



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

UNIT – I – Advanced Optimization Methods – SPRA5301

1. Operation Research

The term Operations Research (OR) was first coined by MC Closky and Trefthen in 1940 in a small town, Bowdsey of UK. The main origin of OR was during the second world war – The military commands of UK and USA engaged several inter-disciplinary teams of scientists to undertake scientific research into strategic and tactical military operations. Their mission was to formulate specific proposals and to arrive at the decision on optimal utilization of scarce military resources and also to implement the decisions effectively. In simple words, it was to uncover the methods that can yield greatest results with little efforts. Thus it had gained popularity and was called “An art of winning the war without actually fighting it”

The name Operations Research (OR) was invented because the team was dealing with research on military operations. The encouraging results obtained by British OR teams motivated US military management to start with similar activities. The work of OR team was given various names in US: Operational Analysis, Operations Evaluation, Operations Research, System Analysis, System Research, Systems Evaluation and so on. The first method in this direction was simplex method of linear programming developed in 1947 by G.B Dantzig, USA. Since then, new techniques and applications have been developed to yield high profit from least costs. Now OR activities has become universally applicable to any area such as transportation, hospital management, agriculture, libraries, city planning, financial institutions, construction management and so forth. In India many of the industries like Delhi cloth mills, Indian Airlines, Indian Railway, etc are making use of OR techniques.

HISTORY OF OR

The term OR coined by Mc.Closkey and Tref in the year 1940 in U.K. OR was first used in military operations for optimum utilization of resources.

YEAR	EVENTS
1940	Term OR was coined by Mc.Closkey and Trefthen in U.K
1949	<ul style="list-style-type: none">• OR unit was set up in India in Hyderabad. (The Regional Research Lab)• OR unit was set up at defence science lab.
1951	<ul style="list-style-type: none">• The National Research Council (NRC) in US formed a committee on OR.• The first book was published called “Methods on OR” by Morse and Kimball.
1952	<ul style="list-style-type: none">• OR Society of America was formed.
1953	<ul style="list-style-type: none">• OR unit was set up in Calcutta in the “Indian Statistical Institute”.
1995	OR society of India was established.

OR gained its significance first in the defence during the World War II (1939-1945) in order to make the best use of limited military resources and win the war. The effectiveness of OR in defence spread interest in Government departments and industry.

CONCEPT AND DEFINITION OF OR

Operations research signifies research on operations. It is the organized application of modern science, mathematics and computer techniques to complex military, government, business or industrial problems arising in the direction and management of large systems of men, material, money and machines. The purpose is to provide the management with explicit quantitative understanding and assessment of complex situations to have sound basics for arriving at best decisions. Operations research seeks the optimum state in all conditions and thus provides optimum solution to organizational problems.

DEFINITION

“OR is defined as the application of Scientific methods, tools and techniques to problems involving the operations of a system so as to provide to those in control of the system, with optimum solutions to the problem”.

“OR is defined as the application of Scientific methods by interdisciplinary team to problems involving control of organized system, so as to provide solutions which serve best to the organization as a whole.

OR is otherwise called as the “Science of use”.

OR is the combination of management principles and mathematical concepts (Quantitative techniques) for managerial decision-making purpose.

CHARACTERISTICS OF OR

- Aims to find solutions for problems of organized systems.
- Aims to provide optimum solution. Optimization means the best minimum or maximum for the criteria under consideration.
- It is the application of scientific methods, tools and techniques.
- Interdisciplinary team approach is used to solve the problems.
- The solutions that serve best to the organization as a whole is taken into consideration.

2. Optimization

Linear Programming Problem: A linear programming problem is one in which we have to find optimal value (maximum or minimum) of a linear function of several variables (called objective function) subject to certain conditions that the variables are non-negative and satisfying by a set of linear inequalities with variables, are sometimes called division variables.

Objective Function: A linear function $z = px + qy$ (p and q are constants) which has to be maximised or minimised, is called an objective function.

Constraints: The linear inequalities or equations or restrictions on the variables of the linear programming problem are called constraints. The conditions $x \geq 0$, $y \geq 0$ are called non-negative restrictions.

Optimal Value: The maximum or minimum value of an objective function is known as its optimal value.

Optimisation Problem: A problem, which seeks to maximise or minimise a linear function subject to certain constraints as determined by a set of linear inequalities, is called an optimisation problem.

Feasible Region: The common region determined by all the constraints including non-negative constraints $x, y \geq 0$ of a linear programming problem is called the feasible region for the problem. The region other than the feasible region is called an infeasible region. The feasible region is always a convex polygon.

Feasible Solutions: Points within and on the boundary of the feasible region represent feasible solutions of the constraints. Any point outside the feasible region is called an infeasible solution.

Optimal Feasible Solution: Any point in the feasible region that gives the optimal value of the objective function is called the optimal feasible solution.

Bounded and Unbounded Region: A feasible region of a system of linear inequalities is said to be bounded, if it can be enclosed within a circle. Otherwise, it is called unbounded.

Optimization is the way of life. We all have finite resources and time and we want to make the most of them. From using your time productively to solving supply chain problems for your company – everything uses optimization. It's an especially interesting and relevant topic in data science.

3. Linear programming (LP)

Linear programming (LP) is one of the simplest ways to perform optimization. It helps you solve some very complex optimization problems by making a few simplifying assumptions. As an analyst, you are bound to come across applications and problems to be solved by Linear Programming.

For some reason, LP doesn't get as much attention as it deserves while learning data science. So, I thought let me do justice to this awesome technique. I decided to write an article that explains Linear programming in simple English. I have kept the content as simple as possible. The idea is to get you started and excited about Linear Programming.

Note- If you want to learn this in a course format, we have curated this free course for you- [Linear Programming for Data Science Professionals](#)

Now, what is linear programming? Linear programming is a simple technique where we depict complex relationships through linear functions and then find the optimum points. The important word in the previous sentence is depicted. The real relationships might be much more complex – but we can simplify them to linear relationships.

Applications of linear programming are everywhere around you. You use linear programming at personal and professional fronts. You are using linear programming when you are driving from home to work and want to take the shortest route. Or when you have a project delivery you make strategies to make your team work efficiently for on-time delivery.

Example of a linear programming problem

Let's say a FedEx delivery man has 6 packages to deliver in a day as shown in figure 1. The warehouse is located at point A. The 6 delivery destinations are given by U, V, W, X, Y, and Z. The numbers on the lines indicate the distance between the cities. To save on fuel and time the delivery person wants to take the shortest route.

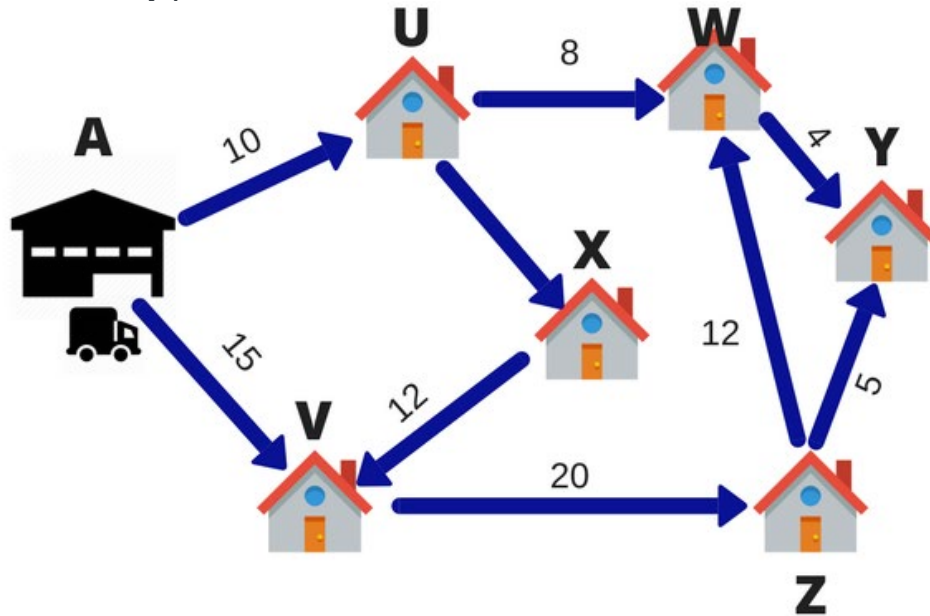


Figure 1. Allocation and Scheduling

So, the delivery person will calculate different routes for going to all the 6 destinations and then come up with the shortest route. This technique of choosing the shortest route is called linear programming.

In this case, the objective of the delivery person is to deliver the parcel on time at all 6 destinations. The process of choosing the best route is called Operation Research. Operation research is an approach to decision-making, which involves a set of methods to operate a system. In the above example, my system was the Delivery model.

Linear programming is used for obtaining the most optimal solution for a problem with given constraints. In linear programming, we formulate our real-life problem into a mathematical model. It involves an objective function, linear inequalities with subject to constraints.

Is the linear representation of the 6 points above representative of the real-world? Yes and No. It is an oversimplification as the real route would not be a straight line. It would likely have multiple turns, U-turns, signals and traffic jams. But with a simple assumption, we have reduced the complexity of the problem drastically and are creating a solution that should work in most scenarios.

Formulating a problem –

Let's manufacture some chocolates

Example: Consider a chocolate manufacturing company that produces only two types of chocolate – A and B. Both the chocolates require Milk and Choco only. To manufacture each unit of A and B, the following quantities are required:

Each unit of A requires 1 unit of Milk and 3 units of Choco

Each unit of B requires 1 unit of Milk and 2 units of Choco

The company kitchen has a total of 5 units of Milk and 12 units of Choco. On each sale, the company makes a profit of

Rs 6 per unit A sold

Rs 5 per unit B sold.

Now, the company wishes to maximize its profit. How many units of A and B should it produce respectively?

Solution: The first thing I'm gonna do is represent the problem in a tabular form for better understanding.

	Milk	Choco	Profit per unit
A	1	3	Rs 6
B	1	2	Rs 5
Total	5	12	

Let the total number of units produced by A be = X

Let the total number of units produced by B be = Y

Now, the total profit is represented by Z

The total profit the company makes is given by the total number of units of A and B produced multiplied by its per-unit profit of Rs 6 and Rs 5 respectively.

Profit: $\text{Max } Z = 6X + 5Y$ which means we have to maximize Z.

The company will try to produce as many units of A and B to maximize the profit. But the resources Milk and Choco are available in a limited amount.

As per the above table, each unit of A and B requires 1 unit of Milk. The total amount of Milk available is 5 units. To represent this mathematically,

$$X + Y \leq 5$$

Also, each unit of A and B requires 3 units & 2 units of Choco respectively. The total amount of Choco available is 12 units. To represent this mathematically,

$$3X + 2Y \leq 12$$

Also, the values for units of A can only be integers.

So we have two more constraints, $X \geq 0$ & $Y \geq 0$

For the company to make maximum profit, the above inequalities have to be satisfied. This is called formulating a real-world problem into a mathematical model.

Common terminologies used in Linear Programming

Let us define some terminologies used in Linear Programming using the above example.

Decision Variables: The decision variables are the variables that will decide my output. They represent my ultimate solution. To solve any problem, we first need to identify the decision variables. For the above example, the total number of units for A and B denoted by X & Y respectively are my decision variables.

Objective Function: It is defined as the objective of making decisions. In the above example, the company wishes to increase the total profit represented by Z. So, profit is my objective function.

Constraints: The constraints are the restrictions or limitations on the decision variables. They usually limit the value of the decision variables. In the above example, the limit on the availability of resources Milk and Choco are my constraints.

Non-negativity restriction: For all linear programs, the decision variables should always take non-negative values. This means the values for decision variables should be greater than or equal to 0.

The process to formulate a Linear Programming problem

Let us look at the steps of defining a Linear Programming problem generically:

Identify the decision variables

Write the objective function

Mention the constraints

Explicitly state the non-negativity restriction

For a problem to be a linear programming problem, the decision variables, objective function and constraints all have to be linear functions.

If all the three conditions are satisfied, it is called a Linear Programming Problem.

4. Linear Programs by Graphical Method

A linear program can be solved by multiple methods. In this section, we are going to look at the Graphical method for solving a linear program. This method is used to solve a two-variable linear program. If you have only two decision variables, you should use the graphical method to find the optimal solution.

A graphical method involves formulating a set of linear inequalities subject to the constraints. Then the inequalities are plotted on an X-Y plane. Once we have plotted all the inequalities on a graph the intersecting region gives us a feasible region. The feasible region explains what all values our model can take. And it also gives us the optimal solution.

Let's understand this with the help of an example.

Example: A farmer has recently acquired a 110 hectares piece of land. He has decided to grow Wheat and barley on that land. Due to the quality of the sun and the region's excellent climate, the entire production of Wheat and Barley can be sold. He wants to know how to plant each variety in the 110 hectares, given the costs, net profits and labor requirements according to the data shown below figure 2:

Variety	Cost (Price/Hec)	Net Profit (Price/Hec)	Man-days/Hec
Wheat	100	50	10
Barley	200	120	30

The farmer has a budget of US\$10,000 and availability of 1,200 man-days during the planning horizon. Find the optimal solution and the optimal value.

Solution: To solve this problem, first we gonna formulate our linear program.

Formulation of Linear Problem

Step 1: Identify the decision variables

The total area for growing Wheat = X (in hectares)

The total area for growing Barley = Y (in hectares)

X and Y are my decision variables.

Step 2: Write the objective function

Since the production from the entire land can be sold in the market. The farmer would want to maximize the profit for his total produce. We are given net profit for both Wheat and Barley. The farmer earns a net profit of US\$50 for each hectare of Wheat and US\$120 for each Barley.

Our objective function (given by Z) is, $\text{Max } Z = 50X + 120Y$

Step 3: Writing the constraints

1. It is given that the farmer has a total budget of US\$10,000. The cost of producing Wheat and Barley per hectare is also given to us. We have an upper cap on the total cost spent by the farmer. So our equation becomes:

$$100X + 200Y \leq 10,000$$

2. The next constraint is the upper cap on the availability of the total number of man-days for the planning horizon. The total number of man-days available is 1200. As per the table, we are given the man-days per hectare for Wheat and Barley.

$$10X + 30Y \leq 1200$$

3. The third constraint is the total area present for plantation. The total available area is 110 hectares. So the equation becomes,

$$X + Y \leq 110$$

Step 4: The non-negativity restriction

The values of X and Y will be greater than or equal to 0. This goes without saying.

$$X \geq 0, Y \geq 0$$

We have formulated our linear program. It's time to solve it.

Solving an LP through Graphical method

Since we know that $X, Y \geq 0$. We will consider only the first quadrant.

To plot for the graph for the above equations, first I will simplify all the equations.

$100X + 200Y \leq 10,000$ can be simplified to $X + 2Y \leq 100$ by dividing by 100.

$10X + 30Y \leq 1200$ can be simplified to $X + 3Y \leq 120$ by dividing by 10.

The third equation is in its simplified form, $X + Y \leq 110$.

Plot the first 2 lines on a graph in the first quadrant (like shown below)

The optimal feasible solution is achieved at the point of intersection where the budget & man-days constraints are active. This means the point at which the equations $X + 2Y \leq 100$ and $X + 3Y \leq 120$ intersect gives us the optimal solution.

The values for X and Y which gives the optimal solution is at (60,20).

To maximize profit the farmer should produce Wheat and Barley in 60 hectares and 20 hectares of land respectively.

The maximum profit the company will gain is,

$$\begin{aligned} \text{Max } Z &= 50 * (60) + 120 * (20) \\ &= \text{US\$5400} \end{aligned}$$

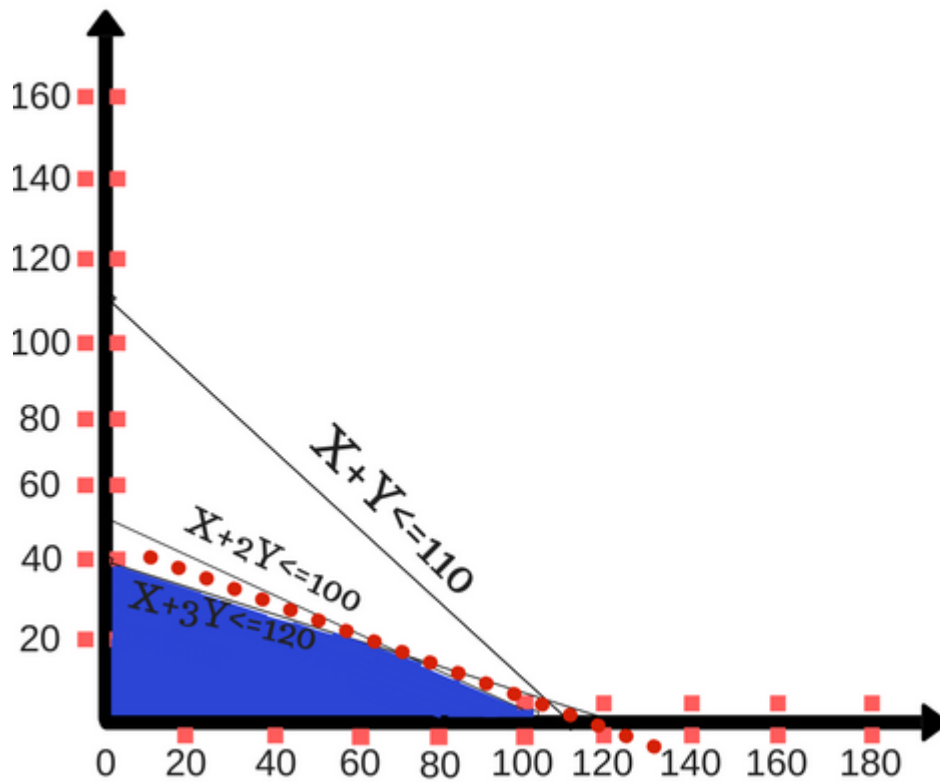


Figure 2. Graphical Method

Therefore from the output, we see that the organization should produce 88 units of toy A and 20 units of toy B and the maximum profit for the organization will be Rs.2600.

5. Simplex Method

Simplex Method is one of the most powerful & popular methods for linear programming. The simplex method is an iterative procedure for getting the most feasible solution. In this method, we keep transforming the value of basic variables to get maximum value for the objective function.

A linear programming function is in its standard form if it seeks to maximize the objective

$$Z = c_1 x_1 + c_2 x_2 + \dots + c_n x_n$$

function.

subject

to

$$a_{11} x_1 + a_{12} x_2 + \dots + a_{1n} x_n \leq b_1$$

constraints,

$$a_{21} x_1 + a_{22} x_2 + \dots + a_{2n} x_n \leq b_2$$

$$a_{m1} x_1 + a_{m2} x_2 + \dots + a_{mn} x_n \leq b_m$$

where, $x_i \geq 0$ and $b_i \geq 0$. After adding slack variables, the corresponding system of constraint equation is,

$$a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n + s_1 = b_1$$

$$a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_n + s_2 = b_2$$

$$a_{m1}x_1 + a_{m2}x_2 + \dots + a_{mn}x_n + s_m = b_m \quad \text{where, } s_i \geq 0$$

The variables, s_1, s_2, \dots, s_m are called slack variables. They are non-negative numbers that are added to remove the inequalities from an equation.

The above explanation gives the theoretical explanation of the simplex method. Now, I am gonna explain how to use the simplex method in real life using Excel.

Example: The advertising alternatives for a company include television, newspaper and radio advertisements. The cost for each medium with its audience coverage is given below.

Mode	Television	Newspaper	Radio
Cost per advertisement (\$)	2000	600	300
Audience per advertisement	100,000	40,000	18,000

The local newspaper limits the number of advertisements from a single company to ten. Moreover, in order to balance the advertising among the three types of media, no more than half of the total number of advertisements should occur on the radio. And at least 10% should occur on television. The weekly advertising budget is \$18,200. How many advertisements should be run in each of the three types of media to maximize the total audience?

Solution: First I am going to formulate my problem for a clear understanding.

Step 1: Identify Decision Variables

Let X_1, X_2, X_3 represent the total number of ads for television, newspaper, and radio respectively.

Step 2: Objective Function

The objective of the company is to maximize the audience. The objective function is given by:

$$Z = 100,000X_1 + 40,000X_2 + 18,000X_3$$

Step 3: Write down the constraints

Now, I will mention each constraint one by one.

It is clearly given that we have a budget constraint. The total budget which can be allocated is \$18,200. And the individual costs per television, newspaper and radio advertisement is \$2000, \$600 and \$300 respectively. This can be represented by the equation,
$$2000X_1 + 600X_2 + 300X_3 \leq 18,200$$

For a newspaper advertisement, there is an upper cap on the number of advertisements to 10. My first constraints are,
$$X_2 \leq 10$$

The next constraint is the number of advertisements on television. The company wants at least 10% of the total advertisements to be on television. So, it can be represented as:

$$X_1 \geq 0.10(X_1 + X_2 + X_3)$$

The last constraint is the number of advertisements on the radio cannot be more than half of the total number of advertisements. It can be represented as
$$X_3 \leq 0.5(X_1 + X_2 + X_3)$$

Now, I have formulated my linear programming problem. We are using the simplex method to solve this. I will take you through the simplex method one by one.

To reiterate all the constraints are as follows. I have simplified the last two equations to bring them in standard form.

$$2000X_1 + 600X_2 + 300X_3 \leq 18,200$$

$$X_2 \leq 10 \quad -9X_1 + X_2 + X_3 \leq 0$$

$$-X_1 - X_2 + X_3 \leq 0$$

We have a total of 4 equations. To balance out each equation, I am introducing 4 slack variables, S_1 , S_2 , S_3 and S_4 .

So our equations are as follows:

$$2000X_1 + 600X_2 + 300X_3 + S_1 = 18,200$$

$$X_2 + S_2 = 10$$

$$-9X_1 + X_2 + X_3 + S_3 = 0$$

$$-X_1 - X_2 + X_3 + S_4 = 0$$

On solving the objective function you will get the maximum weekly audience as 1,052,000. You can follow the tutorial [here](#) to solve the equation. To solve a linear program in excel, follow this tutorial.

6. Applications of Linear Programming

Linear programming and Optimization are used in various industries. The manufacturing and service industry uses linear programming on a regular basis. In this section, we are going to look at the various applications of linear programming.

Manufacturing industries use linear programming for analyzing their supply chain operations. Their motive is to maximize efficiency with minimum operation cost. As per the recommendations from the linear programming model, the manufacturer can reconfigure their storage layout, adjust their workforce and reduce the bottlenecks. Here is a small Warehouse case study of Cequent a US-based company, watch this video for a more clear understanding.

Linear programming is also used in organized retail for shelf space optimization. Since the number of products in the market has increased in leaps and bounds, it is important to understand what does the customer want. Optimization is aggressively used in stores like Walmart, Hypercity, Reliance, Big Bazaar, etc. The products in the store are placed strategically keeping in mind the customer shopping pattern. The objective is to make it easy for a customer to locate & select the right products. This is subject to constraints like limited shelf space, a variety of products, etc.

Optimization is also used for optimizing Delivery Routes. This is an extension of the popular traveling salesman problem. The service industry uses optimization for finding the best route for multiple salesmen traveling to multiple cities. With the help of clustering and greedy algorithm, the delivery routes are decided by companies like FedEx, Amazon, etc. The objective is to minimize the operation cost and time.

Optimizations are also used in Machine Learning. Supervised Learning works on the fundamental of linear programming. A system is trained to fit on a mathematical model of a function from the labeled input data that can predict values from an unknown test data.

Well, the applications of Linear programming don't end here. There are many more applications of linear programming in real-world like applied by Shareholders, Sports, Stock Markets, etc. Go on and explore further.

7. Principle of Simplex Method

As the fundamental theorem of LP problem tells us that at least one basic feasible solution of any LP problem must be optimal, provided the optimal solution of the LP problem exists. Also, the number of basic feasible solutions of the LP problem is finite and at the most nC_m (where, n is number of decision variables and m is the number of constraints in the problem). On the other hand, the feasible solution may be infinite in number. So it is rather

impossible to search for optimal solutions from amongst all feasible solutions. Furthermore, a great labour will also be required in finding out all the basic feasible solutions and select that one which optimizes the objective function. In order to remove this difficulty; a simple method was developed by Dantzig (1947) which is known as Simplex Algorithm. Simplex Algorithm is a systematic and efficient procedure for finding corner point solutions and taking them for optimality. The evaluation of corner points always starts from the point of origin. This solution is then tested for optimality i.e. it tests whether an improvement in the objective function is possible by moving to adjacent corner point of the feasible function space. This iterative search for a better corner point is repeated until an optimal solution if it exists, is determined.

7.1 Basic Terms Involved in Simplex Procedure

The following terms relevant for solving a linear programming problem through simplex procedure are given below:

7.1.1 Standard form of linear programming problem

The standard form of LP problem is to develop the procedure for solving general LP problem. The optimal solution of the standard form of a LP problem is the same as original LP problem. The characteristics of standard form are given in following steps:

- Step 1.** All the constraints should be converted to equations except for the non-negativity restrictions which remain as inequalities (≥ 0).
- Step 2.** The right side element of each constraint should be made non-negative.
- Step 3.** All variables must have non-negative values.
- Step 4.** The objective function should be of maximization form.

4.1.2 Slack variables

If a constraint has less than or equal sign, then in order to make it on equality we have to add something positive to the left hand side. The non-negative variable which is added to the left hand side of the constraint to convert it into equation is called the slack variable. For example, consider the constraints.

$$3X_1 + 5X_2 \leq 2, 7X_1 + 4X_2 \leq 5, X_1, X_2 \geq 0$$

We add the slack variables $S_1 \geq 0, S_2 \geq 0$ on the left hand sides of above inequalities respectively to obtain

$$3X_1 + 5X_2 + S_1 = 2$$

$$7X_1 + 4X_2 + S_2 = 5$$

$$X_1, X_2, S_1, S_2 \geq 0$$

7.1.3 Surplus variables

If a constraint has greater than or equal to sign, then in order to make it an equality we have to subtract something non-negative from its left hand side. The positive variable which is subtracted from the left hand side of the constraint to convert it into equation is called the surplus variable.

For example, consider the constraints.

$$3X_1 + 5X_2 \geq 2, 2X_1 + 4X_2 \geq 5, X_1, X_2 \geq 0$$

We subtract the surplus variables $S_3 \geq 0, S_4 \geq 0$ on the left hand sides of above inequalities respectively to obtain

$$3X_1 + 5X_2 - S_3 = 2$$

$$2X_1 + 4X_2 - S_4 = 5$$

$$X_1, X_2, X_3, X_4 \geq 0$$

7.1.4 Solution to LPP

Any set $X = \{X_1, X_2, X_3, X_{n+m}\}$ of variables is called a solution to LP problem, if it satisfies the set of constraints only.

7.1.5 Feasible solution (FS)

Any set $X = \{X_1, X_2, X_3, X_{n+m}\}$ of variables is called a feasible solution of L.P. problem, if it satisfies the set of constraints as well as non-negativity restrictions.

7.1.6 Basic solution (BS)

For a system of m simultaneous linear equations in n variables ($n > m$), a solution obtained by setting $(n-m)$ variables equal to zero and solving for the remaining variables is called a basic solution. Such m variables (of course, some of them may be zero) are called basic variables and remaining $(n-m)$ zero-valued variables are called non-basic variables.

4.4 Computational Aspect of Simplex Method for Maximization Problem

Step 1: Formulate the linear programming model. If we have n -decision variables X_1, X_2, X_n and m constraints in the problem, then mathematical formulation of L P problem is

$$\text{Maximize } Z = C_1X_1 + C_2X_2 + \dots + C_nX_n$$

Subject to the constraints:

$$a_{11}X_1 + a_{12}X_2 + \dots + a_{1n}X_n \leq b_1$$

$$a_{21}X_1 + a_{22}X_2 + \dots + a_{2n}X_n \leq b_2$$

$$a_{m1}X_1 + a_{m2}X_2 + \dots + a_{mn}X_n \leq b_m$$

$$X_1, X_2, X_n \geq 0$$

Step 2: Express the mathematical model of LP problem in the standard form by adding slack variables in the left hand side of the constraints and assign zero coefficient to these variables in the objective function. Thus we can restate the problem in terms of equations as follows:

Maximize

$$Z = C_1X_1 + C_2X_2 + C_nX_n + 0X_{n+1} + 0X_{n+2} + \dots + 0X_{n+m}$$

Subject to the constraints:

$$a_{11} X_1 + a_{12} X_2 + \dots + a_{1n} X_n + X_{n+1} = b_1$$

$$a_{21} X_1 + a_{22} X_2 + \dots + a_{2n} X_n + X_{n+2} = b_2$$

$$a_{m1} X_1 + a_{m2} X_2 + \dots + a_{mn} X_n + X_{n+m} = b_m$$

$$X_1, X_2, \dots, X_n, X_{n+1}, X_{n+2}, \dots, X_{n+m} \geq 0$$

Step 3: Design the initial feasible solution .An initial basic feasible solution is obtained by setting $X_1=X_2=\dots=X_n=0$. Thus, we get $X_{n+1}=b_1$, $X_{n+2}=b_2$, $X_{n+m}=b_m$.

Step 4: Construct the starting (initial) simplex tableau. For computational efficiency and simplicity, the initial basic feasible solution, the constraints of the standard LP problem as well as the objective function can be displayed in a tabular form, called the *Simplex Tableau* as shown below.

Table 4.1 Initial simplex tableau

C _j (contribution per unit) →			c ₁	c ₂		c _n	0	0		0	Minimum Ratio*
C _b	Basic Variables	Value of Basic Variables	Coefficient Matrix				Identify Matrix				
	B	b(=X _B)	X ₁	X ₂		X _n	X _{n+1}	X _{n+2}		X _{n+m}	
0	s ₁	b ₁	a ₁₁	a ₁₂		a _{1n}	1	0		0	
0	s ₂	b ₂	a ₂₁	a ₂₂		a _{2n}	0	1		0	
.					
.					
.					
0	s _m	b _m	a _{m1}	a _{m2}		a _{mn}	0	0		1	
Contribution loss per unit:		$Z = \sum c_{Bj} a_{ij}$	0	0		0	0	0		0	
Net contribution per units:		$\Delta_j = Z_j - C_j$	c ₁	c ₂		c _n	0	0		0	

* Negative ratio is not to be considered.

The interpretation of the data in the above *Tableau* is given as under. Other simplex tableau will have similar interpretations.

- The first row, called the *objective row* of the simplex table indicates the values of C_j (j subscript refer to the column number) which are the coefficients of the (m + n) variables in the objective function. These coefficients are obtained directly from the objective function and the value C_j would remain the same in the succeeding tables. The second row of the table provides the major column headings for the table

and these column headings remain unchanged in the succeeding tables of the Simplex Method.

- (ii) The first column labelled C_B , also known as *objective column*, lists the coefficient of the current basic variables in the objective function. The second column labelled *Basic variables* points out the basic variables in the basis, and in the initial simplex tableau these basic variables are the slack variables. The third column labelled *Solution values* ($= x_B$), indicates the resources or the solution values of the basic variables.
- (iii) The *body matrix* (under non-basic variables) in the initial simplex tableau consists of the coefficients of the decision variables in the constraint set.
- (iv) The *identity matrix* in the initial simplex tableau represents the coefficient of the slack variables that have been added to the original inequalities to make them equation. The matrix under non-basic variables in the simplex tableau is called *coefficient matrix*. Each simplex tableau contains an identity matrix under the basic variables.
- (v) To find an entry in the Z_j row under a column, we multiply the entries of that column by the corresponding entries of C_B column and add the products, i.e., $Z_j = \sum C_B a_{ij}$.

The Z_j entry under the *Solution column* gives the current value of the objective function.

- (vi) The final row labeled $\Delta_j = Z_j - C_j$ called the *index* (or net evaluation) row, is used to determine whether or not the current solution is optimal. The calculation of $Z_j - C_j$ row simply involves subtracting each Z_j value from the corresponding C_j value for that column, which is written at the top of that column. Each entry in the Δ_j row represents the net contribution (or net marginal improvement) to the objective function that results by introducing one unit of each of the respective column variables.

Step 5: Test the Solution for Optimality. Examine the index row of the above simplex tableau. If all the elements in the index row are positive then the current solution is optimal. If there exist some negative values, the current solution can further be improved by removing one basic variable from the basis and replacing it by some non-basic one.

Step 6: Revision of the Current Simplex Tableau. At each iteration, the Simplex Method moves from the current basic feasible solution to a better basic feasible solution. This involves replacing one current basic variable (called the departing variable) by a new non-basic variable (called the entering variable).

- (a) *Determine which variable to enter into the solution-mix net.* One way of doing this is by identifying the column (and hence the variable) with the most negative number in the Δ_j row of the previous table.
- (b) *Determine the departing variable or variable to be replaced.* Next we proceed to determine which variable must be removed from the basis to pave way for the entering variable. This is accomplished by dividing each number in the quantity

(or solution values) column by the corresponding number in the pivot column selected in (a), *i.e.*, we compute the respective ratios b_1/a_{1j} , b_2/a_{2j} , b_m/a_{mj} (only for those a_{ij} s; $i=1,2, m$ which are strictly positive). These quotients are written in the last column labelled Minimum Ratio of the simplex tableau. The row corresponding to smallest of these non-negative ratios is called the pivot (or key) row and the corresponding basic variable will leave the basis. Let the minimum of $\{ b_1/a_{1j}, b_2/a_{2j}, b_m/a_{mj} ; a_{ij} > 0 \}$ be b_k/a_{kj} , then corresponding variables in the pivot row s_k will be termed as outgoing (or departing) variable in the next tableau to be constructed just after we put an arrow \rightarrow of type to right of k^{th} row of the simplex tableau 1.

- (c) *Identify the pivot number.* The non-zero positive number that lies at the intersection of the pivot column and pivot row of the given table is called the pivot (or key) number. We place a circle around the number.

Step 7: Evaluate (update) the new solution. After identifying the entering and departing variable, find the new basic feasible solution by constructing a new simplex tableau from the current one by using the following steps:

- (a) Compute new values for the pivot row by simply dividing every element of the pivot row by the pivot number.
- (b) New entries in the C_B column and X_B column are entered in the new table of the current solution
- (c) Compute new values for each of the remaining rows by using the following formula

$$\text{New row numbers} = \text{Number in old rows} - \{(\text{corresponding number above or below pivot number}) \times (\text{corresponding number in the row replaced in (a)})\}$$
- (d) Test for optimality. Compute the Z_j and index rows as previously demonstrated in the initial simplex tableau. If all numbers in the index row are either zero or positive, an optimal solution has been made attained. *i.e.*, there is no variable which can be introduced in the solution to cause the objective function value to increase.

4. *Revise the solution.* If any of the numbers in the index ($\Delta_j = Z_j - C_j$) row are negative, repeat the entire steps 5 & 6 again until an optimal solution has been obtained.

The above procedure is illustrated through the following example.

Example 1

A firm produces three products A, B, and C each of which passes through three different departments fabrication, finishing, packaging. Each unit of product A requires 3, 4 and 2 hours respectively, B requires 5, 4 and 4 hours respectively and C requires 2, 4 and 5 hours respectively in 3 departments respectively. Every day 60 hours are available in fabrication department, 72 hours in finishing and 100 hours in packaging department. If unit

contribution of unit A is Rs. 5, Rs. 10 for B and Rs. 3 for C. Then determine number of units of each product so that total contribution to cost is maximized and also determine if any capacity would remain unutilized.

Solution:

Step 1: Formulate this as LPP. Let X_1 , X_2 and X_3 be the number of units produced of the products A, B and C respectively.

Objective function: $\text{Max } Z = 5X_1 + 10X_2 + 3X_3$

Subject to constraints: $3X_1 + 5X_2 + 2X_3 \leq 60$

$$4X_1 + 4X_2 + 4X_3 \leq 72$$

$$2X_1 + 4X_2 + 5X_3 \leq 100 \quad X_1, X_2, X_3 \geq 0$$

Step 2: Now converting into standard form of LPP

$$\text{Max } Z = 5X_1 + 10X_2 + 3X_3 + 0S_1 + 0S_2 + 0S_3$$

$$3X_1 + 5X_2 + 2X_3 + S_1 = 60$$

$$4X_1 + 4X_2 + 4X_3 + S_2 = 72$$

$$2X_1 + 4X_2 + 5X_3 + S_3 = 100 \quad X_1, X_2, X_3, S_1, S_2, S_3 \geq 0$$

where S_1, S_2 and S_3 are slack variables.

Step 3: Find the initial feasible solution. An initial basic feasible solution is obtained by setting $X_1=0$, $X_2=0$ and $X_3=0$. Thus, we get $S_1=60$, $S_2=72$ and $S_3=100$.

Step 4: Construct the starting (initial) simplex tableau.

				C _j	5	10	8	0	0	0		
				→								
	B.V.	C _B	X _B	X ₁	X ₂	X ₃	S ₁	S ₂	S ₃	Minimum Ratio		
R ₁	S ₁	0	60	3	(5)	2	1	0	0	60/5=12→		
R ₂	S ₂	0	72	4	4	4	0	1	0	72/4=18		
R ₃	S ₃	0	100	2	4	5	0	0	1	100/4=25		
				Z _j	0	0	0	0	0	0		
				Δ _j	-5	-10	-8	0	0	0		

Step 5: The most negative value of Δ_j is -10 hence X_2 is the incoming variable (\uparrow) and the least positive minimum ratio is 12 hence S_1 is the outgoing variable (\rightarrow). The element under column X_2 and row R_1 is the key element i.e. 5 so divide each element of row R_1 by 5 (i.e. $R_1^a \rightarrow \frac{R_1}{5}$). Subtract appropriate multiples of this new row from the remaining rows, so

as to obtain zeros in the remaining positions. Performing the row operations $R_2^b - R_2^a \rightarrow 4R_1^a \rightarrow$ and $R_3^b - R_3^a \rightarrow 4R_1^a$ we get the second **Simplex tableau** as

			C _j →	5	10	8	0	0	0	
	B.V.	C _B	X _B	X ₁	X ₂	X ₃	S ₁	S ₂	S ₃	M.R.
R_1^b	X ₂	10	12	3/5	1	2/5	1/5	0	0	12/2/5=30
R_2^b	S ₂	0	24	8/5	0	12/5	-4/5	1	0	24/12/5=10 →
R_3^b	S ₃	0	52	-2/5	0	17/5	-4/5	0	1	52/17/5=15.294
			Z _j	6	10	4	2	0	0	
			Δ_j	1	0	-4	2	0	0	

Step 6: The most negative value of Δ_j is -4 hence X₃ is the incoming variable (↑) and the least positive minimum ratio is 10 hence S₂ is the outgoing variable (↓). The element under column X₂ and row

R₁ is the key element i.e. 5 so divide each element of row R₂^b by 12/5 (i.e. $R^c \rightarrow R^b * 5/12$). Subtract appropriate multiples of this new row from the remaining rows, so as to obtain zeros in the remaining positions. Performing the row operations $R_1^d \rightarrow R_1^c - \frac{2}{5}R_2^c$ and $R_3^d \rightarrow R_3^c - \frac{17}{5}R_2^c$.

We get the third **Simplex tableau** as

			C _j →	5	10	8	0	0	0
	B.V.	C _B	X _B	X ₁	X ₂	X ₃	S ₁	S ₂	S ₃
	X ₂	10	8	1/3	1	0	1/3	-1/6	0
	X ₃	8	10	2/3	0	1	-1/3	5/12	0
	S ₃	0	18	-8/3	0	0	1/3	-17/12	1
			Z _j	26/3	10	8	2/3	5/3	0
			Δ_j	11/3	0	0	2/3	5/3	0

It is apparent from this table that all $\Delta_j = Z_j - C_j$ are positive and therefore an optimum solution is reached. So X₁ = 0, X₂ = 8, X₃ = 10

$$Z = 5X_1 + 10X_2 + 8X_3 = 160$$

And also as S₃ is coming out to be 18 so there are 18 hours unutilized in finishing department.

In case the objective function of the given LP problem is to be minimized, then we convert it into a problem of maximization by using

Min. Z* = - Max. (-Z). The procedure of finding optimal solution using Simplex Method is illustrated through the following example:



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

UNIT – II – Advanced Optimization Methods – SPRA5301

1. Direct Search

A direct search method is a method which relies only on evaluating $f(x)$ on a sequence x_1, x_2, \dots and comparing values in order to calculate a minimizer of $f(x)$. Direct methods are usually applied in the following circumstance

the function $f(x)$ is not differentiable;

the derivatives of f are complicated to compute or even do not exist;

the function has few variables;

the location of an optimal solution is roughly known.

There are many direct search methods. Here we introduce the most popular five:

Golden section method

1. Fibonacci method
2. Hooke and Jeeves' method
3. Spendley, Hext and Himsworth's method
4. Nelder and Mead's method

The first two methods deal with a function of a single variable, the rest four deal with a function of several variables.

2. Golden section

The first algorithm that I learned for root-finding in my undergraduate numerical analysis class (MACM 316 at Simon Fraser University) was the bisection method. It's very intuitive and easy to implement in any programming language (I was using MATLAB at the time). The bisection method can be easily adapted for optimizing 1-dimensional functions with a slight but intuitive modification. As there are numerous books and web sites on the bisection method, I will not dwell on it in my blog post.

Instead, I will explain a clever and elegant way to modify the bisection method with the golden ratio that results in faster computation; I learned this method while reading "A First Course in Statistical Programming with R" by John Braun and Duncan Murdoch. Using a script in R to implement this special algorithm, I will illustrate how to minimize a non-differentiable function with the golden section search method. In a later post (for the sake of brevity), I will use the same method to show that the minimizer of the sum of the absolute deviations from a univariate data set is the median. The R functions and script for doing everything are in another post as shown in figure 1.

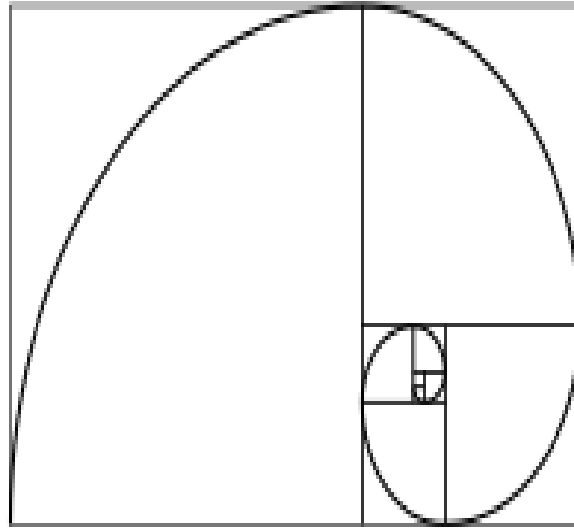


Figure 1. Search Space

The Fibonacci spiral approximates the golden spiral, a logarithmic spiral whose growth factor is the golden ratio. Source: Dicklyon , Minimization with the Bisection Method

Assume that a single-variable continuous function has a unique minimum (and, thus, a unique minimizer) in a closed interval $[a, b]$. If this function is differentiable in $[a, b]$, then calculus can be used easily to find the minimizer. However, if the function has a cusp or a kink, then it's not differentiable at that point, and numerical methods are needed instead. For example, consider the following function as shown in figure 2.

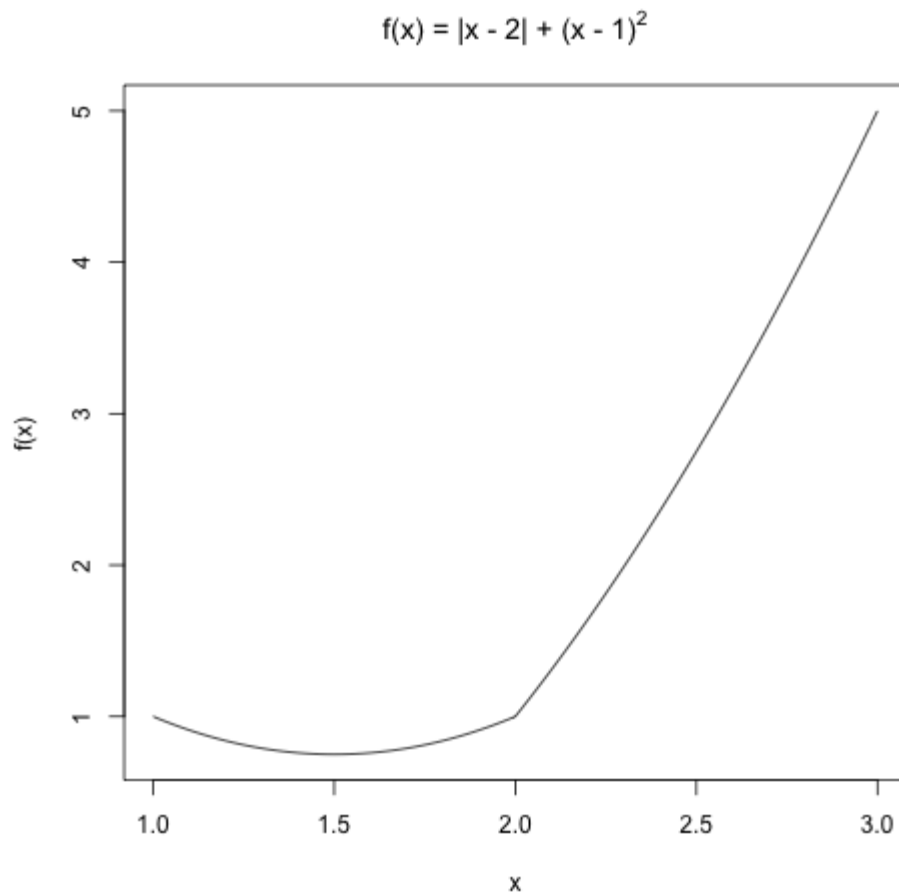


Figure 2. Golden Section Search Space

Within the interval $[1, 3]$, this function has a unique minimizer near $x = 1.5$, but it also has a cusp at $x = 2$. Thus, it is not differentiable at $x = 2$. A simple, stable, but slow numerical method to find the minimizer is the bisection method.

To use the bisection method to find the minimizer, we can

- 1) Pick 2 points, x_1 and x_2 , with $x_1 < x_2$ and both points in $[a, b]$.
- 2) Find $f(x_1)$ and $f(x_2)$.
- 3) a) If $f(x_1) > f(x_2)$, then the minimizer must be to the right of x_1 , because this unique minimizer must be less than $f(x_1)$, and the function is decreasing from x_1 to x_2 .
- b) Otherwise (i.e. $f(x_1) < f(x_2)$), the minimizer must be to the left of x_2 with logically analogous reasoning.
- 4) i) If 3a is true, then the minimizer must be in the interval $[x_1, b]$; repeat the algorithm from Step 1 until the interval length is smaller than some pre-set tolerance.
- ii) If 3b is true, then the minimizer must be in the interval $[a, x_2]$; repeat the algorithm from Step 1 until the interval length is smaller than some pre-set tolerance.

To summarize this algorithm, every iteration begins with

- the boundaries of the interval $[a_j, b_j]$,
- 2 test points for the argument, x_1 and x_2 .

The function is evaluated at x_1 and x_2 , and, based on whether $f(x_1) > f(x_2)$ or $f(x_1) < f(x_2)$, a new set of boundaries and test points are set for the next iteration.

Using the Golden Ratio to Set the Test Points

Since the test points x_1 and x_2 are arbitrarily set, it comes as no surprise that the choice of the test points affects the speed of the computation. A clever way to save computation time is to set the test points to take advantage of some special properties of the golden ratio, which Braun and Murdoch denote with ϕ and has the value

$$(1 + \sqrt{5}) \div 2$$

This number has many special properties that you can easily find on the web. The one that will save computation time is

$$\phi^{-2} = 1 - \phi^{-1}$$

Let the lower and upper bounds of the interval of interest be a and b , respectively. Now, let's set the test points as

$$x_1 = b - (b - a) \div \phi$$

$$x_2 = a + (b - a) \div \phi$$

The advantage of setting the test points as these above values comes when one of the new test points is updated. Suppose that

$$f(x_1) > f(x_2).$$

Then the minimizer must be to the right of x_1 . Thus, x_1 becomes the new lower bound, which I will denote as a' . The beauty of the golden ratio comes in calculating the new lower test point, which I will denote as x'_1 .

$$x'_1 = b - (b - a') \div \phi$$

Since $f(x_1) > f(x_2)$, x_1 becomes the new lower bound.

$$x'_1 = b - (b - x_1) \div \phi$$

Recall that

$$x_1 = b - (b - a) \div \phi$$

Substituting the right-hand side of this above equation for x_1 in the calculation of x'_1 , some very simple algebraic manipulation yields

$$x'_1 = b - (b - a) \div \phi^2$$

Now, taking advantage of that special property of the golden ratio, $\phi^{-2} = 1 - \phi^{-1}$, some more simple algebraic manipulation yields

$$x'_1 = a + (b - a) \div \phi = x_2$$

Thus, the new lower test point is just the old upper test point. This saves computation time, because we don't need to compute a new lower test point; we can just recycle the value that we knew from the old upper test point!

Similar logic for the case $f(x_1) < f(x_2)$ will show that the new upper test point is simply the old lower test point. Notationally, if

$$f(x_1) < f(x_2),$$

then

$$x'_2 = x_1.$$

Using the Script to Numerically Minimize a Function

In another post, you can find the function for implementing this special golden search method and the script for minimizing the above function with the cusp. This script is basically the same as the one written by Braun and Murdoch on Pages 134-135 in their book. My script used slightly more self-evident variable names and included debugging statements to help me to identify why my function was not working properly when I first wrote it. Debugging statements are statements that show the values of key variables being computed in the flow of a script (and often within loops or branches) to identify the point in the flow at which the script stopped working properly. My mathematical and statistical programming skills became much better when I began using debugging statements. They are good for debugging, but should generally be commented out when the script is corrected and working properly. Since the problems that I am solving are quite simple, I kept them in the script and will show part of the output to illustrate why they are useful.

Recall the function with a cusp that I plotted above.

$$f(x) = |x - 2| + (x - 1)^2$$

Using my function, the initial boundaries 1 and 3, and an absolute tolerance of 10^{-5} , here is what my function, `golden.section.search()`, returned in the first iteration. `> golden.section.search(f, 1, 3, 1e-5)`

Iteration # 1

`f1 = 0.8196601`

`f2 = 1.763932`

`f2 > f1`

`New Upper Bound = 2.236068`

`New Lower Bound = 1`

`New Upper Test Point = 1.763932`

`New Lower Test Point = 1.472136`

This function used 26 iterations to get the final answer.

Iteration # 26

$f1 = 0.75$

$f2 = 0.75$

$f2 > f1$

New Upper Bound = 1.500003

New Lower Bound = 1.499995

New Upper Test Point = 1.5

New Lower Test Point = 1.499998

Final Lower Bound = 1.499995

Final Upper Bound = 1.500003

Estimated Minimizer = 1.499999

3. Fibonacci Search Technique

In computer science, the Fibonacci search technique is a method of searching a sorted array using a divide and conquer algorithm that narrows down possible locations with the aid of Fibonacci numbers. Compared to binary search where the sorted array is divided into two equal-sized parts, one of which is examined further, Fibonacci search divides the array into two parts that have sizes that are consecutive Fibonacci numbers. On average, this leads to about 4% more comparisons to be executed, but it has the advantage that one only needs addition and subtraction to calculate the indices of the accessed array elements, while classical binary search needs bit-shift, division or multiplication, operations that were less common at the time Fibonacci search was first published. Fibonacci search has an average- and worst-case complexity of $O(\log n)$.

The Fibonacci sequence has the property that a number is the sum of its two predecessors. Therefore the sequence can be computed by repeated addition. The ratio of two consecutive numbers approaches the Golden ratio, 1.618... Binary search works by dividing the seek area in equal parts (1:1). Fibonacci search can divide it into parts approaching 1:1.618 while using the simpler operations.

If the elements being searched have non-uniform access memory storage (i. e., the time needed to access a storage location varies depending on the location accessed), the Fibonacci search may have the advantage over binary search in slightly reducing the average time needed to access a storage location. If the machine executing the search has a direct mapped CPU cache, binary search may lead to more cache misses because the elements that are accessed often tend to gather in only a few

cache lines; this is mitigated by splitting the array in parts that do not tend to be powers of two. If the data is stored on a magnetic tape where seek time depends on the current head position, a tradeoff between longer seek time and more comparisons may lead to a search algorithm that is skewed similarly to Fibonacci search.

Fibonacci search is derived from Golden section search, an algorithm by Jack Kiefer (1953) to search for the maximum or minimum of a uni modal function in an interval.

4. Algorithm

Let k be defined as an element in F , the array of Fibonacci numbers. $n = F_m$ is the array size. If n is not a Fibonacci number, let F_m be the smallest number in F that is greater than n .

The array of Fibonacci numbers is defined where $F_{k+2} = F_{k+1} + F_k$, when $k \geq 0$, $F_1 = 1$, and $F_0 = 0$.

To test whether an item is in the list of ordered numbers, follow these steps:

Set $k = m$.

If $k = 0$, stop. There is no match; the item is not in the array.

Compare the item against element in F_{k-1} .

If the item matches, stop.

If the item is less than entry F_{k-1} , discard the elements from positions $F_{k-1} + 1$ to n . Set $k = k - 1$ and return to step 2.

If the item is greater than entry F_{k-1} , discard the elements from positions 1 to F_{k-1} . Renumber the remaining elements from 1 to F_{k-2} , set $k = k - 2$, and return to step 2.

Alternative implementation (from "Sorting and Searching"):

Given a table of records R_1, R_2, \dots, R_N whose keys are in increasing order $K_1 < K_2 < \dots < K_N$, the algorithm searches for a given argument K . Assume $N+1 = F_{k+1}$

Step 1. [Initialize]

$i \leftarrow F_k$, $p \leftarrow F_{k-1}$, $q \leftarrow F_{k-2}$ (throughout the algorithm, p and q will be consecutive Fibonacci numbers)

Step 2. [Compare]

If $K < K_i$, go to Step 3; if $K > K_i$ go to Step 4; and if $K = K_i$, the algorithm terminates successfully.

Step 3. [Decrease i]

If $q=0$, the algorithm terminates unsuccessfully. Otherwise set $(i, p, q) \leftarrow (i - q, q, p - q)$ (which moves p and q one position back in the Fibonacci sequence); then return to Step 2

Step 4. [Increase i]

If $p=1$, the algorithm terminates unsuccessfully. Otherwise set $(i, p, q) \leftarrow (i + q, p - q, 2q - p)$ (which moves p and q two positions back in the Fibonacci sequence); and return to Step 2

The two variants of the algorithm presented above always divide the current interval into a larger and a smaller subinterval. The original algorithm, however, would divide the new interval into a smaller and a larger subinterval in Step 4. This has the advantage that the new i is closer to the old i and is more suitable for accelerating search

5. Bi Section Method

The method is also called the interval halving method, the binary search method or the dichotomy method. This method is used to find root of an equation in a given interval that is value of 'x' for which $f(x) = 0$.

The method is based on The Intermediate Value Theorem which states that if $f(x)$ is a continuous function and there are two real numbers a and b such that $f(a) \cdot f(b) < 0$ and $f(b) < 0$, then it is guaranteed that it has at least one root between them.

Steps:

Find middle point $c = (a + b)/2$.

If $f(c) == 0$, then c is the root of the solution.

Else $f(c) \neq 0$

If value $f(a) \cdot f(c) < 0$ then root lies between a and c . So we recur for a and c

Else If $f(b) \cdot f(c) < 0$ then root lies between b and c . So we recur b and c .

Else given function doesn't follow one of assumptions.

Since root may be a floating point number, we repeat above steps while difference between a and b is less than a value (A very small value) as shown in figure 3.

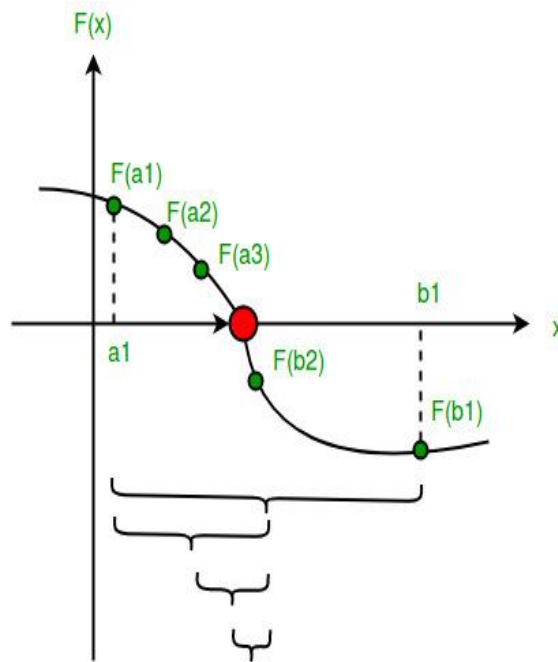


Figure 3. Bi Section

In mathematics, the bisection method is a root-finding method that applies to any continuous functions for which one knows two values with opposite signs. The method consists of repeatedly bisecting the interval defined by these values and then selecting the subinterval in which the function changes sign, and therefore must contain a root. It is a very simple and robust method, but it is also relatively slow. Because of this, it is often used to obtain a rough approximation to a solution which is then used as a starting point for more rapidly converging methods. The method is also called the interval halving method, the binary search method, or the dichotomy method. For polynomials, more elaborated methods exist for testing the existence of a root in an interval (Descartes' rule of signs, Sturm's theorem, Budan's theorem. They allow extending bisection method into efficient algorithms for finding all real roots of a polynomial; see Real-root isolation. The method is applicable for numerically solving the equation $f(x) = 0$ for the real variable x , where f is a continuous function defined on an interval $[a, b]$ and where $f(a)$ and $f(b)$ have opposite signs. In this case a and b are said to bracket a root since, by the intermediate value theorem, the continuous function f must have at least one root in the interval (a, b) . At each step the method divides the interval in two by computing the midpoint $c = (a+b) / 2$ of the interval and the value of the function $f(c)$ at that point. Unless c is itself a root (which is very unlikely, but possible) there are now only two possibilities: either $f(a)$ and $f(c)$ have opposite signs and bracket a root, or $f(c)$ and $f(b)$ have opposite signs and bracket a root. The method selects the subinterval that is guaranteed to be a bracket as the new interval to be used in the next step. In this way an interval that contains a zero of f is reduced in width by 50% at each step. The process is continued until the interval is sufficiently small. Explicitly, if $f(a)$ and $f(c)$ have opposite signs, then the method sets c as the new value for b , and if $f(b)$ and $f(c)$

have opposite signs then the method sets c as the new a . (If $f(c)=0$ then c may be taken as the solution and the process stops.) In both cases, the new $f(a)$ and $f(b)$ have opposite signs, so the method is applicable to this smaller interval.

6. Exhaustive Method

For discrete problems in which no efficient solution method is known, it might be necessary to test each possibility sequentially in order to determine if it is the solution. Such exhaustive examination of all possibilities is known as exhaustive search, direct search, or the "brute force" method. Unless it turns out that NP-problems are equivalent to P-problems, which seems unlikely but has not yet been proved, NP-problems can only be solved by exhaustive search in the worst case.



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

UNIT – III – Advanced Optimization Methods – SPRA5301

1. Decision Tree Analysis

A Decision Tree Analysis is a graphic representation of various alternative solutions that are available to solve a problem. The manner of illustrating often proves to be decisive when making a choice. A Decision Tree Analysis is created by answering a number of questions that are continued after each affirmative or negative answer until a final choice can be made. A Decision Tree Analysis is a scientific model and is often used in the decision making process of organizations. When making a decision, the management already envisages alternative ideas and solutions. By using a decision tree, the alternative solutions and possible choices are illustrated graphically as a result of which it becomes easier to make a well-informed choice. This graphic representation is characterized by a tree-like structure in which the problems in decision making can be seen in the form of a flowchart, each with branches for alternative choices.

The Decision Tree Analysis makes good use of the ‘what if’ thought. There are several alternatives that consider both the possible risks and benefits that are brought about by certain choices. The possible alternatives are also made clearly visible and therefore the decision tree provides clarity with respect to the consequences of any decisions that will be made.

Representation

There are several ways in which a decision tree can be represented. This Analysis is commonly represented by lines, squares and circles. The squares represent decisions, the lines represent consequences and the circles represent uncertain outcomes. By keeping the lines as far apart as possible, there will be plenty of space to add new considerations and ideas.

The representation of the decision tree can be created in four steps:

Describe the decision that needs to be made in the square.

Draw various lines from the square and write possible solutions on each of the lines.

Put the outcome of the solution at the end of the line. Uncertain or unclear decisions are put in a circle. When a solution leads to a new decision, the latter can be put in a new square.

Each of the squares and circles are reviewed critically so that a final choice can be made as shown in figure 1 and 2.

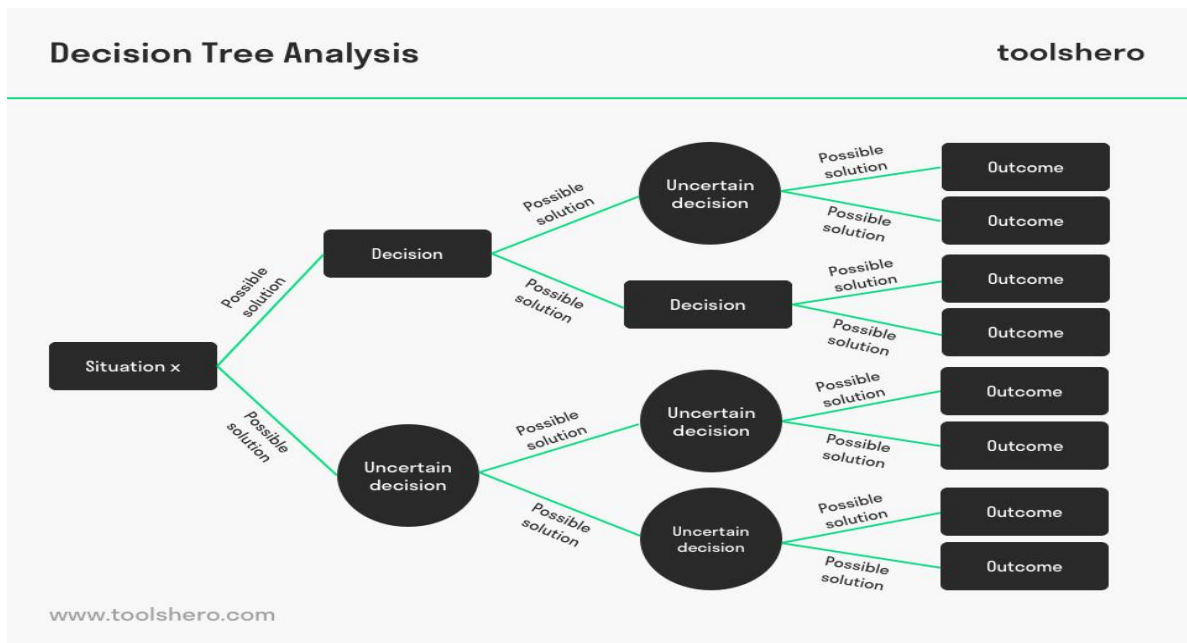


Figure 1. Decision Tree Model 1

2. Decision Tree Analysis example

Suppose a commercial company wishes to increase its sales and the associated profits in the next year.

The different alternatives can then be mapped out by using a decision tree. There are two choice for both increase of sales and profits: 1- expansion of advertising expenditure and 2- expansion of sales activities. This creates two branches. Two new choices arise from choice 1, namely 1-1 a new advertising agency and 1-2 using the services of the existing advertising agency. Choice 2 presents two follow-up choices in turn; 2-1-working with agents or 2-2- using its own sales force.

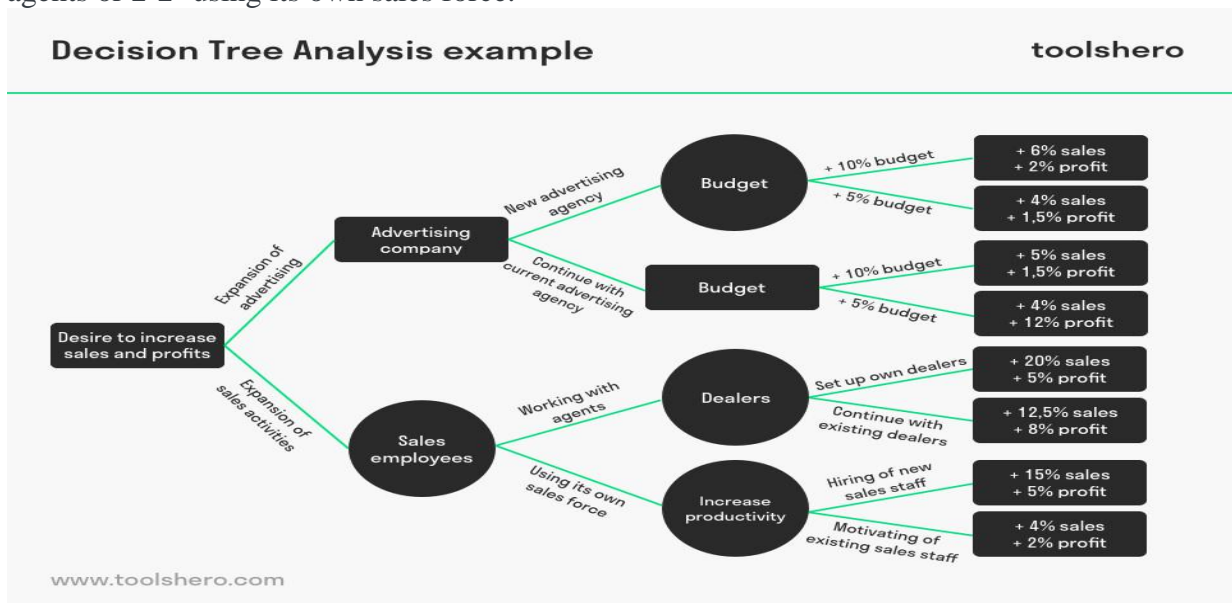


Figure 2. Decision Tree Model 2

The above example illustrates that, in all likelihood, the company will opt for 1-2-2, because the forecast of this decision is that profits will increase by 12%. This Analysis is particularly useful in situations in which it is considered desirable to develop various alternatives of decisions in a structured manner as this will present a clear substantiation. This method is increasingly used by medical practitioners and technicians as it enables them to make a diagnosis or determine car problems.

3. Utility Theory

Utility theory bases its beliefs upon individuals' preferences. It is a theory postulated in economics to explain behavior of individuals based on the premise people can consistently rank order their choices depending upon their preferences. Each individual will show different preferences, which appear to be hard-wired within each individual. We can thus state that individuals' preferences are intrinsic. Any theory, which proposes to capture preferences, is, by necessity, abstraction based on certain assumptions. Utility theory is a positive theory that seeks to explain the individuals' observed behavior and choices. The distinction between normative and positive aspects of a theory is very important in the discipline of economics. Some people argue that economic theories should be normative, which means they should be prescriptive and tell people what to do. Others argue, often successfully, that economic theories are designed to be explanations of observed behavior of agents in the market, hence positive in that sense. This contrasts with a normative theory, one that dictates that people should behave in the manner prescribed by it. Instead, it is only since the theory itself is positive, after observing the choices that individuals make, we can draw inferences about their preferences. When we place certain restrictions on those preferences, we can represent them analytically using a utility function—a mathematical formulation that ranks the preferences of the individual in terms of satisfaction different consumption bundles provide. Thus, under the assumptions of utility theory, we can assume that people behaved as if they had a utility function and acted according to it. Therefore, the fact that a person does not know his/her utility function, or even denies its existence, does not contradict the theory. Economists have used experiments to decipher individuals' utility functions and the behavior that underlies individuals' utility.

To begin, assume that an individual faces a set of consumption “bundles.” We assume that individuals have clear preferences that enable them to “rank order” all bundles based on desirability, that is, the level of satisfaction each bundle shall provide to each individual. This rank ordering based on preferences tells us the theory itself has ordinal utility—it is designed to study relative satisfaction levels. As we noted earlier, absolute satisfaction depends upon conditions; thus, the theory by default cannot have cardinal utility, or utility that can represent the absolute level of satisfaction. To make this theory concrete, imagine that consumption bundles comprise food and clothing for a week in all different

combinations, that is, food for half a week, clothing for half a week, and all other possible combinations.

4. The utility theory then makes the following assumptions:

Completeness: Individuals can rank order all possible bundles. Rank ordering implies that the theory assumes that, no matter how many combinations of consumption bundles are placed in front of the individual, each individual can always rank them in some order based on preferences. This, in turn, means that individuals can somehow compare any bundle with any other bundle and rank them in order of the satisfaction each bundle provides. So in our example, half a week of food and clothing can be compared to one week of food alone, one week of clothing alone, or any such combination. Mathematically, this property wherein an individual's preferences enable him or her to compare any given bundle with any other bundle is called the completeness property of preferences.

More-is-better: Assume an individual prefers consumption of bundle A of goods to bundle B. Then he is offered another bundle, which contains more of everything in bundle A, that is, the new bundle is represented by αA where $\alpha = 1$. The more-is-better assumption says that individuals prefer αA to A, which in turn is preferred to B, but also A itself. For our example, if one week of food is preferred to one week of clothing, then two weeks of food is a preferred package to one week of food. Mathematically, the more-is-better assumption is called the monotonicity assumption on preferences. One can always argue that this assumption breaks down frequently. It is not difficult to imagine that a person whose stomach is full would turn down additional food. However, this situation is easily resolved. Suppose the individual is given the option of disposing of the additional food to another person or charity of his or her choice. In this case, the person will still prefer more food even if he or she has eaten enough. Thus under the monotonicity assumption, a hidden property allows costless disposal of excess quantities of any bundle.

Mix-is-better: Suppose an individual is indifferent to the choice between one week of clothing alone and one week of food. Thus, either choice by itself is not preferred over the other. The "mix-is-better" assumption about preferences says that a mix of the two, say half-week of food mixed with half-week of clothing, will be preferred to both stand-alone choices. Thus, a glass of milk mixed with Milo (Nestlé's drink mix), will be preferred to milk or Milo alone. The mix-is-better assumption is called the "convexity" assumption on preferences, that is, preferences are convex.

Rationality: This is the most important and controversial assumption that underlies all of utility theory. Under the assumption of rationality, individuals' preferences avoid any kind of circularity; that is, if bundle A is preferred to B, and bundle B is preferred to C, then A is

also preferred to C. Under no circumstances will the individual prefer C to A. You can likely see why this assumption is controversial. It assumes that the innate preferences (rank orderings of bundles of goods) are fixed, regardless of the context and time.

If one thinks of preference orderings as comparative relationships, then it becomes simpler to construct examples where this assumption is violated. So, in “beats”—as in A beat B in college football. These are relationships that are easy to see. For example, if University of Florida beats Ohio State, and Ohio State beats Georgia Tech, it does not mean that Florida beats Georgia Tech. Despite the restrictive nature of the assumption, it is a critical one. In mathematics, it is called the assumption of transitivity of preferences.

Whenever these four assumptions are satisfied, then the preferences of the individual can be represented by a well-behaved utility function. The assumption of convexity of preferences is not required for a utility function representation of an individual’s preferences to exist. But it is necessary if we want that function to be well behaved. Note that the assumptions lead to “a” function, not “the” function. Therefore, the way that individuals represent preferences under a particular utility function may not be unique. Well-behaved utility functions explain why any comparison of individual people’s utility functions may be a futile exercise (and the notion of cardinal utility misleading). Nonetheless, utility functions are valuable tools for representing the preferences of an individual, provided the four assumptions stated above are satisfied. For the remainder of the chapter we will assume that preferences of any individual can always be represented by a well-behaved utility function. As we mentioned earlier, well-behaved utility depends upon the amount of wealth the person owns. Utility theory rests upon the idea that people behave as if they make decisions by assigning imaginary utility values to the original monetary values. The decision maker sees different levels of monetary values, translates these values into different, hypothetical terms (“utils”), processes the decision in utility terms (not in wealth terms), and translates the result back to monetary terms. So while we observe inputs to and results of the decision in monetary terms, the decision itself is made in utility terms. And given that utility denotes levels of satisfaction, individuals behave as if they maximize the utility, not the level of observed dollar amounts.

The analytic hierarchy process (AHP) is a structured technique for organizing and analyzing complex decisions, based on mathematics and psychology. It was developed by Thomas L. Saaty in the 1970s who partnered with Ernest Forman to develop Expert Choice in 1983, and has been extensively studied and refined since then. It represents an accurate approach for quantifying the weights of decision criteria. Individual experts’ experiences are utilized to estimate the relative magnitudes of factors through pair-wise comparisons. Each of the respondents has to compare the relative importance between the

two items under special designed questionnaire (note that while most of the surveys adopted the five point likert scale, AHP's questionnaire is 9 to 1 to 9, see Li et al. (2019))

5. Analytic Hierarchy Process

Analytic Hierarchy Process (AHP) is one of Multi Criteria decision making method that was originally developed by Prof. Thomas L. Saaty. In short, it is a method to derive ratio scales from paired comparisons. The input can be obtained from actual measurement such as price, weight etc., or from subjective opinion such as satisfaction feelings and preference. AHP allow some small inconsistency in judgment because human is not always consistent. The ratio scales are derived from the principal Eigen vectors and the consistency index is derived from the principal Eigen value.

Now let me explain what paired comparison is. It is always easier to explain by an example. Suppose we have two fruits Apple and Banana. I would like to ask you, which fruit you like better than the other and how much you like it in comparison with the other. Let us make a relative scale to measure how much you like the fruit on the left (Apple) compared to the fruit on the right (Banana) as shown figure 3.

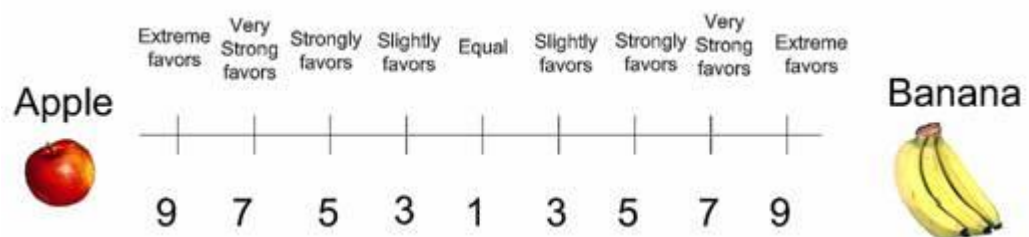


Figure 3. a Paired Comparison Representation

If you like the apple better than banana, you thick a mark between number 1 and 9 on left side, while if you favor banana more than apple, then you mark on the right side.

For instance I strongly favor banana to apple then I give mark like this

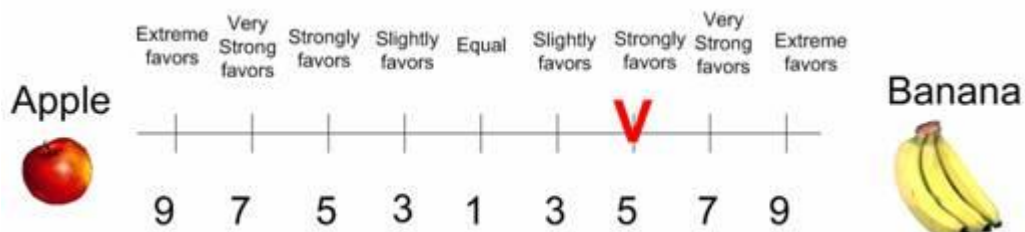


Figure 3. b Paired Comparison Representation

Now suppose you have three choices of fruits. Then the pair wise comparison goes as the following

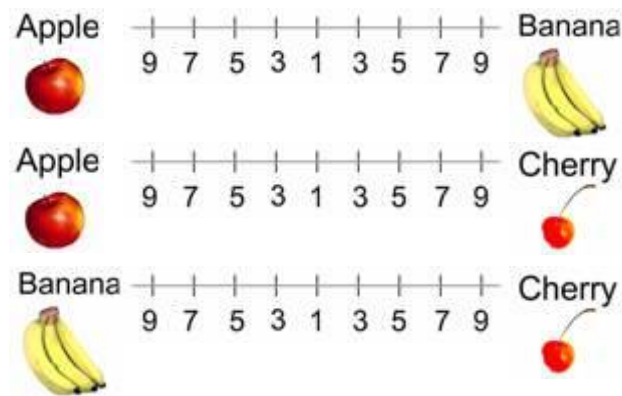


Figure 3. c Paired Comparison Representation

You may observe that the number of comparisons is a combination of the number of things to be compared. Since we have 3 objects (Apple, Banana and Cheery), we have 3 comparisons. Table below shows the number of comparisons.

Number of comparisons

Number of things	1	2	3	4	5	6	7	n
number of comparisons	0	1	3	6	10	15	21	$\frac{n(n-1)}{2}$

The scaling is not necessary 1 to 9 but for qualitative data such as preference, ranking and subjective opinions, it is suggested to use scale 1 to 9.

In the next section you will learn how to analyze this paired comparisons

Making Comparison Matrix (How to make reciprocal matrix?)

By now you know how to make paired comparisons. In this section you will learn how to make a reciprocal matrix from pair wise comparisons.

For example John has 3 kinds of fruits to be compared and he made subjective judgment on which fruit he likes best, like the following

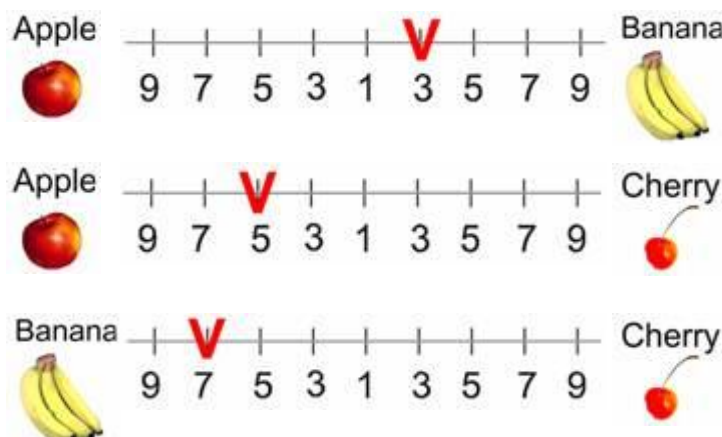


Figure 3. d Paired Comparison Representation

We can make a matrix from the 3 comparisons above. Because we have three comparisons, thus we have 3 by 3 matrix. The diagonal elements of the matrix are always 1 and we only need to fill up the upper triangular matrix. How to fill up the upper triangular matrix is using the following rules:

If the judgment value is on the left side of 1, we put the actual judgment value.

If the judgment value is on the right side of 1, we put the reciprocal value.

Comparing apple and banana, John slightly favor banana, thus we put $\frac{1}{3}$ in the row 1 column 2 of the matrix. Comparing Apple and Cherry, John strongly likes apple, thus we put actual judgment 5 on the first row, last column of the matrix. Comparing banana and cherry, banana is dominant. Thus we put his actual judgment on the second row, last column of the matrix. Then based on his preference values above, we have a reciprocal matrix like this

$$A = \begin{matrix} & \begin{matrix} \text{apple} & \text{banana} & \text{cherry} \end{matrix} \\ \begin{matrix} \text{apple} \\ \text{banana} \\ \text{cherry} \end{matrix} & \begin{bmatrix} 1 & \frac{1}{3} & 5 \\ & 1 & 7 \\ & & 1 \end{bmatrix} \end{matrix}$$

To fill the lower triangular matrix, we use the reciprocal values of the upper diagonal. If a_{ij} is the element of row i column j of the matrix, then the lower diagonal is filled using this formula

$$a_{ji} = \frac{1}{a_{ij}}$$

Thus now we have complete comparison matrix

$$A = \begin{matrix} & \begin{matrix} \text{apple} & \text{banana} & \text{cherry} \end{matrix} \\ \begin{matrix} \text{apple} \\ \text{banana} \\ \text{cherry} \end{matrix} & \begin{bmatrix} 1 & \frac{1}{3} & 5 \\ 3 & 1 & 7 \\ \frac{1}{5} & \frac{1}{7} & 1 \end{bmatrix} \end{matrix}$$

Notice that all the element in the comparison matrix are positive, or $a_{ij} > 0$.

Next section will discuss about how you will use this matrix.

6. Priority Vectors

Having a comparison matrix, now we would like to compute priority vector, which is the normalized Eigen vector of the matrix. If you would like to know what the meaning of Eigen vector and Eigen value is and how to compute them manually, go to my other tutorial and then return back here. The method that I am going to explain in this section is only an approximation of Eigen vector (and Eigen value) of a reciprocal matrix. This approximation is actually worked well for small matrix $n \leq 3$ and there is no guarantee that the rank will not

reverse because of the approximation error. Nevertheless it is easy to compute because all we need to do is just to normalize each column of the matrix. At the end I will show the error of this approximation.

Suppose we have 3 by 3 reciprocal matrix from paired comparison

$$A = \begin{matrix} & \begin{matrix} \text{apple} & \text{banana} & \text{cerry} \end{matrix} \\ \begin{matrix} \text{apple} \\ \text{banana} \\ \text{cerry} \end{matrix} & \begin{bmatrix} 1 & \frac{1}{3} & 5 \\ 3 & 1 & 7 \\ \frac{1}{5} & \frac{1}{7} & 1 \end{bmatrix} \end{matrix}$$

We sum each column of the reciprocal matrix to get

$$A = \begin{matrix} & \begin{matrix} \text{apple} & \text{banana} & \text{cerry} \end{matrix} \\ \begin{matrix} \text{apple} \\ \text{banana} \\ \text{cerry} \\ \text{sum} \end{matrix} & \begin{bmatrix} 1 & \frac{1}{3} & 5 \\ 3 & 1 & 7 \\ \frac{1}{5} & \frac{1}{7} & 1 \\ \frac{21}{5} & \frac{31}{21} & 13 \end{bmatrix} \end{matrix}$$

Then we divide each element of the matrix with the sum of its column, we have normalized relative weight. The sum of each column is 1.

$$A = \begin{matrix} & \begin{matrix} \text{apple} & \text{banana} & \text{cerry} \end{matrix} \\ \begin{matrix} \text{apple} \\ \text{banana} \\ \text{cerry} \\ \text{sum} \end{matrix} & \begin{bmatrix} \frac{5}{21} & \frac{7}{31} & \frac{5}{13} \\ \frac{15}{21} & \frac{21}{31} & \frac{7}{13} \\ \frac{1}{21} & \frac{3}{31} & \frac{1}{13} \\ 1 & 1 & 1 \end{bmatrix} \end{matrix}$$

The normalized principal Eigen vector can be obtained by averaging across the rows

$$w = \frac{1}{3} \begin{bmatrix} \frac{5}{21} + \frac{7}{31} + \frac{5}{13} \\ \frac{15}{21} + \frac{21}{31} + \frac{7}{13} \\ \frac{1}{21} + \frac{3}{31} + \frac{1}{13} \end{bmatrix} = \begin{bmatrix} 0.2828 \\ 0.6434 \\ 0.0738 \end{bmatrix}$$

The normalized principal Eigen vector is also called priority vector . Since it is normalized, the sum of all elements in priority vector is 1. The priority vector shows relative weights among the things that we compare. In our example above, Apple is 28.28%, Banana is 64.34% and Cherry is 7.38%. John most preferable fruit is Banana, followed by Apple and Cheery. In this case, we know more than their ranking. In fact, the relative weight is a ratio scale that we can divide among them. For example, we can say that John likes banana 2.27 (=64.34/28.28) times more than apple and he also like banana so much 8.72 (=64.34/7.38) times more than cheery.

Aside from the relative weight, we can also check the consistency of John's answer. To do that, we need what is called Principal Eigen value. Principal Eigen value is obtained from the summation of products between each element of Eigen vector and the sum of columns of the reciprocal matrix.

$$\lambda_{max} = \frac{21}{5} (0.2828) + \frac{31}{21} (0.6434) + 13(0.0738) = 3.0967$$

Computation and the meaning of consistency are explained in the next section.

As a note, I put the comparison matrix into Matlab to see how different is the result of numerical computation of Eigen value and Eigen vector compared to the approximation above.

$$\mathbf{A} = \begin{bmatrix} 1 & \frac{1}{3} & 5 \\ 3 & 1 & 7 \\ \frac{1}{5} & \frac{1}{7} & 1 \end{bmatrix}$$

$$[\mathbf{W}, \boldsymbol{\lambda}] = \text{eig}(\mathbf{A})$$

We get three Eigen vectors concatenated into 3 columns of matrix \mathbf{W}

$$\mathbf{W} = \begin{bmatrix} 0.3928 & -0.1964 + 0.3402i & -0.1964 - 0.3402i \\ 0.9140 & 0.9140 & 0.9140 \\ 0.1013 & -0.0506 - 0.0877i & -0.0506 + 0.0877i \end{bmatrix}$$

The corresponding Eigen values are the diagonal of matrix $\boldsymbol{\lambda}$

$$\boldsymbol{\lambda} = \begin{bmatrix} 3.0649 & 0 & 0 \\ 0 & -0.0324 + 0.4448i & 0 \\ 0 & 0 & -0.0324 - 0.4448i \end{bmatrix}$$

The largest Eigen value is called the Principal Eigen value, that is $\lambda_{\max}^* = 3.0649$ which is very close to our approximation $\lambda_{\max} = 3.0967$ (about 1% error). The principal Eigen vector is the Eigen vector that corresponds to the highest Eigen value.

$$\bar{\mathbf{w}} = \begin{bmatrix} 0.3928 \\ 0.9140 \\ 0.1013 \end{bmatrix}$$

The sum is 1.4081 and the normalized principal Eigen vector is

$$\mathbf{w}^* = \begin{bmatrix} 0.2790 \\ 0.6491 \\ 0.0719 \end{bmatrix}$$

This result is also very close to our approximation

$$\mathbf{w} = \begin{bmatrix} 0.2828 \\ 0.6434 \\ 0.0738 \end{bmatrix}$$

Thus the approximation is quite good.

Thus the sum of Eigen vector is not one. When you normalized an Eigen vector, then you get a priority vector. The sum of priority vector is one.

In next section you will learn how to make use of information of principal eigen value to measure whether the opinion is consistent.

Consistency Index and Consistency Ratio

What is the meaning that our opinion is consistent? How do we measure the consistency of subjective judgment? At the end of this section will be able to answer those questions.

Let us look again on John's judgment that we discussed in the previous section. Is John judgment consistent or not?

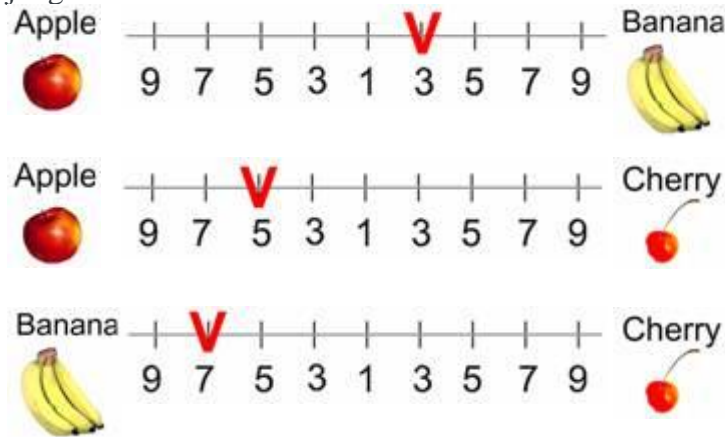


Figure 3. e Paired Comparison Representation

First he prefers Banana to Apple. Thus we say that for John, Banana has greater value than Apple. We write it as $B \succ A$.

Next, he prefers Apple to Cherry. For him, Apple has greater value than Cherry. We write it as $A \succ C$.

Since $B \succ A$ and $A \succ C$, logically, we hope that $B \succ C$ or Banana must be preferable than Cherry. This logic of preference is called transitive property. If John answers in the last comparison is transitive (that he like Banana more than Cherry), then his judgment is consistent. On the contrary, if John prefers Cherry to Banana then his answer is inconsistent. Thus consistency is closely related to the transitive property.

A comparison matrix A is said to be consistent if $a_{ij} a_{jk} = a_{ik}$ for all i, j and k . However, we shall not force the consistency. For example, $B \succ A$ has value $3 \succ 1$ and $A \succ C$ has value $5 \succ 1$, we shall not insist that $B \succ C$ must have value $15 \succ 1$. This too much consistency is undesirable because we are dealing with human judgment. To be called consistent, the rank can be transitive but the values of judgment are not necessarily forced to multiplication formula $a_{ij} a_{jk} = a_{ik}$.

Prof. Saaty proved that for consistent reciprocal matrix, the largest Eigen value is equal to the number of comparisons, or $\lambda_{max} = n$. Then he gave a measure of consistency, called Consistency Index as deviation or degree of consistency using the following formula

$$CI = \frac{\lambda_{\max} - n}{n-1}$$

Thus in our previous example, we have $\lambda_{\max} = 3.0967$ and three comparisons, or $n=3$, thus the consistency index is

$$CI = \frac{\lambda_{\max} - n}{n-1} = \frac{3.0967 - 3}{2} = 0.0484$$

Knowing the Consistency Index, the next question is how do we use this index? Again, Prof. Saaty proposed that we use this index by comparing it with the appropriate one. The appropriate Consistency index is called Random Consistency Index (**RI**).

He randomly generated reciprocal matrix using scale $\frac{1}{9}, \frac{1}{8}, \dots, 1, \dots, 8, 9$ (similar to the idea of Bootstrap) and get the random consistency index to see if it is about 10% or less. The average random consistency index of sample size 500 matrices is shown in the table below

Random Consistency Index (**RI**)

n	1	2	3	4	5	6	7	8	9	10
RI	0	0	0.58	0.9	1.12	1.24	1.32	1.41	1.45	1.49

Then, he proposed what is called Consistency Ratio, which is a comparison between Consistency Index and Random Consistency Index, or in formula

$$CR = \frac{CI}{RI}$$

If the value of Consistency Ratio is smaller or equal to 10%, the inconsistency is acceptable. If the Consistency Ratio is greater than 10%, we need to revise the subjective judgment.

For our previous example, we have **CI** = 0.0484 and **RI** for $n=3$ is 0.58, then we have

$$CR = \frac{CI}{RI} = \frac{0.0484}{0.58} = 8.3\% < 10\%$$

. Thus, John's subjective evaluation about his fruit preference is consistent.

So far, in AHP we are only dealing with paired comparison of criteria or alternative but not both. In next section, I show an example to use both criteria and alternative in two levels of AHP.

7. Illustrative example

In this section, figure 4 shows an example of two levels AHP. The structure of hierarchy in this example can be drawn as the following

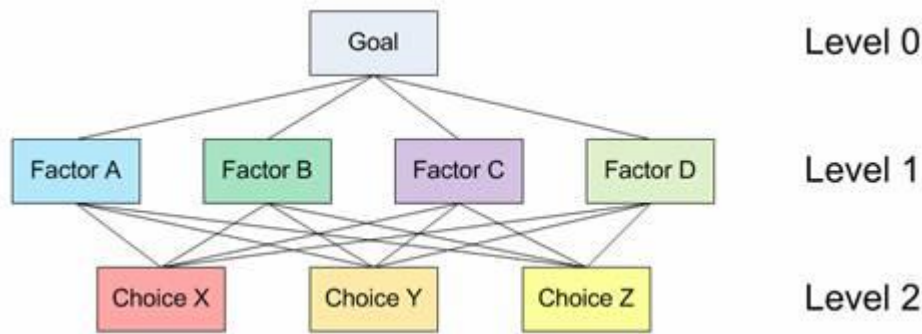


Figure 4 AHP Levels

Level 0 is the goal of the analysis. Level 1 is multi criteria that consist of several factors. You can also add several other levels of sub criteria and sub-sub criteria but I did not use that here. The last level (level 2 in figure above) is the alternative choices. You can see again Table 1 for several examples of Goals, factors and alternative choices. The lines between levels indicate relationship between factors, choices and goal. In level 1 you will have one comparison matrix corresponds to pair-wise comparisons between 4 factors with respect to the goal. Thus, the comparison matrix of level 1 has size of 4 by 4. Because each choice is connected to each factor, and you have 3 choices and 4 factors, then in general you will have 4 comparison matrices at level 2. Each of these matrices has size 3 by 3. However, in this particular example, you will see that some weight of level 2 matrices are too small to contribute to overall decision, thus we can ignore them.

Based on questionnaire survey or your own paired comparison, we make several comparison matrices. Click [here](#) if you do not remember how to make a comparison matrix from paired comparisons. Suppose we have comparison matrix at level 1 as table below. The yellow color cells in upper triangular matrix indicate the parts that you can change in the spreadsheet. The diagonal is always 1 and the lower triangular matrix is filled using

formula
$$a_{ji} = \frac{1}{a_{ij}}$$

Table 9: Paired comparison matrix level 1 with respect to the goal

Criteria	A	B	C	D	Priority Vector
A	1.00	3.00	7.00	9.00	57.39%
B	0.33	1.00	5.00	7.00	29.13%
C	0.14	0.20	1.00	3.00	9.03%
D	0.11	0.14	0.33	1.00	4.45%
Sum	1.59	4.34	13.33	20.00	100.00%

$\lambda_{max} = 4.2692$, $CI = 0.0897$, $CR = 9.97\% < 10\%$ (acceptable)

The priority vector is obtained from normalized Eigen vector of the matrix. Click [here](#) if you do not remember how to compute priority vector and largest Eigen value λ_{max} from a comparison matrix. CI and CR are consistency Index and Consistency ratio respectively, as I have explained in previous section. For your clarity, I include again here some part of the computation:

$$\lambda_{max} = (0.5739)(1.59) + (0.2913)(4.34) + (0.0903)(13.33) + (0.0445)(20) = 4.2692$$

$$CI = \frac{\lambda_{max} - n}{n-1} = \frac{4.2692 - 4}{3} = 0.0897$$

$$CR = \frac{CI}{RI} = \frac{0.0897}{0.90} = 9.97\% < 10\%$$

(Thus, OK because quite consistent)

Random Consistency Index (RI)

Suppose you also have several comparison matrices at level 2. These comparison matrices are made for each choice, with respect to each factor.

Paired comparison matrix level 2 with respect to Factor A

Choice	X	Y	Z	Priority Vector
X	1.00	1.00	7.00	51.05%
Y	1.00	1.00	3.00	38.93%
Z	0.14	0.33	1.00	10.01%
Sum	2.14	2.33	11.00	100.00%

$$\lambda_{max} = 3.104, CI = 0.05, CR = 8.97\% < 10\% \text{ (acceptable)}$$

Paired comparison matrix level 2 with respect to Factor B

Choice	X	Y	Z	Priority Vector
X	1.00	0.20	0.50	11.49%
Y	5.00	1.00	5.00	70.28%
Z	2.00	0.20	1.00	18.22%
Sum	8.00	1.40	6.50	100.00%

$$\lambda_{max} = 3.088, CI = 0.04, CR = 7.58\% < 10\% \text{ (acceptable)}$$

We can do the same for paired comparison with respect to Factor C and D. However, the weight of factor C and D are very small (look at Table 9 again, they are only about 9% and 5% respectively), therefore we can assume the effect of leaving them out from further consideration is negligible. We ignore these two weights as set them as zero. So we do not use the paired comparison matrix level 2 with respect to Factor C and D. In that case, the weight of factor A and B in Table 9 must be adjusted so that the sum still 100%

$$\text{Adjusted weight for factor A} = \frac{57.39\%}{57.39\% + 29.13\%} = 0.663$$

$$\text{Adjusted weight for factor B} = \frac{29.13\%}{57.39\% + 29.13\%} = 0.337$$

Then we compute the overall composite weight of each alternative choice based on the weight of level 1 and level 2. The overall weight is just normalization of linear combination of multiplication between weight and priority vector.

$$X = (0.663)(51.05\%) + (0.337)(11.49\%) = 37.72\%$$

$$Y = (0.663)(38.93\%) + (0.337)(70.28\%) = 49.49\%$$

$$Z = (0.663)(10.01\%) + (0.337)(18.22\%) = 12.78\%$$

Overall composite weight of the alternatives

	Factor A	Factor B	Composite Weight
(Adjusted) Weight	0.663	0.337	
Choice X	51.05%	11.49%	37.72%
Choice Y	38.93%	70.28%	49.49%
Choice Z	10.01%	18.22%	12.78%

For this example, we get the results that choice Y is the best choice, followed by X as the second choice and the worst choice is Z. The composite weights are ratio scale. We can say that choice Y is 3.87 times more preferable than choice Z, and choice Y is 1.3 times more preferable than choice X.

We can also check the overall consistency of hierarchy by summing for all levels, with weighted consistency index (CI) in the nominator and weighted random consistency index (RI) in the denominator. Overall consistency of the hierarchy in our example above is given by

$$\overline{CR} = \frac{\sum_i w_i CI_i}{\sum_i w_i RI_i} = \frac{0.0897(1) + 0.05(0.663) + 0.04(0.337)}{0.90(1) + 0.58(0.663) + 0.58(0.337)} = 0.092 < 10\%$$

(Acceptable)

Final Remark

By now you have learned several introductory methods on multi criteria decision making (MCDM) from simple cross tabulation, using rank, and weighted score until AHP. Using Analytic Hierarchy Process (AHP), you can convert ordinal scale to ratio scale and even check its consistency.



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE

www.sathyabama.ac.in

UNIT – IV – Advanced Optimization Methods – SPRA5301

1. Constrained and Unconstrained Optimization

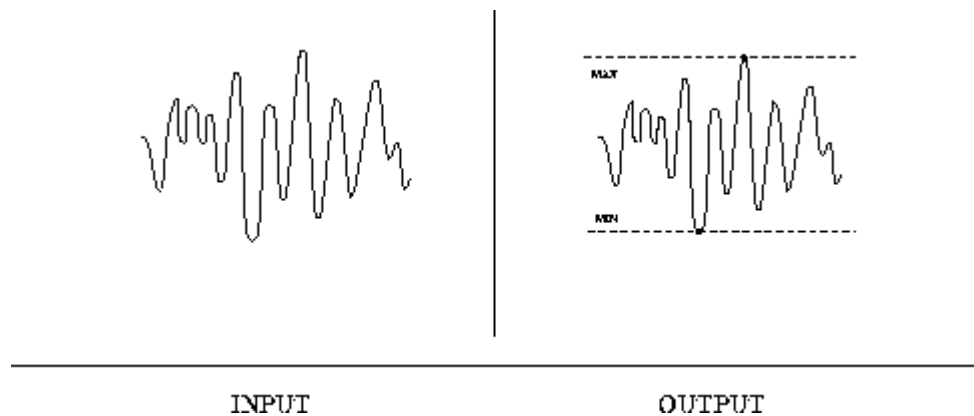


Figure 1. Constrained and Unconstrained Optimization

Input description: A function $f(x_1, \dots, x_n)$.

Problem description: What point $p = (p_1, \dots, p_n)$ maximizes (or minimizes) the function f ?

Discussion: Most of this book concerns algorithms that optimize one thing or another. This section considers the general problem of optimizing functions where, due to lack of structure or knowledge, we are unable to exploit the problem-specific algorithms seen elsewhere in this book.

Optimization arises whenever there is an objective function that must be tuned for optimal performance. Suppose we are building a program to identify good stocks to invest in. We have available certain financial data to analyze, such as the price-earnings ratio, the interest and inflation rates, and the stock price, all as a function of time t as shown in figure 1. The key question is how much weight we should give to each of these factors, where these weights correspond to coefficients of a formula:

$$\text{stock-goodness}(t) = c_1 \times \text{price}(t) + c_2 \times \text{interest}(t) + c_3 \times \text{PE-ratio}(t) + c_4 \times \text{inflation}(t)$$

We seek the numerical values c_1, c_2, c_3, c_4 whose stock-goodness function does the best job of evaluating stocks. Similar issues arise in tuning evaluation functions for game playing programs such as chess.

Unconstrained optimization problems also arise in scientific computation. Physical systems from protein structures to particles naturally seek to minimize their "energy functions." Thus programs that attempt to simulate nature often define energy potential functions for the possible configurations of objects and then take as the ultimate configuration the one that minimizes this potential.

Global optimization problems tend to be hard, and there are lots of ways to go about them. Ask the following questions to steer yourself in the right direction:

Am I doing constrained or unconstrained optimization? - In unconstrained optimization, there are no limitations on the values of the parameters other than that they maximize the value of f . Often, however, there are costs or constraints on these parameters. These constraints make certain points illegal, points that might otherwise be the global optimum. Constrained optimization problems typically require mathematical programming approaches like linear programming, discussed in Section

Is the function I am trying to optimize described by a formula or data? - If the function that you seek to optimize is presented as an algebraic formula (such as the minimum of $f(n) = n^2 - 6n + 2^{n+1}$), the solution is to analytically take its derivative $f'(n)$ and see for which points p' we have $f'(p') = 0$. These points are either local maxima or minima, which can be distinguished by taking a second derivative or just plugging back into f and seeing what happens. Symbolic computation systems such as Mathematica and Maple are fairly effective at computing such derivatives, although using computer algebra systems effectively is somewhat of a black art. They are definitely worth a try, however, and you can always use them to plot a picture of your function to get a better idea of what you are dealing with.

How expensive is it to compute the function at a given point? - If the function f is not presented as a formula, what to do depends upon what is given. Typically, we have a program or subroutine that evaluates f at a given point, and so can request the value of any given point on demand. By calling this function, we can poke around and try to guess the maxima. Our freedom to search in such a situation depends upon how efficiently we can evaluate f . If f is just a complicated formula, evaluation will be very fast. But suppose that f represents the effect of the coefficients x_1, \dots, x_n on the performance of the board evaluation function in a computer chess program, such that x_1 is how much a pawn is worth, x_2 is how much a bishop is worth, and so forth. To evaluate a set of coefficients as a board evaluator, we must play a bunch of games with it or test it on a library of known positions. Clearly, this is time-consuming, so we must be frugal in the number of evaluations of f we use.

How many dimensions do we have? How many do we need? - The difficulty in finding a global maximum increases rapidly with the number of dimensions (or parameters). For this reason, it often pays to reduce the dimension by ignoring some of the parameters. This runs counter to intuition, for the naive programmer is likely to incorporate as many variables as possible into their evaluation function. It is just too hard to tweak such a complicated function. Much better is to start with the 3 to 5 seemingly most important variables and do a

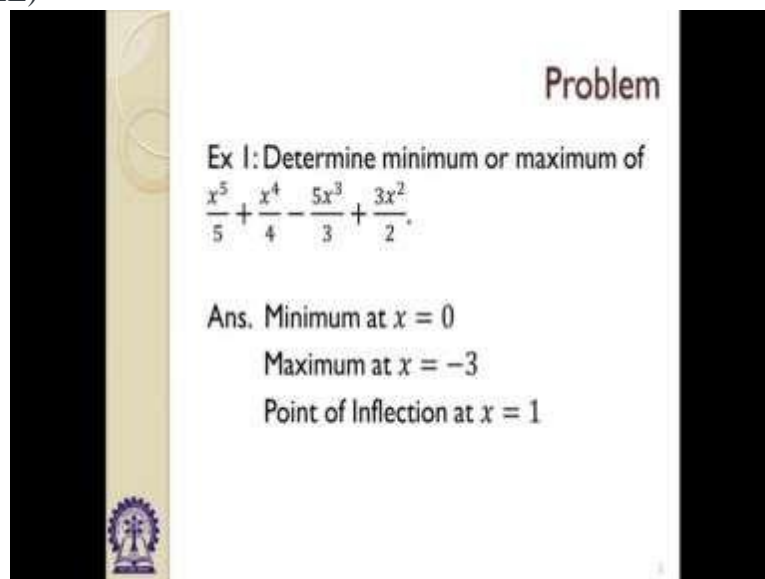
good job optimizing the coefficients for these. Figure 2 represents step by step procedure for problem solving

How smooth is my function? The main difficulty of global optimization is getting trapped in local optima. Consider the problem of finding the highest point in a mountain range. If there is only one mountain and it is nicely shaped, we can find the top by just walking in whatever direction is up. However, if there are many false summits or other mountains in the area, it is difficult to convince ourselves whether we are really at the highest point. Smoothness is the property that enables us to quickly find the local optimum from a given point. We assume smoothness in seeking the peak of the mountain by walking up. If the height at any given point was a completely random function, there would be no way we could find the optimum height short of sampling every single point.

Efficient algorithms for unconstrained global optimization use derivatives and partial derivatives to find local optima, to point out the

Today I will just solve few problems which I explained in the last class regarding the maxima, minima and the saddle point identification of non-linear programming problem. Now again we are concentrating on one dimensional unconstrained non-linear programming problem. Let me take few examples on that.

(Credit to NPTEL)



Problem

Ex I: Determine minimum or maximum of

$$\frac{x^5}{5} + \frac{x^4}{4} - \frac{5x^3}{3} + \frac{3x^2}{2}$$

Ans. Minimum at $x = 0$
Maximum at $x = -3$
Point of Inflection at $x = 1$

Figure 2 a. Problem Solving Steps

This is one of example. The function has been given as $\frac{x^5}{5} + \frac{x^4}{4} - \frac{5x^3}{3} + \frac{3x^2}{2}$. We need to find out the minimum maximum and if there is any saddle point or not.

$$\begin{aligned}
 f'(x) &= x^4 + x^3 - 5x^2 + 3x \\
 &= x(x^3 + x^2 - 5x + 3) \\
 &= x\{x^2(x+3) - 2x(x+3) + (x+3)\} \\
 &= x(x+3)(x-1) \\
 f'(x) &= 0 \quad 0, -3, 1 \\
 f''(x) &= 4x^3 + 3x^2 - 10x + 3 \\
 \text{At } x=0 \quad f''(x) &= 3 > 0 \quad \text{Minimum pt.} \\
 \text{At } x=1 \quad f''(x) &= 0, f'''(x) = 12x^2 + 6x - 10 > 0 \\
 &\quad \text{saddle pt.} \\
 \text{At } x=-3 \quad f''(x) &= -48 < 0 \quad \text{Max.}
 \end{aligned}$$

Credit to NPTEL

Figure 2 b. Problem Solving Steps

Now, if the function $f(x)$ is giving to you, now the process is that now, one thing you have to just see the pattern of the function. First of all, if you can find out the pattern of the function, that would be nice. Now for finding out the pattern we have to judge whether the function has in which part of the interval function has the convexity and which part of the interval function has the concavity. Now once the function has been given us a convex function in certain interval, here there is no restriction on the decision variable that is why this is unrestricted in sign. x can take any value from minus infinity to plus infinity.

Now if you cons, if you just check in which interval function has the convexity. Then we can declare that within that interval there must be certain minimum point. I explained you in the last class regarding this. And if the function has the concave part in the some other region then function must be having the maximum value within that. And if the function is changing from convex to concavity; that means the pattern of the function is just changing in certain point then that must be the saddle point.

Now I am not going into that geometry of the function of these let us try to solve this function first. For doing that thing the necessary conditions suggest such we will find out the first order derivative of the function $f'(x)$. Then we will equate to 0 then we will get the stationary points of these. If we considered the first order a

derivative

$$f'(x) = x^4 + x^3 - 5x^2 + 3x = x(x^3 + x^2 - 5x + 3) = x\{x^2(x+3) - 2x(x+3) + (x+3)\} = x(x+3)(x-1)^2$$

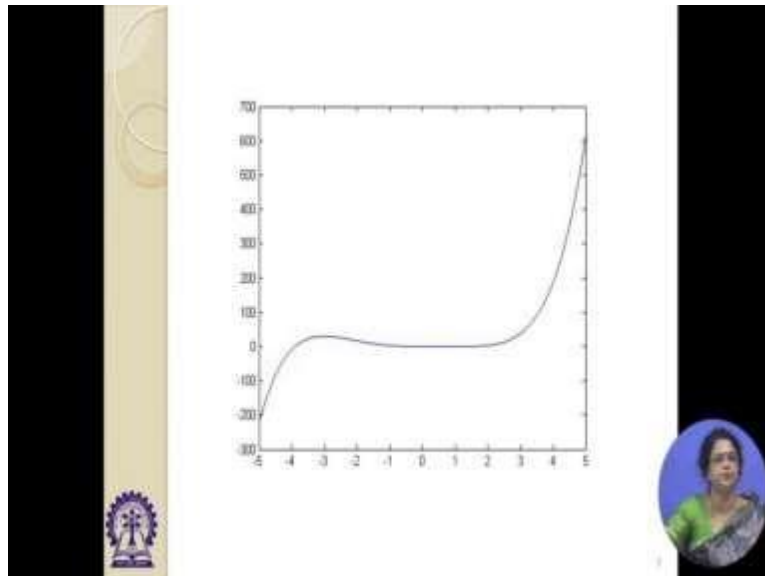
Now, if $f'(x) = 0$, then we must be getting there are three stationary points for this one is 0 another one is -3 and another one is 1. Now for checking the maximum and minimum of the function we will go for the second order derivative of $f(x)$. Then it would be $f''(x) = 4x^3 + 3x^2 - 10x + 3$. Let us see what is happening in individual 3 stationary points. At , we get that $f'(x) = 0$ as well as the $f''(0) = 3 > 0$ that means must be the minimum point all right.

Now, let us go to the next . Then we are getting $f''(1) = 0$ if we consider $f''(1) = 0$. Our sufficient conditions suggest that we will go for third order derivative of x. Now third order derivative is coming as $f'''(x) = 12x^2 + 6x - 10$. At this value is coming $f'''(1) = 8 > 0$. That must be positive since you we see that from the sufficient condition that third order derivative is positive. That is why we cannot we cannot go further we have to declare then these must be a saddle point all right, at must be changing is it is pattern.

Now, let us go for , $f''(-3) = -48 < 0$ that means, at there is a maximum point.

Now, if this is the information we are getting from the necessary and sufficient condition, what we can see from the function that -3, if I just take the interval from -3 to +3 we see at -3 there is a maximum point; that means, the function must be concave. After that it is going to zero; that means we are reaching to the minimum point; that means around 0 that function must be convex. And after that the function is changing it is pattern.

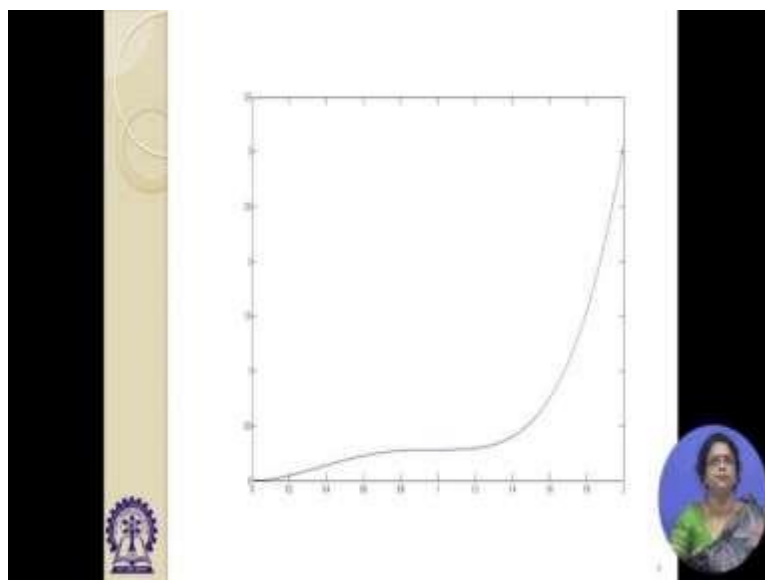
If I just draw it just we see that we will get this function.



(Credit to NPTEL)

Figure 2 c. Problem Solving Steps

Now, what we see here at -3, we are getting the maximum value and at 1, it is changing the pattern and 0 there is a minimum value for the function.



(Credit to NPTEL)

Figure 2 d. Problem Solving Steps

Now, that is why the next level of problem next level of calculations are coming that call the numerical optimization I was mentioning. Now let us take another problem. Now for this problem we are trying to sketch the function.

Solve the Problem:

First then we will go for the maximum minima of the function this is the function for us $f(x) = 2x^6 - 6x^4 + 6x^2 + 10$. Now if I just ask you just tell me how to case this function. One thing we can do that we will first find out the concavity part and the convexity part of this function.

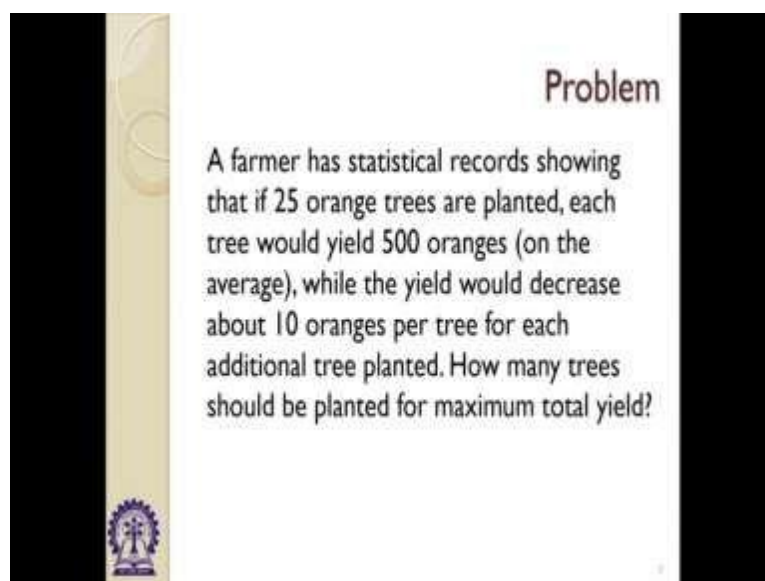
Now, for doing this thing for finding out the convexity or concavity, you know we have to find out the second order derivative of the function. Now if the second order derivative if it is positive, one thing is either convex or concave. What is that? As we know if $f''(x) > 0$; that means, we are getting the convexity of the function. And if $f''(x) < 0$ we are getting the concavity of the function. That is why let us see the value of $f''(x)$. Now the x is again is not persisted. It can move from origin. That is why just to excel I was finding out few values for x and the corresponding $f(x)$ and try to find out the values for $f''(x)$. What we could see here, if we considered the values just you see from -1.2, -1.1, -1, -0.9, if it is moving further at 0 the value is 0, and after it is going to minus then it is going to plus, where it is going the changing it is pattern just look at the excel file you could see the it is changing it pattern at 1. Now we try to draw the picture of it and we could find out this is the picture of $f''(x)$. What we could see in the $f''(x)$. that, after -1 the function value is the $f''(x)$ x is coming negative. After that 0 it is positive. And at 0 we are we will see this is not a function $f(x)$ this is $f''(x)$. That is why must function must be having the concave pattern convex pattern here $f''(x)$ is negative that is why. After the taking we are getting the negative value and after one it is changing from negative to positive. It is not maintaining before one and after one the pattern is not being maintained from convexity to concavity it is going.

That is why I just draw the function after that just you see. We are getting the convex concave part at -3 and at 0 we are getting the convex part of the function.

Again at one it is changing is pattern from minus from the concavity to convexity. That is the thing, now we are applying the classical optimization technique for maxima and minimum. Now if we just look at the function we could see the minimize coming at 0, and maximize coming at -1. And after that we are having the changing pattern at one. That is why the first order derivative. We could see now oh no at -1 also just we could see that it is changing it is pattern, the stationary points are $0, \pm 1$. Now $f''(x)$ is these

and we could see at , if $f''(x)$ is positive that is why it is having the minimum value, all right. And at $x = \pm 1$, $f''(x)$ is coming 0, but $f'''(x)$ is not 0. That is why we can see we can say that seems the aim is here or that is why that must be the point of inflections at point +1 and -1. From the picture of also we can see the same pattern. That is why see if the function is given by judging the convexity concavity part minimum maximum part, we can sketch the function rough sketch we can have of the function all right, but the exact for getting the exact picture of the function we need to study something more than that. That is the asymptote and all other properties, we need to find out then only we can sketch the function better otherwise a rough sketch we will get true this only.

Solve the Problem:



Problem

A farmer has statistical records showing that if 25 orange trees are planted, each tree would yield 500 oranges (on the average), while the yield would decrease about 10 oranges per tree for each additional tree planted. How many trees should be planted for maximum total yield?

Now this is another problem just see. Now a farmer has statistical records showing that if 25 orange trees are planted each tree would yield 500 oranges, but there is information that, if we plant only one tree extra, then 10 oranges per tree will reduce. Now the question is given to you how many tree should be planted for maximum total yield. For normal case we are getting 500 oranges from a tree. Now, but it the next information if we plant one tree extra from each tree 10 oranges will be reduce; that means, we will get from each plant. Now the question is that how many trees should be planted. So, that maximum yield will come.

$$500 - 10$$

Now let us to construct is problem we have to just construct function for solving this. Now let me solve it manually and step by step procedure as shown in figure 3.
(Credit to NPTEL)

$$\begin{aligned}
 &x \rightarrow \text{Number of trees planned in excess of 25} \\
 &(25+x) \rightarrow \text{Total No. of plants} \\
 &\text{Max } (500-10x)(25+x) \\
 &= 12500 + 250x - 10x^2 \\
 &f'(x) = 250 - 20x \\
 &f'(x) = 0 \Rightarrow x = \frac{25}{2} = 12.5 \\
 &f''(x) = -20 < 0 \\
 &\quad \quad \quad \textcircled{x = 12.5}
 \end{aligned}$$

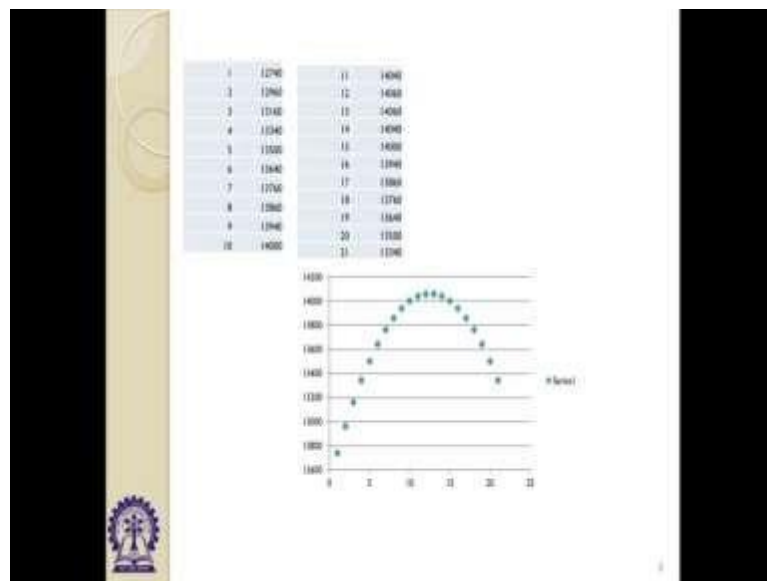
Figure 3 a. Problem Solving Steps

Let me considered the variable x as number of trees the farmer is planting in excess of 25. Because the initially the information has been given that in the normal case is planting 25 plants. That is why in x is we are considering x plants are being planted. If x is equal to 1 then how many how much yield he will get he will get $(500 - 10)(25 + 1)$ because there are 25 plants. Now, if there are x number of plants, then what should be the number of for plants all together it would total number of plants. Now how much yield I will get? If it is normal we will get I will get total yield $(500 - 10x)(25 + x)$. Now I have to maximize this. This is my question, because I have to maximize the total number of yield all right.

Now, if this is a problem for us then just look at the function this function is coming as $12500 + 250x - 10x^2$ all right we have to maximize this function. Now let us apply the classical optimization technique, what we are getting $f'(x) = 250 - 20x$. That is why the stationary points are if I just equate to 0, gives me that $x = \frac{25}{2} = 12.5$ let us find out $f''(x)$ for this. $f''(x) = -20$; That means always it is negative. That is why it is suggested that you plant 12.5 numbers of trees for getting maximum yield.

But you see the problem is such a problem here certainly, there is a restriction on the decision variable x that always x has to be positive. It cannot be negative number of trees cannot be negative. What else you are getting the restriction number of tree should be discrete. That is why this kind of optimization problem is being named as the discrete optimization problem, but you will see if we just find out feasible space the feasible space here we are considering the whole range from x is equal to 0 to 0 , which is totally continuous, but that should not be the always we will have the value for x is 1, 2, 3, 4, 5,... Etcetera integer numbers.

Now, we are getting. That is why this process is not correct process to judge that we are getting the result nice result for it. That is why that is the disadvantage of using the classical optimization technique. Now that is why there is another series of techniques available that is called numerical optimization technique. There is range of variations of the methodologies. Now today I will just discuss one simple methodology for solving through the numerical optimization. And the advantage is that even the space is discrete, we cannot use those methodologies and nicely we can get the solution of it the very well-known method you must have been learned it that is the interval having process.



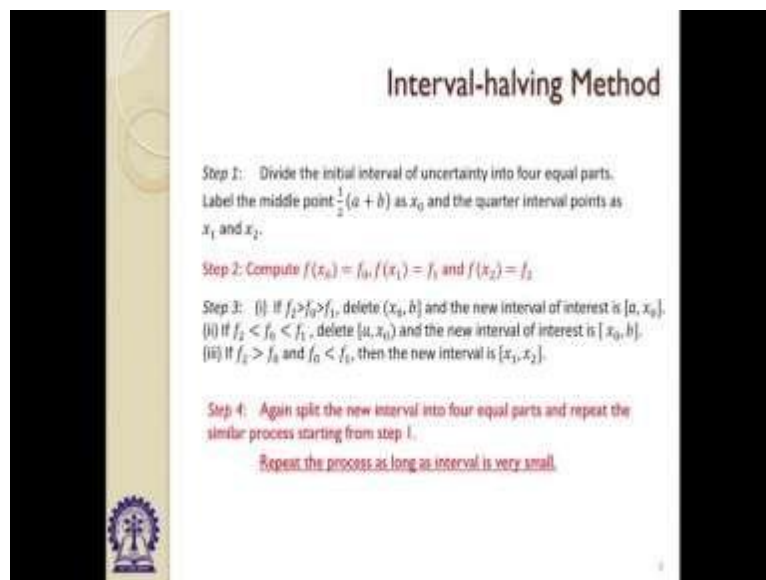
(Credit to NPTEL)

Figure 3 b. Problem Solving Steps

Now, before to that I get certain values I was calculating certain values for affects. And I just draw the picture of it. And certainly we are getting that at we are getting the optimal solution.

That here if we see we are getting the points **0, 1, 2, 3, 4, 5, ... 21** we would calculate. Now one thing is that one thing we can say that within the range of 0 to 21 function is unimodal. That is why whenever we are getting a function it is a better practice for us to just judge the property of the function, let us draw it for if I am having the tool. If I am having any softer for plotting the graph do it otherwise just use excel as I did here. Just simple excel will be sufficient for you. Put the value of x and calculate the value of $f(x)$ and draw it. That is why classical optimization suggested 12.5 it is very much correct.

Figure 3 a. Problem Solving Steps



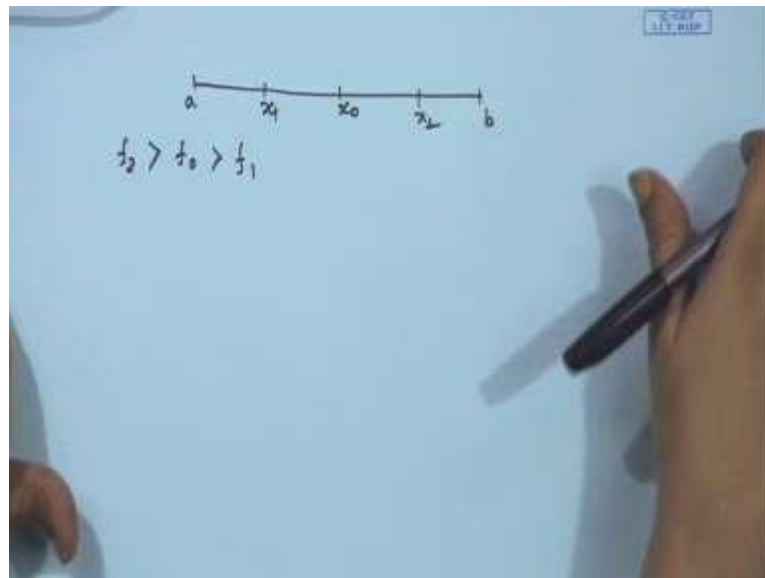
(Credit to NPTEL)

Figure 3 c. Problem Solving Steps

Now, this is not accepted. That is why we are going to the next level of technique optimization technique through which we will solve the problem this is the very well-known methodology. All of you must have been done in numerical analysis interval halving process. There is another name to it that is called the by section method. We will solve the same problem and we will see how nicely we will get the solution for the same problem where the solution will be we can consider as integer all right.

Now let me tell you the steps of interval halving method first. Now the first step it saying that whenever interval halving method is tells you that, as we could see from the function

fact and that always from maximum is there from 0 to 21. That is why it is suggested that let me considered 20. It is suggested that always from 0 to 20 you make the interval half of it. That is why we will get one part $[0, 10]$ and another part $[10, 20]$. After that each part you make you half it, that is why will get $[0, 5]$, $[5, 10]$, $[10, 15]$, $[10, 20]$. That is why initially what you do you take the left point of the interval and the right point on the interval and in between in equally use space 3 points. So, that whole interval can be divided in to 4 parts. The process tells you that the interval is called is the interval of uncertainty, because we do not know that is totally uncertain to us where the optima lie. Only thing we have some information function is unimodal in between 0 to 20.



(Credit to NPTEL)

Figure 3 d. Problem Solving Steps

If I just draw the intervals, that interval will be a, b, x_0, x_1, x_2 . Now we could see here that we are getting the point that is the middle point of $[a, b]$; middle point of $[a, x_0]$ and the middle point of $[x_0, b]$. Just look at the methodology we have written here if $f_2 > f_0 > f_1$ and we are looking for the minimum value of the function certainly we are function is unimodal. That is why there is only one more there only one

Minimum within the interval. That is the minimum cannot lie from within this region from x_2 to b because function is gradually increasing. That is why we will discard a part of the interval, which part we will discard we will discard the part from x_0 to b . And we will get a new interval of uncertainty as a to x_0 all right.

Let us consider the other case that $f_2 < f_0 < f_1$; that means, in the left side function is increasing and then in the right side function is decreasing, but if I just minimize the function certainly, in the left side function minimum cannot lie all right. That is why will discard that part that is why you see we have written the methodology delete a to x_0 and you considered new interval of uncertainty as x_0 to b .

Now, the other case, if we can if we see that $f_2 > f_0$, and $f_0 < f_1$; that means, we are getting the around f_0 we are having the minimum and f_1 one side this having the higher value f_2 one side higher value; that means, the minimum cannot lie below to and the beyond to . That is why will get the new interval of uncertainty as $[x_1, x_2]$. After what we will do once we will get the new interval of uncertainty, that one will consider at the initial interval of uncertainty we repeat the process. We will again find out 4 parts of the interval, we will find out new . we will repeat the process again and again and how long we will do in repeat the process as long as the interval of uncertainty is very small.

Or we have certain target that we want to get accuracy of 10%, accuracy of 5%. Then we can have the better process this process can give us that kind of accuracy. In that way will we will get the solution. Now this is one of the method numerical optimization. What even numerical optimization method we will get we will see everywhere the challenge lies how to select .

Because these are only the guiding points for selecting the minimum value of the function. That is why from the next class I will give you few more nice methodologies. And after learning all the methodologies together we will just compare all the methodologies, that is all for today.



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

UNIT – V – Advanced Optimization Methods – SPRA5301

1. Swarm Intelligence

Swarm Intelligence systems employ large numbers on using fuzzy systems, Tabu Search and Scatter Search, Ant colony algorithm, Multi Response optimization - Gray Relational Analysis.

2. Heuristic Algorithm

A heuristic algorithm is one that is designed to solve a problem in a faster and more efficient fashion than traditional methods by sacrificing optimality, accuracy, precision, or completeness for speed. Heuristic algorithms often times used to solve NP-complete problems, a class of decision problems. In these problems, there is no known efficient way to find a solution quickly and accurately although solutions can be verified when given. Heuristics can produce a solution individually or be used to provide a good baseline and are supplemented with optimization algorithms. Heuristic algorithms are most often employed of agents interacting locally with one another and the environment. Swarm intelligence refers to the collective behavior of decentralized systems and can be used to describe both natural and artificial systems. Specific algorithms for this class of system include the particle swarm optimization algorithm, the ant colony optimization algorithm, and artificial bee colony algorithm. Each of the previous algorithms was inspired by the natural, self-organized behavior of animals.

3. Tabu Search

This heuristic technique uses dynamically generated tabus to guide the solution search to optimum solutions. It examines potential solutions to a problem and checks immediate local neighbors to find an improved solution. The search creates a set of rules dynamically and prevents the system from searching around the same area redundantly by marking rule violating solutions as “tabu” or forbidden. This method solves the problem of local search methods when the search is stuck in suboptimal regions or in areas when there are multiple equally fit solutions.

4. Simulated Annealing

Borrowing the metallurgical term, this technique converges to a solution in the same way metals are brought to minimum energy configurations by increasing grain size. Simulated annealing is used in global optimization and can give a reasonable approximation of a global optimum for a function with a large search space. At each iteration, it probabilistically decides between staying at its current state or moving to another while ultimately leading the system to the lowest energy state.²

5. Genetic Algorithms

Genetic algorithms are a subset of a larger class of evolutionary algorithms that describe a set of techniques inspired by natural selection such as inheritance, mutation, and crossover. Genetic algorithms require both a genetic representation of the solution domain and a fitness function to evaluate the solution domain. The technique generates a population of candidate solutions and

uses the fitness function to select the optimal solution by iterating with each generation. The algorithm terminates when the satisfactory fitness level has been reached for the population or the maximum generations have been reached.

6. Artificial Neural Networks

Artificial Neural Networks (ANNs) are models capable of pattern recognition and machine learning, in which a system analyzes a set of training data and is then able to categorize new examples and data. ANNs are influenced by animals' central nervous systems and brains, and are used to solve a wide variety of problems including speech recognition and computer vision.¹

Support Vector Machines

Support Vector Machines (SVMs) are models with training data used by artificial intelligence to recognize patterns and analyze data. These algorithms are used for regression analysis and classification purposes. Using example data, the algorithm will sort new examples into groupings. These SVMs are involved with machine learning, a subset of artificial intelligence where systems learn from data, and require training data before being capable of analyzing new examples.¹

6.1 Neural Network Theory

Starting with measured data from some known or unknown source, a neural network may be trained to perform classification, estimation, simulation, and prediction of the underlying process generating the data. Hence, neural networks, or neural nets, are software tools designed to estimate relationships in data. An estimated relationship is essentially a mapping, or a function, relating raw data to its features. The Neural Networks package supports several function estimation techniques that may be described in terms of different types of neural networks and associated learning algorithms.

The general area of artificial neural networks has its roots in our understanding of the human brain. In this regard, initial concepts were based on attempts to mimic the brain's way of processing information. Efforts that followed gave rise to various models of biological neural network structures and learning algorithms. This is in contrast to the computational models found in this package, which are only concerned with artificial neural networks as a tool for solving different types of problems where unknown relationships are sought among given data. Still, much of the nomenclature in the neural network arena has its origins in biological neural networks, and thus, the original terminology will be used alongside with more traditional nomenclature from statistics and engineering.

7. Function Approximation

When input data originates from a function with real-valued outputs over a continuous range, the neural network is said to perform a traditional function approximation. An example of an approximation problem could be one where the temperature of an object is to be determined from secondary measurements, such as emission of radiation. Another more trivial example could be to estimate shoe size based on a person's height. These two examples involve models with one input and one output. A more advanced model of the second example might use gender as a second input in order to derive a more accurate estimate of the shoe size.

Pure functions may be approximated with the following two network types:

- Feedforward Neural Networks
- Radial Basis Function Networks

8. Feedforward Neural Networks

Feedforward neural networks (FF networks) are the most popular and most widely used models in many practical applications. They are known by many different names, such as "multi-layer perceptrons." Figure illustrates a one-hidden-layer FF network with inputs x_1, \dots, x_n and output \hat{y} . Each arrow in the figure symbolizes a parameter in the network. The network is divided into layers. The input layer consists of just the inputs to the network. Then follows a hidden layer, which consists of any number of neurons, or hidden units placed in parallel. Each neuron performs a weighted summation of the inputs, which then passes a nonlinear activation function σ , also called the neuron function as shown in figure 1.

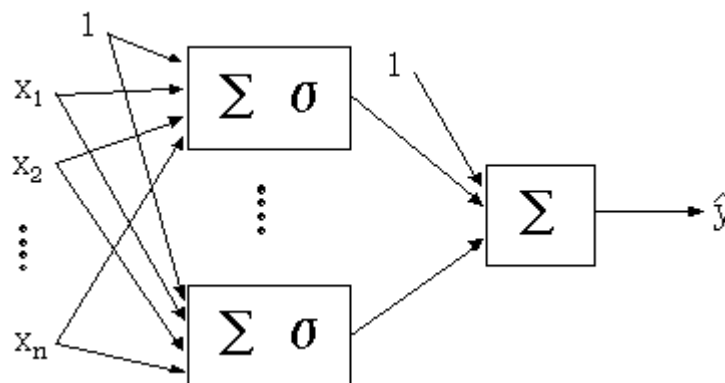


Figure 1. Neural Network

. A feedforward network with one hidden layer and one output. Mathematically the functionality of a hidden neuron is described by

$$\sigma \left(\sum_{j=1}^n w_j x_j + b_j \right)$$

where the weights $\{w_j^i, b_j^i\}$ are symbolized with the arrows feeding into the neuron.

The network output is formed by another weighted summation of the outputs of the neurons in the hidden layer. This summation on the output is called the output layer. In Figure there is only one output in the output layer since it is a single-output problem. Generally, the number of output neurons equals the number of outputs of the approximation problem.

The neurons in the hidden layer of the network in Figure are similar in structure to those of the perceptron, with the exception that their activation functions can be any differential function. The output of this network is given by

$$\hat{y}(\theta) = g(\theta, x) = \sum_{i=1}^{nh} w_i^2 \sigma \left(\sum_{j=1}^n w_{i,j}^1 x_j + b_{i,1}^1 \right) + b^2 \quad (8)$$

where n is the number of inputs and nh is the number of neurons in the hidden layer. The variables $\{w_{i,j}^1, b_{j,i}^1, w_i^2, b^2\}$ are the parameters of the network model that are represented collectively by the parameter vector θ . In general, the neural network model will be represented by the compact notation $g(\theta, x)$ whenever the exact structure of the neural network is not necessary in the context of a discussion.

Some small function approximation examples using an FF network can be found in Section 5.2.

Note that the size of the input and output layers are defined by the number of inputs and outputs of the network and, therefore, only the number of hidden neurons has to be specified when the network is defined. The network in Figure 2.5 is sometimes referred to as a three-layer network, counting input, hidden, and output layers. However, since no processing takes place in the input layer, it is also sometimes called a two-layer network. To avoid confusion this network is called a one-hidden-layer FF network throughout this documentation.

In training the network, its parameters are adjusted incrementally until the training data satisfy the desired mapping as well as possible; that is, until $\hat{y}(\theta)$ matches the desired output y as closely as possible up to a maximum number of iterations. The nonlinear activation function in the neuron is usually chosen to be a smooth step function. The default is the standard sigmoid as shown in figure 2

$$\text{Sigmoid}[x] = \frac{1}{1 + e^{-x}} \quad (9)$$

that looks like this.

```
<< NeuralNetworks`
In[1]:=Plot[Sigmoid[x], {x, -8, 8}]
```

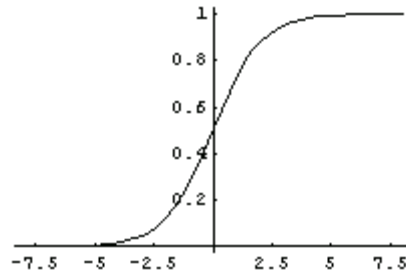


Figure 2. Performance Evaluation

The FF network in Figure is just one possible architecture of an FF network. You can modify the architecture in various ways by changing the options.

9. Multilayer Networks

The package supports FF neural networks with any number of hidden layers and any number of neurons (hidden neurons) in each layer. In Figure a multi-output FF network with two hidden layers is shown in figure.

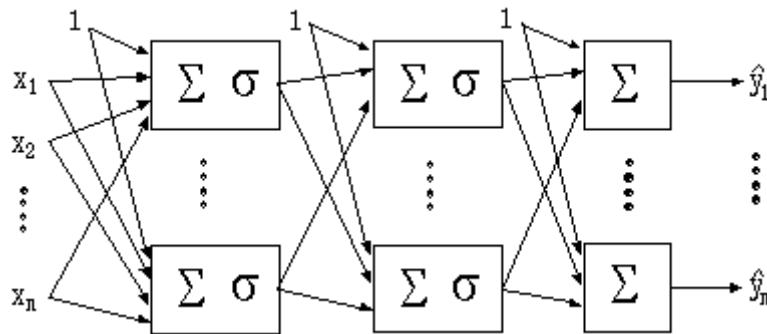


Figure 3. Multilayer Network

A multi-output feed forward network with two hidden layers.

The number of layers and the number of hidden neurons in each hidden layer are user design parameters. The general rule is to choose these design parameters so that the best possible model with as few parameters as possible is obtained. This is, of course, not a very useful rule, and in practice you have to experiment with different designs and compare the results, to find the most suitable neural network model for the problem at hand. For many practical applications, one or two hidden layers will suffice. The recommendation is to start with a linear model; that is, neural networks with no hidden layers, and then go over to networks with one hidden layer but with no more than five to ten neurons. As a last step you should try two hidden layers.

10.Radial Basis Function Networks

After the FF networks, the radial basis function (RBF) network comprises one of the most used network models. Figure illustrates an RBF network with inputs x_1, \dots, x_n and output \hat{y} . The arrows in the figure symbolize parameters in the network. The RBF network consists of one hidden layer of basis functions, or neurons. At the input of each neuron, the distance between the neuron center and the input vector is calculated. The output of the neuron is then formed by applying the basis function to this distance. The RBF network output is formed by a weighted sum of the neuron outputs and the unity bias shown in figure 4.

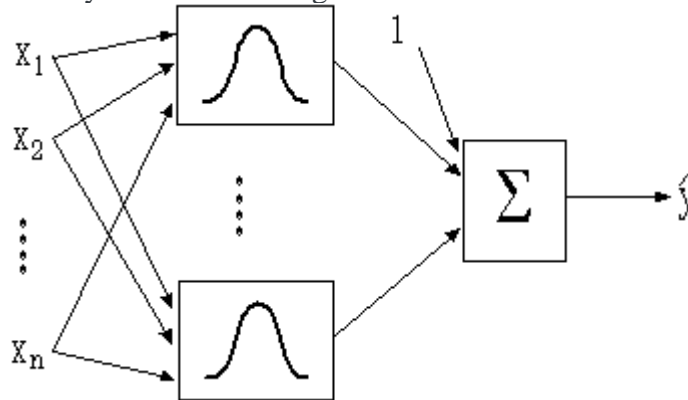


Figure 4. RBF network with one output.

The RBF network in Figure is often complemented with a linear part. This corresponds to additional direct connections from the inputs to the output neuron. Mathematically, the RBF network, including a linear part, produces an output given by

$$\hat{y}(\theta) = g(\theta, x) = \sum_{i=1}^{nb} w_i^2 e^{-\lambda_i^2 (x - w_i^1)^2} + w_{nb+1}^2 + x_1 x_1 + \dots + x_n x_n \quad (10)$$

where nb is the number of neurons, each containing a basis function. The parameters of the RBF network consist of the positions of the basis functions w_i^1 , the inverse of the width of the basis functions λ_i , the weights in output sum w_i^2 , and the parameters of the linear part x_1, \dots, x_n . In most cases of function approximation, it is advantageous to have the additional linear part but it can be excluded by using the options.

The parameters are often lumped together in a common variable θ to make the notation compact. Then you can use the generic description $g(\theta, x)$ of the neural network model, where g is the network function and x is the input to the network.

In training, the network the parameters are tuned so that the training data fits the network model Eq. (2.10) as well as possible. In Eq. (2.10) the basis function is chosen to be the Gaussian bell

function. Although this function is the most commonly used basis function, other basis functions may be chosen. Also, RBF networks may be multi-output as illustrated in Figure 2.8.

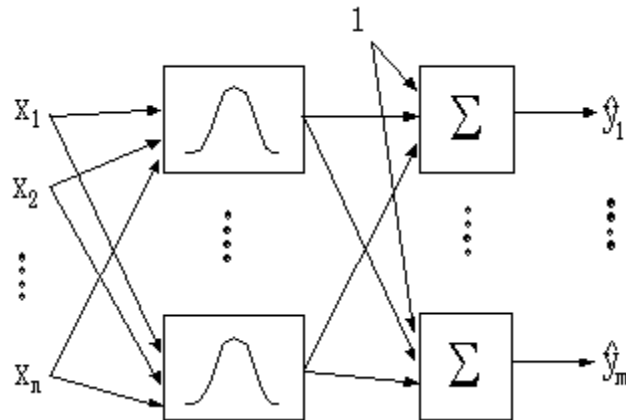


Figure 5. Multi-output RBF network

FF networks and RBF networks can be used to solve a common set of problems. The built-in commands provided by the package and the associated options are very similar. Problems where these networks are useful include:

- Function approximation
- Classification
- Modeling of dynamic systems and time series

11.Dynamic Neural Networks

Techniques to estimate a system process from observed data fall under the general category of system identification. Figure 6 illustrates the concept of a system.

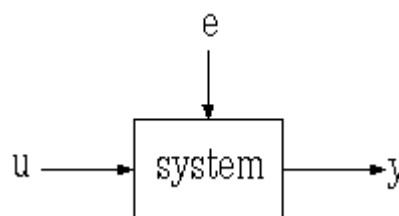


Figure 6. System

A system with input signal u , disturbance signal e , and output signal y

Loosely speaking, a system is an object in which different kinds of signals interact and produce an observable output signal. A system may be a real physical entity, such as an engine, or entirely abstract, such as the stock market. There are three types of signals that characterize a system, as indicated. The output signal $y(t)$ of the system is an observable/measurable signal, which you want to understand and describe. The input signal $u(t)$ is an external measurable

signal, which influences the system. The disturbance signal $e(t)$ also influences the system but, in contrast to the input signal, it is not measurable. All these signals are time dependent.

In a single-input, single-output (SISO) system, these signals are time-dependent scalars. In the multi-input, multi-output (MIMO) systems, they are represented by time-dependent vectors. When the input signal is absent, the system corresponds to a time-series prediction problem. This system is then said to be noise driven, since the output signal is only influenced by the disturbance $e(t)$. The Neural Networks package supports identification of systems with any number of input and output signals. A system may be modeled by a dynamic neural network that consists of a combination of neural networks of FF or RBF types, and a specification of the input vector to the network. Both of these two parts have to be specified by the user. The input vector, or regressor vector, which it is often called in connection with dynamic systems, contains lagged input and output values of the system specified by three indices: n_a , n_b , and n_k . For a SISO model the input vector looks like this:

$$\begin{aligned} x(t) = & [y(t-1) \dots y(t-n_a) \\ & u(t-n_k) \dots u(t-n_k-n_b+1)]^T \end{aligned} \quad (22)$$

Index n_a represents the number of lagged output values; it is often referred to as the order of the model. Index n_k is the input delay relative to the output. Index n_b represents the number of lagged input values. In a MIMO case each individual lagged signal value is a vector of appropriate length. For example, a problem with three outputs and two inputs $n_a=\{1,2,1\}$, $n_b=\{2,1\}$, and $n_k=\{1,0\}$ gives the following regressor:

$$\begin{aligned} x(t) = & [y_1(t-1) \ y_2(t-1) \ y_2(t-2) \\ & y_2(t-1) \ u_1(t-1) \ u_1(t-2) \ u_2(t)] \end{aligned}$$

For time-series problems, only n_a has to be chosen.

The neural network part of the dynamic neural network defines a mapping from the regressor space to the output space. Denote the neural network model by $g(\theta, \cdot)$ where θ is the parameter vector to be estimated using observed data. Then the prediction $\hat{y}(t)$ can be expressed as

$$\hat{y}(t) = g(\theta, x(t)) \quad (23)$$

Models with a regressor like Eq. (2.0) are called ARX models, which stands for AutoRegressive with eXtra input signal. When there is no input signal $u(t)$, its lagged valued may be eliminated from Eq. (2.0), reducing it to an AR model. Since the mapping $g(\theta, \cdot)$ is based on neural networks, the dynamic models are called neural ARX and neural AR models, or neural AR(X) as

short form for both them. Figure 2.10 shows a neural ARX model, based on a one-hidden-layer FF network.

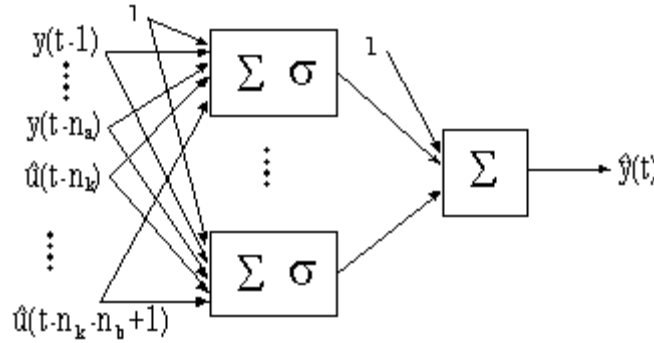


Figure 7. Neural ARX Model

The special case of an ARX model where no lagged outputs are present in the regressor (that is, when $n_a=0$ in Eq. (2.0)), is often called a Finite Impulse Response (FIR) model.

Depending on the choice of the mapping $g(\theta, \cdot)$ you obtain a linear or a nonlinear model using an FF network or an RBF network.

Although the disturbance signal $e(t)$ is not measurable, it can be estimated once the model has been trained. This estimate is called the prediction error and is defined by

$$\hat{e}(t) = y(t) - \hat{y}(t) \quad (24)$$

A good model that explains the data well should yield small prediction errors. Therefore, a measure of $\hat{e}(t)$ may be used as a model-quality index.

12. Hopfield Network

In the beginning of the 1980s Hopfield published two scientific papers, which attracted much interest. This was the starting point of the new era of neural networks, which continues today.

Hopfield showed that models of physical systems could be used to solve computational problems. Such systems could be implemented in hardware by combining standard components such as capacitors and resistors.

The importance of the different Hopfield networks in practical application is limited due to theoretical limitations of the network structure but, in certain situations, they may form interesting models. Hopfield networks are typically used for classification problems with binary pattern vectors.

The Hopfield network is created by supplying input data vectors, or pattern vectors, corresponding to the different classes. These patterns are called class patterns. In an n -dimensional data space the class patterns should have n binary components $\{1, -1\}$; that is, each class pattern corresponds to a corner of a cube in an n -dimensional space. The network is then used to classify distorted patterns into these classes. When a distorted pattern is presented to the

network, then it is associated with another pattern. If the network works properly, this associated pattern is one of the class patterns. In some cases (when the different class patterns are correlated), spurious minima can also appear. This means that some patterns are associated with patterns that are not among the pattern vectors.

Hopfield networks are sometimes called associative networks since they associate a class pattern to each input pattern.

The Neural Networks package supports two types of Hopfield networks, a continuous-time version and a discrete-time version. Both network types have a matrix of weights W defined as

$$W = \frac{1}{n} \sum_{i=1}^D \xi_i \xi_i^T \quad (25)$$

where D is the number of class patterns $\{\xi_1, \xi_2, \dots, \xi_D\}$, vectors consisting of ± 1 elements, to be stored in the network, and n is the number of components, the dimension, of the class pattern vectors.

Discrete-time Hopfield networks have the following dynamics:

$$x(t+1) = \text{Sign}[W x(t)] \quad (26)$$

Eq. (2.26) is applied to one state, $x(t)$, at a time. At each iteration the state to be updated is chosen randomly. This asynchronous update process is necessary for the network to converge, which means that $x(t) = \text{Sign}[W x(t)]$.

A distorted pattern, $x(0)$, is used as initial state for the Eq. (2.0), and the associated pattern is the state toward which the difference equation converges. That is, starting with $x(0)$ and then iterating Eq. (2.0) gives the associated pattern when the equation converged.

For a discrete-time Hopfield network, the energy of a certain vector x is given by

$$E(x) = -x W x^T \quad (27)$$

It can be shown that, given an initial state vector $x(0)$, $x(t)$ in Eq. (2.26) will converge to a value having minimum energy. Therefore, the minima of Eq. (2.27) constitute possible convergence points of the Hopfield network and, ideally, these minima are identical to the class patterns $\{\xi_1, \xi_2, \dots, \xi_D\}$. Hence, one can guarantee that the Hopfield network will converge to some pattern, but one cannot guarantee that it will converge to the right pattern.

Note that the energy function can take negative values; this is, however, just a matter of scaling. Adding a sufficiently large constant to the energy expression it can be made positive.

The continuous Hopfield network is described by the following differential equation

$$\frac{dx(t)}{dt} = -x(t) + W \sigma[x(t)] \quad (28)$$

where $x(t)$ is the state vector of the network, W represents the parametric weights, and σ is a nonlinearity acting on the states $x(t)$. The weights W are defined in Eq. (2.25). The differential equation, Eq. (2.28), is solved using an Euler simulation.

To define a continuous-time Hopfield network, you have to choose the nonlinear function σ . There are two choices supported by the package, Saturated Linear and the default nonlinearity of Tanh.

For a continuous-time Hopfield network, defined by the parameters given in Eq. (2.25), one can define the energy of a particular state vector x as

$$E(x) = -\frac{1}{2} x^T W x + \sum_{i=1}^m \int_0^{x_i} \sigma^{-1}(t) dt \quad (29)$$

As for the discrete-time network, it can be shown that given an initial state vector $x(0)$, the state vector $x(t)$ in Eq. (2.28) converges to a local energy minimum. Hence, the minima of Eq. (2.29) constitute the possible convergence points of the Hopfield network and ideally these minima are identical to the class patterns $\{\xi_1, \xi_2, \dots, \xi_p\}$. However, there is no guarantee that the minima will coincide with this set of class patterns.