NSGA-II
Multi-objective Optimization for Bevel Gears

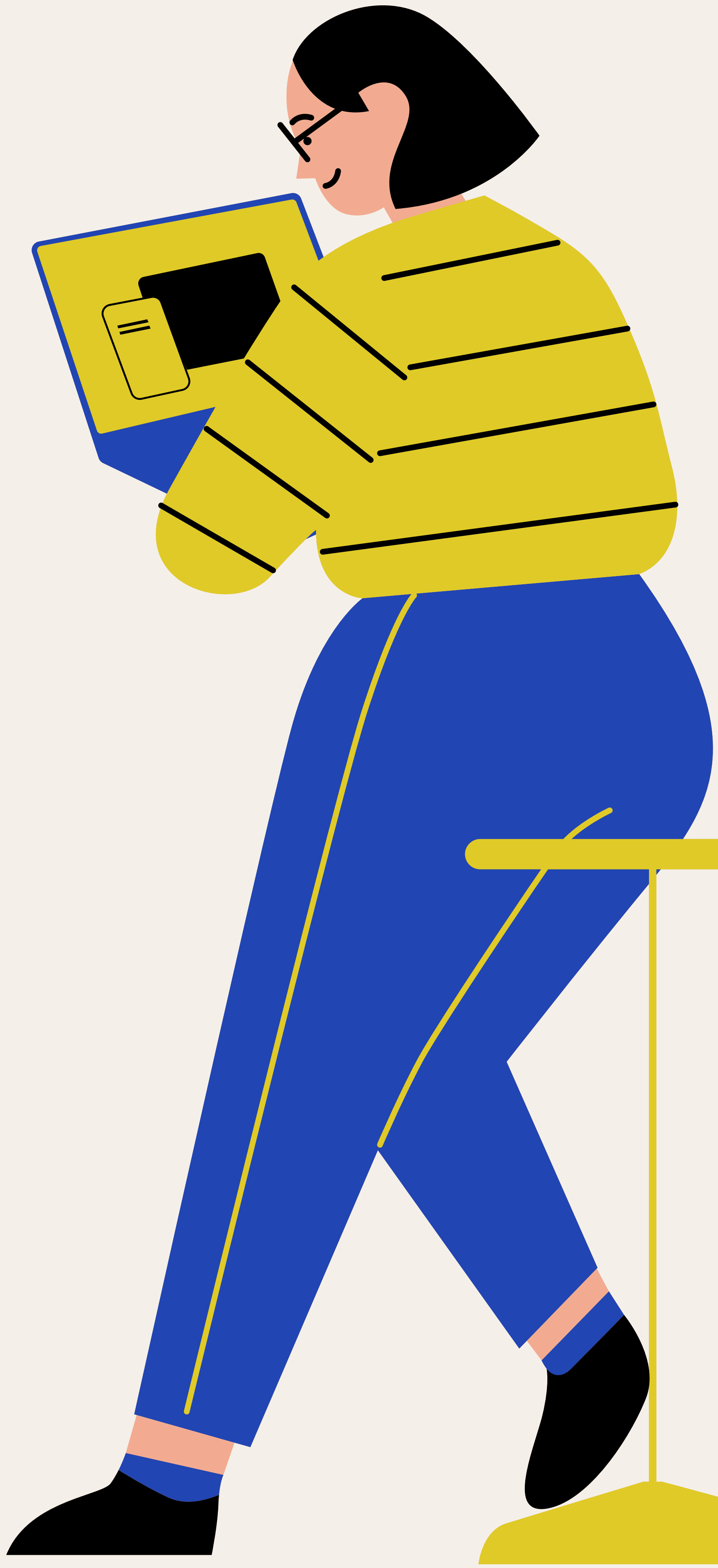
ME 232 KDoM
Group B8

Ayush Singh (22B2203)
Shahu Patil (22B2120)
Rithvik Subash (22B2146)

Project Description

Background Information on the Project

- Objective: Develop a Python application for given objectives optimization on bevel gear design.
- Motivation: Effective mechanical systems rely on well-designed gears.
- Solution: Utilized computational algorithms to automate gear design.
- Benefits: Reduced errors, faster design iterations, and customizable designs.



Objectives and research questions

- To investigate existing algorithms and principles for bevel gear design.
- Determine the impact of gear specifications (module, number of teeth, pressure angle) on gear performance.
- Evaluate the influence of design parameters (shaft angles, gear ratio) on gear efficiency.
- To optimize the **weights**, **efficiency** and the **pitch cone distance** of gear pair.
- Validate the program's user interface for intuitive parameter entry and result interpretation.

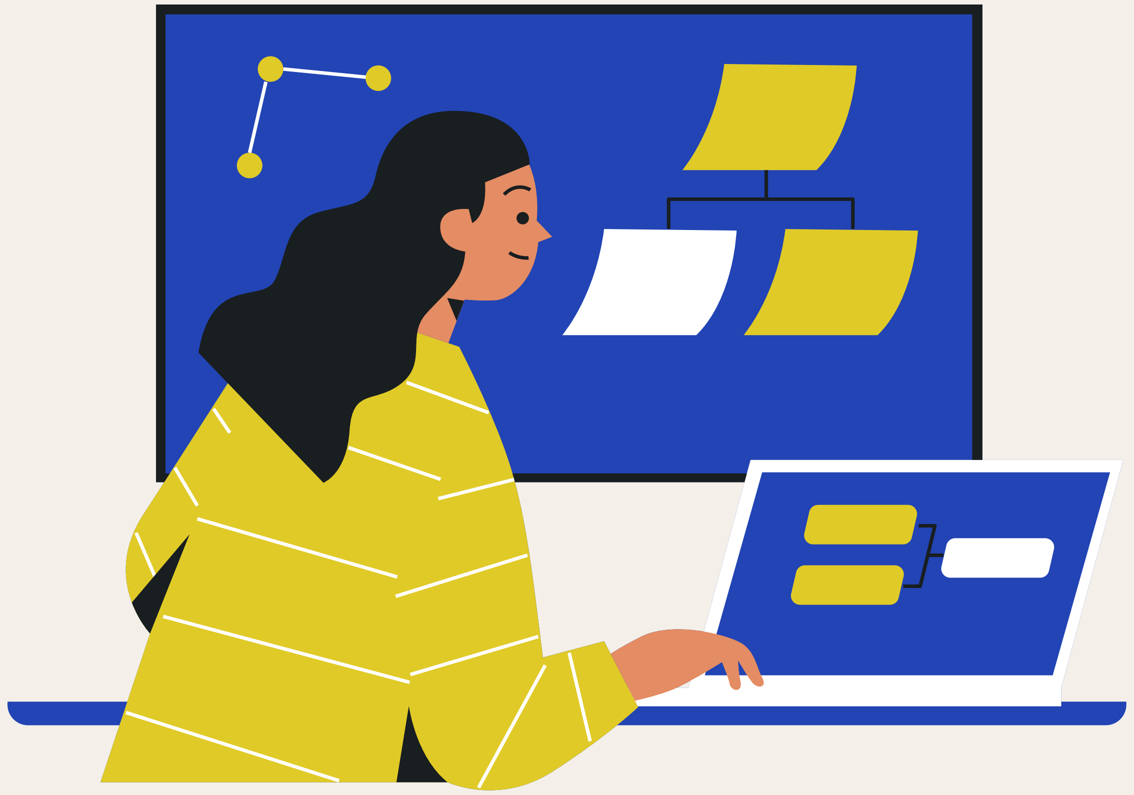
Purpose

- To provide the best optimized parameters for the solutions pairs of the variables.

- Implement the optimization algorithm to find the best set of the pareto frontal solutions.

- To improve the design process by reducing errors and design time compared to manual techniques

Problem Solving Approach



Objective Optimization:

- Objectives: We aimed to optimize three critical parameters of gear pairs: weights, efficiency, and pitch cone distance.

- Significance: Weight reduction enhances overall system efficiency, while maximizing efficiency ensures minimal power loss. Pitch cone distance affects gear meshing, crucial for smooth operation.

Finding the best optimization algorithm is promising

Algorithm and the approach used!

NSGA starts by **initializing a population** of individuals, each representing a potential solution to the optimization problem. These individuals typically **encode candidate solutions as chromosomes or vectors in a search space.**

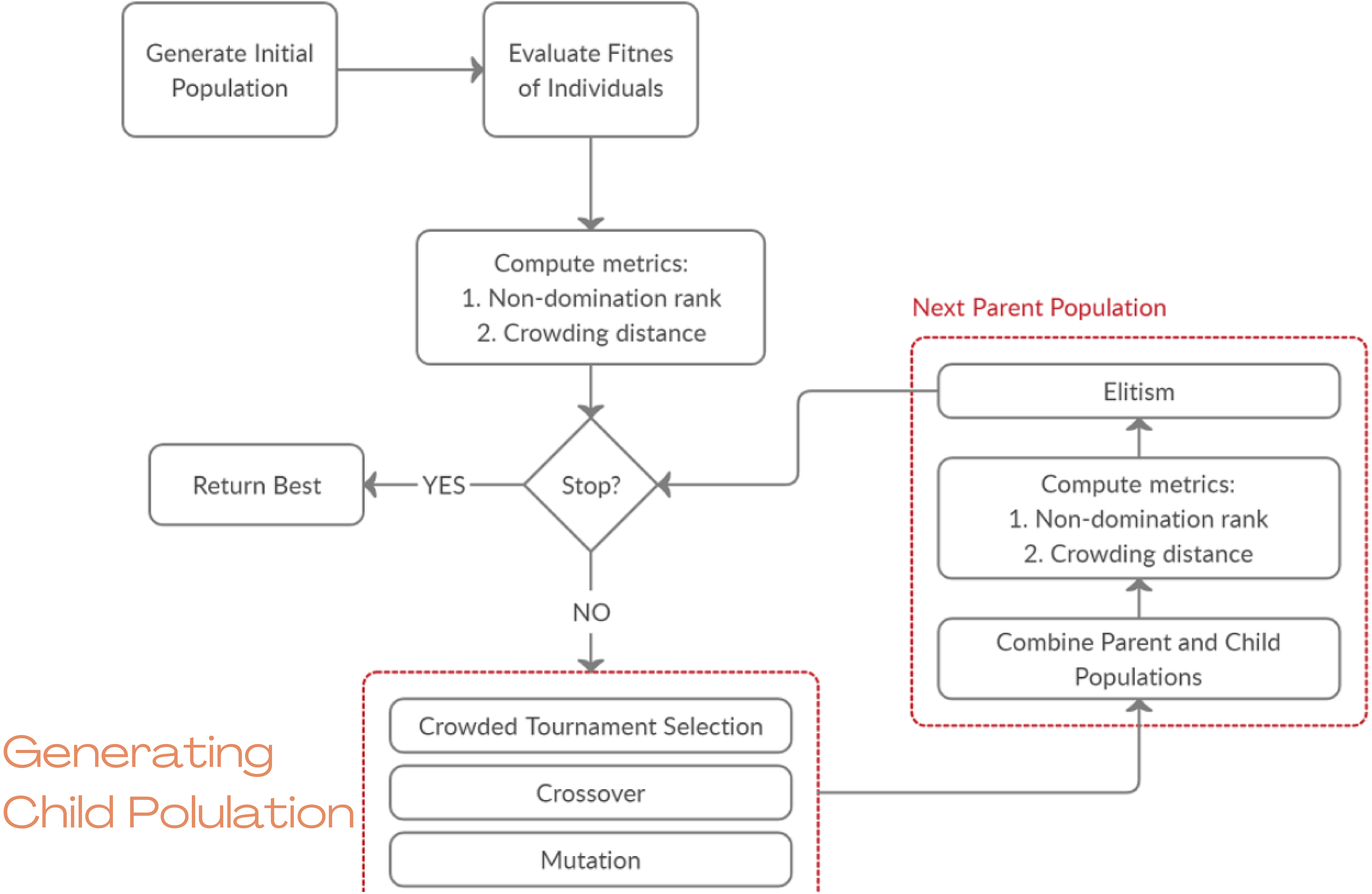
$$\text{Minimize} \quad \left(f_1(x), f_2(x), \dots, f_M(x)\right)^T,$$

$$\text{subject to} \quad g_j(x) \geq 0, \quad j = 1,2,\dots,J$$

$$h_k(x) = 0, \quad k = 1,2,\dots,K,$$

$$x_i^{(L)} \leq x_i \leq x_i^{(U)}, \quad i = 1,2,\dots,n.$$

General optimization problem with the objective functions and the constrains



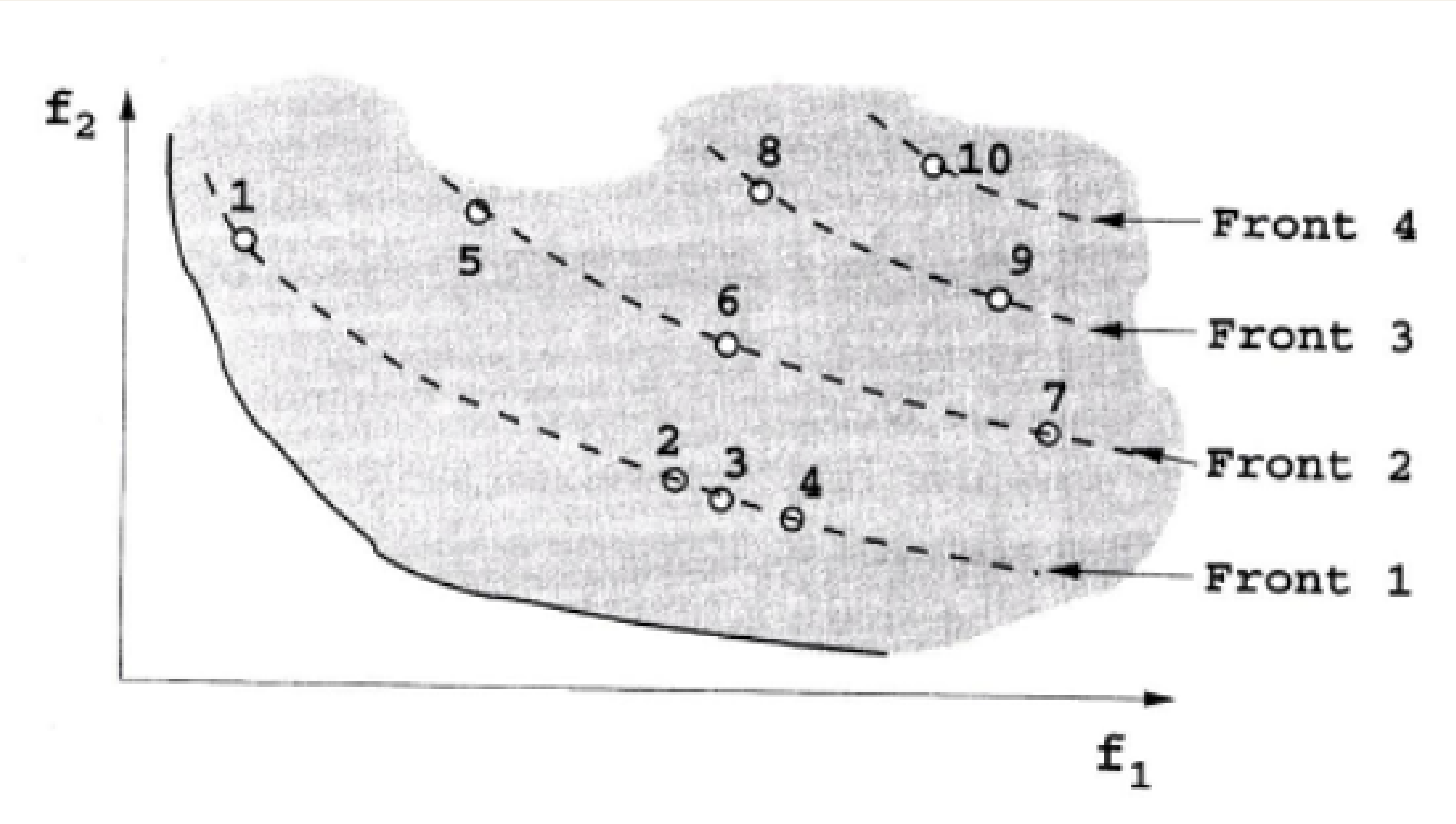
Flow Chart of the NSGA Algorithm

NSGA-II Algorithm

Non dominated genetic sorting algorithm

Non-dominated Sorting:

- Data points of population are sorted into different levels of non-domination based on Pareto dominance.
- Individuals that are not dominated by any other individuals are placed in the first level, forming the Pareto front.

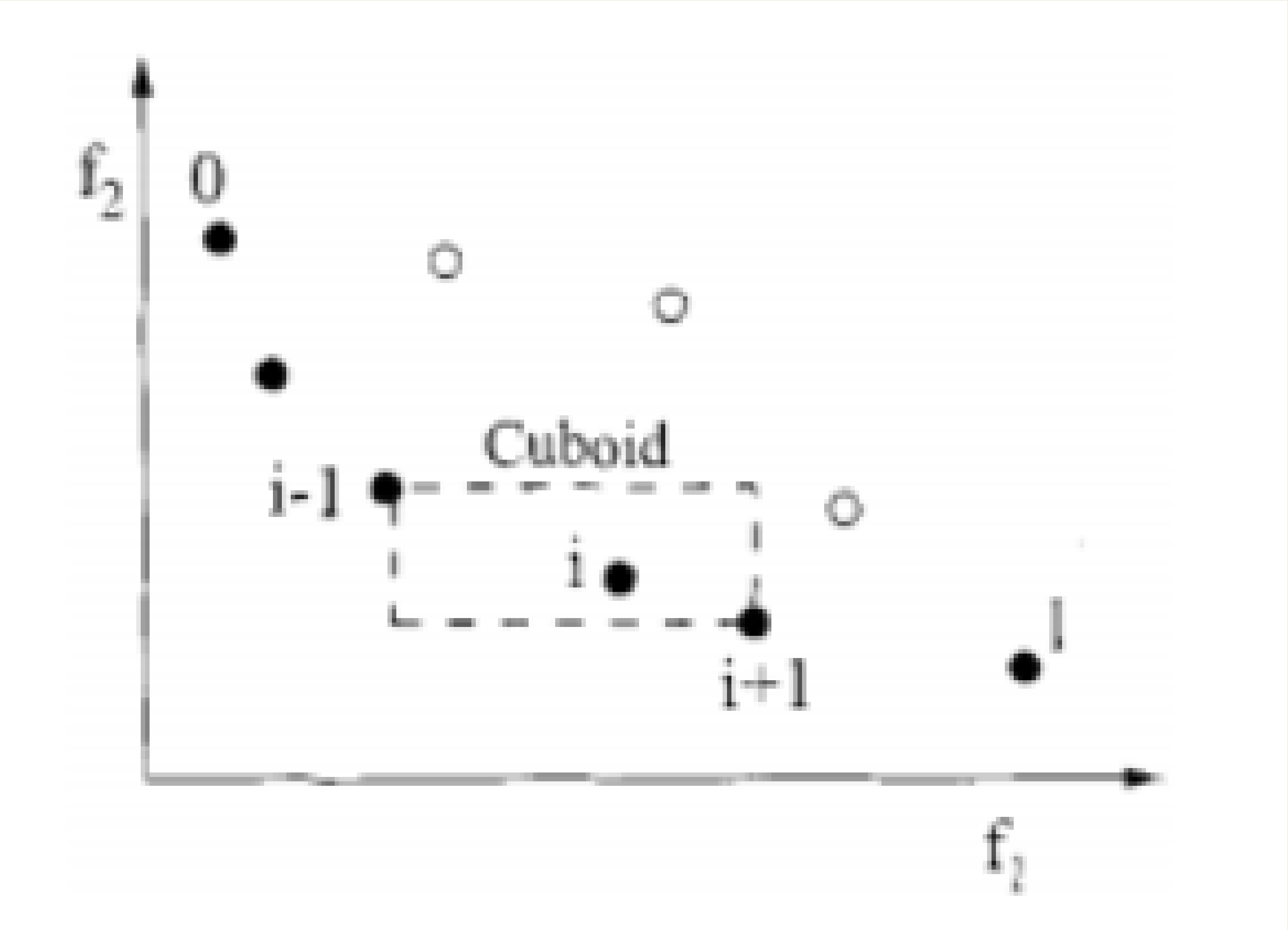


Solution divided in fronts

NSGA-II Algorithm

Non dominated genetic sorting algorithm

After ranking the data points of the population on the basis of the front they are then allotted the crowding distance values.




```
crowding_distance_assignment(I):
  l = |I|                                # number of solutions in I
  for each i, set I[i]_distance = 0      # initialize distance
  for each objective m
    I = sort(I,m)                        # sort using each objective value
    I[1]_distance = I[l]_distance = ∞    # so that boundary points always selected
    for i = 2 to (l − 1)                 # for all other points
      I[i]_distance = I[i]_distance + (I[i+1].m − I[i−1].m)/(f_m^max − f_m^min)
```

$$\mathcal{I}[i]_{distance} = \mathcal{I}[i]_{distance} + \frac{\mathcal{I}[i+1].m - \mathcal{I}[i-1].m}{f_m^{max} - f_m^{min}}$$

Hyper-cuboid of a data point

Crowding Binary Tournament Operator

1. Initialization:
   - We randomly selected two individuals from the combined population of parents and offspring.
2. Comparison:
   - Compared the two selected individuals based on their dominance relationship and crowding distance.
   - If one individual dominated the other it was chosen as the winner.
   - If there is no dominance relationship between the two individuals, the one with the higher crowding distance was selected.
   - If the crowding distances are equal, any of the individuals was chosen randomly.
3. Repetition:
   - Repeated the process until enough individuals are selected to form the next generation.
4. Survivor Selection:
   - The selected individuals become part of the survivor population for the next generation.

Crossover Operator

It helps in the combining genetic information from parent solutions to produce offspring solutions with potentially improved characteristics.

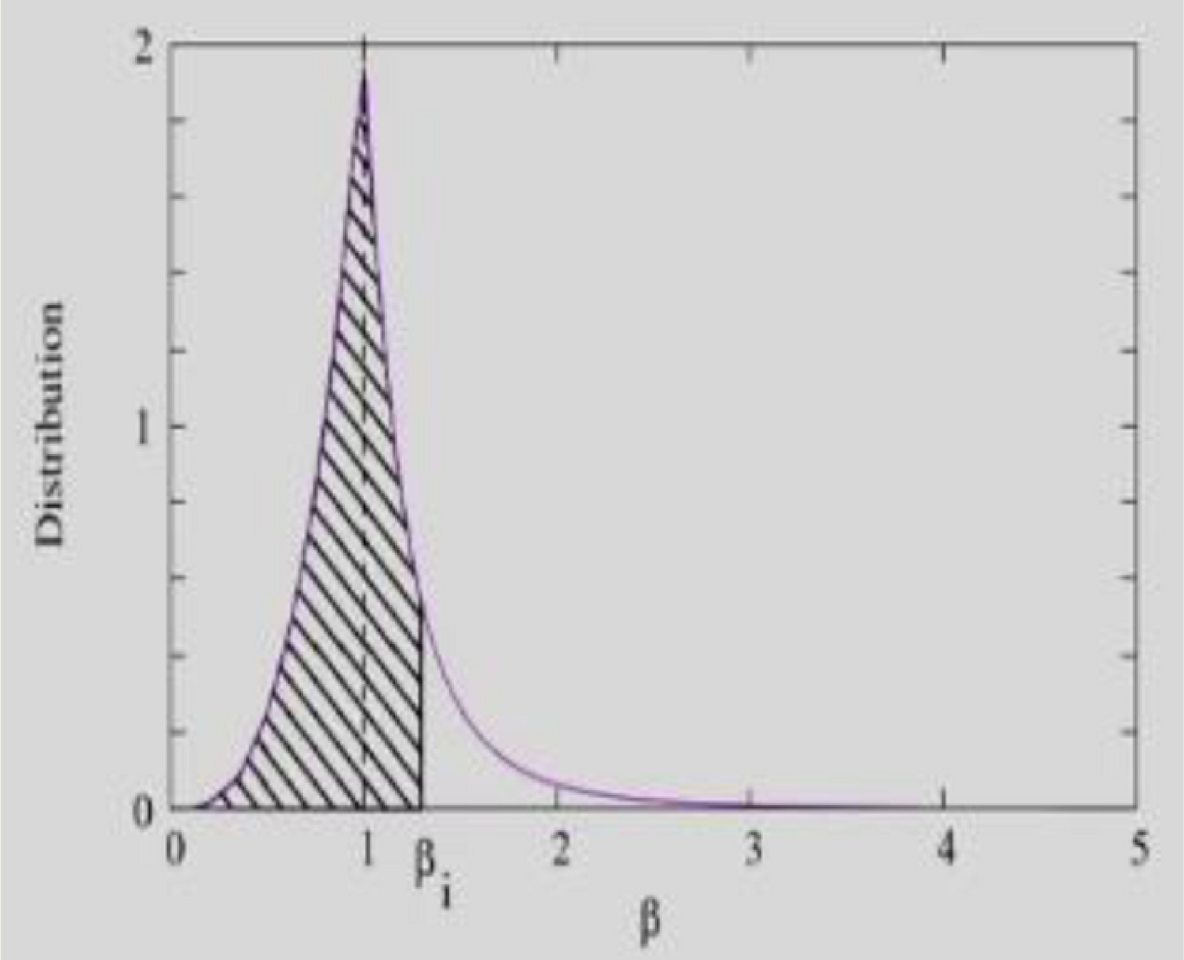
- Selection of Parents based on their fitness or other selection criteria.
- Crossover Point Determination which determines where the genetic material will be exchanged between the parents.
- Crossover Operation
- Offspring Generation.
- Application of Crossover Rate
- Repeat until a sufficient number of offspring solutions are generated to form the next generation of the population.

$$p(\beta_i) = \begin{cases} 0.5(\eta_c + 1)\beta_i^{\eta_c}, & \text{if } \beta_i \leq 1 \\ 0.5(\eta_c + 1)\dfrac{1}{\beta_i^{\eta_c+2}}, & \text{otherwise} \end{cases}$$

$$\beta_i = \begin{cases} (2u_i)^{\frac{1}{\eta_c+1}} & \text{if } u_i \leq 0.5 \\ \left(\dfrac{1}{2(1-u_i)}\right)^{\frac{1}{\eta_c+1}} & \text{otherwise} \end{cases}$$

$$x_i^{(1,t+1)} = 0.5\left[\left(x_i^{(1,t)} + x_i^{(2,t)}\right) - \beta_i\left(x_i^{(2,t)} - x_i^{(1,t)}\right)\right]$$

$$x_i^{(2,t+1)} = 0.5\left[\left(x_i^{(1,t)} + x_i^{(2,t)}\right) + \beta_i\left(x_i^{(2,t)} - x_i^{(1,t)}\right)\right]$$



Mutation Operator

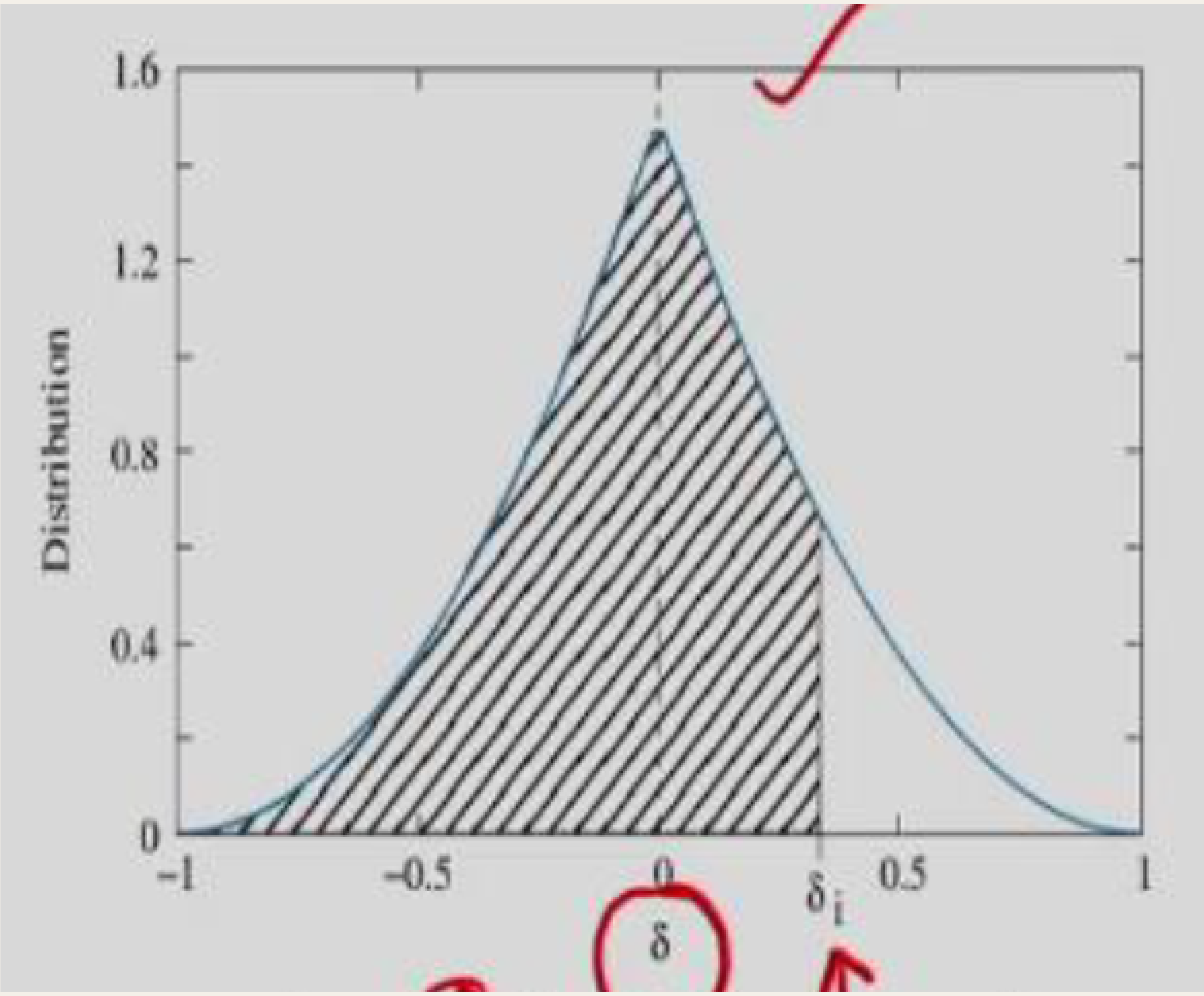
It maintains diversity within the population and prevent premature convergence to suboptimal solutions by exploring new regions of the search space.

- Selection of Parents based on their fitness or other selection criteria.
- Crossover Point Determination which determines where the genetic material will be exchanged between the parents.
- Crossover Operation
- Offspring Generation.
- Application of Crossover Rate
- Repeat until a sufficient number of offspring solutions are generated to form the next generation of the population.

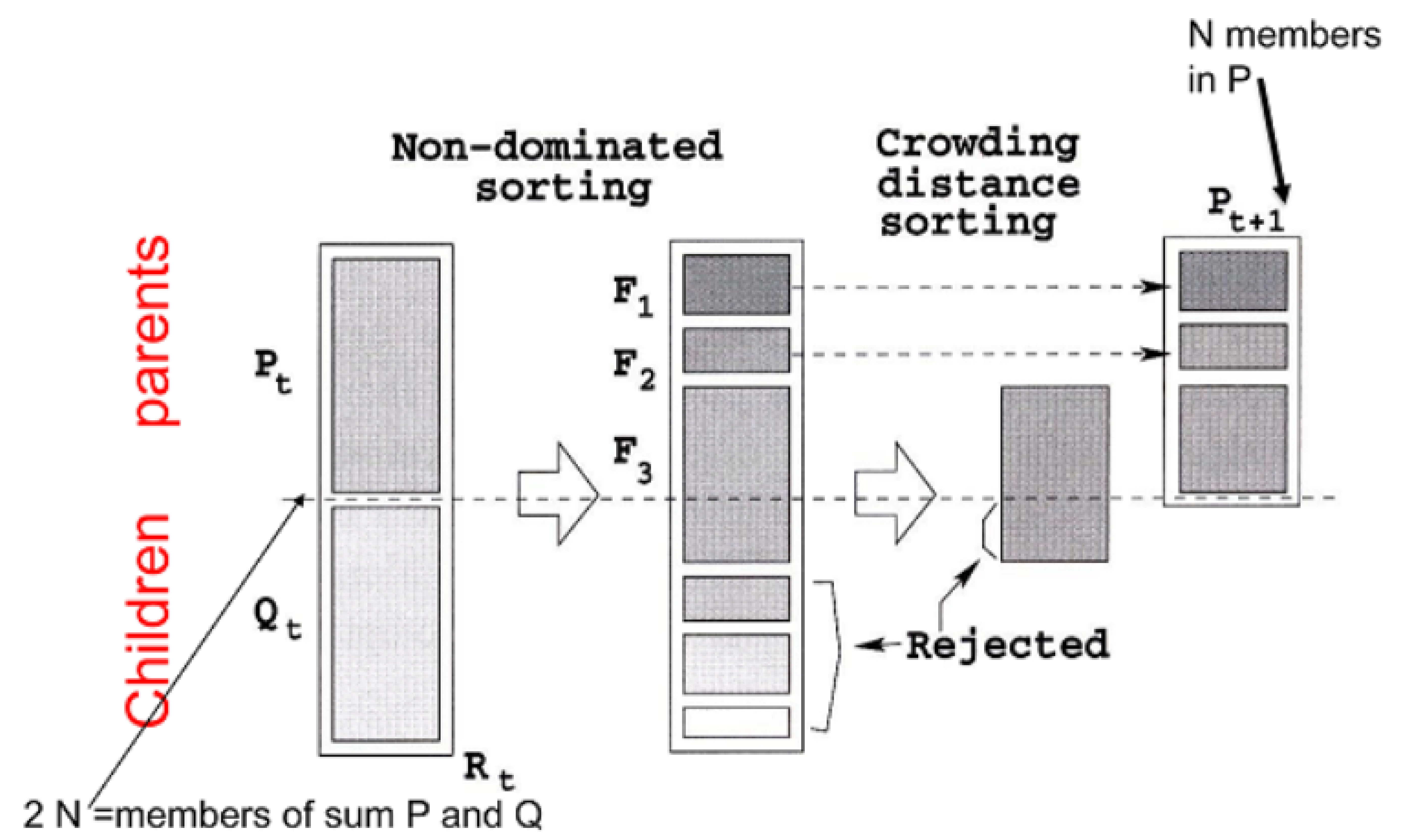
$$y_i^{(1,t+1)} = x_i^{(1,t+1)} + (x_i^{(U)} - x_i^{(L)})\bar{\delta}_\iota$$

$$P(\delta) = 0.5(\eta_m + 1)(1 - |\delta|)^{\eta_m} :$$

$$\bar{\delta}_\iota = \begin{cases} (2r_i)^{\frac{1}{\eta_m+1}} - 1, & if\ r_i < 0.5 \\ 1 - [2(1 - r_i)]^{\frac{1}{\eta_m+1}}, & if\ r_i \geq 0.5 \end{cases}$$



Survival or Elimination



Elitism in NSGA

Implementation

```
def efficiency(mt, z1):
    hs = 5.778 * (math.sqrt(abs(4.778 * z1 / (4.778 * z1 - 1.6888) ** 2 - 0.883)) - 0.342)
    ht = 1.2093 * (math.sqrt(abs(z1 / (z1 - 1.3467) ** 2 - 0.883)) - 0.342)
    power_loss = 3.8781 * ((hs ** 2 + ht ** 2) / (hs + ht))
    return 100 - power_loss

def pitch_cone_distance(mt,z1):
  return 2.44*mt*z1

def check_sigmab(mt,z1):
  sigmab= 974286.17*z1**2/(mt**3*(5.938*z1**2-16.643*z1+11.66)*(0.154*z1-0.603))
  return sigmab<=430

def check_sigmac(mt,z1):
  sigmac= 179401.3/(2.4368*z1*mt**(1.5)-3.415*mt**(1.5))
  return sigmac<=1100

def check_average_modeule_size(mt,z1):

  limit= 6.304/((0.154*z1-0.603)**0.333)
  return mav>=limit

def weight(pinion_teeth, gear_teeth, transverse_module, density):

  pinion_weight = 42.438*density*(np.power(transverse_module,3))*pinion_teeth
  gear_weight = 68.52*density*(np.power(transverse_module,3))*gear_teeth
  total_weight = pinion_weight + gear_weight
  return total_weight
```

Obj 1

Minimisation of weight of the spiral bevel gear pair:

$$f_1 = \text{Total Weight } W = W_1 + W_2$$

Where, $W_1 = \text{Weight of pinion} = 42.438\, \rho m_t^3 z_1$

$$W_2 = \text{Weight of gear} = 68.52\, \rho\, m_t^3 z_2$$

Obj 2

Maximisation of efficiency of gear pair:

$$f_2 = \text{Efficiency}\,\eta = 100 - P_L$$

Where P_L = Power Loss, which is given by the following equation:

$$50\, f \left\{ \frac{\cos + \cos}{\cos \phi_n} \right\} \cos^2 \beta \, \frac{(H_s^2 + H_t^2)}{(H_s + H_t)}$$

To calculate H_s and H_t, following equations (8) and (9) are used

$$H_S = (i+1)\left\{ \left[\sqrt{\left(\frac{R_o}{R}\right)^2 - \cos^2 \varnothing_n} \right] - \sin \varnothing_n \right\}$$

$$H_t = \left(\frac{i+1}{i}\right)\left\{ \left[\sqrt{\left(\frac{r_o}{r}\right)^2 - \cos^2 \varnothing_n} \right] - \sin \varnothing_n \right\}$$

$$R_o = R + \text{one addendum}$$

One addendum for 20° full depth involute system = *One average Module* $= m_{av}$

Where,

$$m_{av} = m_t \left(\frac{\Psi_y - 0.5}{\Psi_y} \right)$$

Obj 3

Minimisation of pitch cone distance of gear pair:
Eqn. 10 represents this objective function.

$$f3 = Rc = 0.5 m_t z_1 \sqrt{i^2 + 1}$$

Constraint Equations

Here we have the four inequality constraints and the one equality constraints.

These constraints are essential in order to prevent the material failure.

```
def bending_stress(transverse_module, twisting_moment, face_width, density, youngs_modulus):
    R_b = face_width / 2  # Radius of gear
    R_i = R_b - 1.7 * transverse_module  # Inside radius of gear
    M = twisting_moment  # Twisting moment
    b = face_width
    n_v = 1  # Dynamic factor (assume 1 for simplicity)

    # Calculating bending stress
    sigma_b = ((0.7 * 1 * (R_b / R_i + 1)) + (0.5 * transverse_module * b * n_v * R_i / M)) * math.sqrt(R_b / R_i)

    # Checking against allowable bending stress
    allowable_bending_stress = density * youngs_modulus / safety_factor  # Adjust safety_factor as needed
    return sigma_b, sigma_b <= allowable_bending_stress

def crushing_stress(transverse_module, twisting_moment, density, youngs_modulus):
    R_b = face_width / 2  # Radius of gear
    R_i = R_b - 1.7 * transverse_module  # Inside radius of gear
    M = twisting_moment  # Twisting moment

    # Calculating crushing stress
    sigma_c = (0.72 * (transverse_module ** 1.5) * R_b) / (2 * M * R_i)

    # Checking against allowable crushing stress
    allowable_crushing_stress = density * youngs_modulus / safety_factor  # safety_factor can be adjusted as needed
    return sigma_c, sigma_c <= allowable_crushing_stress
```

(a) Bending stress

$$\sigma_b \le [\sigma_b]$$

$$\sigma_b = \left(\frac{0.7R\sqrt{(i^2+1)}[M_t]}{(R-0.5b)^2 bm_n y_v} \right)$$

(b) Crushing stress

$$\sigma_c \le [\sigma_c]$$

$$\sigma_c = \frac{0.72}{(R-0.5b)} \sqrt{\frac{(i^2 \pm 1)^3}{ib} E\, [M_t]}$$

(c) Cone distance

$$R_{\min} \le R$$

$$\frac{41.4885}{(0.357Z_1 - 0.5)^{\frac{2}{3}}} \le R$$

(d) Average Module

$$m_{av} \ge 1.15 cos\beta_{av} \sqrt[3]{\frac{[M_t]}{y_{v[\sigma_b]\psi_m Z_1}}}$$

Additionally gear ratio is kept constant.

(e) Gear ratio

$$i = 4.778$$

Result and Discussions

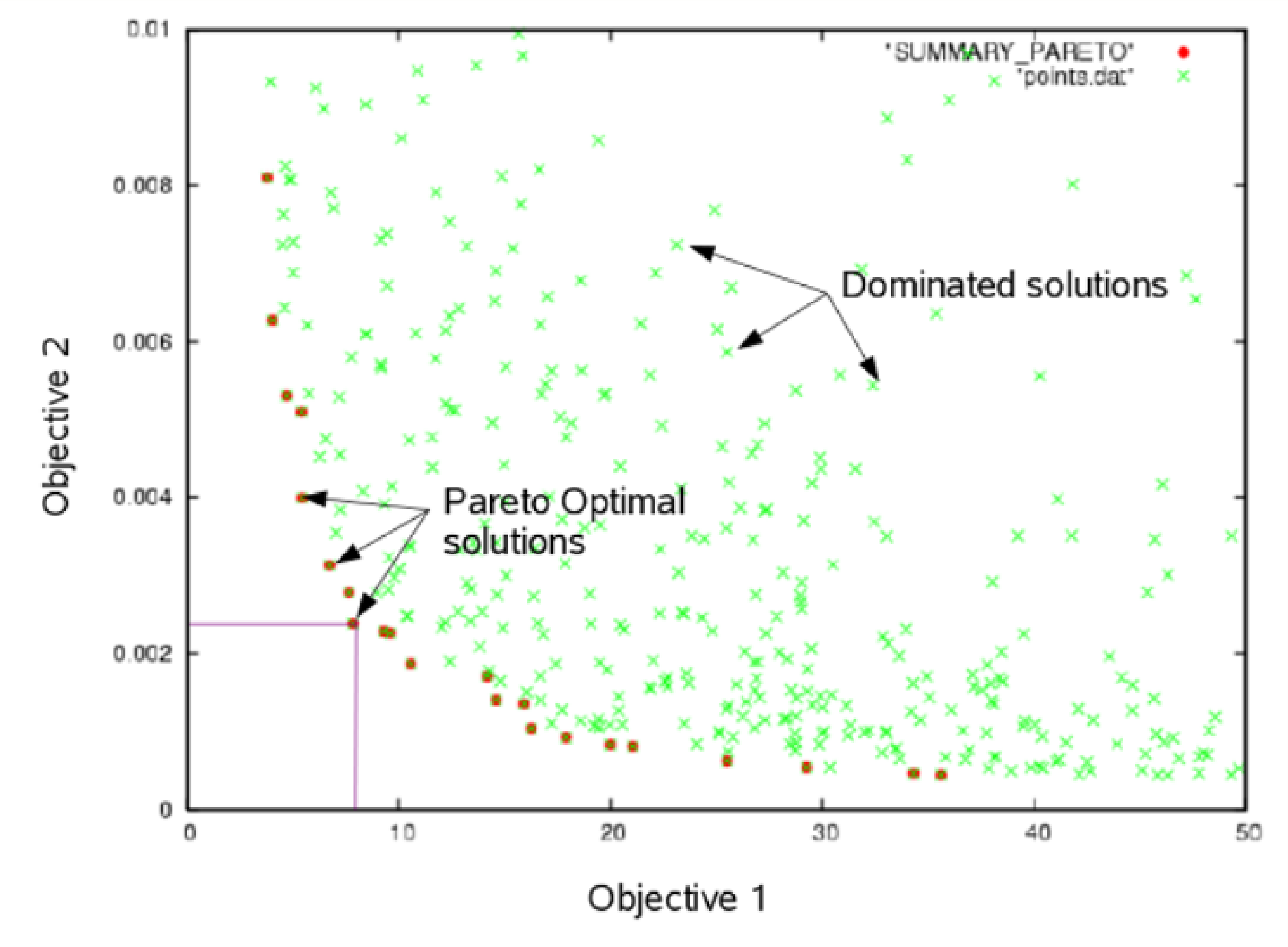
After running the NSGA algorithm for the given constraints and the given material properties, we get the pareto frontal solutions.

```
def cone_distance(pinion_teeth, gear_teeth, transverse_module):
    Z = pinion_teeth
    R = gear_teeth
    R_min = min(R)
    return 41.4885 * (R - 0.5 * Z) / (R - 1)

def average_module(face_width, pinion_teeth, gear_teeth):
    Z = pinion_teeth
    R = gear_teeth
    m = face_width / 8
    return m >= 1.15 * (Z + R) / (Z * R)

def gear_ratio():
    return 4.778  # Constant value

safety_factor = 1.5  # Can be adjusted as needed
```



Trends for the Titanium metal

Weight: Lower weight.
Steady decrease as the gear dimensions increase.

Efficiency: High mechanical efficiency.
High values across different gear sizes.

Bending Stress (σ_b): High bending stresses.

Crushing Stress (σ_c): High crushing stress values.
Increasing values with larger gear dimensions

Cone Distance: Consistent trend across different gear sizes.

Average Module: Consistent trend with gear dimensions.

Conclusion

Summary of the Present State of the Project:

- We have achieved significant progress in developing an automated bevel gear design application.
- Key functionalities, such as user input handling, algorithm execution, and output generation, have been successfully implemented.
- Initial testing has demonstrated the feasibility and effectiveness of the automated design process.

Limitations/Challenges Faced:

- We were not able to find the derivation of the conventional equations of the objective functions which we took reference from the research paper.
- Resource constraints, including limited computational resources and development time, impacted the pace of progress and testing cycles

Special Comments Based on Our Experience:

- This project has provided valuable insights into the intricacies of gear design and the complexities of developing automated design tools.
- This project has also highlighted the importance of adaptability and flexibility in the face of evolving requirements and constraints.
- We learned to embrace change and iterate on our designs based on feedback and new insights gained during the development process.

Thanks

Ayush Singh (22B2203)
Shahu Patil (22B2120)
Rithvik Subash (22B2146)