.

4.1 Brute Force Approach

A brute force approach systematically evaluates all possible artist groupings to determine the optimal combination. Given m available artists, this method requires evaluating 2^m possible selections. While this guarantees the optimal solution, the exponential growth in search space makes it computationally infeasible for large values of m.

4.2 Greedy Algorithms

Greedy methods are computationally efficient but may result in suboptimal solutions due to their local decision-making nature. They work as incremental enhancements of the optimal solution by choosing the next best artist that improves the objective function while considering the cost constraints incurred by the chosen artist. The algorithms were programmed in Python, using helper functions to calculate the best score of each greedily chosen artist. If the group's value exceeded the best score of the selected heuristic, the artist was added to the group. The process would reach its conclusion if no new artist was picked for the group. We implemented two greedy heuristic approaches that use different choices to assign an optimal solution.

4.2.1 Maximizing Skill Contribution

Firstly, we considered creating a group of the most skilled artists that can be added with the available budget. At each step of the algorithm, the artist that adds the highest marginal increase in skill score is selected, subject to budget constraints.

4.2.2 Maximizing Efficiency

The previous greedy algorithm did not factor cost-effectiveness when the group could have a cheaper monetary value of comparative skill caliber based on the budget allocated. Hence, artists were ranked based on skill-to-cost ratio, prioritizing those that provide the highest skill contribution per unit cost.

4.2.3 Comparison of Greedy Heuristics

Comparing the two greedy algorithms, we found that maximizing the efficiency at each step produces significantly worse results than maximizing skill contribution. We believe this has to do with the inability of the second model to deal with the regularizer, as it tends to add in efficient low-cost artists during its first few iterations despite the quadratic regularizer favoring solutions with a smaller number of high-cost artists.

4.3 Simulated Annealing

Simulated annealing is a probabilistic optimization technique that allows exploration of the solution space while avoiding local optima. The algorithm iteratively modifies the artist selection, accepting new solutions based on the random addition or removal of an artist in the selected group. Expanding on the randomness factor, we accepted new solutions based on an energy function, which decayed over time by an exponential cooling schedule, to determine the new group. This energy function created either a random chance to choose artists that improved the optimality of the solution, or a small probability to instead decrease the optimality. As the algorithm continued to find the global maximum of the group's objective value function, the change in 'temperature' of this function is slowly reduced to zero. Due to the method's random nature, we rerun the solver multiple times to ensure the convergence into a global optimal solution. This method results in an algorithmically efficient solution that is satisfactory with larger amounts of data.

4.4 Integer Programming

We used Integer Programming to model the objective function and constraints for mathematical optimization using the PuLP library. While our primary approach employed an L2 norm formulation for other methods, the quadratic penalty term in the L2 formulation presented challenges for integer programming solvers, which typically handle linear constraints and objectives. To accommodate these limitations, we reformulated our problem using an L1 norm approach for the integer programming implementation. We modeled the basic structure as a 0-1 Knapsack problem, where each artist represents an item with an associated cost. For added complexity, our inclusion of the collaboration matrix was leveraged to determine group compatibility. While straightforward to implement in our other methods, the collaboration matrix introduced a non-linear relationship to the problem. To adhere to linear programming requirements, we introduced binary decision variables and additional constraints that ensured some artists would be selected together while others would not, based on their collaboration values. This modification allowed us to solve the problem using standard integer programming techniques while maintaining the essential structure of our optimization model. The necessity of this L1 reformulation for integer programming ultimately led to valuable comparative insights between L1 and L2 approaches, revealing how different penalty formulations fundamentally affect group composition and optimization outcomes

5 Results

First we contrasted each method's optimal groups based on four factors: the selected collaborators in the group, the size of the group, the cost of the group and the optimal value obtained from the objective function (Group Score) rounded to 2 decimal places. Our analysis examines both L1 and L2 formulations to understand how different regularization approaches influence optimization outcomes.

5.1 Analysis of Optimal Group's Properties

5.1.1 L1 Formulation

Method	**Group Formed**	**Group Size**	**Group Score**	**Group Cost**
Brute Force	Ariana Grande, Justin Bieber, Drake, Ed Sheeran, David Guetta, Travis Scott, Eminem, Sabrina Carpenter, Kanye West, Dua Lipa, Maroon 5, Future, Sia, Calvin Harris	14	953.0.0	99957
Greedy Algorithm 1	The Weeknd, Billie Eilish, SZA, Bad Bunny, Ariana Grande, Justin Bieber, Drake, Ed Sheeran, Travis Scott, Sabrina Carpenter, Post Malone	11	784.5	95761
Greedy Algorithm 2	The Weeknd, Bad Bunny, Drake, Ed Sheeran, Travis Scott, Eminem, Sabrina Carpenter, Post Malone, Kanye West, Dua Lipa, Future, Sia, Calvin Harris	13	874.5	98138
Simulated Annealing	Ariana Grande, Justin Bieber, Drake, Ed Sheeran, David Guetta, Travis Scott, Eminem, Sabrina Carpenter, Kanye West, Dua Lipa, Maroon 5, Future, Sia, Calvin Harris	14	761.0	99928
Integer Programming	Coldplay, Ariana Grande, Justin Bieber, Ed Sheeran, David Guetta, Travis Scott, Eminem, Sabrina Carpenter, Post Malone, Kanye West, Maroon 5, Future, J Balvin, Sia	14	953.0	99957

Table 1: Optimal Pop Groups formed by Method

The implementation of L1 regularization (linear cohesion penalty) in our objective function produces substantially different results compared to the quadratic (L2). With the L1 formulation, all methods select significantly larger artist groups, ranging from 11-14 members, compared to 4-6 members (refer to Table 2) under quadratic regularization.

All optimization methods under the L1 formulation utilized nearly the entire available budget (95-99%), suggesting that the linear penalty provides insufficient disincentive against adding more artists when additional talent is available within budget constraints. The brute force and integer programming methods achieved identical optimal scores (953.0), with simulated annealing finding a solution with a slightly lower score (761.0) but similar composition.

5.1.2 L2 Formulation

Method	Group Formed	Group Size	Group Score	Group Cost
Brute Force	Kendrick Lamar, SZA, Rihanna, Taylor Swift, Justin Bieber, Ed Sheeran	6	58.18	54179
Greedy Algorithm 1	The Weeknd, SZA, Rihanna, Taylor Swift, Justin Bieber, Ed Sheeran	6	57.82	55817
Greedy Algorithm 2	Rihanna, Travis Scott, Sabrina Carpenter, Kanye West	4	50.23	30549
Simulated Annealing	Kendrick Lamar, SZA, Rihanna, Taylor Swift, Justin Bieber, Ed Sheeran	6	58.18	54179

Table 2: Optimal Pop Groups formed by Method

The brute force and simulated annealing methods both converged on an identical optimal solution consisting of 6 artists (Kendrick Lamar, SZA, Rihanna, Taylor Swift, Justin Bieber, Ed Sheeran) with the highest group score of 58.18. This confirms that simulated annealing successfully located the global optimum for this problem.

Greedy Algorithm 1 produced a very similar group to the optimal solution, but replaced Kendrick Lamar with The Weeknd, resulting in a slightly lower score of 57.82. This minor difference suggests that Greedy Algorithm 1 provides a highly competitive approximation with minimal computational expense.

Greedy Algorithm 2, which prioritizes cost efficiency, delivered the smallest group with just 4 members (Rihanna, Travis Scott, Sabrina Carpenter, Kanye West) and a correspondingly lower group score of 50.23. While this solution achieved only 86.3% of the optimal value, it did so at significantly lower cost (30,549 vs. 54,179), highlighting its potential utility in severely budget-constrained scenarios.

5.2 Comparison of Group's Skill Score Across Methods

5.2.1 L1 Formulation

Figure 1 illustrates the skill breakdown comparison across different optimization methods for the L1 formulation. All methods seem to have similar range in results with Integer Programming producing the same results as Brute Force, with identical skill scores in all categories. This demonstrates the effectiveness of Integer Programming in finding optimal or near-optimal solutions for the L1 formulation despite the computational complexity reduction. Simulated Annealing performs consistently well across most skill categories, closely matching the results of Brute Force and Integer Programming.

The Dance skill category shows the lowest scores across all methods (ranging from 14.2 to 17.1), indicating that this skill may be less emphasized or more difficult to maximize within the constraints of the L1 formulation regardless of the optimization approach used.

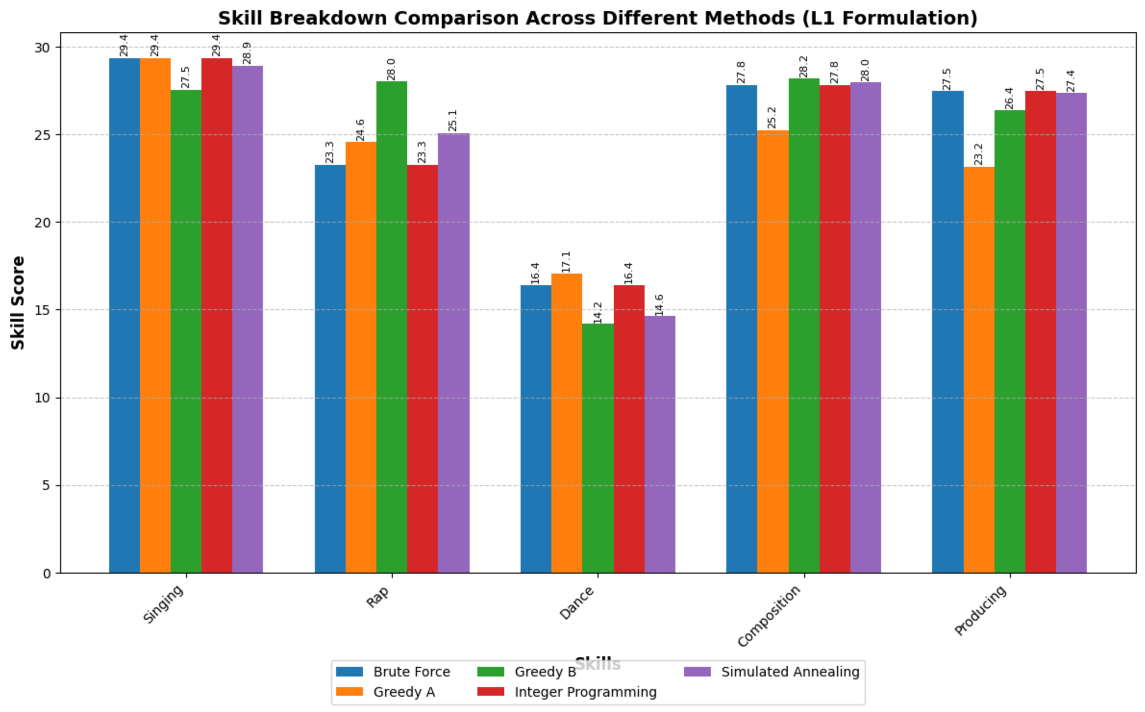

Figure 1: Skills of Optimal Artist Group by Method (L1)

5.2.2 L2 Formulation

Figure 2 reveals several intriguing patterns in how different optimization methods prioritize artistic skills. Comparatively, to the skill breakdown for the L1 formulation, the breakdown across methods for L2 is less consistent. There also appears to be a significant imbalance in dance capabilities across all optimization approaches in this formulation as well, with this skill consistently scoring lowest regardless of method. This suggests a potential gap in the talent pool - even when optimizing for the best possible groups, dance remains relatively underrepresented among top streaming artists.

Interestingly, while the Brute Force and Simulated Annealing methods found identical artist groups, their skill profile shows a strategic balance between commercial appeal (strong singing at 21.5) and creative control (solid composition at 19.8). This balanced approach likely explains why these methods achieved the optimal solution - they effectively navigate the trade-off between performance skills and creative capabilities.

Greedy Algorithm 1's notable emphasis on singing skills (23.3) suggests it favors immediately recognizable commercial talent, which aligns with its “take the best available artist now” approach. In contrast, Greedy Algorithm 2's uniformly lower skills across all categories demonstrates how cost constraints fundamentally change the optimization landscape, forcing compromises that impact all skill dimensions rather than sacrificing any single capability.

These patterns reveal that different optimization strategies do not just affect efficiency and group size - they fundamentally shape the artistic identity and capability profile of the resulting collaborations, with significant implications for both creative output and commercial potential.

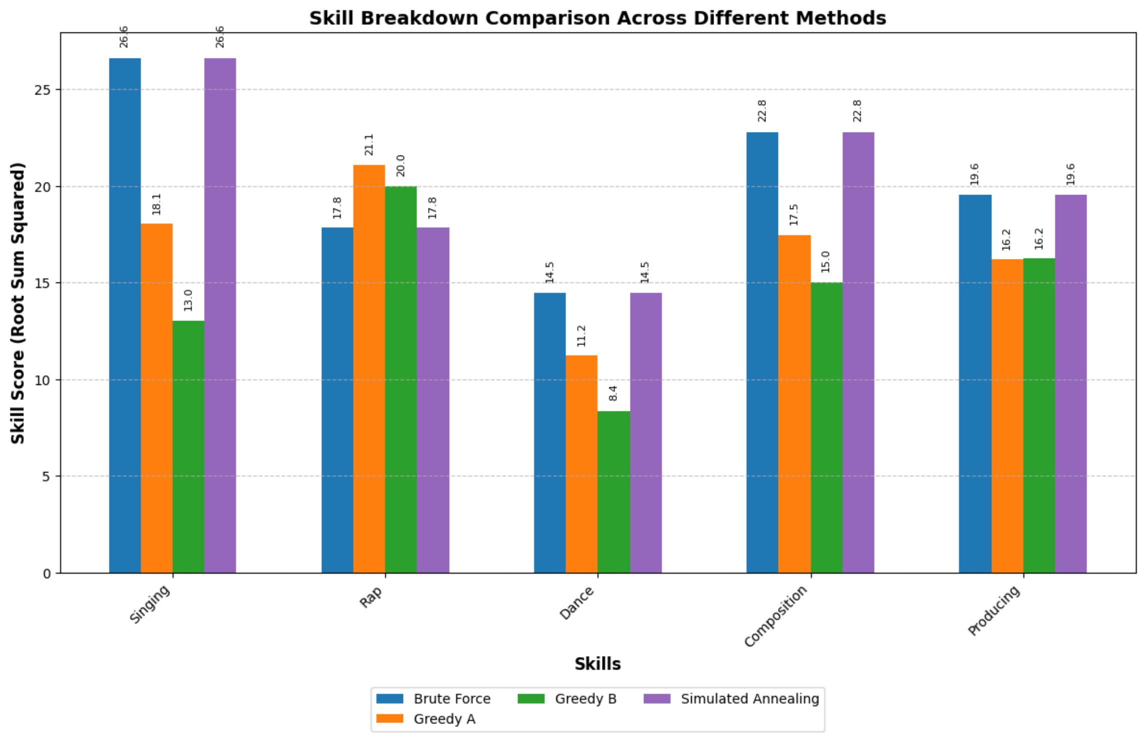

Figure 2: Skills of Optimal Artist Group by Method (L2)

5.3 Analysis on effect of Cohesion Penalty

The relatively small group sizes (4-6 members) observed in most methods in the L2 formulation can be attributed to the level of cohesion penalty strength ($\lambda = 4.5$) used in our model. This value for λ was selected after empirical testing across a range of values from 0 to 10 and chosen because it creates realistic group sizes (4-6 members) that align with successful real-world music groups while still allowing enough diversity of skills.

Figure 3 illustrates how the optimal group size and group score (objective function value) changes as we vary the cohesion penalty strength (λ). We used simulated annealing for this analysis since it produces results nearly identical to brute force but with significantly less computational expense. As shown in the graph, without any penalty ($\lambda = 0$), the optimal solution includes 12 artists, essentially maximizing raw skill and collaboration value without considering group dynamics. As λ increases, we see a steady decrease in group size, with our chosen value of $\lambda = 4.5$ yielding a 6-member group. This relationship demonstrates the direct control the penalty term provides over group size while still allowing the optimization to select the most compatible and skilled artists within that constraint.

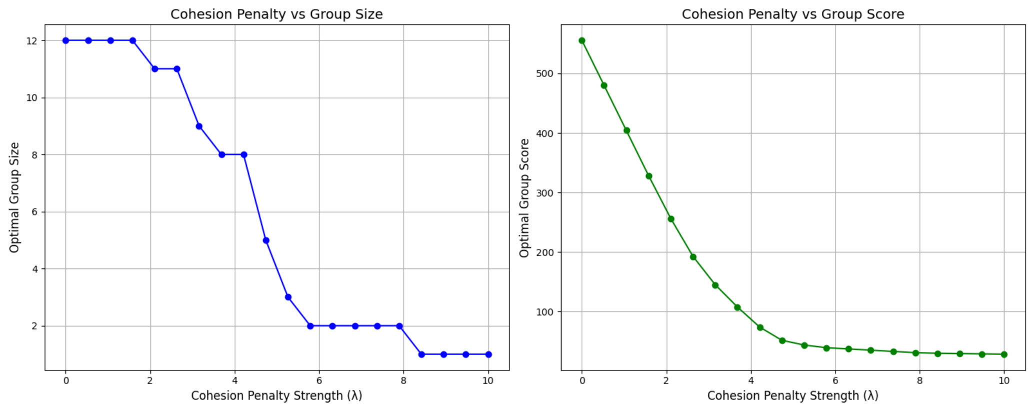

Figure 3: Effect of Cohesion Penalty Strength in L2 Formulation

Unlike the L2 formulation, which shows a responsive relationship between cohesion penalty and group size, figure 4 reveals a key limitation of the L1 formulation. As shown in the left graph, the L1 formulation maintains a constant optimal group size of 14 members regardless of the cohesion penalty strength (λ). This indicates that the L1 penalty term fails to effectively control group size in a nuanced manner.

The right graph further illustrates this limitation, showing only a linear decrease in group score as λ increases, without any corresponding adjustment in group composition. This suggests that the L1 formulation merely penalizes the objective function value without actually influencing the structural characteristics of the optimal solution.

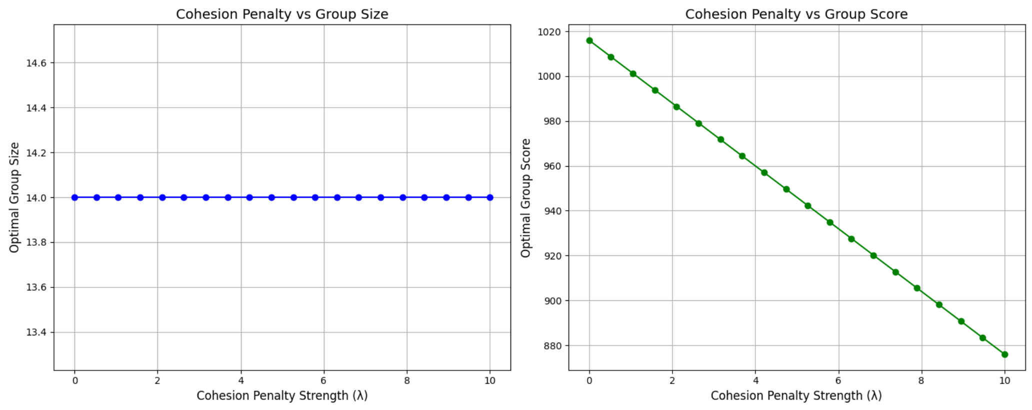

Figure 4: Effect of Cohesion Penalty Strength in L1 Formulation

These results demonstrate that the choice between L1 and L2 regularization fundamentally changes the optimization outcome. The L2 formulation provides fine-grained control over group size by balancing skill maximization against collaboration challenges and allows us to model the diminishing returns of adding additional members more accurately, reflecting real-world group dynamics where coordination costs increase non-linearly with group size. In contrast, the L1 formulation's binary behavior makes it unsuitable for modeling the subtle tradeoffs involved in collaborative group formation.

5.4 Performance Comparison

Table 3 provides a performance comparison of each method under the L2 formulation, including time taken to compute the solution and number of solutions evaluated. Furthermore we split simulated annealing into two, the total time taken and number of solutions checked over all the simulations, as well as the mean for each individual simulation.

Method	**Time taken (s)**	**Number of Solutions checked**
Brute Force	1310.42	33554432
Greedy Algorithm 1	0.01	154
Greedy Algorithm 2	0.01	115
Ensemble Simulated Annealing (sample size 100)	58.12	1000000
Mean Simulated Annealing	0.58	10000

Table 3: Comparison of Method Complexity

The results reveal striking differences in computational efficiency. The brute force method required approximately 22 minutes (1310.42 seconds) to evaluate around 33.5 million possible combinations. In contrast, both greedy algorithms completed in just 0.01 seconds after checking only 154 and 115 solutions, respectively. Simulated annealing achieved the optimal solution in 58.12 seconds—a 95.6% reduction in computation time compared to brute force—while integer programming required 50.52 seconds.

Most notably, the performance gap between brute force and more efficient methods expands exponentially as the problem size increases. The brute force approach, while guaranteed to find the optimal solution, is impractical for optimization problems with many variables due to exponential time complexity. Adding just one more artist would double the time taken to approximately 45 minutes, while 30 artists would require roughly 12 hours. This exponential growth makes brute force completely infeasible for larger datasets, highlighting why approximation methods like simulated annealing, which found the same optimal solution in just 58 seconds, are essential for practical applications in combinatorial optimization.

6 Further Remarks

Much of our data was estimated, such as the salary of each artist, which could be improved with real-life datasets. Additionally, we note that the skill score for each artist is subjective and more complex than a linear scoring system ranging from 1 to 10. Furthermore, with larger groups of artists, the collaboration matrix also increases exponentially (m^2 size). While we chose some specific features of an artist's skill level, it could be interesting to see how extra features modify these results. For example, we could factor in the amount of years the artist has been active, their listener demographic, etc.

For further computational exploration, we could try a genetic algorithm, involving selecting and 'mutating' artists to try to find an optimal group, and compare it to our current methods, specifically with simulated annealing. On the other hand, we could also try optimizing hyperparameters, i.e. cohesion penalty strength, ensemble simulated annealing sample size, etc.

7 Appendix

Below is an example of the data collected for each artist.

Artist	Listeners	Singing	Rap	Dance	Composition	Producing
Bruno Mars	147,914,735	10	7	7	8	7
The Weeknd	125,937,923	10	9	4	8	7
Lady Gaga	124,595,645	9	1	6	6	7
Kendrick Lamar	109,561,228	4	10	3	9	8
Billie Eilish	107,907,213	10	3	3	9	8

Table 4: Data of the 5 artists with the most listeners

Below is an example of a collaboration matrix, for a group of 4 artists.

$$\begin{array}{c|cccc} & A & B & C & D \\ \hline A & \text{NaN} & 9 & 4 & 8 \\ B & 9 & \text{NaN} & 9 & 8 \\ C & 4 & 9 & \text{NaN} & 4 \\ D & 8 & 8 & 4 & \text{NaN} \end{array}$$

Table 4: Sample Collaboration Matrix

References

[1] Spotify Top Artists by Monthly Listeners. Kworb.net. (2025). `https://kworb.net/spotify/listeners.html#google_vignette`.