Module 5: Multi Objective Optimization & Hybrid Systems

- Multi-objective optimization problem.
- Principles of Multi-objective optimization,
- Dominance and Pareto-optimality.
- Optimality conditions.
- Neuro-fuzzy hybrid systems.
- Genetic – neuro hybrid systems.

Module 5: Multi Objective Optimization & Hybrid Systems

- MOOP-Linear &Nonlinear, Convex & Non-Convex
- Principles of MOO-Illustrating Pareto Optimal Solutions
- Objectives in MOO
- Dominance & Pareto-Optimality
  - Concept of Domination
  - Properties of Dominance Relation, Pareto Optimality
  - Procedure for finding a non-dominated set
- Optimality Conditions
- Neuro Fuzzy hybrid system-Classification& Characteristics
- Genetic –neuro hybrid systems

Multi-objective optimization problem.

5.1 Multi-objective optimization problem

- Most real-world problems involve the simultaneous optimization of several objective functions.
- Generally, these objective functions are measured in different units, often competing and conflicting.
- MOOP deals with more than one objective function that is to be minimized or maximized
- The MOO or multi-objective optimization refers to finding the optimal solution values of more than one desired goal.
- These optimal solutions are known as Pareto-optimal solutions.

Multi-objective optimization problem

- MOOP Involves more than one objective function that is to be minimized or maximized
- Multi-objective optimization having such conflicting objective functions gives rise to a set of optimal solutions, instead of one optimal solution, because no solution can be considered to be better than any other with respect to all objectives.
- Minimizing cost while maximizing comfort while buying a car, and maximizing performance at the same time as minimizing fuel consumption and emission of pollutants of a vehicle are examples of multi-objective optimization problems involving two and three objectives, respectively.

Multi-objective optimization problem

- Multi-objective optimization problems with a number of objectives and a number of equality and inequality constraints can be **formulated** as:

  Minimize / Maximize $\quad f_i(x),\ i = 1, \ldots, N_{obj}$

  Subject to: $\quad g_k(x) = 0,\ k = 1, \ldots, K \quad h_l(x) \le 0,\ l = 1, \ldots, L$

- Where,
  - $f_i(x)$ is the objective function,
  - x is a decision vector that represents a solution
  - N_{obj} is the number of objectives, and
  - K and L are number of equality and inequality constraints respectively.

MOOP Mathematical form

- Multi-objective optimization problems with a number of objectives and a number of equality and inequality constraints can be **formulated** as:

Mathematically

$$
\begin{aligned}
&\text{min/max } f_m(\mathbf{x}), && m = 1, 2, \ldots, M \\
&\text{subject to } g_j(\mathbf{x}) \ge 0, && j = 1, 2, \ldots, J \\
&h_k(\mathbf{x}) = 0, && k = 1, 2, \ldots, K \\
&\underbrace{x_i^{(L)}}_{\text{lower bound}} \le x_i \le \underbrace{x_i^{(U)}}_{\text{upper bound}}, && i = 1, 2, \ldots, n
\end{aligned}
$$

- A solution x is a vector of n decision variables $x = \{x_1, x_2, x_3, \ldots x_n\}$
- Associated with the problem are J inequality and K equality constraints.
- The term $g_j(x)$ and $h_k(x)$ are called constraint functions.
- If any solution x satisfies all constraints and variable bounds, it is known as a feasible solution. The set of all feasible solutions is called the feasible region, or S

Multi-objective optimization problem

There are three components in any optimization problem:

F: Objectives

minimize (maximize) $f_i(x_1, x_2, \cdots, x_n)$, $i = 1, 2, \cdots, m$

S: Constraints

Subject to

$g_j(x_1, x_2, \cdots, x_n)$, $ROP_j\ C_j$, $j = 1, 2, \cdots, l$

V: Design variables

$x_k\ ROP_k\ d_k$, $k = 1, 2, \cdots, n$

Note :

1. For a multi-objective optimization problem (MOOP), $m \geq 2$
2. Objective functions can be either minimization, maximization or both.

formal specification of MOOP

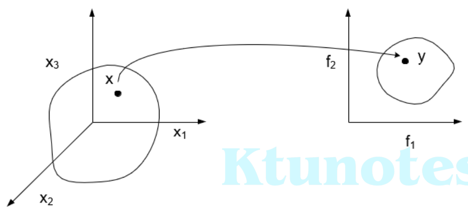
Let us consider, without loss of generality, a multi-objective optimization problem with n decision variables and m objective functions

Minimize $y = f(x) = [y_1 \in f_1(x), y_2 \in f_2(x), \cdots, y_k \in f_m(x)]$

where
$$x = [x_1, x_2, \cdots, x_n] \in X$$
$$y = [y_1, y_2, \cdots, y_n] \in Y$$

Here : x is called the decision vector
y is called an objective vector
X is called a decision space
Y is called an objective space

Illustration: Decision space and objective space



Decision Space X

Objective Space Y

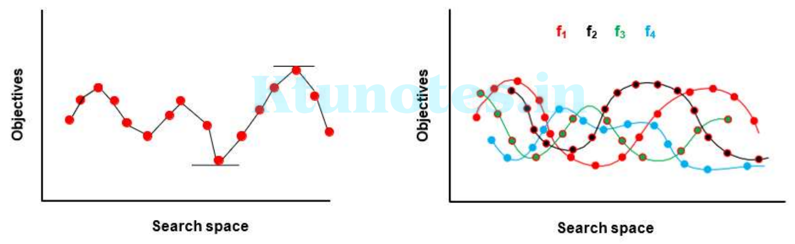
- For each solution x in the decision variable space, there exists a point in the objective space(Z) denoted by $f(x)=z=\{z_1,z_2,\ldots,z_m\}$
- The mapping takes place between an n-dimensional decision vector(x) and M dimensional objective vector(y).

Thus, solving a MOOP implies to search for x in the decision space (X) for an optimum vector (y) in the objective space (Y).

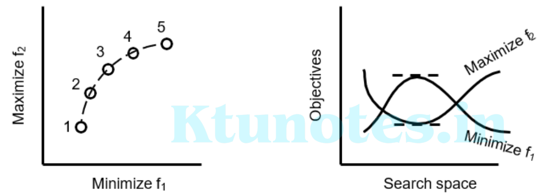
Illustration: Decision space and objective space

- Optimization problems with a number of **objective functions** to be satisfied.
- The objective functions may be **conflicting** with one another.
- In order to simplify the solution process, additional objective functions are usually handled as **constraints**.
- **Multiple objective functions** are handled at the same time.

Illustration: Single vs. multiple objectives



MOOP: Trade-off and conflicts in solutions



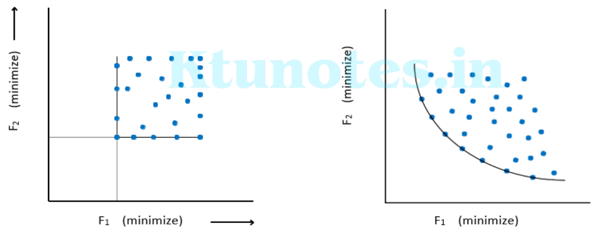
Trade-off optimal solution

Conflicting objectives

MOOP: Illustration: ideal solution vs. real solution

It is observed that in many real-life problems, we hardly have a situation in which all the $f_i(\bar{x})$ have a minimum in $\bar{X}$ at a common point $\bar{x}^*$.

This is particularly true when objective functions are conflicting in their interests.



Ideal Situation

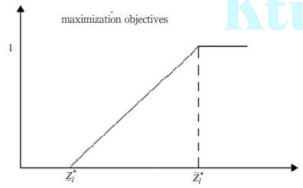
Real Situation

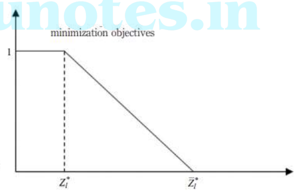
5.1.1 Linear and Non linear MOOP

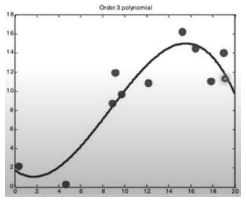
- If all the objective functions and constraints are linear in the MOOP, then the problem is defined as a Multi-Objective Linear Problem(MOLP)
- If any of the objective functions or constraints are nonlinear , then the resulting problem is called a nonlinear Multi-Objective optimization Problem(nonlinear MOOP)
- Like linear programming problems, MOLP has many theoretical properties
- For nonlinear problems, the solution techniques often do not have convergence proofs.
- Since most of the real-world Multi-Objective optimization Problems are nonlinear in nature, we do not assume any particular structure of the objective and constraint functions here.

5.1.1 Linear and Non linear MOOP

- If all the objective functions and constraints are linear in the MOOP, then the problem is defined as a Multi-Objective Linear Problem(MOLP)
- If any of the objective functions or constraints are nonlinear , then the resulting problem is called a nonlinear Multi-Objective optimization Problem(nonlinear MOOP)







5.1.1 Linear and Non linear MOOP

- An optimization problem is nonlinear if the objective function $f(x)$ or any of the inequality constraints $c_i(x) \le 0$, $i = 1, 2, \ldots, m$, or equality constraints $d_j(x) = 0$, $j = 1, 2, \ldots, n$, are nonlinear functions of the vector of variables x.
- For example, if x contains the components x_1 and x_2, then the function $3 + 2x_1 - 7x_2$ is linear, whereas the functions $(x_1)^3 + 2x_2$ and $3x_1 + 2x_1x_2 + x_2$ are nonlinear.
- Nonlinear problems arise when the objective or constraints cannot be expressed as linear functions without sacrificing some essential nonlinear feature of the real-world system

5.1.2 Convex and Nonconvex MOOP

A MOOP is convex if all the objective functions and feasible region are convex.

A function $f : \mathbb{R}^n \to \mathbb{R}$ *is a convex function if for any two pair of solutions* $\mathbf{x}^{(1)}, \mathbf{x}^{(2)} \in \mathbb{R}^n$*, the following condition is true:*

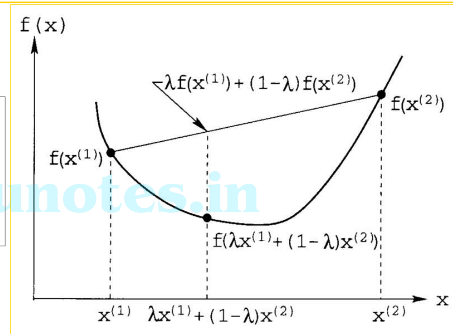
$$f\left(\lambda \mathbf{x}^{(1)} + (1-\lambda)\mathbf{x}^{(2)}\right) \le \lambda f(\mathbf{x}^{(1)}) + (1-\lambda) f(\mathbf{x}^{(2)}), \quad \textit{for all } 0 \le \lambda \le 1.$$

properties of a convex function:

1. The linear approximation of $f(\mathbf{x})$ at any point in the interval $[\mathbf{x}^{(1)}, \mathbf{x}^{(2)}]$ always *underestimates* the actual function value.
2. The Hessian matrix of $f(\mathbf{x})$ is positive definite for all $\mathbf{x}$.
3. For a convex function, a local minimum is always a global minimum.[1]

5.1.2 Convex and Nonconvex MOOP

$$f\left(\lambda \mathbf{x}^{(1)} + (1-\lambda)\mathbf{x}^{(2)}\right) \le \lambda f(\mathbf{x}^{(1)}) + (1-\lambda) f(\mathbf{x}^{(2)}), \quad \textit{for all } 0 \le \lambda \le 1.$$



A convex function is illustrated. A line joining function values at two points $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$ always estimates a large value of the true convex function.

5.1.2 Convex and Nonconvex MOOP [another definition]

A MOOP is convex if all the objective functions and feasible region are convex.

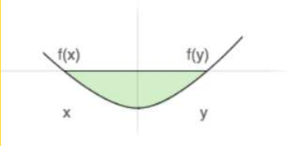
Definition Let $\mathbf{x} \in \mathcal{X}$ and $\mathbf{y} \in \mathcal{X}$.

Let $0 \leq \lambda \leq 1$. A function $f : \mathbb{R}^n \to \mathbb{R}$ is **convex** over $\mathcal{X}$ if

$$f(\lambda \mathbf{x} + (1-\lambda)\mathbf{y}) \leq \lambda f(\mathbf{x}) + (1-\lambda) f(\mathbf{y}).$$

The function is called strictly convex if "$\leq$" is replaced by "$<$".

Geometrically, a function is convex if a line segment drawn from any point (x, f(x)) to another point (y, f(y)) -- called the chord from x to y -- lies on or above the graph of f, as in the picture

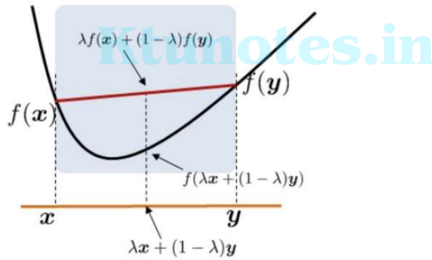


5.1.2 Convex and Nonconvex MOOP

Let $\mathbf{x} \in \mathcal{X}$ and $\mathbf{y} \in \mathcal{X}$. Let $0 \leq \lambda \leq 1$.

A function $f : \mathbb{R}^n \to \mathbb{R}$ is **convex** over $\mathcal{X}$ if

$f(\lambda \mathbf{x} + (1-\lambda)\mathbf{y}) \leq \lambda f(\mathbf{x}) + (1-\lambda) f(\mathbf{y})$.



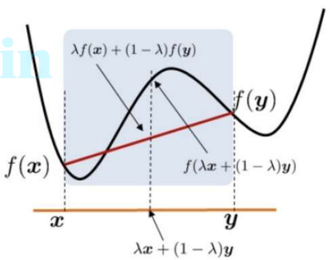
5.1.2 Convex and Nonconvex MOOP

Let $\mathbf{x} \in \mathcal{X}$ and $\mathbf{y} \in \mathcal{X}$. Let $0 \leq \lambda \leq 1$.

A function $f : \mathbb{R}^n \to \mathbb{R}$ is Non-Convex over $\mathcal{X}$ if

$f(\lambda \mathbf{x} + (1-\lambda)\mathbf{y}) > \lambda f(\mathbf{x}) + (1-\lambda) f(\mathbf{y})$.

A non-convex function "curves up and down" -- it is neither convex nor concave.
A familiar example is the sine function.
If a function g(x) is nonconvex , the set of solutions satisfying g(x)>=0



Principles of Multi-objective optimization,

Principles of Multi-objective optimization,

- Real-world problems have **more than one objective function**, each of which may have a different individual optimal solution.
- Different in the optimal solutions corresponding to different objectives because **the objective functions are often conflicting** (competing) with each other.
- Set of trade-off optimal solutions instead of one optimal solution, generally known as "***Pareto-Optimal***" solutions (named after Italian economist Vilfredo Pareto (1906)).
- No one solution can be considered to be better than any other with respect to all objective functions. The **non-dominant solution** concept.

Principles of Multi-objective optimization

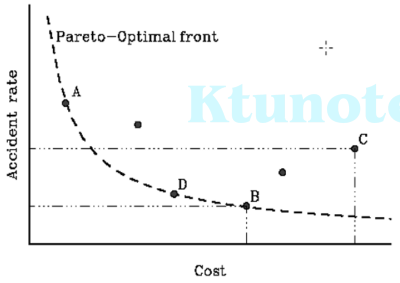
- Is the optimization of different objective functions at the same time, thus at the end the algorithm return *n* different optimal values which is different to return one value in a normal optimization problem.
- Thus, there is more than one objective function

Pareto - optimal solutions and **Pareto - optimal front**

- Pareto - optimal solutions: The optimal solutions found in a multiple-objective optimization problem
- Pareto - optimal front: the curve formed by joining all these solutions (Pareto - optimal solutions)

Principles of Multi-objective optimization

- Simple car design example: two objectives - cost and accident rate – both of which are to be minimized.



A, B, D - One objective can only be improved at the expense of at least one other objective!

Principles of Multi-objective optimization

- point A represents a solution that incurs a near-minimum cost but is highly accident-prone (predisposed).
- On the other hand, point B represents a solution that is costly but is near the least accident-prone.
- **One cannot really say whether solution A is better than solution B** or vice versa because one solution is better than the other in one objective but is worse in the other.
- **solution C is not optimal because there exists another solution D in the search space, which is better than solution C in both objectives**
- One cannot conclude about an absolute hierarchy of solutions **A, B, D**, or any other solution in the set.

Principles of Multi-objective optimization

- These solutions are known as **Pareto-Optimal solutions** (named after Italian economist Vilfredo Pareto (1906))
- The set of the best compromise solutions is referred to as the **Pareto-ideal set, characterized by the fact** that starting from a solution within the set, **one objective can only be improved at the expense of at least one other objective.**
- In front of the Pareto-Optimal front are un-attainable solutions corresponding to the optimal of both objectives.
- **The area behind the Pareto-Optimal front is known as feasible search space or feasible design space.**

Objectives of Multi-objective optimization

- A multi-objective optimization algorithm must achieve:
  1. *Guide the search towards the global Pareto-Optimal front.*
  2. *Maintain solution diversity in the Pareto-Optimal front.*
- The first goal is mandatory in any optimization task.
- When solutions converges close to the true optimal solutions, that one can be assured of their near optimality properties.
- The second goal is entirely specific to MOOP.
- since MOEA deals with two spaces-decision variable space and objective space-'diversity' among solutions can be defined in both of these spaces.

Objectives of Multi-objective optimization

- For example, two solutions' are diverse, in the decision variable space, if their Euclidean distance(length of a line segment between the two points) in the decision variable space is large.
- Similarly, two solutions are diverse in the objective space, if their Euclidean distance in the objective space is large.
- Diversity in one space usually means diversity in the other space, this may not be so, in all problems.
- In such complex and nonlinear problems, it is then the task to find a set of solutions having a good diversity in the desired space.

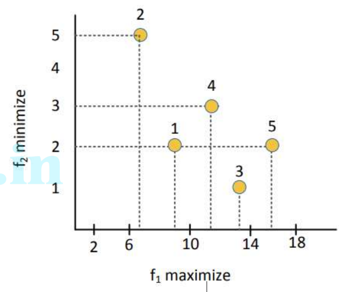
Dominance and Pareto-optimality

Dominance and Pareto-optimality

- Most MOOP uses the concept of dominance in their search.
- Concept of Dominance
- Properties of Dominance Relation
- Pareto Optimality

Dominance

- In the single-objective optimization problem, the superiority of a solution over other solutions is easily determined by comparing their objective function values
- In a multi-objective optimization problem, the goodness of a solution is determined by the **dominance**



Here, 1 dominates 2, 5 dominates 1 etc

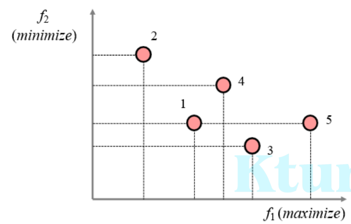
Dominance

- In a multi-objective optimization problem, the goodness of a solution is determined by the **dominance**

Dominance Test

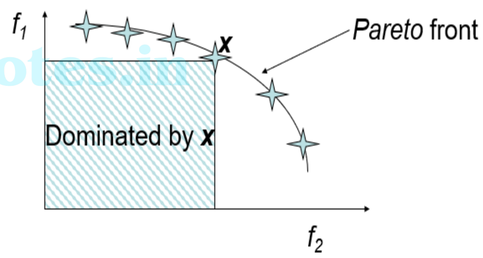
- x_1 dominates x_2, if
  - Solution x_1 is no worse than x_2 in all objectives
  - Solution x_1 is strictly better than x_2 in at least one objective
- x_1 dominates x_2 $\Longleftrightarrow$ x_2 is dominated by x_1

Example of Dominance



- 1 Vs 2: 1 dominates 2 [Minimize]
- 1 Vs 5: 5 dominates 1 [Maximize]
- 1 Vs 4: Neither solution dominates

we say *x* dominates *y* if it is at least as good on all criteria and ***better*** on at least one



Properties of dominance relation

- This dominance relation satisfies four binary relation properties.

Reflexive :

The dominance relation is **NOT** reflexive.

- Any solution x does not dominate itself.
- Condition II of definition 3 does not allow the reflexive property to be satisfied.

Symmetric :

The dominance relation also **NOT** symmetric

- $x \preceq y$ does not imply $y \preceq x$.

Properties of dominance relation

Antisymmetric :

- Dominance relation **can not be** antisymmetric

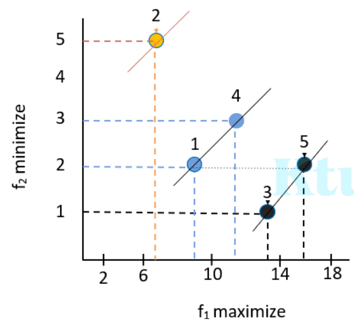
Transitive :

- The dominance relation is **TRANSITIVE**
  - If $x \preceq y$ and $y \preceq z$, then $x \preceq z$.

Pareto optimal Solution

- **Non-dominated solution set**
- Given a set of solutions, the non-dominated solution set is a set of all the solutions that are not dominated by any member of the solution set
- The non-dominated set of the entire feasible decision space is called the **Pareto-optimal set**
- The boundary defined by the set of all points mapped from the Pareto optimal set is called the **Pareto- optimal front**

Pareto optimality



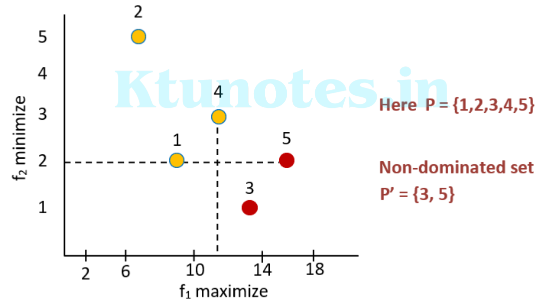
Non-dominated front

- Consider solutions 3 and 5
- Solution 5 is better than solution 3 with respect to f_1 while 5 is worse than 3 with respect to f_2.
- Hence, we can not conclude that 5 dominates 3 nor 3 dominated 5.
- In other words, we can not say that two solutions 3 and 5 are better.
- {3,5} can be considered a non-dominated front

Non-dominated set

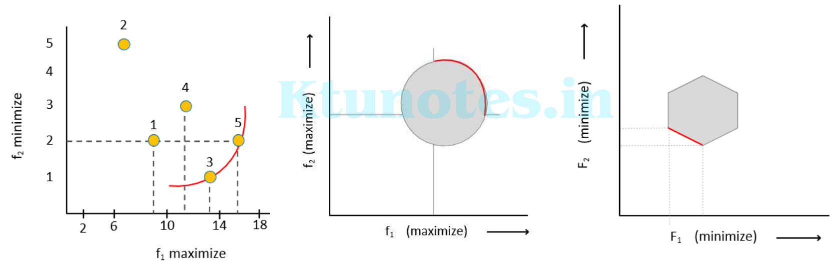
Non-dominated set

Among a set of solutions P, the non-dominated set of solutions P' are those which are not dominated by any member of the set P.

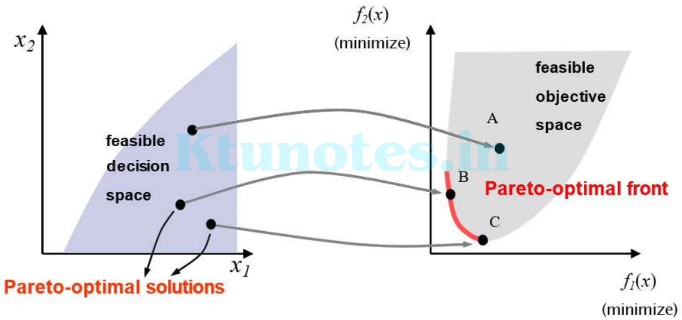


Non-dominated set

The non-dominated set concept is applicable when there is a trade-off in solutions.

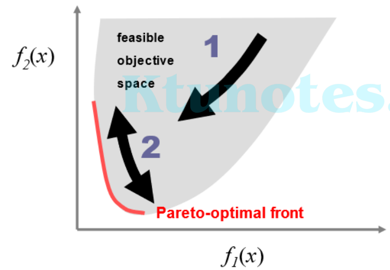


Graphical Depiction of Pareto Optimal Solution



Graphical Depiction of Pareto Optimal Solution

- Find a set of solutions as close as possible to Pareto-optimal front
- To find a set of solutions as diverse as possible

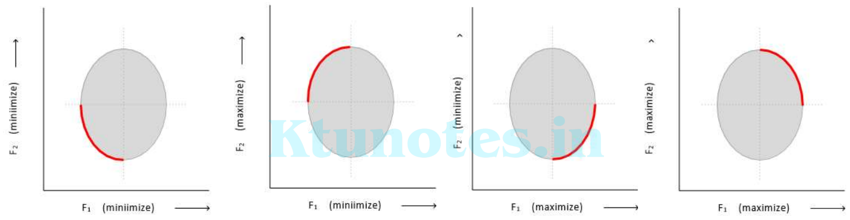


Pareto optimal set

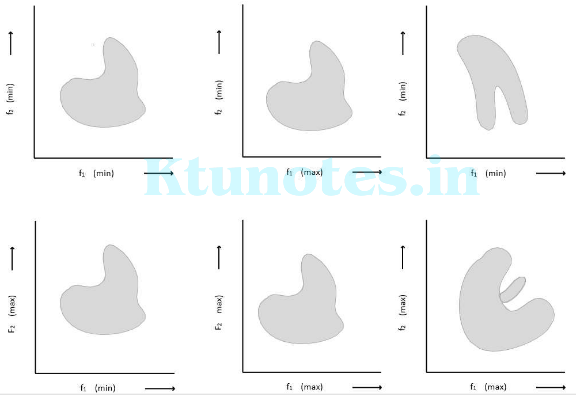
Pareto optimal set
When the set P is the entire search space, that is $P = S$, the resulting non-dominated set P' is called the Pareto-optimal set.

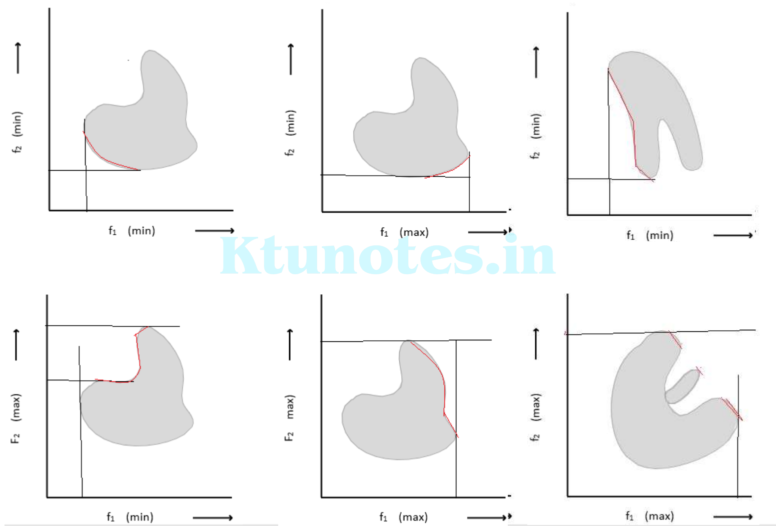
The following figures show the Pareto optimal set for a set of feasible solutions over an entire search space under four different situations with two objective functions f_1 and f_2.

Pareto optimal set



Pareto optimal set

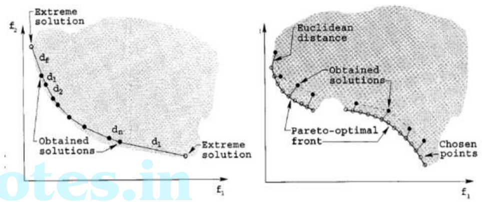




Pareto Optimality

- Solutions along the line are all non-dominated solutions.
- Dominated solutions are inside the line as there is another solution on the line with at least one objective that is better.
- The line is the Pareto-optimal front and the solutions on it are called Pareto-optimal.
- All Pareto-optimal solutions are

non-dominated



Pareto-Optimal Front

It is important to find solutions as close as possible to the Pareto front and as far along it as possible.

Example of Pareto Optimal solution

- There are 9 types of air tickets with different time and cost details.
- We need to choose from them with objective functions of minimum cost and minimum time.
- If we compare air tickets A and B – we see that while

A is better from a Time point of view, and B is better from the Cost angle. A and B thus form a non-dominated set.

- However, if we compare B and C, we see that B is equal to C in time, but better than C in Cost. Hence, we can say that B "dominates" C .
- So, as long as B is a feasible option, there is no reason to choose C.

Ticket Type	Time (Hours)	Cost (1000 Won)
A	2	7.5
B	3	6
C	3	7.5
D	4	5
E	4	6.5
F	5	4.5
G	5	6
H	5	7
I	6	6.5

Example of Pareto Optimal solution

- There are 9 types of air tickets with different time and cost details.
- We need to choose from them with objective functions of minimum cost and minimum time.
- Find the dominance of A, by comparing A, with B, C, D, E, F, G, H, I.
- A→B- Non dominating,
- A→C- A Dominates C
- A→D- Non dominating ,A→E- Non dominating
- A→F- Non dominating, A→G- Non dominating
- A→H- Non dominating, A→I- Non dominating

Ticket Type	Time (Hours)	Cost (1000 Won)
A	2	7.5
B	3	6
C	3	7.5
D	4	5
E	4	6.5
F	5	4.5
G	5	6
H	5	7
I	6	6.5

Example of Pareto Optimal solution

- Similarly Find the dominance Set of B, by comparing B, with C, D, E, F, G, H, I.
- B→C- B Dominates C
- B→D- Non dominating- B Dominates E
- B→F- Non-dominating, B→G- B Dominates G
- B→H- B Dominates H, B→I- B Dominates I
- B dominates C, E, G, H, I
- Similarly No dominance set for C, E, G, H, I
- D Dominates E, G, H, I
- F dominates G, H, I

Ticket Type	Time (Hours)	Cost (1000 Won)
A	2	7.5
B	3	6
C	3	7.5
D	4	5
E	4	6.5
F	5	4.5
G	5	6
H	5	7
I	6	6.5

Example of Pareto Optimal solution

- 9 air tickets to choose from them with objective functions of **minimum cost** and **minimum time**.

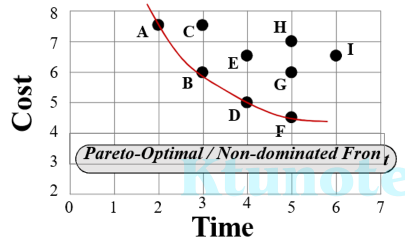
A "dominates" C
B "dominates" C, E, G, H, I
D "dominates" E, G, H, I
F "dominates" G, H, and I

Considering any two air tickets between A, B, D, or F, we find that they do not dominate over each other!

- Thus, A, B, D, and F form a **"Non-dominated set"**.
- Hence, A, B, D, and F will form the **Pareto-Optimal Front** or **Non-Dominated Front** and they are the **Pareto-Optimal solutions**.

Ticket Type	Time (Hours)	Cost (1000 Won)
A	**2**	**7.5**
B	**3**	**6**
C	**3**	**7.5**
D	4	5
E	4	6.5
F	5	4.5
G	5	6
H	5	7
I	6	6.5

Example of Pareto Optimal solution



Ticket Type	Time (Hours)	Cost (1000 Won)
A	**2**	**7.5**
B	**3**	**6**
C	**3**	**7.5**
D	4	5
E	4	6.5
F	5	4.5
G	5	6
H	5	7
I	6	6.5

A, B, D and F form a "Non-dominated set" and are "Pareto-Optimal solutions."

Pareto Vs Non Pareto optimization Techniques

Non-Pareto Techniques

- Approaches that do not incorporate **directly** the concept of Pareto optimum.
- Incapable of producing **certain portions** of the Pareto Front.
- Efficient and easy to implement, but appropriate to handle only **a few** objectives.

Pareto Techniques

- Suggested originally by **Goldberg** (1989) to solve multi-objective problems.
- Use of **non-dominated ranking** and selection to move the population towards the **Pareto Front**.
- Require a **ranking procedure** and the technique to **maintain diversity** in the population.

Pareto Vs Non Pareto optimization Techniques

Non-Pareto Techniques

- Approaches that do not incorporate **directly** the concept of Pareto optimum.
- Incapable of producing **certain portions** of the Pareto Front.
- Efficient and easy to implement, but appropriate to handle only **a few** objectives.

Pareto Techniques

- Suggested originally by **Goldberg** (1989) to solve multi-objective problems.
- Use of **non-dominated ranking** and selection to move the population towards the **Pareto Front**.
- Require a **ranking procedure** and the technique to **maintain diversity** in the population.

Pareto Vs Non Pareto optimization Techniques

Non-Pareto Techniques

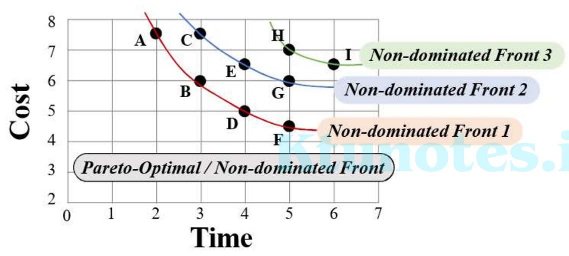
- Aggregating approaches
- Vector evaluated genetic algorithm (VEGA)
- Lexicographic ordering
- The C-constraint method

Pareto Techniques

- Multi-objective genetic algorithm (MOGA)
- **Non-dominated sorting genetic algorithm-II (NSGA-II)**
- Multi-objective particle swarm optimization (MOPSO)
- Pareto evolution archive strategy (PAES)
- Strength Pareto evolutionary algorithm (SPEA-II)

Non-dominated Sorting Genetic Algorithm-I

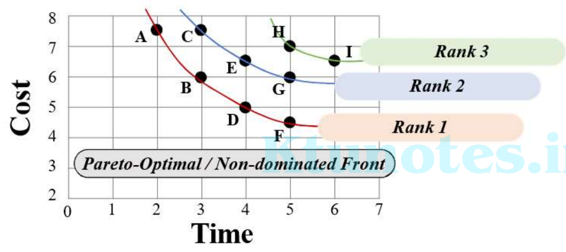
Non-dominated Sorting



Ticket Type	Time (Hours)	Cost (1000 Won)
A	2	7.5
B	3	6
C	3	7.5
D	4	5
E	4	6.5
F	5	4.5
G	5	6
H	5	7
I	6	6.5

A, B, D and F form a "Non-dominated set" and are "Pareto-Optimal solutions."

Non-dominated Sorting



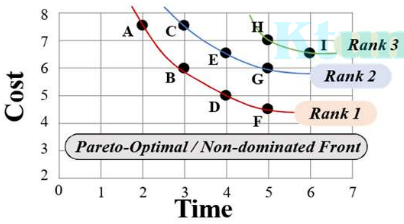
Ticket Type	Time (Hours)	Cost (1000 Won)
A	2	7.5
B	3	6
C	3	7.5
D	4	5
E	4	6.5
F	5	4.5
G	5	6
H	5	7
I	6	6.5

A, B, D and F form a "Non-dominated set" and are "Pareto-Optimal solutions."

Non-dominated Rank Comparison

- Given two solutions i and j, **i is preferred to j if:**

  rank (i) < **rank** (j)



Fitness assignment:

- **Non-dominated Front 1** are **ranked 1** and have the **highest** fitness.
- Solutions in the **same front** have the **same rank** and **same fitness**.
- A solution with lower **non-dominated rank** is preferred over others.

When two solutions have the same non-dominated rank (belong to the same front), the one located in a less crowded region of the front is preferred.

Optimality conditions

Optimality Conditions

The following condition is known as the necessary condition for Pareto-optimality.

Theorem 1. *(Fritz–John necessary condition). A necessary condition for* $\mathbf{x}^*$ *to be Pareto-optimal is that there exist vectors* $\boldsymbol{\lambda} \geq \mathbf{0}$ *and* $\mathbf{u} \geq \mathbf{0}$ *(where* $\boldsymbol{\lambda} \in \mathbb{R}^M$, $\mathbf{u} \in \mathbb{R}^J$ *and* $\boldsymbol{\lambda}, \mathbf{u} \neq \mathbf{0}$*) such that the following conditions are true:*

1. $\sum_{m=1}^{M} \lambda_m \nabla f_m(\mathbf{x}^*) - \sum_{j=1}^{J} u_j \nabla g_j(\mathbf{x}^*) = \mathbf{0}$, *and*
2. $u_j g_j(\mathbf{x}^*) = 0$ *for all* $j = 1, 2, \ldots, J$.

Optimality Conditions

For an unconstrained MOOP, the above theorem requires the following condition:

$$\sum_{m=1}^{M} \lambda_m \nabla f_m(\mathbf{x}^*) = \mathbf{0}$$

to be necessary for a solution to be Pareto-optimal.

Optimality Conditions

Theorem 2. *(Karush–Kuhn–Tucker sufficient condition for Pareto-optimality).* *Let the objective functions be convex and the constraint functions of the problem be nonconvex.*

Let the objective and constraint functions be continuously differentiable at a feasible solution $\mathbf{x}^*$

A sufficient condition for

$\mathbf{x}^*$ *to be Pareto-optimal is that there exist vectors* $\boldsymbol{\lambda} > \mathbf{0}$ *and* $\mathbf{u} \geq \mathbf{0}$ *(where* $\boldsymbol{\lambda} \in \mathbb{R}^M$ *and* $\mathbf{u} \in \mathbb{R}^J$ *) such that the following equations are true:*

1. $\sum_{m=1}^{M} \lambda_m \nabla f_i(\mathbf{x}^*) - \sum_{j=1}^{J} u_j \nabla g_j(\mathbf{x}^*) = \mathbf{0}$, *and*
2. $u_j g_j(\mathbf{x}^*) = 0$ *for all* $j = 1, 2, \ldots, J$.

Hybrid systems

- A Hybrid system is an intelligent system that is framed by combining at least two intelligent technologies like Fuzzy Logic, Neural networks, Genetic algorithms, reinforcement learning, etc.
- The combination of different techniques in one computational model makes these systems possess an extended range of capabilities.
- These systems are capable of reasoning and learning in an uncertain and imprecise environment. These systems can provide human-like expertise like domain knowledge, adaptation in noisy environments, etc.
- **Types of Hybrid Systems:**
  - Neuro-Fuzzy Hybrid systems
  - Neuro Genetic Hybrid systems
  - Fuzzy Genetic Hybrid systems

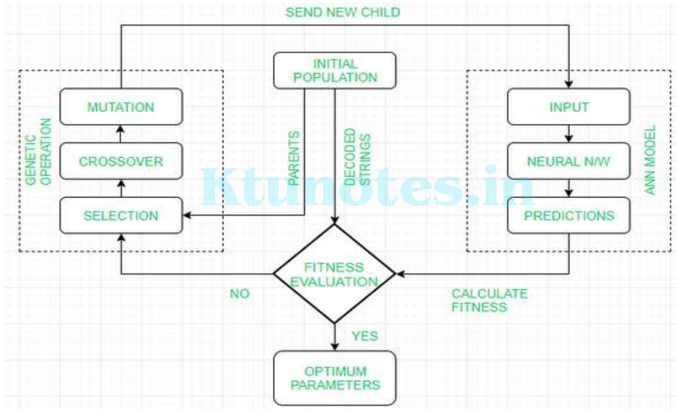
Hybrid systems

Genetic neuro hybrid systems

Genetic neuro hybrid systems

- A Neuro Genetic hybrid system is a system that combines **Neural networks**: which are capable to learn various tasks from examples, classify objects and establish relations between them, and a **Genetic algorithm**: which serves important search and optimization techniques.
- Genetic algorithms can be used to improve the performance of Neural Networks and they can be used to decide the connection weights of the inputs.
- These algorithms can also be used for topology selection and training networks.
- **Applications:**
  - Face recognition
  - DNA matching
  - Animal and human research
  - Behavioural system

Genetic neuro hybrid systems



Genetic neuro hybrid systems

- **Working Flow:**
- GA repeatedly modifies a population of individual solutions. GA uses three main types of rules at each step to create the next generation from the current population:
  - **Selection** to select the individuals, called parents, that contribute to the population of the next generation
  - **Crossover** to combine two parents to form children for the next generation
  - **Mutation** to apply random changes to individual parents in order to form children
- GA then sends the new child generation to the ANN model as a new input parameter.
- Finally, calculating the fitness by the developed ANN model is performed.

Genetic neuro hybrid systems

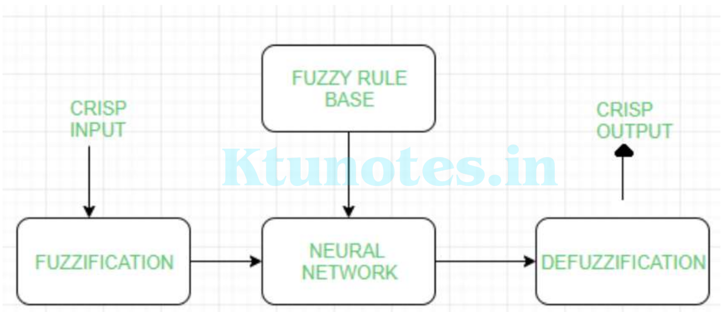
- **Advantages:**
  - GA is used for topology optimization i.e. to select a number of hidden layers, number of hidden nodes, and interconnection pattern for ANN.
  - In Gas, the learning of ANN is formulated as a weight optimization problem, usually using the inverse mean squared error as a fitness measure.
  - Control parameters such as learning rate, momentum rate, tolerance level, etc are also optimized using GA.
  - It can mimic the human decision-making process.
- **Disadvantages:**
  - Highly complex system.
  - Accuracy of the system is dependent on the initial population.
  - Maintenance costs are very high.

Neuro-fuzzy hybrid systems

Neuro-fuzzy hybrid systems

- The Neuro-fuzzy system is based on a fuzzy system which is trained on the basis of the working of neural network theory.
- The learning process operates only on the local information and causes only local changes in the underlying fuzzy system.
- A neuro-fuzzy system can be seen as a 3-layer feedforward neural network.
- The first layer represents input variables, the middle (hidden) layer represents fuzzy rules and the third layer represents output variables.
- Fuzzy sets are encoded as connection weights within the layers of the network, which provides functionality in processing and training the model.

Neuro-fuzzy hybrid systems



Neuro-fuzzy hybrid systems

- **Working flow:**
- In the input layer, each neuron transmits external crisp signals directly to the next layer.
- Each fuzzification neuron receives a crisp input and determines the degree to which the input belongs to the input fuzzy set.
- The fuzzy rule layer receives neurons that represent fuzzy sets.
- An output neuron combines all inputs using fuzzy operation UNION.
- Each defuzzification neuron represents the single output of the neuro-fuzzy system.

Neuro-fuzzy hybrid systems

- **Advantages:**
- It can handle numeric, linguistic, logic, etc kind of information.
- It can manage imprecise, partial, vague, or imperfect information.
- It can resolve conflicts through collaboration and aggregation.
- It has self-learning, self-organizing and self-tuning capabilities.
- It can mimic the human decision-making process.
- **Disadvantages:**
- Hard to develop a model from a fuzzy system
- Problems in finding suitable membership values for fuzzy systems
- Neural networks cannot be used if training data is not available.

Neuro-fuzzy hybrid systems

- **Applications:**
  - Student Modelling
  - Medical systems
  - Traffic control systems
  - Forecasting and predictions