To Build or Not To Build: Determining a Quantitative Metric for Land Planning and Allocation

Zixuan Li , Yan Xiao , Ziang Li , Jocelyn Wang

Abstract:

Land planning is crucial to ensure that urban development occurs with consideration to the economic, social, and environmental interests of a community. Many conflicting factors must often be considered to adhere to optimal land planning. In this paper, our team makes a quantitative decision metric that can analyze these factors and determine the "best" choice from a given set of development options and the allocation of those choices. First, linear programming is used to determine two "best" development options: one that maximizes both economic and social factors and one that minimizes negative environmental factors while maximizing social. The maxima and minima from linear programming are then applied to the Technique for Order of Preference by Similarity to the Ideal Solution to obtain a third real-world "overall best" option that balances economic and environmental factors with a desired weighting. A genetic algorithm is then used to determine the optimal positioning of the three established "bests" by analyzing opportunity costs based on an environmental degradation penalty index. Finally, the Cobb-Douglas Function is used to conduct a short- and long-term analysis of each result's profit by solving differential equations about inflation. This model is then applied to the parcel of land in Victory, NY, using data obtained from research. The ideal option and positioning are found to be 267 acres of a sports complex in the northern half of the land, 129 acres of regenerative farm directly west of the sports complex, 344 acres of a solar array in the southernmost region of the land, and 1 acre of agritourism center on the eastern side of the land. Conducting a sensitivity analysis on our model reveals that the linear programming results are most affected by the area and societal benefit restrictions but that the TOPSIS results remain relatively stable regardless of the changing parameters. Our model is adjusted to account for Micron Technology, Inc. building a nearby fabrication facility. As this facility brings more jobs and thus more people, the profit of facilities that involve tourism will increase. However, nature-based facilities will suffer detriment due to pollution caused by the facility. With these adjustments, the model is re-run, and the results are compared to the previous results. In this scenario, there would be a greater area of the solar array and agritourist center, a smaller sports complex, no regenerative farm, and 128 acres of ranch. Finally, the generalizability of our model is discussed by first discussing its application in Shenzhen, China, and then widening the scope to any location in any country. Our model will provide the most implementable results in rural environments due to its quantitative nature that cannot consider complicated urban planning laws but that the model can be applied to nearly any scenario as long as data is provided.

Keywords: Linear Programming, TOPSIS, Genetic Algorithm, Cobb-Douglas Function, Differential Equation

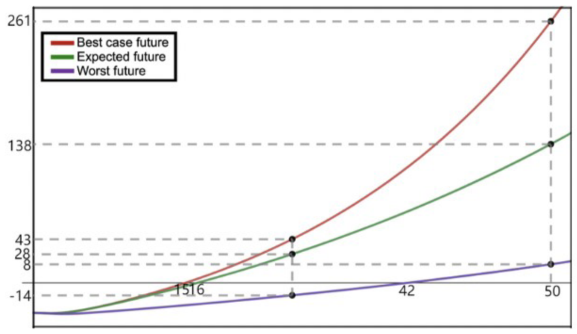
LETTER TO DECISION MAKERS

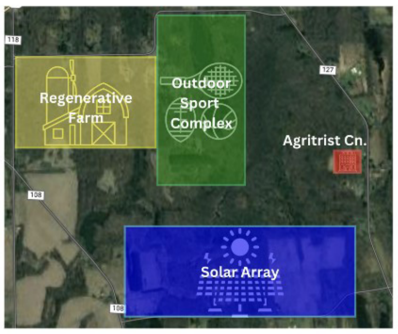
April 2023 | New York, USA

Dear Decision Makers,

Our team has now arrived at a recommendation for the best use of the land in Victory, NY, obtained through a quantitative mathematical model. The model is extremely comprehensive and arrived at its conclusion after a thorough consideration of profit, environmental degradation, carbon emissions, employment, attraction of tourism, recreational offers, and societal benefits.

From our summary sheet, you are likely already aware of our recommended distribution for the land. The image to the right hopefully provides further clarity on that recommendation.





This recommended distribution balances environmental and economic factors in a 1-to-2 ratio. If you were to follow our model's recommendations exactly, your annual profit would be $9,047,920 USD and the annual CO2 emissions of the finished land developments would be -1,095 tons. Year-by-year profit analysis, including best- and worst-case scenario analyses according to inflation and efficiency, are shown on the graph above. As can be seen, this recommendation can be trusted economically; even the worst-case scenario returns millions in profit by 50 years, and in a best-case scenario, you will be able to make over 250 million in profit at the 50-year mark.

It is worth noting that both the solar array and agritourist center appeared multiple times in our model's recommendation. This did not change whether the model was told to prioritize environmental factors or economic factors, and the pattern remained even when Micron Tech's fab was taken into consideration. Because of this, we highly recommend building at least a **solar array and agritourist center** on the land according to the recommendations.

This summarizes the most pertinent conclusions of our team and our model. Whatever your final decision is regarding this land, we hope you will be able to make an easier decision knowing what the numbers say.

Thank you for your trust in us.

Sincerely,

Team 2023001

1 Introduction

Whether to build or not to build is the question that plagues land planners and decision-makers day and night (and tomorrow, and tomorrow, and tomorrow). Regional planning is an essential aspect of land planning, and its importance has only increased in recent years as urban sprawl continues to grow [3]. On top of this, increasing awareness of issues like climate change has caused land planners to become more mindful of how they design their plans and what they should consider during the design process. With this in mind, community leaders and business planners consulted us to help decide the "best" use of a 3km^2 (741.316 acres) parcel of land 50 km from Syracuse.

This paper begins by creating a model to determine the "best" development option. Research is conducted to determine economic, environmental, and social criteria/sub-criteria influencing development options. These variables are then applied in linear programming, TOPSIS, and genetic algorithms to create the base model. Next, the model is applied to the parcel of land in New York by doing further research and collecting data. Sensitivity analysis is also conducted on the model to test the adaptability and reliability of its result.

We then explain how our model will be affected by the Micron Tech., Inc. fabrication facility being built not too far from the parcel of land. A new set of calculations are also performed for this scenario. Finally, the generalizability of our model is evaluated by discussing how it would perform in an environment familiar to our team (Shenzhen, China) and in other international contexts.

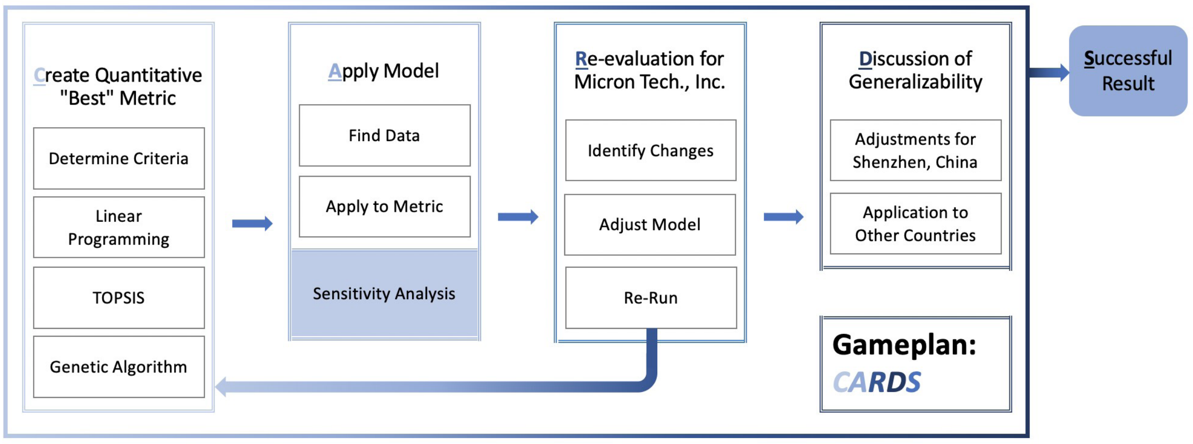


Figure 1 Flow Chart of Process

2 Preliminary Assumptions and Definitions

2.1 Variable Definitions

Table 1 Variable Definitions

Notations	Descriptions	Notations	Descriptions
P	Annual profit	O_i	Facility *i*'s employment index
A_i	Attraction index of facility *i*	E	Annual carbon emission
R_i	Facility *i*'s recreational index	S_i	Facility *i*'s societal service index
αipr	Annual profit of facility *i*	xA_i	Area of facility *i*
ϵce_i	Annual carbon emission of facility *i*	x_b	Highest annual profit possible
y_b	Least annual carbon emission possible	h	Developers' desired weighting
A_{si}	Facility *s*'s area in land type *i*	F	Fixed cost
n	Number of Facilities	A_j	Area of facilities in land type *j*

2.2 Assumptions

Assumption 1: The “best” development option can be determined quantitatively. This assumption is the basis of our model and also dramatically simplifies it. Additionally, many diverse quantitative measures are already available to employ in modeling, so limiting the factors used to quantitative ones will still provide a comprehensive result.
Assumption 2: All global, local, and environmental developments in the near and far future will not suddenly change and will remain relatively predictable. This includes extreme weather phenomena, sudden tragedies, and unforeseen economic and political changes. The nature of these things makes them unpredictable and rare, and therefore burdensome - and arguably unnecessary due to their rarity - to model.
Assumption 3: The available budget for initial development is 50 million USD. Looking at land in and around Syracuse, even if we take the lower averages, this plot of land should cost far more than 50 million USD. Additionally, many of these development options require lots of money to build. Since the decision-makers have considered these options, it is reasonable to assume that we have at least 50 million USD to put towards developing this plot of land.
Assumption 4: The developments on this land have been given the green light by the appropriate party (i.e., local or state government, permit association, etc.). Nearly all possible development options require some building permit, development permit, or a similar green light. Thus, for the sake of considering all of these options in our model, we assume that all possibilities have already received the necessary permits.
Assumption 5: The placement of land types is visually distinguishable, as indicated in the provided satellite map. For simplicity and straightforwardness, we assume that the land types correspond with the visual cues in the satellite map, i.e., flat-looking yellow-green is cropland, an uneven patch of green is forest, etc.
Assumption 6: The best use of a cross-country skiing trail during the rest of the year is as a hiking trail. Cross-country skiing being only available for three months of the year gives this option a considerable disadvantage, causing it to be automatically neglected by nearly any quantitative model. Thus, to level the playing field, we assume that the trail is open during the non-winter months as a hiking trail.

3 Task 1: Creating the Model

This task requires us to create a quantitative decision metric that can define the “best” use of the land. [6] To do this, we use a combination of linear programming, TOPSIS, and genetic algorithm:

- Linear Programming - Used to obtain the maxima and minima of criteria [13]. We first use research to identify the most important measures (in other words, the requirements that affect the development options the most). Then, formulae and restrictions are created to model these criteria standardized quantitatively.
- TOPSIS - We apply the maximum and minimum from linear programming to deduce the best overall alternative in TOPSIS [11]. Plotting each development option into the model and measuring distances between each option and the best/worst case reveals a “best” option (the option closest to the best point).
- Genetic Algorithm - Used to determine the distribution and location of facilities specified in TOPSIS [9]. We invite the three “best” options (highest performing overall and by environmental and economic criteria) to participate in genetic modeling. Thus, the most ideal distribution for the land is finalized, and the “best” plan for the land is finalized.

3.1 Initial Research

To conduct research, the sources consulted are primarily government or official sources when possible (either Syracuse, New York State, or US National). If this is impossible, we look for data from relevant sources, such as solar panel companies, for solar array information. Our data sources are listed in Table 2.

Table 2 Data Sources and Uses

Information	Source	Site
Agricultural data: income, income:land trend	US Department of Agriculture	https://www.usda.gov
Geographical and Environmental data	City of Syracuse	https://www.syr.gov/Home
Economic information	Observatory of Economic Complexity (OEC)	https://oec.world
Solar Array information	LG Corporation	https://www.lg.com/global
income statistics	Forbes Finance Council	https://www.forbes.com

Agriculture finance information	Office of the New York State Comptroller	https://www.osc.state.ny.us
Carbon Emissions data	New York Academy of Sciences	https://nyaspubs.onlinelibrary.wiley.com
Agritrst, cross-country ski and crop info	Ontario Ministry (OMAFRA)	http://omafra.gov.on.ca

From this research, seven factors stand out as criteria that would affect the determination of the "best" development option: profit, employment opportunities, the attraction of tourism, carbon emissions, environmental degradation, recreational offers, and societal benefits.

3.2 Criteria Identification

The final determined criteria can fit into three main categories: economic, environmental, and social, as shown in Figure 2.

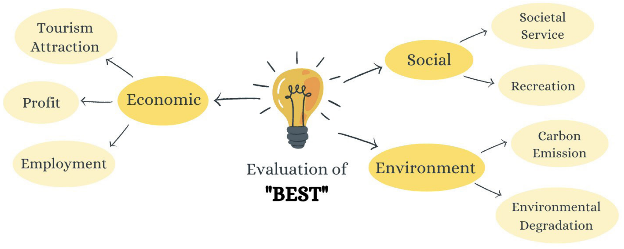


Figure 2 Quantitative Criteria

- Economic Factors:
  - Profit: Referenced by P. This includes the profit generated by all facilities while extracting the cost required to produce goods or services concerning time and area coverage.
  - Employment Opportunities: Referenced by O_i. This is an evaluative index on the employment opportunities provided by land development i.
  - Attraction of Tourism: Referenced by A_i. This is an evaluative index on the tourist and resident attraction value of development i. An increase in the interest will undoubtedly be a booster to the local and possibly even the state economy.
- Environmental Factors:
  - Carbon Emissions: Referenced by E. It is essential to minimize carbon emissions because they contribute to climate change that poses a significant threat to the environment, economy, and human societies worldwide.
  - Environmental Degradation: Referenced by E_W, E_F, E_D, and E_C. This is a penalty-based index utilized in the genetic algorithm. It considers the effect of biodiversity reduction, soil erosion, and all forms of pollution on each land type.
- Social Factors:
  - Recreation: Referenced by R_i. Citizens' enjoyment will also matter in this deciding factor. Some facilities offer recreational activities, while others do not. That will affect the locals' acceptance of the facilities and thus affect aspects such as the attraction of tourism and employment opportunities.
  - Societal Benefits: Referenced by S_i. Other than recreation, there are also other facilities that a society yearns for, such as education, clean water, nutritious crops, or other similar aspects.

3.3 Linear Programming for Preliminary Planning

In the first step of this model, linear programming is used to incorporate criteria that can be modeled linearly. These criteria can be either objective functions to be maximized or minimized or restrictions on those functions. The variables involved are assumed to be continuous. The resulting solution is definitive and represents the best possible solution, given the available resources and restrictions imposed.

To facilitate the chosen criteria in linear programming, social factors are incorporated into restrictions on economic and environmental. In other words, they are not independent objective functions that must be optimized.

3.3.1 Economic

The profit should be maximized in modeling the aforementioned economic criteria in linear programming. The development option that yields the highest economic results should be the "best" in that aspect, as it would have the highest profit, provide the most employment opportunities, and attract the most tourists and residents.

$$\max P = \sum_{i=1}^{n}\left(\alpha_i^{pr} \times x_i^A\right) \tag{1}$$

$$s.t.\begin{cases} \sum_{i=1}^{n} x_i^A \le 741.316 \\ \sum_{i=1}^{n} S_i \times x_i^A \ge 3000 \\ \sum_{i=1}^{n} R_i \times x_i^A \ge 2000 \\ \sum_{i=1}^{n} O_i \times x_i^A \ge 4500 \\ \sum_{i=1}^{n} A_i \times x_i^A \ge 1500 \end{cases} \tag{2}$$

where P is the profit; α_i^{pr} is the annual profit of facility i; x^A_i is the area of facility i in acres; S_i is the societal index of facility i; R_i is the recreational index of facility i; O_i is

the employment opportunity index of facility i; and A_i is the tourism and residence attraction index of facility i.
Restriction Explanation

- $\sum_{i=1}^{n} x_i^A \leq 741.316$: As a basic measure, the first restriction placed on this confirms that the facility does not exceed the available land
- $\sum_{i=1}^{n} O_i \times x_i^A \geq 4500$ $\sum_{i=1}^{n} A_i \times x_i^A \geq 1500$ area.
- Restrictions are also written to ensure that employment opportunities (O_i) and tourism attraction (A_i) are sufficient to provide tangible economic benefits.
- $\sum_{i=1}^{n} S_i \times x_i^A \geq 3000$ And $\sum_{i=1}^{n} R_i \times x_i^A \geq 2000$: Finally, the societal factors are incorporated into the restrictions. $\sum_{i=1}^{n} S_i \times x_i^A \geq 3000$ Sets the bar for societal benefits to the local community and $\sum_{i=1}^{n} R_i \times x_i^A \geq 2000$ recreational opportunities.

3.3.2 Environmental

The environmental criteria - only carbon emissions in this case - should be as low as possible (i.e., minimized) to deduce the "best" option under this criteria.

$$\min E = \sum_{i=1}^{n} \left(\epsilon_i^{ce} \times x_i^A \right) \tag{3}$$

$$s.t.\$ \begin{cases} \sum_{i=1}^{n} x_i^A \leq 741.316 \\ \sum_{i=1}^{n} S_i \times x_i^A \geq 3000 \\ \sum_{i=1}^{n} R_i \times x_i^A \geq 2000 \end{cases} \tag{4}$$

where E is the annual carbon emission of the land model in kg and ϵ_i^{ce} is the annual carbon emission of facility i in kg · acres^{-1}.

Restriction Explanation

Because there is only one environmental factor being modeled with linear programming, the restrictions are comprised of only

- $\sum_{i=1}^{n} x_i^A \leq 741.316$ the basic area restriction, and
- $\sum_{i=1}^{n} S_i \times x_i^A \geq 3000$ and $\sum_{i=1}^{n} R_i \times x_i^A \geq 2000$ the societal restrictions, as described previously.

3.4 TOPSIS for Working Model

With a finished linear programming model, we can use TOPSIS to create a working model that can define the one "best" alternative.

3.4.1 Introduction to TOPSIS

TOPSIS (Technique for Order of Preference by Similarity to Ideal Solution) [2] is a multi-criteria decision-making method used to evaluate the best alternative from a set of available options. It is based on the assumption that the best alternative is the one that has the shortest distance from the positive ideal solution (i.e., the best alternative that maximizes the criteria) and the longest distance from the negative least ideal solution (i.e., the worst alternative that minimizes the criteria).

To apply the method, a set of criteria is first defined. Weights are then assigned to each criterion based on the criterion's relative importance. Then, a matrix including each alternative and criterion is constructed. The matrix is also normalized to account for differences in the criteria scale. Finally, the distance between each alternative and the best and worst solutions is measured, and the alternatives are ranked based on their closeness to the positive ideal solution.

3.4.2 TOPSIS in Our Model

The TOPSIS model in this report will be done on a 2D coordinate plane, with the x-axis measuring annual economic profit and the y-axis measuring annual carbon emissions, as calculated in the linear programming section.

1. Normalization: The x value x_b represents the highest annual, and the y value y_b represents the least annual carbon emission. It must also be considered that the range of profits could drastically differ from the range of carbon emissions. To standardize the relationship between the two, the environmental value will be modified (multiplied by r) to emulate the former.
$$\frac{|x_b|}{|y_b|} = r \tag{5}$$
2. Weighting: To weigh the economic and environmental factors according to their influence, the economic aspect will also be adjusted to be h times that of the environmental if the decision maker so wishes (if not, then h = 1). This is proposed to account for the fact that each stakeholder will have different beliefs about which factors are more critical in determining the "best" development option.
3. Best and Worst: Taking into account these preliminary factors, a formula is created to determine the best (positive Ideal) and worst (negative least ideal) possibilities I_b and I_w, respectively:
$$I_b=(x_b|h\times\max(P),y_b|r\times\min(E)),$$
$$I_w=(x_w|h\times\min(P),y_w|r\times\max(E)) \tag{6}$$
4. Candidates Selection: To reduce the time complexity of the model, 19 points are then manually determined, of which at least one is the best option. These points are set on a spectrum ranging from heavily considering environmental factors (5:95%) to heavily considering economic factors (95:5%). The 19 points (x_ω,y_ω) are each determined using another linear programming

system that combines economics and environmental considerations.

$$\max \sum_{i=1}^{n} x_i^A \left(h \times \omega \times \alpha_i^{pr} - r \times (1-\omega) \times \epsilon_i^{ce}\right) \quad (7)$$

$$s.t. \begin{cases} \sum_{i=1}^{n} x_i^A \leq 741.316 \\ \sum_{i=1}^{n} S_i \times x_i^A \geq 3000 \\ \sum_{i=1}^{n} R_i \times x_i^A \geq 2000 \\ \sum_{i=1}^{n} O_i \times x_i^A \geq 4500 \\ \sum_{i=1}^{n} A_i \times x_i^A \geq 1500 \end{cases} \quad (8)$$

where $\omega \in \{0.05, 0.1, 0.15, ..., 0.95\}$. From the afore written formulae, the calculation for x_ω and y_ω can be formed:

$$x_\omega = h \times \sum_{i=1}^{n} x_i^A \alpha_i^{pr}, \; y_\omega = r \times \sum_{i=1}^{n} x_i^A \epsilon_i^{ce} \quad (9)$$

5. Distance Calculation: Finally, with x_ω and y_ω, distance calculations - the defining aspect of TOPSIS begin. Using the distance formula, the distance between development option ω at (x_ω,y_ω) and the ideal and least ideal possibilities are calculated as

$$d_{\omega w} = \sqrt{(x_\omega - x_b)^2 + (y_\omega - y_b)^2}, \quad d_{\omega b} = \sqrt{(x_\omega - x_w)^2 + (y_\omega - y_w)^2}. \quad (10)$$

6. Judging Index Calculation: Both distance calculations are repeated for all 19 manually determined points. After the process is finished and results are recorded, a judging index s_ω is created based on the distance of point each point (x_ω,y_ω) from the ideal and least ideal possibilities:

$$s_\omega = \frac{d_{\omega w}}{d_{\omega w} + d_{\omega b}}, 0 \leq s_\omega \leq 1 \quad (11)$$

Note that $s_\omega = 1$ if and only if the option is exactly the ideal alternative, and $s_\omega = 0$ if and only if the option is exactly the least ideal alternative.

7. Ranking: All results are taken and ranked according to s_ω, and the highest-ranking overall option is chosen along with the highest-performing economic and environmental options from linear programming to participate in the next step, the genetic algorithm (GA).

3.5 Genetic Algorithm (GA) for Application

Next, the positioning of each facility must be calculated. This is because, although the maximum profit of different facilities has now been shown, evaluating their placement for minimum environmental damage is still necessary according to the environmental degradation criteria determined in Section 3.2.

3.5.1 Process of Genetic Algorithm

GA is a type of optimization algorithm that uses a heuristic search technique to find the optimal solution to a given problem by mimicking the process of natural selection and evolution. The genetic algorithm goes through the following steps to achieve this:

- **Initialization:** First, a possible solution (the initial instance) is found and fed to the program in the form of "DNA," a binary sequence indicating all the required input, just like all the other generations. The DNA data is a string that constrains all variables subject to change,
- **Conversion:** The binary sequence in the DNA is converted to decimal variables and fitted across the range of all possible data.
- **Selection:** The "fitness" of the DNA is evaluated based on how successful it is in solving the objective function. If the DNA sequence has a high fitness value, it gets a higher chance to survive and reproduce the next generation of offspring.
- **Reproduction:** This is done through crossover, where one parent selected according to fitness has its DNA replaced at random points by another randomly selected parent.
- **Mutation:** During this process, mutations may occur by arbitrarily choosing a DNA bit from a child and reverting it.
- **Repetition until convergence:** The steps described above are repeated for a fixed number of generations (at which point the final-generation child with the highest fitness is selected) or indefinitely until a satisfactory solution is found.

GA is particularly suitable for problems that involve non-linear objective functions and large solution spaces, and the detail of GA is shown in Algorithm 1.

Algorithm 1: Genetic Algorithm for Finding Best Solution

```
Input: An instance $\delta$, DNA length $\beta$, reproduction rate $\alpha$, mutation rate $\gamma$, population size
       $\sigma$, generation number $\omega$
Output: Fittest DNA string found $P_{best}$

// Initialize
1 Objective function OF(x), Current Population $P$, Current Fitness $F$;
2 Generate $\sigma$ DNA genes, each being 60 bytes long, and save them to $P$;
3 for $i \leftarrow 0$ to $\omega$ do
```

```
    // Calculate Fitness
4   for $j \leftarrow 0$ to $\sigma$ do
5       $F[j] \leftarrow OF(P[j])$;
    // Selection according to Fitness
6   for $j \leftarrow 0$ to $\sigma$ do
7       Choose DNA $P[\theta]$ with replacement $Rwithprobability^{F[\theta]}/\sum_{k=0}^{\sigma-1} F[k]$;
8       Randomly choose DNA $P[\kappa]$ with replacement;
        // Reproduction
9       if $randint \in [0,1] \leq \alpha$ then
10          Randomly choose $k$;
11          set $P[\theta][k] \leftarrow P[\kappa][k]$;
    // Mutation
12  for $j \leftarrow 0$ to $\sigma$ do
13      for $k \leftarrow 0$ to $60$ do
14          if $randint \in [0,1] \leq \gamma$ then
15              Revert $P[j][k]$ from 0 to 1 or from 1 to 0;
  // Return
16 return $P_{best} \leftarrow \max F_i$;
```

3.5.2 Application of Genetic Algorithm

We put the map into a flat Cartesian coordinate system to simplify the model. Assume that the point at the bottom left corner of the map is the origin point (0,0), and the top right corner will be (20, 17.5). Hence, each unit of land is calculated to equal $\frac{Area_{land}}{Area_{plane}} = 2.709$ acres of land.

Since the profit of the facilities is constant with the calculated area of TOPSIS, the genetic algorithm will focus on an aspect called opportunity cost. Opportunity cost is defined as the loss of potential gain from other alternatives when one is chosen. In our model, this cost is a number between 0 and 30, with a lower number indicating a more precious environment.

The opportunity cost [10] of each land type is formed by 3 different aspects - Pollution, Soil Erosion, and Biodiversity. Each has its specific environmental cost (except shrubs, which will not be considered due to its mere one percent coverage). E_W represents the environmental cost of wetland, E_F of forest, E_D of developed land, and E_C of crop land.

The objective function - the fitness evaluator - is as follows:

$$\max \sum_{m \in \mathcal{S}} A_m E_m \tag{12}$$

$$s.t. \begin{cases} \left|x_{si} - x_{sj}\right| \geq \dfrac{\Delta x_{si} + \Delta x_{sj}}{2} \\ \left|y_{si} - y_{sj}\right| \geq \dfrac{\Delta y_{si} + \Delta y_{sj}}{2} \\ \left(x_{si} \pm \Delta x_{si}, y_{si} \pm \Delta y_{si}\right) \in M \\ \left(x_{si} \pm \Delta x_{si}, y_{si} \mp \Delta y_{si}\right) \in M \end{cases} \tag{13}$$

where m is the land type; s is the type of facility; k_s is the number of facility s; A_si refers to the area of facility s in land type i; S is the set of {Wetland, Forest, Developed,Crop}; and (x_{si},y_{si}),(x_{sj},y_{sj}) is the center of the facilities (assuming all facilities are rectangular with sides parallel to either the x-axis or y-axis). Note that in programming, the input is taken in the bottom left corner, and the center is then calculated by adding half of the height and width. Δx_{si} is the width of the rectangle and Δy_{si} is the length; and A_W ,A_F ,A_D,A_C refers to the area of facilities in wetland, forest, developed area, and cropland, respectively.

Restrictions Explanations

- $\left|x_{si} - x_{sj}\right| \geq \frac{\Delta x_{si} + \Delta x_{sj}}{2}$ and $\left|y_{si} - y_{sj}\right| \geq \frac{\Delta y_{si} + \Delta y_{sj}}{2}$ ensure that the facilities do not overlap with each other, and
- $\left(x_{si} \pm \Delta x_{si} y_{si} \pm \Delta y_{si}\right) \cup \left(x_{si} \pm \Delta x_{si} y_{si} \mp \Delta y_{si}\right) \in M$ ensures that the facilities lie inside the map.

The detail of the results is further explained in Section 4.2.3, and the complete code for GA is demonstrated in Appendix C.

3.6 Short- and Long-term Considerations

To incorporate the short-term and long-term into consideration, we focus on total profit, which is equal to the entire production value minus the total cost. In other words, the total profit of a specific distribution of facilities will be equal to the total cost of maintenance subtracted from the value of all goods produced in a year. Thus, to find a real profit, the equation.

$T_p(t) = P(t) - C(t)$ (14)

is used, where $T_p(t)$ is the total profit; P(t) is the total production value accumulated through t years, and C(t) is the total cost accumulated through t years.

- Total Production: The total production value can be measured by the convolution of the Cobb-Douglas production function [8] and an exponential function:

$$P(t) = Y(t) * (1+\alpha)^t = \int_0^t Y(u)(1+\alpha)^{t-u}\,\mathrm{d}u \quad (15)$$

where α is the rate of inflation. Y (t) is the Cobb-Douglas production function

$Y(t) = A \times L(t)^g \times K(t)^j$ (16)

where A is the efficiency constant; g and j are economic constants. The efficiency constant is a measure of total factor productivity. The economic constant measures capital and labor output elasticity (percentage change of output).

Additionally, L(t) is the total labor with respect to time, where labor is the working time. L(t) can be expressed as

$$L(t) = -\frac{a(L_i - L_0)}{t+a} + L_i \quad (17)$$

where a is the labor constant; L_0 is the total hours of labor at t = 0; L_i is the total hours of labor at t = ∞.

K(t) is the real capital with respect to time, which can be expressed as

$$\begin{cases} \dfrac{\mathrm{d}K}{\mathrm{d}t} = r\left(1 - \dfrac{K}{K_i}\right)K \\ K(0) = K_0 \end{cases} \quad (18)$$

where r is the growth constant; K_0 is the total hours of labor at t = 0; and K_i is the total hours of labor at t = ∞. The growth constant measures the speed at which facility construction is completed.

From this, the solution to the function is concluded to be:

$$K(t) = \frac{K_i}{1 + \left(\dfrac{K_i}{K_0} - 1\right)e^{-rt}} \quad (19)$$

- Total Cost: The total cost consists of the fixed cost (cost to build) and the accumulated operating cost (cost of maintenance and operations). The operating cost can be measured by the convolution of a logarithmic function and an exponent function. The total cost function C(t) is:

$$C(t) = F + log(bt+1) * (1+\alpha)^t =$$

$$F + \int_0^t log(bu+1)(1+\alpha)^{t-u}\,\boldsymbol{du} \quad (20)$$

where F is the fixed cost, and b is a constant.

- Total Profit: Plug all functions K(t) and C(t) into the main function, and the result will be:

$$T_p(t) = A\int_0^t \left(-\frac{a(L_i - L_0)}{t+a} + L_i\right)^g \left(\frac{K_i}{1 + \left(\dfrac{K_i}{K_0} - 1\right)e^{-rt}}\right)^j$$

$$(1+\alpha)^{t-u}\,\mathrm{d}u - F - \int_0^t log(bu+1)(1+\alpha)^{t-u}\,\boldsymbol{du} \quad (21)$$

From this, the function can be graphed, and visualizations can be made for the short-term and long-term profit fluctuation based on labor input, capital input, inflation rate, and fixed cost. The numerical solution and the details of the results are further explained in section 4.2.4.

4 Task 2: Application and Sensitivity Analysis

We first revisit the sources listed under Section 3.1 and gather concrete data to incorporate into the finished model. From this, the "best" options out of those listed by the decision-makers are determined using linear programming and TOPSIS. Their positioning is also calculated using a genetic algorithm. The short- and long-term profit analysis is then applied to the genetic algorithm results. Finally, a sensitivity analysis is conducted on our model to evaluate its reliability and the sensitivity of the results.

4.1 Data Collection

Using the sources in Table 2, we determined concrete numbers for each variable and each development option. Our final master data table with all of these values can be seen in Table 3.

Table 3 Data of Different Facilities

Measurement	Sports Comp.	Ski/Trail	Crop F.	Ranch	Regen. F.	Sol. Arr.	Agritrst Cn.	Agrvltc F.
P_j ($\$ \cdot acre^{-1}$)	9038.46	254.95	179.28	562.51	319.30	19155.41	3999.91	215.09
E_j ($kg \cdot acre^{-1}$)	200	50	-25	500	-20	-3330	-2	-2030
O_i	9.5	1.0	7.5	8.5	8.5	2.5	9.0	7.5
A_i	9.5	8.0	1.0	1.0	1.5	1.0	7.0	1.0
S_i	3.0	4.0	8.0	9.0	9.0	3.0	6.0	7.0
R_i	10.0	9.0	1.0	2.0	2.0	1.0	9.0	2.0

The Employment Opportunity Index O_i, Tourism Attraction Index A_i, Societal Benefit Index S_i, and Recreational Index R_i are all determined manually as a number from 1.0 to 10.0 via a thorough analysis of all available sources on each topic. The environmental degradation penalties E_W , E_D, E_F , and E_C are also determined using this method. The exact values can be seen in Table 14. The carbon emission data E_j are also rounded to the nearest integer.

4.2 The "Best" Option

The data described above are applied to the linear programming systems defined in Section 3.3, the TOPSIS process in Section 3.4, and the genetic algorithm introduced in Section 3.5. In this way, the "best" land development options and their distribution and positioning are determined.

4.2.1 Linear Programming Results

They were using the linear programming systems defined in Section 3.3 and the data described in Section 4.1 economic and environmental criteria results are calculated.

The optimal result with maximum annual profit, at $9 079 300, can be seen in Table 4; and the optimal result with minimum annual carbon emissions, at -1 869 100kg, can be seen in Table 5. These two results will advance to be employed in the genetic algorithm analysis.

The calculated minima for both economic and environmental have also been listed in these two tables. The two minima are used to calculate the least ideal possibility in TOPSIS.

Table 4 Economic Max./Min. Facilities ($9079300 and $144090 in Annual Profit)

Facilities	Sports Comp.	Ski/Trail	Crop F.	Ranch	Regen. F.	Sol. Arr.	Agritrst Cn.	Agrvltc F.
Economic Ideal (*acres*)	267	0	0	129	0	344	1	0
Economic Least Ideal (*acres*)	0	158	579	0	0	0	0	0

Table 5 Environmental Min./Max. Facilities (-1869100kg and +351000kg Annual CO_2 Emissions)

Facilities	Sports Comp.	Ski/Trail	Crop F.	Ranch	Regen. F.	Sol. Arr.	Agritrst Cn.	Agrvltc F.
Environment Ideal (acres)	0	0	0	0	0	510	147	84
Environment Least Ideal (acres)	65	0	0	676	0	0	0	0

4.2.2 TOPSIS Results

These results are then applied to the TOPSIS model described in Section 3.4. We first set r to 4.86 to standardize the relationship between environmental and economic based on the ranges found in the data.

$$\frac{|x_b|}{|y_b|} \approx 4.86 \tag{22}$$

h is also set to 2, making the final ratio/weighting between environmental to economic factors 1 : 2. This is because we believe that, although the environment is a significant factor, comparing solely carbon emissions to the entirety of all the economic benefits would be an unfair comparison. There are 3 economic factors and 1 environmental, so a 1-env. To 2-econ. Weighting best reflects our beliefs.

With r as 4.86 and h as 2, the coordinates of the ideal possibility are $A_b = (18158600, -9083826)$, and the coordinates of the least ideal possibility are $A_w = (288180, 1705860)$.

Next, the 19 manually determined points are plotted into TOPSIS and ranked according to ω, the result of which can be seen in Table 6 below.

Table 6 Alternatives According to Judging Index s_ω; ranked best-worst

Weighting	0.95 - 0.45 Econ.	0.4 Econ.	0.35 - 0.25 Econ.	0.2 Econ.	0.15 - 0.1 Econ.	0.05 Econ.
Index ω	0.463650	0.457908	0.457321	0.431908	0.431316	0.430778

The top-performing development option from TOPSIS, which will be modeled in GA, along with the two results above from linear programming, can be seen in Table 7. This option, with an annual profit of \$9047920 and annual CO_2 emissions of −1094700kg, is found at 0.95 - 0.45 Econ.

Table 7 Distribution of Top Option, according to TOPSIS

Facilities	Sports Comp.	Ski/Trail	Crop F.	Ranch	Regen. F.	Sol. Arr.	Agritrst Cn.	Agrvltc F.
Best option (*acres*)	267	0	0	0	129	344	1	0

4.2.3 Genetic Algorithm Results

We then use the genetic algorithm to determine the positioning of facilities for each of the three ideal results. This is done by following the GA process as described in Section 3.5.

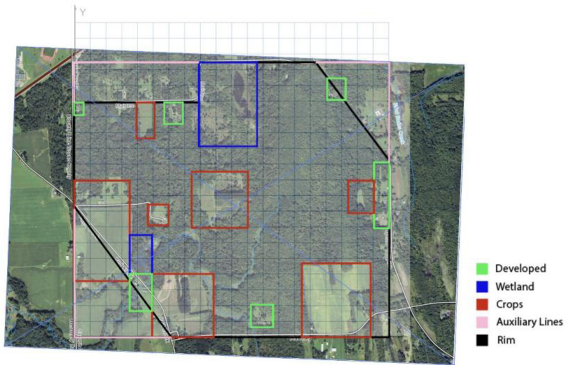


Figure 3 The Annotated and Plotted Map of All Land Types

- **Cartesian coordinate system:** The map is first plotted in the 2D Cartesian plane. The placement and area of each land type are assumed from the given satellite map view and outlined using rectangles. The distribution of each land type except forest is as follows: the forest is the remainder of the area. A limitation of this approach is the inability to accurately note the environment with rectangles, especially when this map could be more visually distinguishable. The tables in Appendix A specify the exact (x,y) coordinates of each rectangle's diagonal vertices.
- **Initialization:** As described in Section 3.5.1, all inputs are fed to the genes in the form of DNA, which, in our instance, contains the lower left corner and width of the rectangles. This is the only information needed to calculate the length of the rectangle with its area predetermined.
- Conversion: In this model, the DNA is 60 bits long. It is separated into 12 sections, each with a length of 5, storing data up to 31. The first 4 sets of 5 (the first 20) manifest the x coordinate of the rectangle's bottom left vertex. The second 20 are similar but for y, and the last 20 indicate each facility's width.

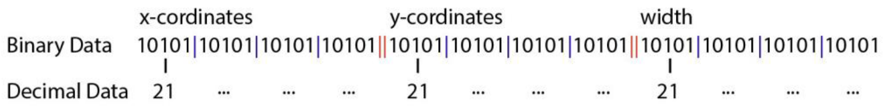


Figure 4 An Example of DNA String

- Objective Function Constraints:
- – No Overlaps: The overlapping area function is created following the restrictions described in Section 3.5.2, ensuring that facilities do not overlap with each other. To follow the rule, the code will ensure the overlapping area between two rectangles is 0. The overlapping area code can be seen in lines [31~49] of Appendix C.
- – No out-of-bounds: To ensure that land developments lie within the boundary, new restrictions are set to ensure:(a) the lower left corner of a facility does not exceed the hypotenuse of the lower left triangle; (b) the upper right corner does not exceed the hypotenuse of the upper right triangle.Both triangles are shown in pink in Figure 3. To incorporate this into the objective function, two linear inequalities are used. Let (*x,y*) be the lower left corner, A be the area, and w be the width. This gives the equation of the lower left hypotenuse $y = \frac{-8.5}{6}x + 8.5$ and the upper right hypotenuse as $y = \frac{-7.5}{4.5}x + 44.33333$. The constraints are thus:

$$\begin{gathered} y \geq \frac{-8.5}{6}x + 8.5 \\ y + \frac{A}{w} \leq \frac{-7.5}{4.5}(x + w) + 44.33333 \end{gathered} \tag{23}$$

If any of the constraints are not met by a string of DNA, then it is immediately abandoned with its fitness set to 0. Otherwise, if a string of DNA meets all of the constraints, its fitness value is calculated according to the objective function (see line [84~224] of Appendix C).

- **Objective Function:** Table 14 lists the values of the environmental degradation penalties E_W , E_D, E_F , and E_C to be employed in the objective function. To maximize the fitness of the DNA, the environmental degradation factor should have an inverse relationship with the preciousness of the environment. Thus, the environmental factor is 30 − in total.
- **Selection, Reproduction, and Mutation:** DNA sequences then undergo the process of Genetic Algorithm as written in Section 3.5. Their code appears in Appendix C.
- **Additional Variable - Occupation Rate:** Since TOPSIS grants the ratio between areas, there is one more variable called **occupation rate** that the decision-makers must decide. This variable dictates the proportion of the total land area the development is going to occupy. Let this variable be named γ. Then:

$$A_{cordinate} = A_{acres} \times 2.709\gamma. \tag{24}$$

- **Application:** The initial instance is generated in two ways, either randomly generated or hand-drawn. A random generation will be faster with a lower occupation rate, but hand-drawing will be more efficient when the occupation rate is 0.5 or higher. The facilities are graphed according to the DNA string to verify and debug the initial and final DNA (see Appendix B).

GA also relies on user inputs to determine the size of the DNA (choice and justification described above), population, crossover rate, and more. These are all determinate factors to the generation's success since the genetic algorithm only promises the maximum to the highest extent of convergence given the user inputs. The user inputs used in our instance are listed below.

Table 8 User Inputs

DNA size	Population	Cross Rate	Mutation Rate	Generations	width/x/y bounds
60	1000	0.8	0.002	400	[1,20] [0,20] [0,17.5]

- **Results:** With the aforementioned user inputs, the genetic algorithm runs across three different occupation rates throughout the three different models of facilities proposed by TOPSIS and linear programming.

Table 9 Resulting Fitness of All 3 Cases, calculated via the objective function

Occupation rate	TOPSIS Best	Economic Best	Environmental Best
0.4	1620.07338413	1246.89606331	1530.62376101
0.5	1899.65084277	1473.23451274	1761.06401523
0.6	2020.18783949	1455.66527764	1996.19740878

The distribution of facilities on the land, as calculated from the TOPSIS Best option at an occupation rate of 0.5, is provided in Figure 5. The result's DNA string, graph, and converted DNA, among others, are listed in Table 10. Similar maps and tables can be seen for the rest of the results in Appendix B. The overall evaluation of the proposed land planning and allocation is in Figure 6.

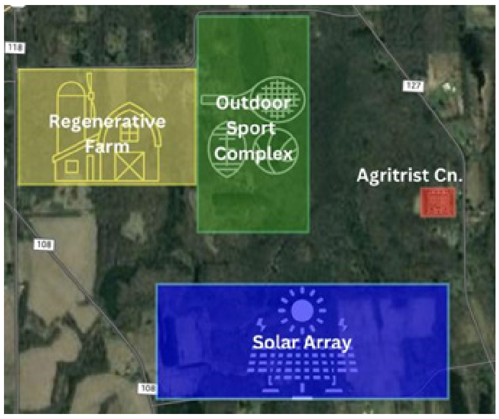


Figure 5 Map of GA Best Option with 0.5 Occupation Rate

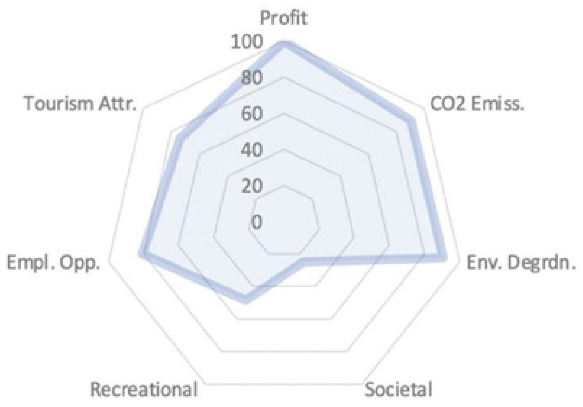
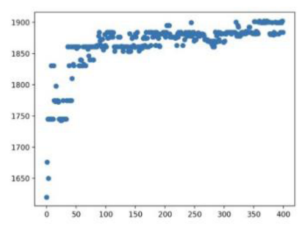


Figure 6 Evaluation Result

Table 10 DNA-related Data of TOPSIS Best Option with 0.5 Occupation Rate

Occupation rate	Fitness	Best DNA string	Learning Curve	Converted DNA string in the form of [x, y, width]
0.5	1899.7	0 1 1 0 1 0 0 0 0 0 0 1 0 1 0 1 1 0 0 1 0 1 1 1 0 1 0 1 0 1 0 0 0 0 0 0 1 1 1 1 0 1 0 0 0 0 1 1 0 0 1 0 1 0 0 0 0 0 0 1		Sports Complex:[7.7, 7.9, 5.2] Regen. Farm: [0.0, 11.9, 7.7] Solar Array: [6.5, 0.0, 12.9] Agritourist C.: [19.1, 8.5, 0.6]

4.2.4 Short- and Long-term Results

After applying all the obtained data into Equation (21) in Section 3.6, the function of total profit is:

$$T_p(t) = A\int_0^t \left(-\frac{500}{t+10}+50\right)^{0.3} \left(\frac{30}{1+\left(\frac{30}{2.6}-1\right)e^{-t}}\right)^{0.7} (1+\alpha)^{t-u}$$

$$du - 30 - \int_0^t log(12u+1)(1+\alpha)^{t-u}\, du$$

Where α is the inflation rate, and A is the efficiency constant that measures the ratio of output and input of labor and capital, respectively. In Figure 7, the projected futures include best-case, expected, and worst-case futures. These projections are determined by adjusting the aforementioned constants (A and α).

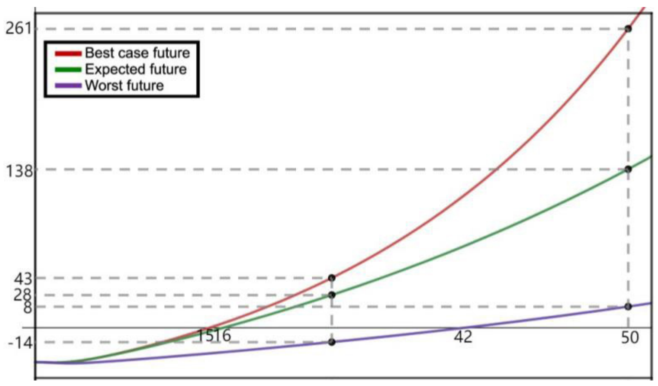


Figure 7 The Projected Future

In the short term, profits for all cases will be negative as construction costs must be covered, but the production value is limited due to less labor and capital input. However, in the long term, profits will grow from inflation rates and high production value. Better case scenarios project higher profits.

The best-case scenario predicts that fixed costs will be covered in year 15, resulting in a total profit of 43 million by year 25 and 261 million by year 50. The expected future, calculated using the most likely inflation and efficiency values, forecasts that fixed costs will be covered in year 16, leading to a total profit of 28 million by year 25 and 138 million by year 50. In the worst-case scenario, fixed costs will be covered in year 42, and the total profit will be 7 million by year 50. Despite this, however, the model's results are considered dependable profit-wise because even the worst-case scenario yields a positive total profit within 50 years.

4.3 Sensitivity Analysis

Sensitivity analysis must be conducted to assess the degree of uncertainty and variability and identify the parameters that impact the results most. This information can then be used to refine the model, optimize its parameters, or identify areas where further data collection may be needed.

4.3.1 Sensitivity Analysis of Linear Programming

Two aspects can be tested and manipulated to conduct a sensitivity analysis [4] of linear programming: tight constraints [1] and shadow price [7].

Tight restrictions refer to constraints met with equality in the objective value, thus limiting it. The restrictions in linear programming will always form a convex polygon, and the objective function is, in essence, another line on the coordinate plane. The objective function will always intersect the polygon at one of its corners to obtain the most optimal value. Tight restrictions are the inequalities, which, when plotted on the coordinate plane, form the corner intersected by the objective function. On the other hand, loose restrictions are those that do not influence the result whatsoever. Nonetheless, the polygon might shift in shape, thus altering the tight or loose status of the restrictions.

Shadow price refers to the change in the objective value that results from a difference in the range of the tight restrictions. As previously mentioned, tight restrictions are those that form the corner where the objective function will intersect to reach the optimal solution. Thus, any change in their range will result in a return value dependent on the magnitude of the change.

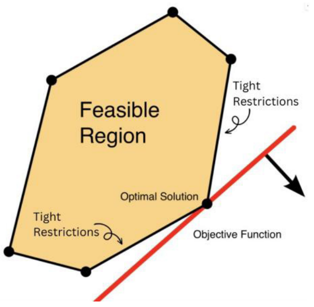


Figure 8 Linear Programming Demonstration in 2D; image taken from [5]

Tight constraints and shadow prices are calculated for the environmental and economic linear programming systems. First, three restrictions are tested as tight constraints for **the environmental linear programming system.** They are as follows:

1. the primary area restriction $\sum_{i=1}^{n} x_i^A \leq 741.316$;

2. the societal benefit index restriction $\sum_{i=1}^{n} S_i \times x_i^A \geq 3000$;

3. the recreation index restriction $\sum_{i=1}^{n} R_i \times x_i^A \geq 2000$.

As is shown in Figure 9, the sensitivity analysis graph for the area restriction shows a linear decrease in the range observed (67.5% to 132.5%), suggesting that as the restrictive range increases, the carbon emissions decrease. At 67.5%, the restrictions oppose each other, making further optimization impossible.

The sensitivity analysis graph for societal benefit restrictions is piece-wise, comprising three distinct regions with different slopes. The slope of 0 in the first part (50% to 74.5%) indicates a loose restriction before 74.5%. The second part's positive slope means the return value from a percentage increase is unfavorable. The third phase has a steeper slope, demonstrating a more unfavorable return value.

The sensitivity analysis graph for recreational opportunity restrictions is also piece-wise, with an almost unnoticeable increase in the slope at the 140.5% point. It demonstrates unfavorable return values for percentage increases in the constraining constant.

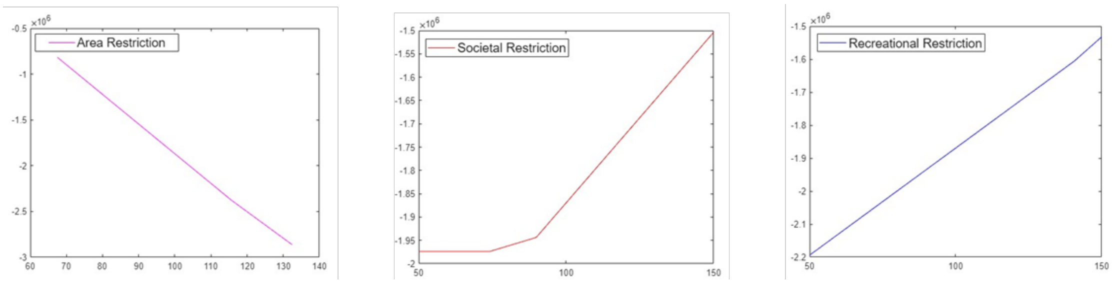


Figure 9 Sensitivity Analysis and Shadow Prices for Environmental Linear Programming System

In **the economic linear programming system**, the restrictions tested as tight constraints include:

1. the basic area restriction $\sum_{i=1}^{n} x_i^A \le 741.316$;

2. the societal benefit index restriction $\sum_{i=1}^{n} S_i \times x_i^A \ge 3000$;

3. the third restriction is not the recreation index but the employment opportunity index restriction $\sum_{i=1}^{n} O_i \times x_i^A \ge 4500$.

As shown in Figure 10, these graphs almost precisely mirror the shadow price graphs of the tight environmental constraints. Where decreases were seen previously, increases are seen here, and vice-versa. However, there are still a few exceptions.

- In the societal restriction of the economic section, the graph exhibits a longer second phase with a steeper slope in addition to a smoother transition from phases 2 to 3.
- The economic employment opportunity sensitivity graph differs from the environment's recreation restriction. This graph has a slope of 0 from 50% to 69.5%, indicating a loose restriction status there. After that, however, it has a sustained slope inside the observed range, with a direct relationship between the decrease in profit and the increase in the constraining constant.

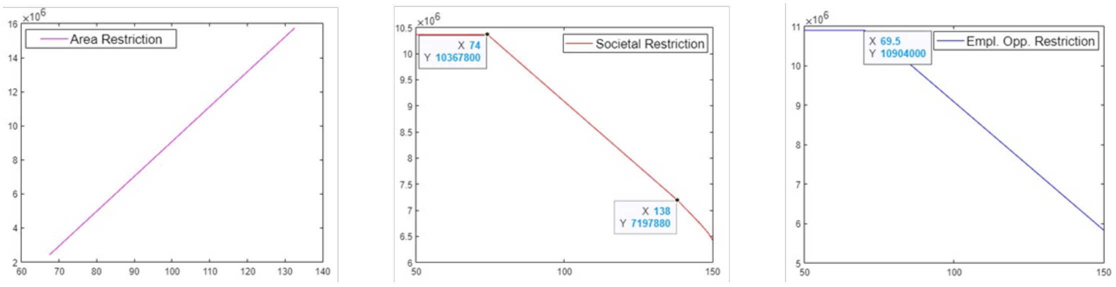


Figure 10 Sensitivity Analysis and Shadow Prices for Economic Linear Programming System

In conclusion, to reduce carbon emissions, either societal or recreational constraining constants should be decreased, or area constraining constants should be increased. To increase profits, societal and employment opportunity constraining constants should be reduced, or area constraining constants should be increased. However, adjusting these factors would reduce the respective benefits provided by the land.

4.3.2 Sensitivity Analysis of TOPSIS

One of the most significant uncertainty factors in TOPSIS model is the land developer's opinion on the weighting factor between the environment and the economy, which is assumed to be 0.33 Env. :0.66 Econ. in our model.

To test and analyze the sensitivity of the model proposed, the weighting factor is changed from (0.2 : 0.8) ~ (0.8 : 0.2), increasing by increments of (+0.1 : −0.1).

However, although the judging indexes changed, the overall ranking stayed mostly the same. Most notably, the first two choices stayed the same throughout, as seen in Figure 11. This demonstrates the reliability and stability of TOPSIS and, thus, the reliability and stability of its suggestion of the "best" land development option.

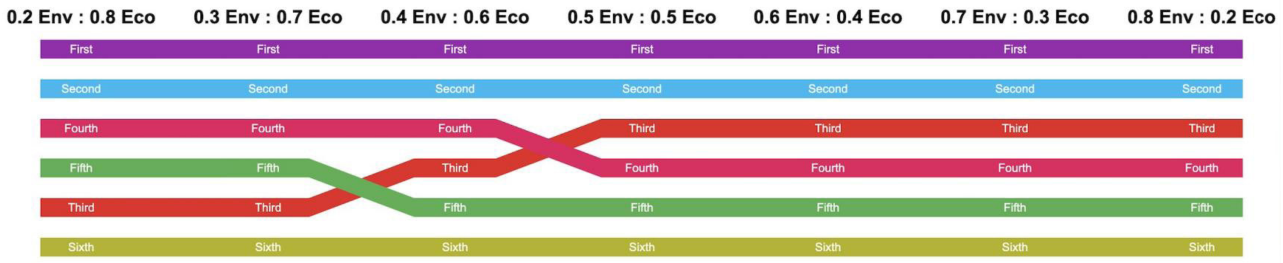


Figure 11 TOPSIS Result Fluctuation Under Distinct Weighting

5 Task 3: Re-evaluation for Micron Tech., Inc.

In October 2022, it was announced that Micron Technology, Inc. will build a large semiconductor fabrication facility (fab) in Clay, a town just north of Syracuse. The fab is projected to bring many more jobs and thus many more people. To account for this fab's changes to the local community, we re-evaluate the affected criteria and re-run our model based on this.

5.1 Affected Factors

The question announces the establishment of a new large semiconductor fabricator(fab) near the land being modeled, which is expected to significantly impact local employment, production value, and tourism attraction. To comprehensively assess the impact of the new fab on our metrics, one needs to carefully consider these factors' influence on the data and model.

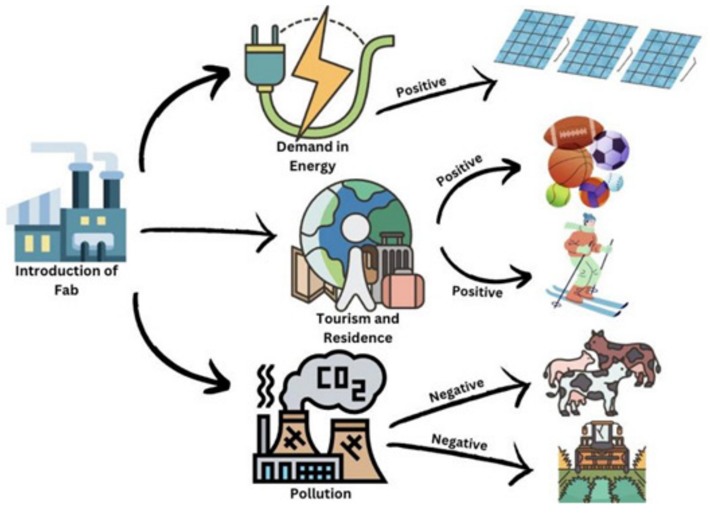


Figure 12 Affected Factors

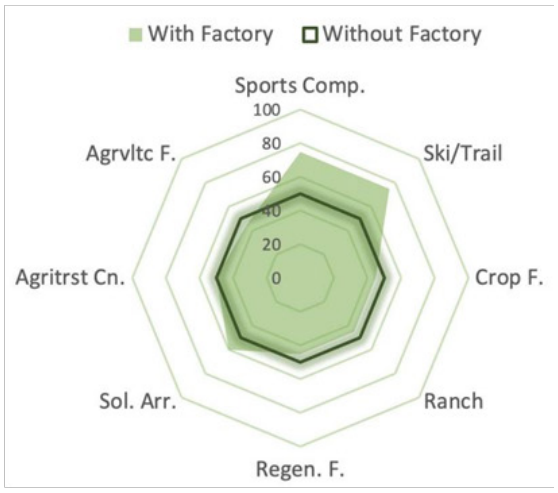


Figure 13 Adjusted Profits

5.1.1 Change in Facility Profit

- **Solar array:** Solar array heavily relies on energy requirement in the local community. Introducing the fab will increase the energy demand, thereby increasing the profit of solar arrays.
- **Crop farms, agrivoltaic farms, regenerative farms, and ranches:** As food quality plays a decisive factor in Americans' selection for food, the newly built fab will significantly decrease the profit of these facilities as the demand for the product reduces.
- **Sports Complex, Cross-country ski/trail:** Both of these facilities' profit relies heavily on attraction and the living conditions of their local community. As this fad is estimated to introduce 49,000 more jobs with a high annual salary of over $100,000, a significant rise in profit can be predicted.
- **Agritourist Center:** The Agritourist Center's resultant influence is a double-edged sword. The introduction of fab boosts its tourism attraction while decreasing the demand for its agricultural goods.

Table 11 Estimated Adjustment to the Facility Profit, Contrast in Figure 13

Measurement	Sports Comp.	Ski/Trail	Crop F.	Ranch	Regen. F.	Sol. Arr.	Agritrst Cn.	Agrvltc F.
P_j (\$acre^{-1})	13557.69 ↑	382.43 ↑	161.35 ↓	509.26 ↓	287.37 ↓	21070.95 ↑	3999.91 −	193.58 ↓

5.1.2 Change in Restrictions

Many restrictions in the linear programming systems are based on the land's property. With the introduction of the fab, some of these factors need to be reconsidered.

- **Employment Index Restriction Removed:** As this new fab introduces 9000 direct jobs with 100,000 annual salaries and more than 40,000 indirect jobs, the restriction for the employment index $\sum_{i=1}^{n} O_i \times x_i^A \geq 4500$ can be removed.
- **Tourism Attraction Index Restriction Reduced:** This fab greatly increases tourism and resident attraction, so the attraction index limit is reduced to 1000. $\sum_{i=1}^{n} A_i \times x_i^A \geq 1000$

5.2 New Plan

Using the same linear programming and TOPSIS with minor changes to the models discussed above, one could determine the ideal environmental, economical, and overall.

Table 12 Proposed Plan in the Presence of the Fab

Facilities	Sports Comp.	Ski/Trail	Crop F.	Ranch	Regen. F.	Sol. Arr.	Agritrst Cn.	Agrvltc F.
Environment Ideal (*acres*)	0	0	0	0	0	510−	147−	84−
Economic Ideal (*acres*)	123↓	0	0	128↑	0↓	487↑	3↑	0
Overall Ideal (*acres*)	123↓	0	0	128↑	0↓	487↑	3↑	0
Economic Ideal and Overall Ideal: • Annual profit: $12006300↑ • Annual Carbon Emission: −1533120kg↓	**Environment Ideal:** • Annual profit: $11350400− • Annual Carbon Emission: −1869100kg−							

With the given TOPSIS statistics, the Genetic Algorithm can calculate the facilities' optimal placements, as shown in Figure 14. For specific data, see Appendix B Table 21)

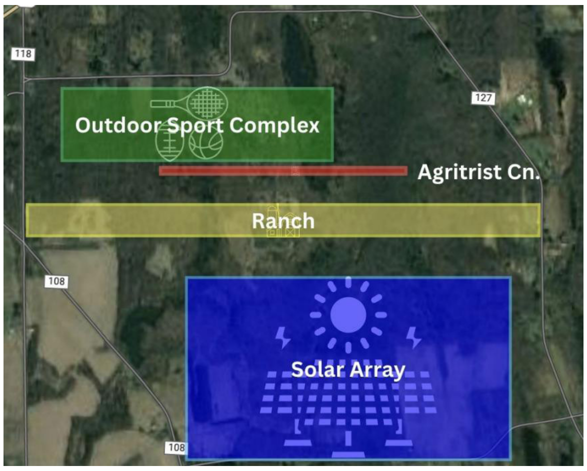


Figure 14 Distribution of Facilities With Fab

In conclusion, with the introduction of the new fab that brings forward many opportunities and changes, the annual profit will see a significant rise of over 30% and the annual carbon emission of over 45%. However, alongside these benefits is a decrease in the environmental degradation index of 11%.

6 Task 4: Generalizability

The most distinct aspect of this piece of land lies in the fact that the land is in a rural environment. This means that there are far fewer restrictions around the use of the land, which allows purely mathematical modeling to approach an answer that is far more applicable in real life. Thus, in considering the generalizability of this model, one must also keep in mind that it will be the most realistically applicable in rural environments due to its very nature.

6.1 Familiar Contexts

(Note: The land in New York will be called "original"; our familiar ground will be called "new".)

Some of our team members have ancestry in Shenzhen, China, so Shenzhen will be the familiar context for discussing this model's applicability. The new land can be seen below, under Figure 15. It is 3km^2 in size and lies roughly 26km east of the city center. Many new considerations exist for this plot, as can be seen in Table 13.

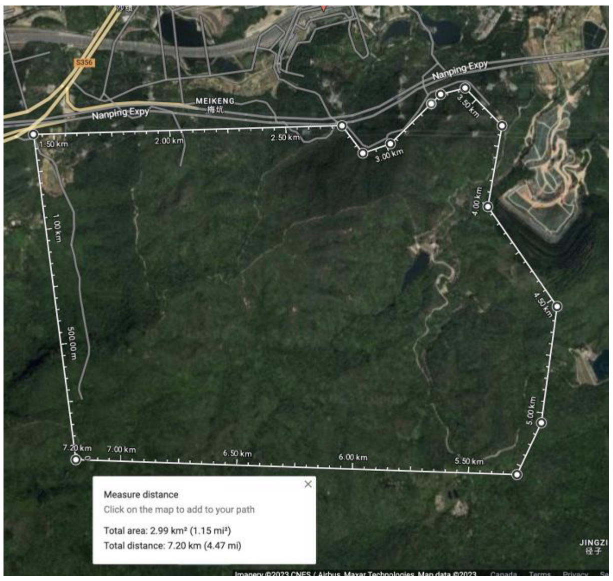


Figure 15 New Land in Shenzhen

Table 13 Affected Criteria

Factor	Change(s)
Pop.Density	Adjust restrictions for P,Oi,Ai
Climate	Remove the cross-country sking option

Landscape	Adjustments to genetic algorithm and land distribution
Urbanization	Restrictions around development size and emissions

- **Population Density:** The population of Shenzhen is far greater than Syracuse's, with 12.59 million for the former and 0.15 million for the latter [12]. As was the case in Task 3, more people means that developments that attract people will increase profit, and developments that don't will decrease. Restrictions around P, O_i, and A_i should be adjusted accordingly.
- **Climate:** The climate in Shenzhen is subtropical, whereas the climate around Syracuse is continental [12]. This has implications for any building type that includes outdoor requirements. Additionally, a subtropical climate makes it impossible for cross-country skiing to be viable at all.
- **Landscape:** The original plot was only somewhat close to certain freshwater lakes and had five different land types: forest, crop, wetland, developed, and shrub. The new plot, however, is very close to the South China Sea and has three land types: forest, wetland, and developed. The genetic algorithm must be modified to consider only these three land types, and changes must be made to accommodate the new percentage distribution of each.
- **Urbanization:** Since the new land is so close to the Shenzhen city center, urban planning will heavily influence it. New restrictions should be placed around the maximum size of a given development, and social factors R_i and S_i should be adjusted to reflect an urban society's different needs and wants. Restrictions around carbon emissions E should also be tighter to reflect the increased seriousness of pollution in cities.

6.2 Land in Other Countries

Many of the same parallels remain when considerations are expanded to an international level. As long as considerations and adjustments are made to reflect the characteristics of the plot of land (such as urban/rural setting, climate, landscape, and unique factors), our model can be applied and can present a solution. This reflects the versatility of linear programming: as long as variables are related linearly, the model will produce results.

7 Conclusion and Evaluation

7.1 Evaluation of Strengths and Weaknesses

Strengths

- Linear programming allows maximizing the land's potential within given constraints, ensuring the output is as ideal as possible. The simplex algorithm is also highly efficient, drastically reducing the calculation time.
- TOPSIS provides a straightforward method for finding an optimal solution when distinct or conflicting criteria are present and require a desired weighting factor.
- Genetic algorithm is powerful due to its ability to perform a global search, even in complex and non-linear search spaces. It also converges to a good solution thanks to the diversity maintained within the population.

Weaknesses

- Linear programming is limited to problems with linear and continuous relationships between decision variables and the objective function. Additionally, it is highly sensitive to input data, so small differences in data can result in significant changes in output.
- While TOPSIS assumes normalized data without outliers, outliers can still affect rankings and lead to incorrect conclusions. Furthermore, because it requires the selection of a weighting factor, TOPSIS results introduce subjectivity, making them less reliable objectively.
- Genetic algorithms can be time-consuming, requiring many function evaluations to find a good solution, leading to severe time complexity. Additionally, they can get stuck in local optima, where the algorithm finds a suboptimal solution that is better than its neighbors but not the global optimum.

7.2 Conclusion

This paper aims to develop a comprehensive, quantitative approach to determining the optimal planning and allocation of land. Our team achieves this by proposing a base mathematical model that integrates linear programming, TOPSIS, and genetic algorithm. This model considers seven factors that belong to either economic or social benefits or environmental detriments. Furthermore, a short and long-term analysis model incorporating key factors like inflation, labor cost, and operating cost is included to evaluate the time-based feasibility of plans proposed by the base model. Results show that the model successfully determines the most optimal planning and allocation of land, even in unfamiliar situations, such as new fabrication facilities built nearby or densely populated urban areas. A sensitivity analysis also reveals the stability of TOPSIS results and possible approaches to enhance each criterion in linear programming further. These findings demonstrate the versatility and applicability of the proposed approach in a variety of contexts, as well as its reliability in producing results.

References

[1] Richard Battye, Bjoern Garbrecht, and Adam Moss. Tight constraints on f-and d-term hybrid inflation scenarios. Physical Review D, 81(12):123512, 2010.

[2]Majid Behzadian, S Khanmohammadi Otaghsara, Morteza Yazdani, and Joshua Ignatius. A state-of-the-art survey of topics applications. Expert Systems with applications, 39(17):13051–13069, 2012.

[3]Edward J Blakely. Urban planning for climate change. 2007.

[4]H Christopher Frey and Sumeet R Patil. Identification and review of sensitivity analysis methods. Risk analysis, 22(3):553–578, 2002.

[5]Wikimedia Commons. File:linear optimization in a 2-dimensional polytope.svg — wikimedia commons, the free media repository, 2022. [Online; accessed April-2023].

[6]Juan Angel Demerutis and Magdalena Vicu´ na. Urban and regional planning in Latin America and the Caribbean. The Routledge Handbook of Urban Studies in Latin America and the Caribbean, pages 357–382, 2023.

[7]Jean Dreze and Nicholas Stern.` Policy reform, shadow prices, and market prices. Journal of public economics, 42(1):1–45, 1990.

[8]Tomas J Havr' anek.' Cobb–douglas production function. In Dictionary of Ecological Economics, pages 71–71. Edward Elgar Publishing, 2023.

[9]Seyedali Mirjalili and Seyedali Mirjalili. Genetic algorithm. Evolutionary Algorithms and Neural Networks: Theory and Applications, pages 43–55, 2019.

[10]Stephen Palmer and James Raftery. Opportunity cost. Bmj, 318(7197):1551–1552, 1999.

[11]Jason Papathanasiou, Nikolaos Ploskas, Jason Papathanasiou, and Nikolaos Ploskas. Topsis. Multiple Criteria Decision Aid: Methods, Examples and Python Implementations, pages 1–30, 2018.

[12]Jianfa Shen. Urban growth and sustainable development in Shenzhen city 1980-2006. Open Environmental Sciences Journal, 2(1), 2008.

[13]Robert J Vanderbei et al. Linear programming. Springer, 2020.

Appendix A Substantiating Tables and Data

Table 14 Environmental Degradation Penalties

	Wetland	Developed	Forest	Crop
Biodiversity	8	1	9	2
Soil Erosion	7	0	10	2
Pollution	9	0	8	1
Total	24	1	27	5
E Factor	6	29	3	25

Table 15 Developed Coordi-nates

lower-left vertex	higher-right vertex
(0, 3.5)	(3.5, 10)
(5, 0)	(9, 4)
(4.5, 7)	(6, 8.5)
(7.5, 7)	(11, 10.5)
(4, 12.5)	(5, 15)
(14.5, 0)	(19, 5)
(17.5, 8)	(19, 10)

Table 16 Crop Land Coordinates

lower-left vertex	higher-right vertex
(3.5, 1.5)	(5, 4)
(11.5, 0.5)	(12.5, 2)
(0, 14)	(0.5, 15)
(5.5, 13.5)	(7, 15)
(19, 7)	(20, 11)
(16, 15)	(17, 16.5)

Table 17 Wetland Coordinates

lower-left vertex	higher-right vertex
(3.5, 4)	(5, 6.5)
(8, 12)	(11.5, 18.5)

Appendix B All learning Curves and Final DNA of Genetic Algorithm

Table 18 Further DNA-related Data of the TOPSIS Overall Best option with 0.4 - 0.6 Occupation Rate

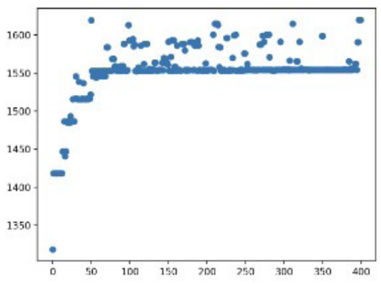
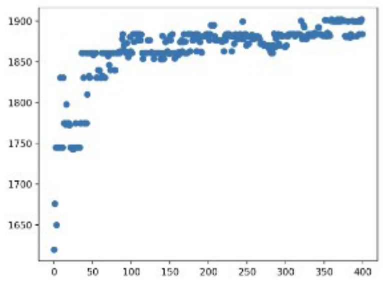
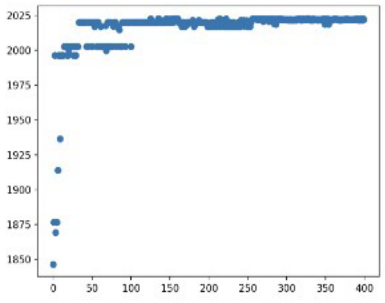
Occupation rate	Fitness	Best DNA string	Learning Curve	Converted DNA string in the form of [x, y, width]
0.4	1620.1	0 1 0 0 1 0 0 0 0 0 0 1 0 1 0 1 1 0 0 1 0 1 1 0 1 1 0 0 0 0 0 0 0 0 0 0 1 1 0 1 0 1 0 0 0 0 0 1 0 1 1 0 1 0 0 0 0 1 0 1		Sports Comp. [5.8, 7.3, 5.2] Regen. F. [0.0, 9.0, 3.2] Sol. Arr. [6.5, 0.0, 12.9] Agritrst Cn. [16.1, 7.3, 3.2]
0.5	1899.7	0 1 1 0 1 0 0 0 0 0 0 1 0 1 0 1 1 0 0 1 0 1 1 1 0 1 0 1 0 1 0 0 0 0 0 0 1 1 1 1 0 1 0 0 0 0 1 1 0 0 1 0 1 0 0 0 0 0 0 1		Sports Comp. [7.7, 7.9, 5.2] Regen. F. [0.0, 11.9, 7.7] Sol. Arr. [6.5, 0.0, 12.9] Agritrst Cn. [19.1, 8.5, 0.6]
0.6	2020.2	0 1 1 0 1 0 0 0 0 0 0 1 0 1 0 1 1 0 1 1 0 1 1 0 0 1 0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 1 0 0 1 0 1 0 0 0 1 0 1 0 0 0 0 0 1 1		Sports Comp.[8.4, 6.8, 5.8] Regen. F. [0.0, 9.0, 5.2] Sol. Arr. [6.5, 0.0, 12.9] Agritrst Cn. [17.4, 8.5, 1.9]

Table 19 Further DNA-related Data of the Economic best option with 0.4 - 0.6 Occupation Rate

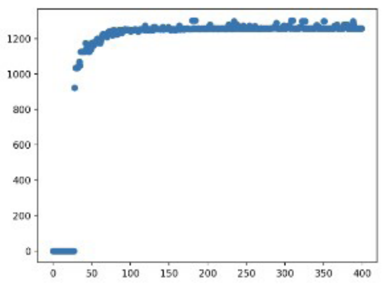
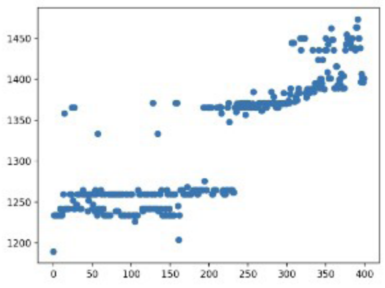
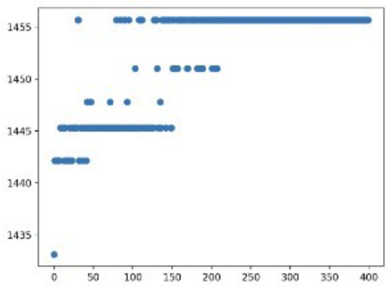
Occupation rate	Fitness	Best DNA string	Learning Curve	Converted DNA string in the form of [x, y, width]
0.4	1246.9	0 1 1 1 0 0 0 0 0 0 1 0 1 1 0 0 1 0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 1 0 1 0 0 0 0 1 1 1 0 1 0 0 1		Sol. Arr.[0.0, 9.0, 12.9] Agritrst Cn. [14.2, 0.0, 4.5] Agrvltc F. [6.5, 0.0, 5.8]
0.5	1473.2	0 1 1 1 1 0 1 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 1 0 1 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 1 1 0 1 1 0 1 1 1 1 0 0 1 1 1 1		Sol. Arr. [5.2, 2.3, 14.2] Agritrst Cn. [0.6, 9.0, 19.4] Agrvltc F. [10.3, 0.0, 9.7]
0.6	1455.7	0 1 0 1 0 0 1 0 0 0 0 0 1 1 0 1 0 0 1 1 1 1 1 1 0 0 0 1 0 0 1 0 0 1 1 1 0 1 0 1 0 0 1 1 1 1 0 1 1 1 0 1 1 0 0 0 0 1 1 1		Sol. Arr. [5.2, 2.3, 14.8] Agritrst Cn. [3.9, 10.7, 7.7] Agrvltc F. [12.3, 11.9, 4.5]

Table 20 Further DNA-related Data of the Environmental best option with 0.4 - 0.6 Occupation Rate

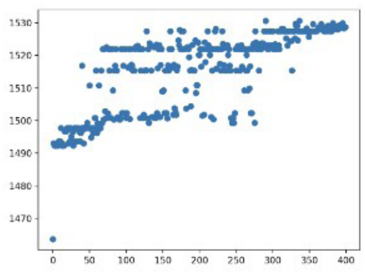
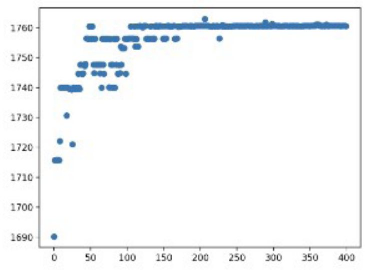
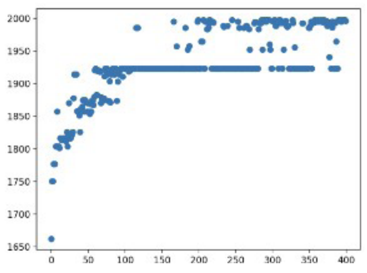
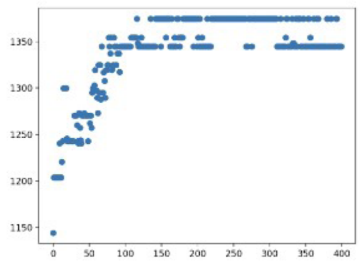
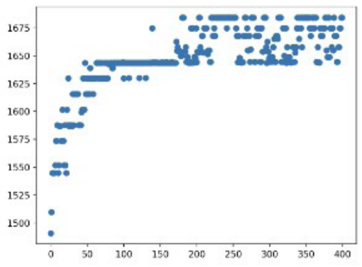
Occupation rate	Fitness	Best DNA string	Learning Curve	Converted DNA string in the form of [x, y, width]
0.4	1530.6	0 0 1 0 0 0 0 0 0 0 0 1 0 1 0 0 1 1 0 1 0 1 0 1 1 1 0 1 1 1 0 0 0 0 0 1 0 0 1 0 1 0 0 0 0 1 0 0 0 0 1 0 0 1 1 0 0 0 1 0		Sport Comp. [2.6, 6.2, 10.3] Ranch [0.0, 13.0, 10.3] Sol. Arr. [6.5, 0.0, 12.3] Agritrst Cn. [8.4, 10.2, 1.3]
0.5	1761.1	0 0 1 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 0 0 0 1 0 1 0 1 0 1 1 0 0 0 0 0 0 1 1 0 1 0 1 0 0 0 0 1 0 0 0 1 1 0 0 1 1 0 0 0 1 0		Sports Comp. [2.6, 5.6, 10.3] Ranch [0.0, 12.4, 11.0] Sol. Arr. [6.5, 0.0, 12.3] Agritrst Cn. [0.0, 14.7, 1.3]
0.6	1996.2	1 0 1 1 0 0 1 0 0 1 0 0 0 0 1 0 0 1 1 0 0 0 0 0 0 0 0 0 0 1 0 1 1 1 0 0 1 1 0 1 0 1 0 0 1 0 1 0 1 1 1 0 0 0 1 0 0 0 1 1		Sports Comp. [14.2, 0.0, 5.8] Ranch [5.8, 0.6, 7.1] Sol. Arr.[0.6, 7.9, 11.0] Agritrst Cn. [3.9, 7.3, 1.9]

Table 21 Further DNA-related Data of different best option with 0.5 Occupation Rate with the addition of factory

Occupation rate & Cases	Fitness	Best DNA string	Learning Curve	Converted DNA string in the form of [x, y, width]
Environmental 0.5	1374.6	0 1 1 0 0 0 0 0 0 1 0 0 1 0 0 1 1 0 0 0 0 0 0 1 0 0 1 1 1 0 0 1 0 0 1 0 0 0 0 0 0 0 1 0 0 1 0 1 0 1 1 0 0 0 0 0 0 1 0 1		Sol. Arr. [0.6, 7.9, 13.5] Agritrst Cn. [2.6, 5.1, 10.3] Agrvltc F.[15.5, 0.0, 3.2]
Economic / All 0.5	1684.1	0 0 0 1 0 0 0 0 0 0 0 1 0 1 0 0 1 0 0 0 1 0 1 1 0 1 0 0 0 0 0 0 0 0 0 1 0 1 0 1 1 0 0 0 0 1 1 1 1 1 1 0 0 1 1 0 1 1 1 1		Sports Comp. [1.3, 12.4, 10.3] Ranch [0.0, 9.0, 20.0] Sol. Arr. [6.5, 0.0, 12.3] Agritrst Cn. [5.2, 11.9, 9.7]

Appendix C Genetic Algorithm Code

Table 22 Appendix Code of GA Processes

GA Process	Code Lines
Objective Function	[84~224]
Conversion	[230~244]
Natural Selection	[247~254]
Reproduction	[257~262]
Mutation	[265~269]

```
"""
Visualize Genetic Algorithm to find a maximum point in a function.
""" import numpy as np import matplotlib.pyplot as plt
DNA_SIZE = 5 * 3 * 4                    # DNA length
POP_SIZE = 1000 # population size
CROSS_RATE = 0.8                        # mating probability (DNA crossover)
MUTATION_RATE = 0.002 # mutation probability N_GENERATIONS = 400 length_bound = [1, 20]      # x upper
and lower bounds x_bound = [0, 20] y_bound = [0, 17.5]
# Python program to find total area of two
# overlapping Rectangles
# Returns Total Area of two overlap
# rectangles
def binary_to_decimal(binary_string): decimal = 0 for i in range(len(binary_string)): digit = int(binary_string[i]) power =
    4 - i decimal += digit * (2 ** power)
    return decimal
def overlappingArea(l1, r1, l2, r2): x = 0 y = 1
    ''' Length of intersecting part i.e start from max(l1[x], l2[x]) of x-coordinate and end at min(r1[x], r2[x]) x-coordinate
        by subtracting start from end we get required lengths '''
    x_dist = (min(r1[x], r2[x]) max(l1[x], l2[x]))
    y_dist = (min(r1[y], r2[y]) max(l1[y], l2[y]))
    areaI = 0 if x_dist > 0 and y_dist > 0:
        areaI = x_dist * y_dist return areaI
#all environmental factors
Ew = 30 - 24
Ed = 30 - 1
Ef = 30 - 27 Ec = 30 - 5
osc_area = 267 * 0.6 / 2.709 rf_area = 129 * 0.6 / 2.709 sa_area = 344 * 0.6 / 2.709 ac_area = 1 * 0.6 / 2.709
CRO = [[[4, 12.5], [5, 15]], [[17.5, 8], [19, 10]],
       [[7.5, 7], [11, 10.5]],
       [[4.5, 7], [6, 8.5]],
       [[0, 3.5], [3.5, 10]],
       [[14.5, 0], [19, 5]],
       [[5, 0], [9, 4]]]
dev = [[[16, 15], [17, 16.5]],
        [[0, 14], [0.5, 15]],
        [[5.5, 13.5], [7,15]],
        [[19, 7], [20, 11]],
        [[3.5, 1.5], [5, 4]],
        [[11.5, 0.5], [12.5, 2]]]
wet = [[[8, 12], [11.5, 18.5]],
           [[3.5, 4], [5, 6.5]]]
squareconstrains = [[[0, 15], [8, 17.5]],
                    [[0, 17.5], [20, 1000]],
                    [[20, 0], [1000, 17.5]]]
def F(osc_x, rf_x, sa_x, ac_x, osc_y, rf_y, sa_y, ac_y, osc_width, rf_width, sa_width, ac_width):
    global osc_area, rf_area, sa_area, ac_area, dev, CRO, wet, Ew, Ec, Ef, Ed total = 0 if(osc_width == 0): return 0
    if (rf_width == 0): return 0
    if (sa_width == 0): return 0
    if (ac_width == 0):
        return 0
    osc_length = osc_area / osc_width rf_length = rf_area / rf_width sa_length = sa_area / sa_width ac_length = ac_area
    / ac_width if(osc_y < -8.5/6 * osc_x + 8.5): return 0 if (rf_y < -8.5 / 6 * rf_x + 8.5): return 0
    if (ac_y < -8.5 / 6 * ac_x + 8.5): return 0
    if (sa_y < -8.5 / 6 * sa_x + 8.5):
        return 0
    if (osc_y + osc_length > -7.5 / 4.5 * (osc_x + osc_width) + 44.33333):
```

```
    return 0
if (rf_y + rf_length > -7.5 / 4.5 * (rf_x + rf_width) + 44.33333): return 0
if (ac_y + ac_length > -7.5 / 4.5 * (ac_x + ac_width) + 44.33333): return 0
if (sa_y + sa_length > -7.5 / 4.5 * (sa_x + sa_width) + 44.33333): return 0
osc = [[osc_x, osc_y], [osc_x + osc_width, osc_y + osc_length]] rf = [[rf_x, rf_y], [rf_x + rf_width, rf_y + rf_
length]] sa = [[sa_x, sa_y], [sa_x + sa_width, sa_y + sa_length]]
ac = [[ac_x, ac_y], [ac_x + ac_width, ac_y + ac_length]]
#Outdoor Sport Complex area = osc_area
#developed for i in range(0, 6):
    total += Ed * overlappingArea(osc[0], osc[1], dev[i][0], dev[i][1]) area -= overlappingArea(osc[0], osc[1],
    dev[i][0], dev[i][1]) area -= overlappingArea(osc[0], osc[1], dev[i][0], dev[i][1])
#Crop for i in range(0, 7):
    total += Ed * overlappingArea(osc[0], osc[1], cro[i][0], cro[i][1]) area -= overlappingArea(osc[0], osc[1],
    cro[i][0], cro[i][1])
#Wetland for i in range(0, 2):
    total += Ed * overlappingArea(osc[0], osc[1], wet[i][0], wet[i][1]) area -= overlappingArea(osc[0], osc[1],
    wet[i][0], wet[i][1])
#taking off the part that is not in the map for j in range(0, 3):
    if (overlappingArea(osc[0], osc[1], squareconstrains[j][0], squareconstrains[j][1]) > 0):
        return 0
total += area * Ef
# Regenetive farm area = rf_area # developed for i in range(0, 6):
    total += Ed * overlappingArea(rf[0], rf[1], dev[i][0], dev[i][1]) area -= overlappingArea(rf[0], rf[1], dev[i][0],
    dev[i][1])
# Crop for i in range(0, 7):
    total += Ec * overlappingArea(rf[0], rf[1], cro[i][0], cro[i][1]) area -= overlappingArea(rf[0], rf[1], cro[i][0],
    cro[i][1])
# Wetland for i in range(0, 2):
    total += Ew * overlappingArea(rf[0], rf[1], wet[i][0], wet[i][1]) area -= overlappingArea(rf[0], rf[1], wet[i][0],
    wet[i][1])
# taking off the part that is not in the map for j in range(0, 3):
    if (overlappingArea(rf[0], rf[1], squareconstrains[j][0], squareconstrains[j][1]) > 0):
        return 0
total += area * Ef
#Solar array area = sa_area # developed for i in range(0, 6):
    total += Ed * overlappingArea(sa[0], sa[1], dev[i][0], dev[i][1]) area -= overlappingArea(sa[0], sa[1], dev[i][0],
    dev[i][1])
# Crop for i in range(0, 7):
    total += Ec * overlappingArea(sa[0], sa[1], cro[i][0], cro[i][1]) area -= overlappingArea(sa[0], sa[1], cro[i][0],
    cro[i][1])
# Wetland for i in range(0, 2):
    total += Ew * overlappingArea(sa[0], sa[1], wet[i][0], wet[i][1]) area -= overlappingArea(sa[0], sa[1], wet[i][0],
    wet[i][1])
# taking off the part that is not in the map for j in range(0, 3):
    if (overlappingArea(sa[0], sa[1], squareconstrains[j][0], squareconstrains[j][1]) > 0):
```

```
                return 0
        total += area * Ef
        # agriculture center area = ac_area # developed for i in range(0, 6):
            total += Ed * overlappingArea(ac[0], ac[1], dev[i][0], dev[i][1]) area -= overlappingArea(ac[0], ac[1], dev[i][0],
            dev[i][1])
        # Crop for i in range(0, 7):
            total += Ec * overlappingArea(ac[0], ac[1], cro[i][0], cro[i][1]) area -= overlappingArea(ac[0], ac[1], cro[i][0],
            cro[i][1])
        # Wetland for i in range(0, 2):
            total += Ew * overlappingArea(ac[0], ac[1], wet[i][0], wet[i][1]) area -= overlappingArea(ac[0], ac[1], wet[i]
            [0], wet[i][1])
        # taking off the part that is not in the map for j in range(0, 3):
            if (overlappingArea(ac[0], ac[1], squareconstrains[j][0], squareconstrains[j][1]) > 0):
                return 0
        total += area * Ef
        #check if 4 rectangles touch each other if(overlappingArea(ac[0], ac[1], sa[0], sa[1]) > 0): return 0
        if (overlappingArea(osc[0], osc[1], sa[0], sa[1]) > 0): return 0
        if (overlappingArea(rf[0], rf[1], sa[0], sa[1]) > 0):
            return 0
        if (overlappingArea(rf[0], rf[1], ac[0], ac[1]) > 0): return 0
        if (overlappingArea(rf[0], rf[1], osc[0], osc[1]) > 0): return 0
        if (overlappingArea(osc[0], osc[1], ac[0], ac[1]) > 0):
            return 0 return total
# find non-zero fitness for selection def get_fitness(pred): return pred
# convert binary DNA to decimal and normalize it to a range(0, 5) def translateDNA(pop):
        newpop = [] for i in range(0, POP_SIZE): newpop.append([])
        for i in range(0, POP_SIZE):
            for j in range(0, DNA_SIZE//3, 5):
                string = str(pop[i][j]) + str(pop[i][j+1]) + str(pop[i][j+2]) + str(pop[i][j+3]) + str(pop[i][j+4])
                newpop[i].append(binary_to_decimal(string) / float(2**5-1) * x_bound[1])
            for j in range(DNA_SIZE//3, DNA_SIZE//3 * 2, 5):
                string = str(pop[i][j]) + str(pop[i][j+1]) + str(pop[i][j+2]) + str(pop[i][j+3]) + str(pop[i][j+4])
                newpop[i].append(binary_to_decimal(string) / float(2**5-1) * y_bound[1])
            for j in range(DNA_SIZE//3 * 2, DNA_SIZE, 5):
                string = str(pop[i][j]) + str(pop[i][j+1]) + str(pop[i][j+2]) + str(pop[i][j+3]) + str(pop[i][j+4])
                newpop[i].append(binary_to_decimal(string) / float(2**5-1) * length_bound[1])
        return newpop
def select(pop, fitness): # nature selection wrt pop's fitness if(fitness.sum() == 0):
            pop = np.random.randint(2, size=(POP_SIZE, DNA_SIZE)) return pop
        else:
            IDX = np.random.choice(np.arrange(POP_SIZE), size=POP_SIZE, replace=True, p=fitness/(fitness.sum()))
        return pop[idx]
def crossover(parent, pop): # mating process (genes crossover) if np.random.rand() < CROSS_RATE:
            i_ = np.random.randint(0, POP_SIZE, size=1)          # select another individual from pop
            cross_points = np.random.randint(0, 2, size=DNA_SIZE).astype(bool) # choose crossover points parent[cross_
```

```
          points] = pop[i_, cross_points]       # mating and produce one child
     return parent
def mutate(child):
     for point in range(DNA_SIZE):
          if np.random.rand() < MUTATION_RATE:
               child[point] = 1 if child[point] == 0 else 0
     return child
pop = np.random.randint(2, size=(POP_SIZE, DNA_SIZE)) # initialize the pop DNA new_row = np.array([1 ,0, 1, 1, 0,
0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0,
     0, 0, 0, 1, 0, 1 ,1, 1, 1, 0, 1,1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0 ,0, 0, 0, 0,
     0, 1]) pop[0] = new_row
a = [] b = [] for _ in range(N_GENERATIONS):
     a.append(_)
     print("This is generation ->", _) F_values = [] list2 = translateDNA(pop) for i in range(0, POP_SIZE):
          list1 = list2[i]
          F_values.append(F(list1[0], list1[1], list1[2], list1[3], list1[4], list1[5], list1[6], list1[7],
                    list1[8], list1[9], list1[10], list1[11])) # compute function value by extracting DNA
     F_values = np.asarray(F_values) print(F_values, "F_values") fitness = get_fitness(F_values) b.append(max(fitness))
     print(max(fitness)) print("Most fitted DNA: ", pop[np.argmax(fitness), :]) pop = select(pop, fitness) pop_copy =
     pop.copy() for parent in pop:
          child = crossover(parent, pop_copy) child = mutate(child) parent[:] = child       # parent is replaced by its
          child
plt.scatter(a, b) plt.show()
```

Appendix D Genetic Algorithm DNA Graph Code

```
string = "1 0 1 1 0 0 1 0 0 1 0 0 0 0 1 0 0 1 1 0 0 0 0 0 0 0 0 0 0 1 0 1 1 1 0 0 1 1 0 1 0 1 0 0
     1 0 1 0 1 1 1 0 0 0 1 0 0 0 1 1"
import turtle import random
# setting up of all the turtles, game chart and screen screen = turtle.Screen() screenW = 1280 screenH = 720 pen = turtle.
Turtle() osc_area = 267 * 0.6 / 2.709 rf_area = 129 * 0.6 / 2.709
sa_area = 344 * 0.6 / 2.709 ac_area = 1 * 0.6 / 2.709
DNA_SIZE = 5 * 3 * 4                    # DNA length
POP_SIZE = 400 # population size
CROSS_RATE = 0.8                  # mating probability (DNA crossover)
MUTATION_RATE = 0.003 # mutation probability N_GENERATIONS = 1000 length_bound = [1, 20]      # x upper
and lower bounds x_bound = [0, 20] y_bound = [0, 17.5]
area = [osc_area, rf_area, sa_area, ac_area] def drawers(l1, l2, r1, r2):
     pen.penup() pen.goto(l1, l2) pen.pendown() pen.goto(l1, r2) pen.goto(r1, r2) pen.goto(r1, l2) pen.goto(l1, l2)
newstring = "" for x in range(0, 119, 2):
     newstring += string[x]
def binary_to_decimal(binary_string): decimal = 0 for i in range(len(binary_string)):
          digit = int(binary_string[i]) power = 4 - i decimal += digit * (2 ** power)
     return decimal
def translateDNA(given): newpop = [[],[],[],[]] for j in range(0, DNA_SIZE//3, 5):
          string = str(given[j]) + str(given[j + 1]) + str(given[j + 2]) + str(given[j+3]) + str(given[j+4])
```

```
            newpop[(j) // 5].append(binary_to_decimal(string) / float(2**5-1) * x_bound[1])
        for j in range(DNA_SIZE//3, DNA_SIZE//3 * 2, 5):
            string = str(given[j]) + str(given[j+1]) + str(given[j+2]) + str(given[j+3]) + str(given[j+4])
            newpop[(j-20) // 5].append(binary_to_decimal(string) / float(2**5-1) * y_bound[1])
        for j in range(DNA_SIZE//3 * 2, DNA_SIZE, 5):
            string = str(given[j]) + str(given[j+1]) + str(given[j+2]) + str(given[j+3]) + str(given[j+4])
            newpop[(j-40) // 5].append(binary_to_decimal(string) / float(2**5-1) * length_bound[1])
        return newpop
array = translateDNA(newstring) print(array) x =10 pen.goto(6 * x, 0) pen.goto(0, 8 * x) pen.goto(0, 15 * x) pen.goto(8
* x, 15 * x) pen.goto(8 * x, 17.5 * x) pen.goto(15.5 * x, 17.5 * x) pen.goto(20 * x, 11 * x) pen.goto(20 * x, 0 * x) pen.
goto(6 * x, 0)
y = x for x in range(0, 4):
        drawrec(y * array[x][0], y * array[x][1], y *(array[x][0] + array[x][2]), y*(array[x][1] + area[x] / array[x][2]))
turtle.done()
```

Appendix E Linear Programming Code

```
% Create optimization variables b = optimvar("b",1,8,"LowerBound",0);
% Set initial starting point for the solver initialPoint.b = zeros(size(b));
% Create problem problem = optimproblem("ObjectiveSense","Maximize");
% Define problem objective problem.Objective = 0.05 * (-4.86) * (200 * b(1) +50*b(2)
        -25*b(3)+500*b(4)-20*b(5)-3330*b(6)-2*b(7)-2030*b(8)) + 0.95 * 2 * (13557.69* b(1)
        +382.43*b(2) + 161.35* b(3)+ 509.26*b(4) + 287.37*b(5)+
        21070.95*b(6)+3999.91*b(7)+193.58*b(8));
% Define problem constraints problem.Constraints.constraint1 = sum(b) <= 741; problem.Constraints.constraint2 = 3 *
b(1) +4*b(2) +8*b(3)+9*b(4)+9*b(5)+3*b(6)+6*b(7)+7*b(8) >=
        3000; problem.Constraints.constraint3 = 10 * b(1) +9*b(2) +b(3)+2*b(4)+2*b(5)+b(6)+9*b(7)+2*b(8) >=
        2000; problem.Constraints.constraint4 = 9.5* b(1) +b(2) +7.5 * b(3)+ 8.5*b(4)+8.5*b(5)+2.5 *
b(6)+9*b(7)+7.5*b(8) >= 4500;
problem.Constraints.constraint5 = 9.5* b(1) +8 * b(2) +b(3)+ b(4)+1.5*b(5)+ b(6)+7*b(7)+b(8) >= 1500;
% Display problem information show(problem);
% Solve problem
[solution,objectiveValue,reasonSolverStopped] = solve(problem,initialPoint);
% Display results solution reasonSolverStopped objectiveValue
% Remove Variable clearvars b initialPoint reasonSolverStopped objectiveValue
```

Appendix F TOPSIS Code

```
#include<iostream> #include<cmath> using namespace std; int main(){ double pr; double eco; double array1[8]; double
bd, wd; double array[8][8] = {
        {123,0,0,128,0,487,3,0},
        {0,0,0,0,0,510,147,84},
        {0,0,0,0,54,536,151,0},
        {0,0,0,0,2,511,147,81},
        {0,0,0,0,0,510,147,84} };
          for(int i = 0; i < 5;i ++){ pr = 13557.69 * array[i][0] + 382.43 * array[i][1] + 161.35 * array[i][2]
```

```
                + 509.26 * array[i][3] + 287.37 * array[i][4] + 21070.95 * array[i][5]
                + 3999.91 * array[i][6] + 193.58 * array[i][7]; eco = 200 * array[i][0] + 50 * array[i][1] - 25 * ar-
            ray[i][2]
+ 500* array[i][3] - 20 * array[i][4] - 3330 * array[i][5] - 2 * array[i][6]
                - 2030 * array[i][7]; cout<<pr<<endl<<eco<<endl; bd = sqrt(pow(2 * (12006000 - pr),2) +
            pow(4.86 * (-1869100 - pr),2)); wd = sqrt(pow(2 * (118250 - pr),2) + pow(4.86 * (351000 - pr),2)); ar-
            ray1[i] = wd/(bd + wd);
      }
      for (int i = 0 ; i < 5; i ++){
            cout<<array1[i]<<endl;
      } return 0;
}
```