NOTES

Decision Models

2223-1-FDS01Q002-FDS01Q003M

Vittorio Haardt
853268
vittoriohaardt@gmail.com

June 22, 2023

Abstract

This module will emphasizes the relevance of data in decision making. The general aim is to develop skills in mathematical modeling and in algorithms and computational methods to solve and analyze decision problems. The course will illustrates how to formulate real world problems using case studies and examples; how to use efficient algorithms – both old and new – for solving these models; and how to evaluate, draw useful conclusions and derive useful planning information from the output of these algorithms.

Contents

Chapter 1

Introduction

1.1 Introduction and Basic Concepts

1.1.1 Analytics

Analytics is the discovery, interpretation, and communication of meaningful patterns in data. Especially valuable in areas rich with recorded information, analytics relies on the simultaneous application of statistics, computer programming and operations research to quantify performance.

Gartner defines Analytics as a term that encompasses a variety of different business intelligence initiatives. Often, the study of historical data is involved to search for trends and analyze decisions or events. The objective is to improve business by adding knowledge that can be used to make changes.

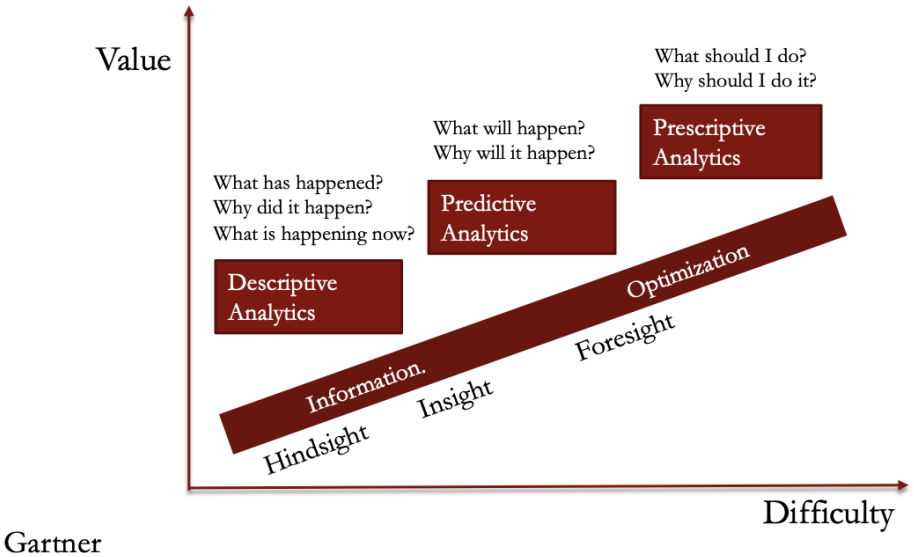
Data is used to build models and create value, and for this purpose, both big data and small data can be used. When we talk about Analytics, we can differentiate between three major areas of study: descriptive, predictive, and prescriptive.

Descriptive Analytics can be used to describe the past, where data structures are used to efficiently save and extract data. Predictive Analytics, on the other hand, uses past data to create predictive models, while prescriptive analytics uses heuristic models, optimizations, and simulations to prescribe optimal actions.

- **Descriptive analytics**
  - Summary statistics
  - Visualizations
  - Clustering

  - etc...
- **Predictive analytics**
  - Linear regression
  - Logistic Regression
  - CART
  - Random Forest
  - etc...
- **Prescriptive analytics**
  - Optimization

The three phases of Analytics refer to different objectives summarized in the following image. The chart also shows the relationship between the added value and the difficulty in tackling the various approaches.



Gartner

Predictive Analytics starts from the descriptive phase, extracting patterns from data, correlating data types to explain short-term behaviors, and estimating linear or nonlinear behaviors. On the other hand, starting from Predictive Analytics, we find *Prescriptive Analytics*. Essentially, it determines what should happen and how to make it happen by identifying those factors that contribute to desirable/undesirable outcomes. An example could be finding a set of prices and an advertising frequency to maximize economic return. Techniques such as linear programming, sensitivity analysis, integer programming, goal programming, nonlinear programming, and simulation programming are usually used. Some of these models will be covered during the course.

1.1.2 Decision Models

Making a decision can be interpreted as an analytical process of selecting a choice from possible options. When we try to make a good decision, we must consider all alternatives and weigh all possible positive and negative choices. For effective selection, we should be able to predict the outcome of all possible options and make the choice based on these predictions.

Decision-Making is a goal-oriented process; we usually make decisions to achieve an objective. We can use Analytics to make decisions.

- **Unstructured problems**
  - Problems that are new, unusual, and for which information is ambiguous or incomplete
  - Problems that require ad-hoc solutions
  - **Non-programmed decisions**
    - Decisions that are unique and non-recurring
    - Decisions that generate unique responses
- **Structured problems**
  - Objectives are clear
  - They are familiar (they have occurred previously)
  - They are easy and completely defined (information regarding the problem is available and complete)
  - **Programmed decisions**
    - Repetitive decisions that can be managed through a routine-based approach, i.e., an approach based on procedures rather than ad-hoc solutions
    - In this case, the difficulty of this process is related to the number of decisions and not the evaluation of decisions

Structured problems are typically operational and based on multiple short-term solutions. Problems can also be tactical, in which case we have medium-term solutions. Finally, unstructured problems, which are typically strategic, are characterized by few possible long-term solutions. Another categorization is related to the condition of **certainty**. When we are faced with a situation in which the results of all possible decisions are known, we are dealing with a condition of certainty that allows us to make accurate decisions. On the contrary, in situations of **uncertainty**, we have no knowledge of the outcome of the various possible choices. In this case, the probabilistic estimation of the result allows us to estimate the probability of the results of the choices, and this condition is called **risk**. A decision model is a symbolic representation of assumptions about reality. A decision model predicts the results of decisions. Models are essential for developing a scientific method for solving a problem and presenting the results. When we talk about optimization, we refer to that part of decision making that is at the center of all machine learning and statistical techniques used in data science. The goal of optimization is to maximize or minimize a certain performance

function relative to some possible solutions that meet a set of constraints. This performance function can be applied to different choices to determine the best one. It is usually achieved through operations research techniques.

Example: Industry 4.0 Industry 4.0 indicates the current trend of automation and data exchange in manufacturing technologies in order to facilitate manufacturing. For example, consider the case of *predictive maintenance* in which sensors generate a multitude of data dealing with indicators of equipment's degradation. *Descriptive analytics* algorithms monitor the current condition of the manufacturing system and provide alerts in cases of abnormal behaviours. This is achieved by comparing the actual measurements of several parameters that constitute indicators of degradation. When they vary from the normal values, an alert triggers the *predictive analytics* algorithms. The alert is evaluated and, if it indicates a potentially hazardous state of the manufacturing equipment, the predictive analytics algorithms generate predictions about the future health state of the manufacturing system, e.g. a prediction about the time-to-failure. On the basis on this prediction, *prescriptive analytics* algorithms are able to provide recommendations about the optimal mitigating actions and the optimal time for their implementation in a way that the expected loss and the risk are minimized.

Example: Transportation The *traffic congestion control* attempts to release the city centers from the traffic jams. Currently, sensors can detect vehicles in corresponding areas. Applying *descriptive* analytics we can use this data along with historical data from traffic monitoring network to derive outcomes such as level of traffic in different areas at different times in an aggregated form and describe traffic flow. These results feed into the *predictive analytics* algorithms can provide predictions about the traffic flow (congestion level) of the system by taking into account contextual information (e.g. peak times). Then these predictions trigger the *prescriptive analytics* algorithms which can suggest real time reactions with the aim to reduce the congestion level proactively (e.g. traffic lights control), or also take strategic decisions on the network viability in order to reduce congestions.

1.2 The Need for Decision Models

Models can be used for technical aspects of decision problems. Other aspects cannot be modeled easily, requiring intuition and judgment, but human judgment and intuition is not always rational. In psychology, two effects are studied: **anchoring effect** and **framing effect**.

The **anchoring effect** arises when trivial factors influence the initial reasoning on a certain problem. As a result, some problems and their solutions are underestimated.

For example in a study, participants were asked to calculate $1 \times 2 \times 3 \times 4 \times 5 \times 6 \times 7 \times 8$ and $8 \times 7 \times 6 \times 5 \times 4 \times 3 \times 2 \times 1$ in 5 seconds. Since the time was not enough to perform all the multiplications, they had to make estimates. Participants who performed the first series of multiplications generally estimated a smaller product (the median of the estimate was 512), while those who performed the second series estimated a higher number (the median was 2250). The actual result is 40320. The reason for these two different estimates is that the initial conditions of the first series led participants to think of smaller numbers in their multiplication.

Errors in human judgment often arise because of what psychologists term anchoring and framing effects associated with decision problems. Anchoring effects arise when a seemingly trivial factor serves as a starting point (or anchor) for estimations in a decision-making problem. Decision makers adjust their estimates from this anchor but nevertheless remain too close to the anchor and usually under-adjust.

The **framing effects** refers to how decision-makers view alternatives in a problem, often from a win-loss perspective. The way a problem is framed often influences choices in irrational ways.

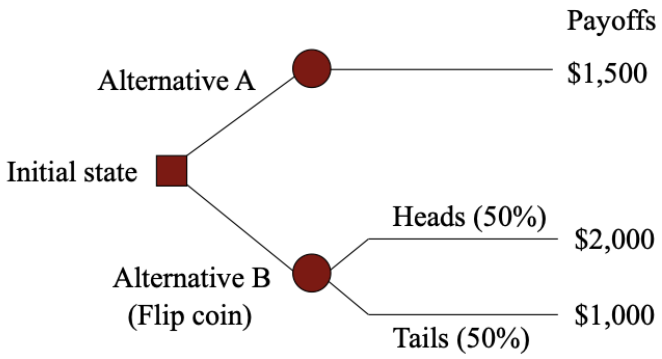
Suppose we have received \$1000 and must choose between:

- Receiving \$500 immediately
- Flipping a coin and receiving \$1000 more if it lands heads, or \$0 if it lands tails

The first option guarantees a win and is the most popular choice. Now, suppose we have received \$2000 and must choose between:

- Immediately giving back \$500
- Flipping a coin and giving back \$0 if it lands heads, or giving back \$1000 if it lands tails

By framing the problem in this way, people who chose the first option in the first scenario and then chose the second option in the second scenario made an irrational choice. The irrationality arises when we view the first option as a guaranteed loss without considering its benefits.



Decision Models help in making good decisions but cannot guarantee that good outcomes will always occur as a result of those decisions. However, using a structured, modeling approach to decision making should produce good outcomes more frequently than making decisions in a more haphazard manner.

Chapter 2

Linear Programming

2.1 Mathematical Programming

Optimization comes from the same root as "optimal", which means best. When you optimize something, you are "making it best". But "best" can vary. If you're a football player, you might want to maximize your running yards,and also minimize your fumbles. If you are a financial investor, you might want to maximize your profit and minimize your risk.

Mathematical optimization is a branch of applied mathematics which is useful in the decision making making process. Both **maximizing** and *minimizing* are types of optimization problems.
Remember that maximizing a linear function is equivalent to minimizing the negative of the same linear function.
Our focus in the first part will be mainly on the modeling and evaluation process rather than on the solution methods. Mathematical Programming is a field of operations research that finds the optimal, or most efficient, way of using limited resources to achieve the objectives of an individual of a business.

An optimization problem has the following characteristics:

- Objectives: An **objective function** expresses the main aim of the model which is either to be minimized or maximized
- Decisions: A set of **unknowns** or **variables** which control the value of the objective function
- Constraints: A set of **constraints** that allow the unknowns to take on certain values but exclude others

The optimization problem is then to find values of the variables that minimize or maximize the objective function while satisfying the constraints.

When both the objective function and the constraint functions are linear, we have a linear programming problem, which can be more properly represented in the following way:

$$\begin{array}{ll} \text{MAX (or MIN):} & f_0(X_1, X_2, ..., X_n) \\ \text{Subject to:} & f_1(X_1, X_2, ..., X_n) <= b_1 \\ & \vdots \\ & f_0(X_1, X_2, ..., X_n) >= b_k \\ & \vdots \\ & f_0(X_1, X_2, ..., X_n) = b_m \end{array}$$

Note that if all the functions in an optimization are linear, the problem is a **Linear Programming** (LP) problem.

Another example of an optimization problem is that of centroid-based clustering, which is to find the centroids of a vector space such that the sum of distances between these centroids and their corresponding points is minimized.

- **PROBLEM**: Given **m points** $\{x_1, x_2, \ldots, x_m\}$ in n-dimensional space R^n, and a **fixed integer *k* of clusters**, determine ***k* "centers"** in R^n, $\{c_1, c_2, \ldots, c_k\}$, such that the sum of the "**distances**" of each point to a nearest cluster center is **minimized**.

- **MODEL:**

$$\min_{c^1,\ldots,c^k} \sum_{i=1}^{m} \min_{\ell=1,\ldots,k} \|x^i - c^\ell\| \quad (1)$$

$$\begin{array}{ll} \underset{c^\ell, t_{i\ell}}{\text{minimize}} & \sum_{i=1}^{m}\sum_{\ell=1}^{k} t_{i\ell} \cdot \|x^i - c^\ell\| \quad (2) \\ \text{subject to} & \sum_{\ell=1}^{k} t_{i\ell} = 1, t_{i\ell} \geq 0,\ i = 1,\ldots,m,\ \ell = 1,\ldots,k \quad (3) \end{array}$$

- For a fixed data point x^i if ℓ^* is the index such that center c^{ℓ^*} is nearest to x^i, then $\ell^*=1$ and $\ell = 0$, $\ell \neq \ell^*$

- **K-Median**: solving (2)-(3) with the 1-norm distance

P. S. Bradley, O. L. Mangasarian, and W. N. Street. Clustering via concave minimization. In Advances in Neural Information Processing Systems, pages 368–374, Cambridge, MA, 1997. MIT Press.

$$\begin{array}{ll} \underset{c,d,t}{\text{minimize}} & \sum_{i=1}^{m}\sum_{\ell=1}^{k} t_{i\ell} \cdot (e^T d_{i\ell}) \\ \text{subject to} & -d_{i\ell} \leq x^i - c^\ell \leq d_{i\ell},\ i = 1,\ldots,m,\ \ell = 1,\ldots,k, \\ & \sum_{\ell=1}^{k} t_{i\ell} = 1,\ t_{i\ell} \geq 0,\ i = 1,\ldots,m,\ \ell = 1,\ldots,k. \end{array}$$

- **K-Means**: solving (2)-(3) with the 2-norm squared distance

MacQueen, J. Some methods for classification and analysis of multivariate observations. In Proceedings of the 5° Berkeley symposium on mathematical statistics and probability (Vol. 1, No. 14, pp. 281-297).

$$\begin{array}{ll} \underset{c,t}{\text{minimize}} & \sum_{\ell=1}^{k}\sum_{i=1}^{m} t_{i\ell} \left(\frac{1}{2}\|x^i - c^\ell\|_2^2\right) \\ \text{subject to} & \sum_{\ell=1}^{k} t_{i\ell} = 1,\ t_{i\ell} \geq 0,\ i = 1,\ldots,m,\ \ell = 1,\ldots,k. \end{array}$$

By changing the distance criterion to be optimized, K-means and K-medians arise.

When both the objective function and the constraint functions are linear, we are facing a linear programming problem, which can be more properly represented as follows:

$$\begin{array}{ll} \text{MAX (or MIN):} & c_1X_1 + c_2X_2 + ... + c_nX_n \\ \text{Subject to:} & a_{11}X_1 + a_{12}X_2 + ... + a_{1n}X_n <= b_1 \\ & \vdots \\ & a_{k1}X_1 + a_{k2}X_2 + ... + a_{kn}X_n >= b_k \\ & \vdots \\ & a_{m1}X_1 + a_{m2}X_2 + ... + a_{mn}X_n = b_m \end{array}$$

Now lets wee a simple example of linear programming.

An Example LP Problem Blue Ridge Hot Tubs produces two types of hot tubs: Aqua-Spas and Hydro-Luxes.

	Aqua-Spa	Hydro-Lux
Pumps	1	1
Labor	9 hours	6 hours
Tubing	12 feet	16 feet
Unit profit	$350	$300

There are 200 pumps, 1566 hours of labor, and 2880 feet of tubing available.

The question is: how many pumps can we produce if we want to maximize profit in the next production cycle? This is a problem that can be formulated using linear programming. This is done through several phases:

1. Understand the problem.
2. Identify the decision variables
   X_1 = number of Aqua-Spas to produce
   X_2 = number of Hydro-Luxes to produce
3. State the objective function as a linear combination of the decision variables
   MAX: $350X_1 + 300X_2$
4. State the constraints as linear combinations of the decision variables
   $1X_1 + 1X_2 <= 200$ } pumps
   $9X_1 + 6X_2 <= 1566$ } labor
   $12X_1 + 16X_2 <= 2880$ } tubing
5. Identify any upper or lower bounds on the decision variables
   $X_1 >= 0$
   $X_2 >= 0$

2.2 Linear Programming Graphical Solution

Let's go back to the mathematical formulation of a previous linear programming problem.

$$
\begin{array}{ll}
\text{MAX:} & 350X_1 + 300X_2 \\
\text{S.T.:} & 1X_1 + 1X_2 <= 200 \\
 & 9X_1 + 6X_2 <= 1566 \\
 & 12X_1 + 16X_2 <= 2880 \\
 & X_1 >= 0 \\
 & X_2 >= 0
\end{array}
$$

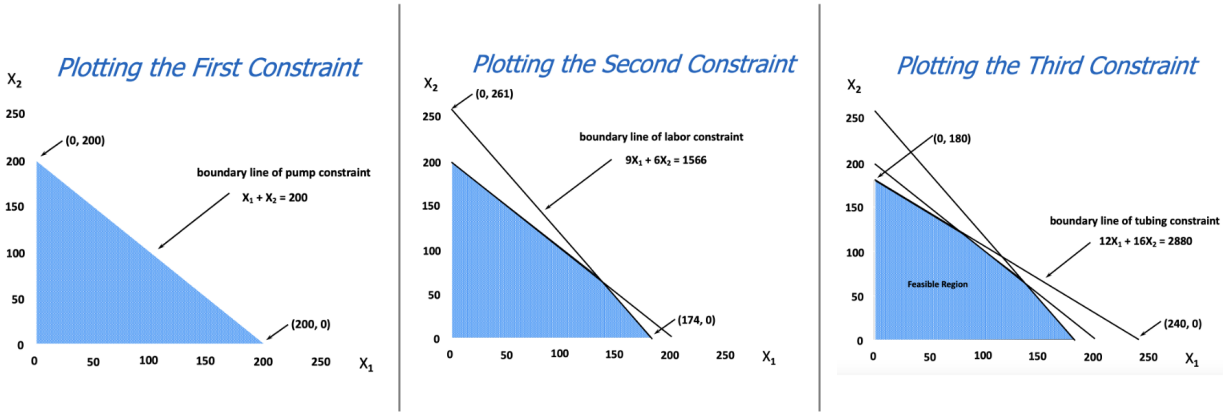
A first intuitive approach is to notice that the Aqua-Spa pump produces the maximum unit profit, so how many Aqua-Spa pumps can we produce? Setting $X_2 = 0$ transforms the system into:

- 1st constraint: $1X_1 <= 200$
- 2nd constraint: $9X_1 <= 1566$ or $X_1 <= 174$
- 3rd constraint: $12X_1 <= 2880$ or $X_1 <= 240$

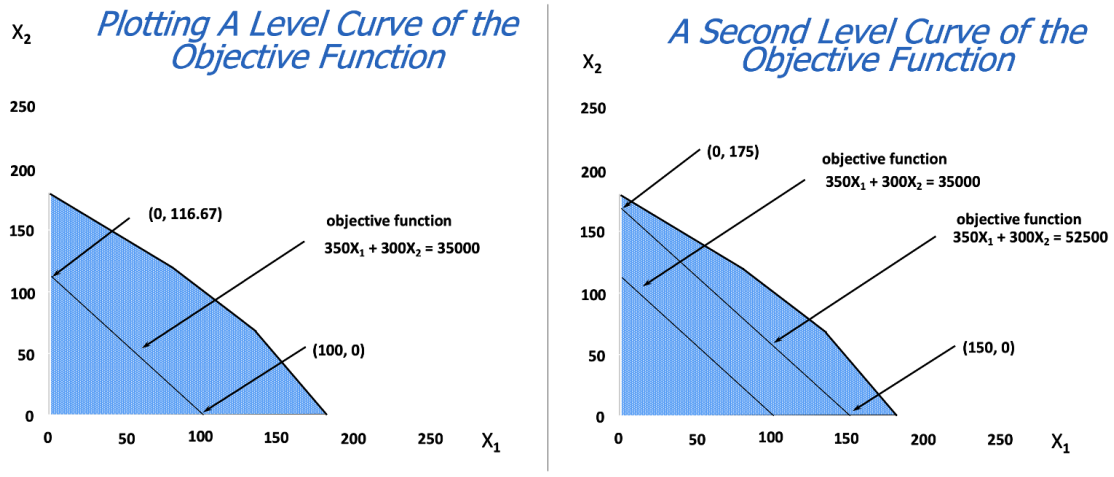
If $X_2 = 0$, the maximum value of X_1 is 174 and the total profit is $\$350 \cdot 174 + \$300 \cdot 0 = \$60,900$. This solution is feasible, but is it optimal?

With this new system, we have that the maximum number of Aqua-Spa pumps we can produce is 174 for a total profit of 60900. We can define a feasible region for our objective function through the constraints, and the best point in the region is also the optimal solution to the problem. For LP problems with two variables, it is simple to identify the feasible region through graphs.

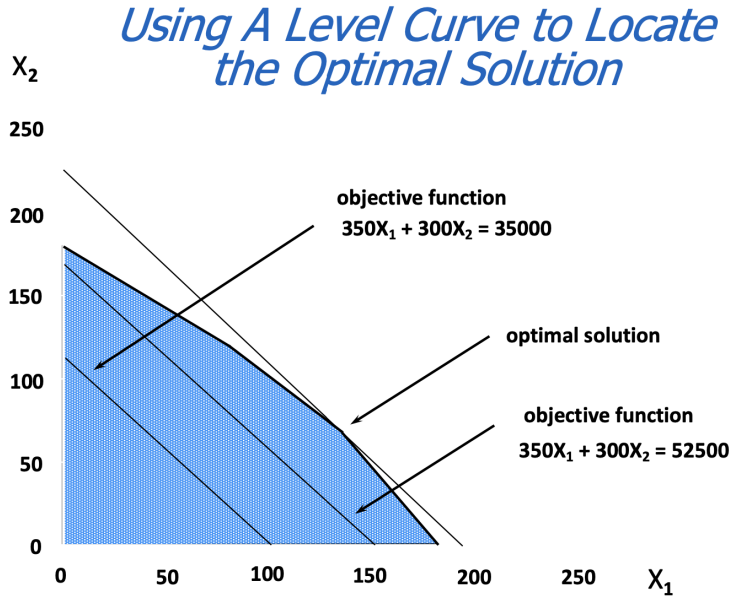
The idea of finding the solution through graphical representation is to iteratively define the region by applying the constraints, which will progressively shrink the final region.



At the end of determining the feasible region, the objective function is then plotted, and the vertex that maximizes the region and therefore the objective function is identified.



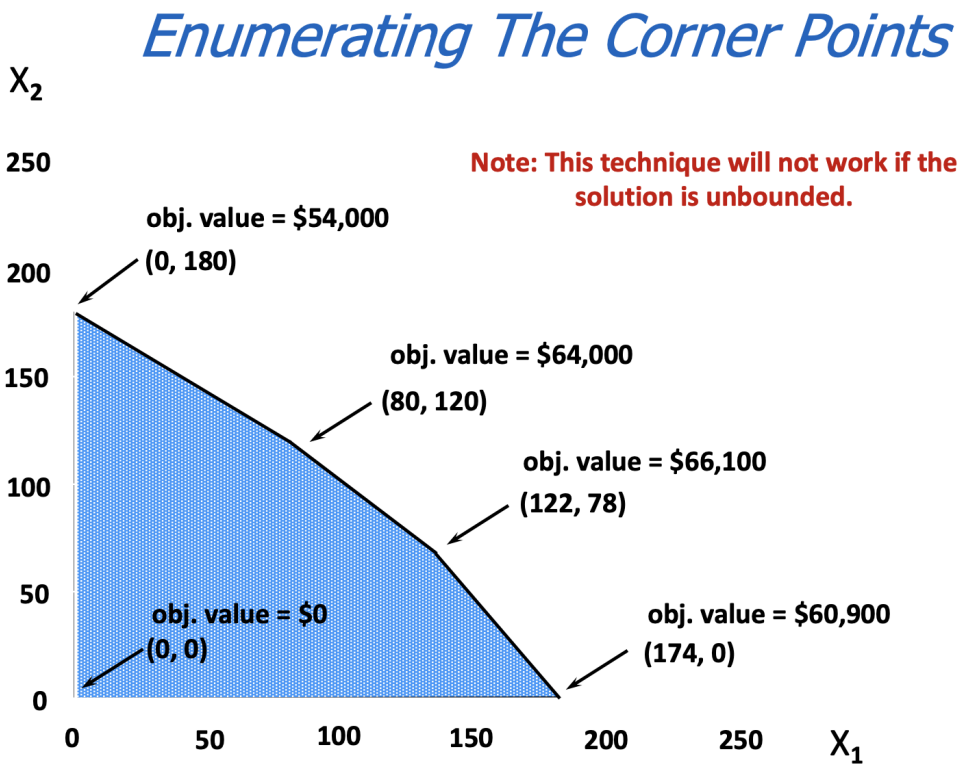
As mentioned before, at the end of this process, the optimal solution is identified.



We can then notice that the intersection between the first and second constraint. Le's now see the calculation for the optimal solution. As wee notice the optimal solution occurs where the "pumps" and "labor" constraints intersect. This occurs where: $X_1+X_2 = 200$ and $9X_1+6X_2 = 1566$. From the first one we have $X_2 = 200-X_1$. And now substituting it for X_2

in second one we have, $9X_1 + 6(200 - X_1) = 1566$ which reduces to $X_1 = 122$. So the optimal solution is, $X_1 = 122$, $X_2 = 200 - X_1 = 78$. Total Profit is $\$350 \cdot 122 + \$300 \cdot 78 = \$66,100$.

The optimal solution will always occur at a vertex of the feasible region, so another way to identify this solution is to identify all possible vertices and calculate the value of the objective function for each vertex. This search procedure could be tedious with a large number of constraints to apply and would fail if the solution is unbounded.



Let's now summarize the **graphical solution to linear programming problems**.

1. Plot the boundary line of each constraint
2. Identify the feasible region
3. Locate the optimal solution by either:
   (a) Plotting level curves
   (b) Enumerating the extreme points

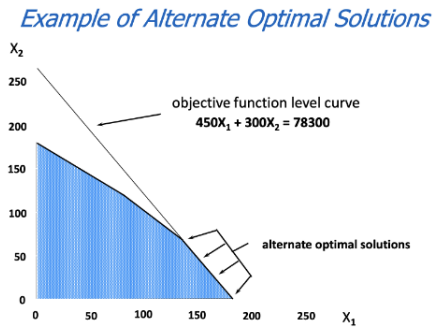
2.2.1 Special Conditions in LP Models

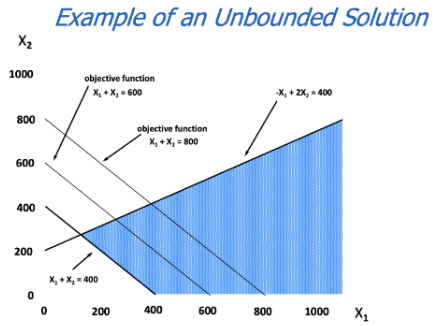
In a linear programming problem, various anomalies can arise, such as the presence of:

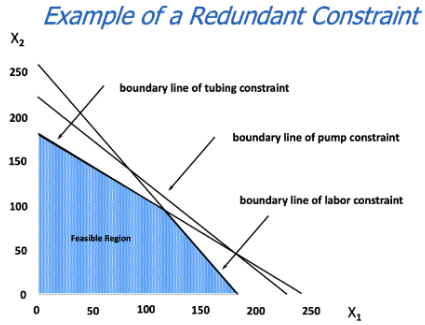
- **alternate optimal solutions**

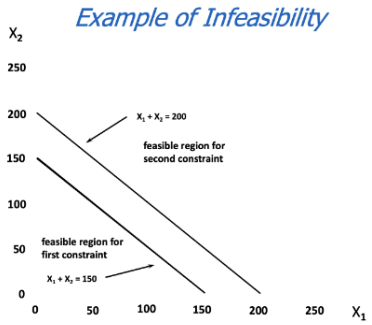
- **redundant constraints**
- **unbounded solutions**
- **infeasibility**

Alternate optimal solutions occur when one of the sides of the feasible region coincides with part of the objective function; in this case, all points on this segment will constitute an optimal solution. If there are constraints that do not contribute in any way to the definition of any of the sides of the feasible region, we have the condition of redundant constraints. When the feasible region extends into an infinite space, we cannot determine optimal vertices, the solutions are infinite. Finally, when the feasible region is empty, there are no possible solutions.









2.3 Linear Programming Assumptions

An LP objective function is linear; this results in the following two implications:

- **Proportionality**: contribution to the objective function from each decision variable is proportional to the value of the decision variable. E.g., contribution to profit from making 4 aqua-spas ($4 \times \$350$) is 4 times the contribution from making 1 aqua-spa ($350).

- **Additivity**: contribution to the objective function value from any decision variable is independent of the values of the other decision variables. E.g., no matter what the value of x_2, the manufacture of x_1 aqua-spas will always contribute $350x_1$ dollars to the objective function.

Analogously, since each constraint is a linear inequality or linear equation, the following implications result:

- **proportionality**: contribution of each decision variable to the left-hand side of each constraint is proportional to the value of the variable. E.g., it takes 3 times as many labor hours ($9 \times 3 = 27$ hours) to make 3 aqua-spas as it takes to make 1 aqua-spa ($9 \times 1 = 9$ hours).
- **Additivity**:the contribution of a decision variable to the left-hand side of a constraint is independent of the values of the other decision variables. E.g., no matter what the value of x_1 (number of aqua-spas produced), the production of x_2 hydro-luxes uses: x_2 pumps, $6x_2$ hours of labor, $16x_2$ feet of tubing.

No economies of scale, the resources required will always be proportional to the number of products to be produced.

There are other two important assumption in linear programming:

- **Divisibility Assumption**: each decision variable is allowed to assume fractional values.
- **Certainty Assumption**: each parameter (objective function coefficient c_j, right-hand side constant b_i of each constraint, and technology coefficient a_{ij}) is known with certainty.

When solving an LP problem, we typically make the assumption that the values of all model coefficients are known with absolute certainty. However, in reality, such certainty is a rare occurrence. This is where sensitivity analysis comes into play as it helps us to address questions regarding the degree to which the optimal solution is sensitive to changes in various coefficients within a model.

2.4 The Simplex Method

The simplex method is a widely used algorithm for solving linear programming (LP) problems. The method works by moving iteratively from one vertex of the feasible region to another along edges, until the optimal solution is found. The simplex method is highly efficient, especially for large-scale problems, and has been the basis for many other optimization algorithms.

To use the simplex method, we first *convert all inequalities to equalities* by *adding* **slack variables** to <= constraints and *subtracting* slack variables from >= constraints.

$$\begin{array}{ll} \text{For example:} & a_{k1}X_1 + a_{k2}X_2 + ... + a_{kn}X_n <= b_k \\ \text{converts to:} & a_{k1}X_1 + a_{k2}X_2 + ... + a_{kn}X_n + S_k = b_k \end{array}$$

$$\begin{array}{ll} \text{And:} & a_{k1}X_1 + a_{k2}X_2 + ... + a_{kn}X_n >= b_k \\ \text{converts to:} & a_{k1}X_1 + a_{k2}X_2 + ... + a_{kn}X_n - S_k = b_k \end{array}$$

Note that slack variables S with coefficient -1 are also called **Surplus variable**. The goal is to find the intersections of the lines that define the vertices of the feasible region, and then to find the optimal vertex.

For our example problem, seen in previous sections, we have:

$$\begin{array}{lll} \text{MAX:} & 350X_1 + 300X_2 & \}\text{profits} \\ \text{S.T.:} & 1X_1 + 1X_2 <= 200 & \}\text{pumps} \\ & 9X_1 + 6X_2 <= 1566 & \}\text{labor} \\ & 12X_1 + 16X_2 <= 2880 & \}\text{tubing} \\ & X_1, X_2, S_1, S_2, S_3 >= 0 & \}\text{non-negativity} \end{array}$$

Note that if there are n variables in a system of m equations (where $n > m$) we can select any m variables and solve the equations (setting the remaining $n - m$ variables to zero.)

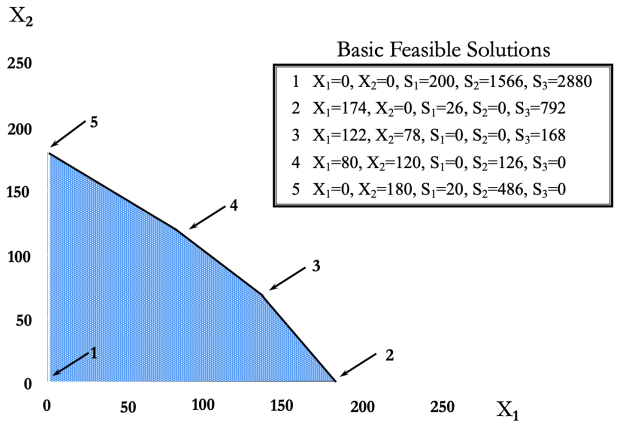
The following list represents the possible distributions between basic variables (the variables we want to solve for in the equation) and non-basic variables (the variables we set to zero for the solution). Not all solutions are feasible, but at least one of these is optimal.

	Basic Variables	Nonbasic Variables	Solution	Objective Value
1	S_1, S_2, S_3	X_1, X_2	X_1=0, X_2=0, S_1=200, S_2=1566, S_3=2880	0
2	X_1, S_1, S_3	X_2, S_2	X_1=174, X_2=0, S_1=26, S_2=0, S_3=792	60,900
3	X_1, X_2, S_3	S_1, S_2	X_1=122, X_2=78, S_1=0, S_2=0, S_3=168	66,100
4	X_1, X_2, S_2	S_1, S_3	X_1=80, X_2=120, S_1=0, S_2=126, S_3=0	64,000
5	X_2, S_1, S_2	X_1, S_3	X_1=0, X_2=180, S_1=20, S_2=486, S_3=0	54,000
6*	X_1, X_2, S_1	S_2, S_3	X_1=108, X_2=99, S_1=-7, S_2=0, S_3=0	67,500
7*	X_1, S_1, S_2	X_2, S_3	X_1=240, X_2=0, S_1=-40, S_2=-594, S_3=0	84,000
8*	X_1, S_2, S_3	X_2, S_1	X_1=200, X2=0, S_1=0, S_2=-234, S_3=480	70,000
9*	X_2, S_2, S_3	X_1, S_1	X_1=0, X_2=200, S_1=0, S_2=366, S_3=-320	60,000
10*	X_2, S_1, S_3	X_1, S2	X_1=0, X_2=261, S_1=-61, S_2=0, S_3=-1296	78,300

* denotes infeasible solutions

The last 5 solutions are not feasible because they do not satisfy the constraint of non-negativity.

Below is a graphical representation of our possible solutions.



To summarize, the Simplex Method works by first identifying any basic feasible solution (also known as an extreme point) for an LP problem, and then moving to an adjacent extreme point if doing so would improve the value of the objective function. The process of moving from one extreme point to an adjacent one involves switching one of the basic variables with one of the nonbasic variables, resulting in a new basic feasible solution for the adjacent extreme point. When no adjacent extreme point yields a better objective function value, the algorithm stops and the current extreme point is deemed optimal.

As the algorithm scans through adjacent points, it may not necessarily take the best path to finding the optimal solution, as illustrated by the previous example where an implementation of the simplex method could have gone from the first solution to the fifth and so on, in reverse order. However, although the algorithm may not follow the best path, it is guaranteed to converge to the optimal solution eventually.

In the previous example, starting from the combination of basic variables (S_1, S_2, S_3) and substituting S_1 with X_1 to obtain the combination (X_1, S_2, S_3) as the basic variables, the simplex method moved from the first to the second solution. This substitution resulted in an improvement in the objective function and was thus chosen as the best substitution for that iteration (as well as for subsequent iterations, as can be seen from the previous table).

2.5 Sensitivity Analysis

When solving an LP problem we assume that values of all model coefficients are known with certainty, but such certainty rarely exists. **Sensitivity analysis** helps answer questions about how sensitive the optimal solution is to changes in various coefficients in a model.

$$\begin{array}{ll} \text{MAX (or MIN):} & c_1X_1 + c_2X_2 + ... + c_nX_n \\ \text{Subject to:} & a_{11}X_1 + a_{12}X_2 + ... + a_{1n}X_n <= b_1 \\ & \vdots \\ & a_{k1}X_1 + a_{k2}X_2 + ... + a_{kn}X_n >= b_k \\ & \vdots \\ & a_{m1}X_1 + a_{m2}X_2 + ... + a_{mn}X_n = b_m \end{array}$$

We want to answer the question of how sensitive a solution is to change in the c_i, a_{ij}, and b_i.

Two different approach are possible. The first one is to change the data and re-solve the model, and sometimes this is the only practical approach; the second one is using solver (simplex method), that also produces sensitivity reports that can answer various questions.

Sensitivity Report

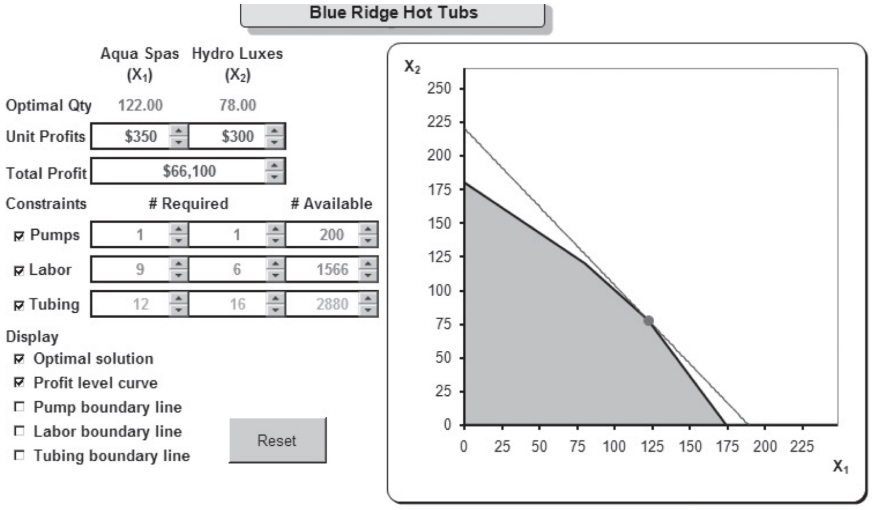
When we compute a solver's sensitivity report we want to answers questions about:

- Amounts by which objective function coefficients can change without changing the optimal solution.
- The impact on the optimal objective function value of changes in constrained resources.
- The impact on the optimal objective function value of forced changes in decision variables.
- The impact changes in constraint coefficients will have on the optimal solution.

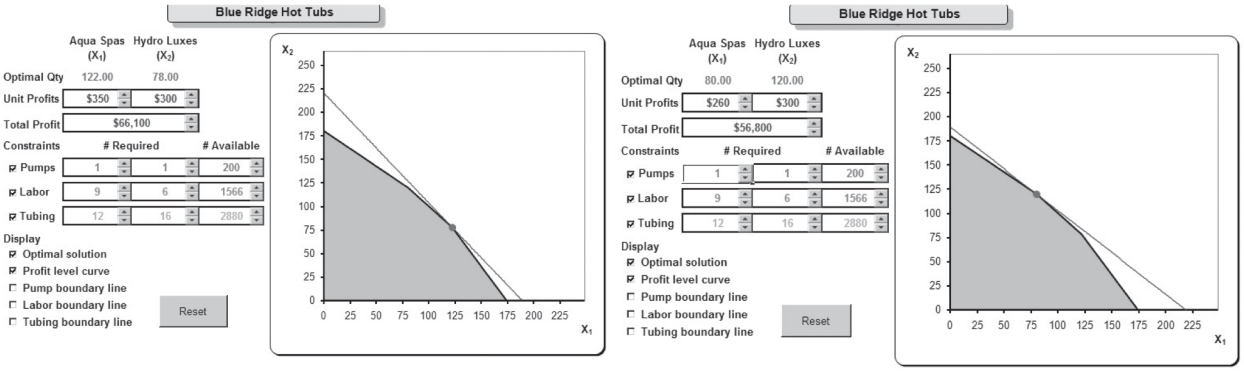
In order to show this method let's use the same example as before.

$$\begin{array}{lll} \text{MAX:} & 350X_1 + 300X_2 & \}\text{profits} \\ \text{S.T.:} & 1X_1 + 1X_2 <= 200 & \}\text{pumps} \\ & 9X_1 + 6X_2 <= 1566 & \}\text{labor} \\ & 12X_1 + 16X_2 <= 2880 & \}\text{tubing} \\ & X_1, X_2, S_1, S_2, S_3 >= 0 & \}\text{non-negativity} \end{array}$$

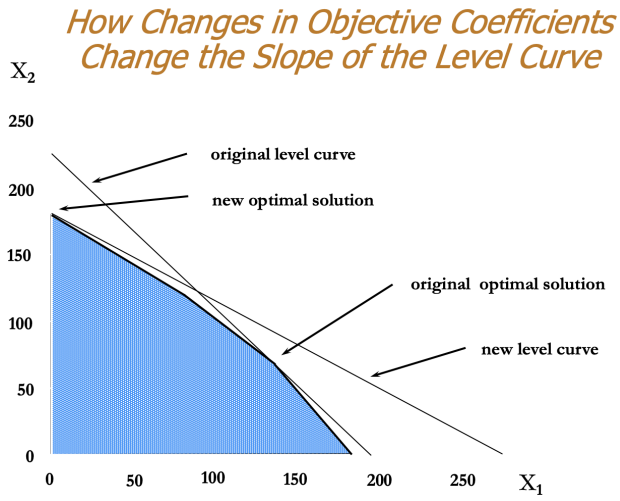
Solving it we get the following:



One thing to do is to identify the so-called binding constraints, which are those constraints whose line rests on the optimal solution. In this case, any change to the constraint will also change the optimal solution. With respect to the previous example, it can be seen that the optimal solution rests on the vertex, which is the intersection of two constraints, Pump and Labor. The third constraint is not used by the optimal solution. In the case of binding constraints, all of the constraint's resources will be used. In fact, from the previous example, 200 pumps and 1566 hours of work will be used. Otherwise, there will be remainders. Using equations with slack variables, as used in the simplex method, we can say that the binding constraint is such that its slack variable is 0. If the slack variables of the constraint is positive, it represents the difference between the LHS and RHS of the constraint (essentially the unused resources). Binding constraints prevent the objective function from achieving higher values. The values of the slack variables indicate the difference between the LHS and RHS of each constraint.



We have already seen how changing the coefficients of the objective function can change the slope, since the slope of the line determines which vertex of the feasible region constitutes the optimal solution.



In order to get the **sensitivity report** we just need this command.

```
printSensitivityObj(model)
## Objs Sensitivity
## 1 C1 300 <= C1 <= 450
## 2 C2 233.333333333333 <= C2 <= 350
```

Name	Final Value	Reduced Cost	Objective Coefficient	Allowable Increase	Allowable Decrease
Number to make Aqua-Spas	122	0	350	100	50
Number to make Hydro-Luxes	78	0	300	50	66.66667

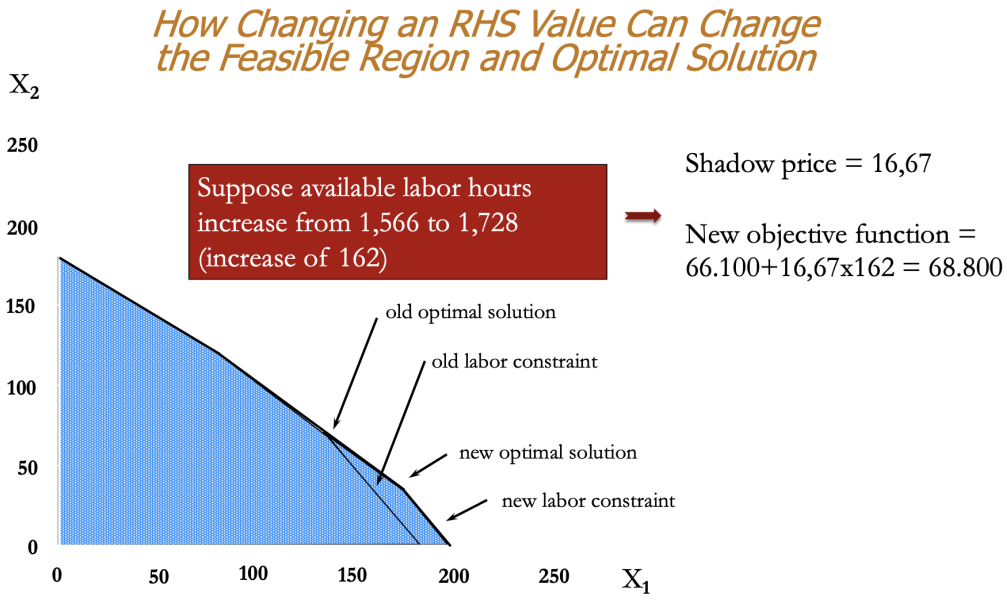
The objective coefficient C1 was 350 therefore it can increase up to \$100 or decrease up to \$50 without changing the optimal solution assuming all other coefficients remain constant. Similarly, the objective function value C2 can increase by \$50 or decrease by \$66.67 without changing the optimal values of the decision variables, assuming all other coefficients remain constant.

These are called **optimality ranges** indicate the amounts by which an objective function coefficient can change without changing the optimal solution, *assuming all other coefficients remain constant.*When objective function coefficient are changed the feasible region remain unchanged. If the range of this variability (increase or decrease) is 0 then an alternate optimal solution exists (if we are not in the special case of a degenerate solution). That is, it means that, also a little change of the coefficient may cause a change of the optimal solution.

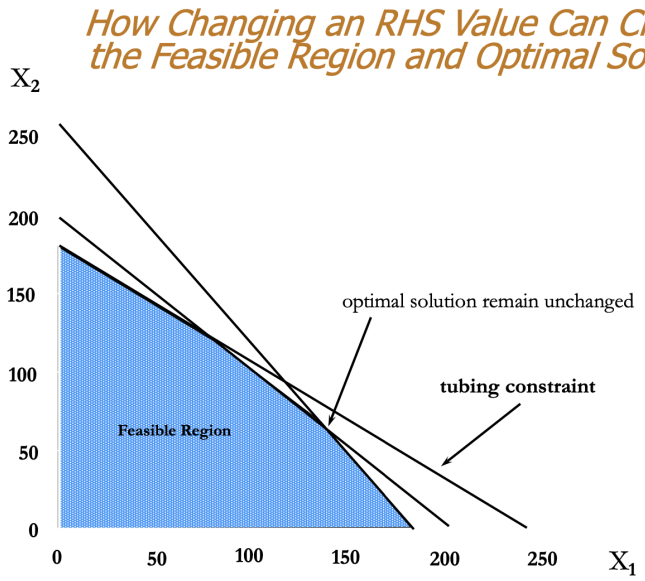
Changes in Constraint RHS Values

The **shadow price** of a constraint indicates the amount by which the objective function value changes given a unit increase in the RHS value of the constraint, *assuming all other coefficients remain constant*. Shadow prices hold only within RHS changes falling within a range, and for nonbinding constraints they are always zero.

Name	Final Value	Shadow Price	Constraint R.H. Side	Allowable Increase	Allowable Decrease
Pumps Req'd Used	200	200	200	7	26
Labor Req'd Used	1566	17	1566	234	126
Tubing Req'd Used	2712	0	2880	1E+30	168



Shadow prices only indicate the changes that occur in the objective function value as the right-hand side (RHS) values change. However, changing a RHS value for a binding constraint not only affects the objective function value but also changes the feasible region and the optimal solution (as shown in the following slide). Therefore, to find the optimal solution after changing a binding RHS value, you must re-solve the problem.



Let's now see other uses of shadow prices.
Suppose a new Hot Tub (the Typhoon-Lagoon) is being considered. It generates a marginal profit of \$320 and requires:

- 1 pump (shadow price = \$200)
- 8 hours of labor (shadow price = \$16.67)
- 13 feet of tubing (shadow price = \$0)

We wonder if it would be profitable to produce any hot tub. The answer is no because:
$\$320 - \$200 * 1 - \$16.67 * 8 - \$0 * 13 = -\$13.33 = No!$

The **reduced cost** for each product equals its per-unit marginal profit minus the per-unit value of the resources it consumes (priced at their shadow prices).

Type of Problem	Optimal Value of Decision Variable	Optimal Value of Reduced Cost
Maximization	at simple lower bound	$<= 0$
	between lower and upper bounds	$= 0$
	at simple upper bound	$>= 0$
Minimization	at simple lower bound	$>= 0$
	between lower and upper bounds	$= 0$
	at simple upper bound	$<= 0$

Let's sum up the key points:

- The shadow prices of resources equate the marginal value of the resources consumed with the marginal benefit of the goods being produced.
- Resources in excess supply have a shadow price (or marginal value) of zero.
- The reduced cost of a product is the difference between its marginal profit and the marginal value of the resources it consumes.
- Products whose marginal profits are less than the marginal value of the goods required for their production will not be produced in an optimal solution.

Analyzing changes in constraint coefficients, we can answer some other questions.
For example suppose a Typhoon-Lagoon required only 7 labor hours rather than 8, we wonder if it is now profitable to produce any. The answer is yes, because:
$\$320 - \$200 * 1 - \$16.67 * 7 - \$0 * 13 = \$3.31 = Yes!$
We also wonder what is the maximum amount of labor Typhoon-Lagoons could require and still be profitable. The answer is that we need
$\$320 - \$200 * 1 - \$16.67 * L3 - \$0 * 13 >= 0$
and it is true if $L + 3 <= \$120/\$16.67 = 7.20$

Simultaneous Changes in Objective Function Coefficients

The **100% Rule** can be used to determine if the optimal solutions changes when more than one objective function coefficient changes. Two cases can occur:

- Case 1: All variables with changed obj. coefficients have nonzero reduced costs.
- Case 2: At least one variable with changed obj. coefficient has a reduced cost of zero.

In the first case, the solution will remain optimal as long as the changes in the coefficients fall within their respective range of admissibility (the ranges that stabilize the so-called shadow prices). In the second case, however, we need to perform a simple algorithm to study the impact on the optimal solution. For each changed coefficient, we calculate the percentage of change with respect to the related admissible range (either for an increase or a decrease). So in the second case for each variable **compute objective function coefficients**:

$$r_j \begin{cases} \frac{\Delta c_j}{I_j} & \text{if } \Delta c_j >= 0 \\ \frac{-\Delta c_j}{D_j} & \text{if } \Delta c_j < 0 \end{cases}$$

where:

- Δc_j: the change made to the coefficient c_j
- I_j: the possible range of increase
- D_j: the possible range of decrease

If more than one objective function coefficient changes, the current solution remains optimal provided the r_j sum to $<= 1$, but if the r_j sum to > 1, the current solution, might remain optimal, but this is not guaranteed.

The solution to an LP problem is **degenerate** if the Allowable Increase of Decrease on any constraint is zero (0).
When the solution is degenerate:

1. The methods mentioned earlier for detecting alternate optimal solutions cannot be relied upon.
2. The reduced costs for the variables may not be unique. Also, the objective function coefficients for the variables must change by at least as much as (and possibly more than) their respective reduced costs before the optimal solution would change.
3. The allowable increases and decreases for the objective function coefficients still hold and, in fact, the coefficients may have to be changed beyond the allowable increase and decrease limits before the optimal solution changes.
4. The given shadow prices and their ranges may still be interpreted in the usual way but they may not be unique. That is, a different set of shadow prices and ranges may also apply to the problem (even if the optimal solution is unique).

Chapter 3

Integer Linear Programming

3.1 Integer Programming

When one or more variables in an LP problem must assume an **integer** value we have an Integer Linear Programming (ILP) problem. ILPs occur frequently when scheduling workers or when manufacturing airplanes. Integer variables also allow us to build more accurate models for a number of common problems.

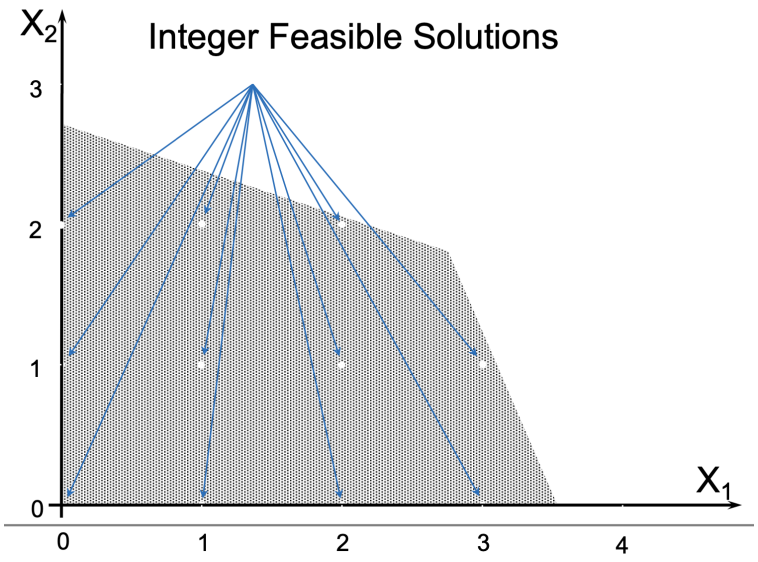
- Original ILP

MAX: $2X_1 + 3X_2$
S.T.: $X_1 + 3X_2 <= 8.25$
$2.5X_1 + X_2 <= 8.75$
$X_1, X_2 >= 0$
X_1, X_2 must be integers

- LP Relaxation

MAX: $2X_1 + 3X_2$
S.T.: $X_1 + 3X_2 <= 8.25$
$2.5X_1 + X_2 <= 8.75$
$X_1, X_2 >= 0$

Integrality conditions are easy to state but make the problem much more difficult (and sometimes impossible) to solve.



When solving an LP relaxation, sometimes you "get lucky" and obtain an integer feasible solution. This was the case in the original Blue Ridge Hot Tubs problem in earlier chapters. But what if we reduce the amount of labor available to 1520 hours and the amount of tubing to 2650 feet?

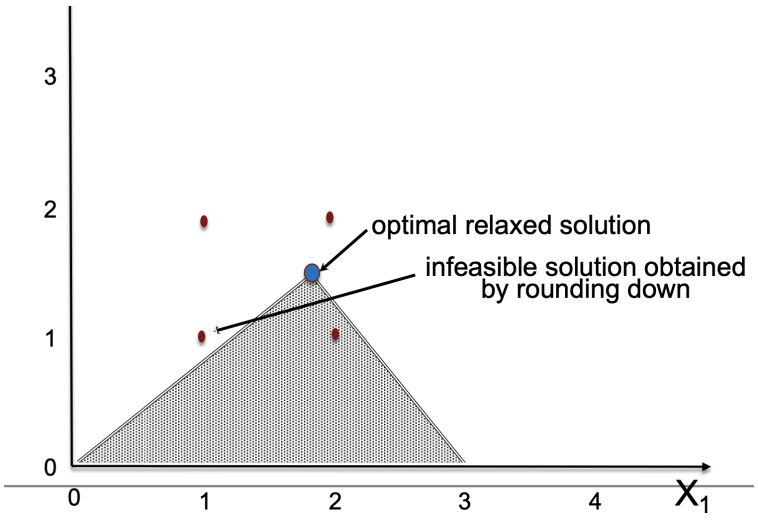
MAX:	$350X_1 + 300X_2$	}profits
S.T.:	$1X_1 + 1X_2 <= 200$	}pumps
	$9X_1 + 6X_2 <= 1520$	}labor
	$12X_1 + 16X_2 <= 2650$	}tubing
	$X_1, X_2 >= 0$	}non-negativity
	X_1, X_2 must be integers	}integrality

Optimal solution for the relazed problem:

$X_1 = 116,9444 \quad X_2 = 77,9167$

Corresponding to a maximum profit of \$64306.

It is tempting to simply round a fractional solution to the closest integer solution, but in general, this does not work because the rounded solution may be infeasible, or the rounded solution may be suboptimal.



There are various methods available to solve ILP, but the most widely used one is known as the **Branch-and-Bound** (B&B) technique. This approach involves solving a sequence of LP problems called *candidate problems*. Although it is theoretically capable of solving any ILP, practically speaking, it often requires a considerable amount of computational effort and time.

Since the B&B method can be quite time-consuming, most ILP packages offer the option of specifying a suboptimality tolerance factor. This enables you to stop once an integer solution is found that is within a certain percentage of the global optimal solution. The bounds obtained from LP relaxations are especially useful in this regard. For instance, let's say that the optimal objective function value of the LP relaxation is \$64,306. In that case, a 95% approximation of this value would be \$61,090. Consequently, any integer solution with an objective function value of \$61,090 or higher must be within 5% of the optimal solution.

In summary, we can draw the following conclusions:

- The optimal solution to an LP relaxation of an ILP problem provides us with an *estimate* of the optimal objective function value.
- For *maximization* problems, the optimal relaxed objective function value is an *upper bound* on the optimal integer value.
- For *minimization* problems, the optimal relaxed objective function value is a *lower bound* on the optimal integer value.

3.2 Branch and Bound

We want to solve the same integer linear programming problem as before:

MAX: $2X_1 + 3X_2$
S.T.: $X_1 + 3X_2 <= 8.25$
$2.5X_1 + X_2 <= 8.75$
$X_1 + X_2 >= 0$
X_1, X_2 must be integers

In order to solve it we use the following algorithm.

Step 1 **INITIALIZATION**

(a) Relax all the integrality conditions in ILP and solve the resulting LP problem. If the optimal solution to the relaxed LP problem happens to satisfy the original integrality conditions, stop because this is the optimal integer solution.

(b) Otherwise, proceed to **step 2**.

- If the problem being solved is a maximization problem, let $Z_{best} = -\infty$.
- If it is a minimization problem, let $Z_{best} = \infty$.

In general Z_{best} represents the objective function value of the best known integer solution as the algorithm proceeds.

Step 2 **BRANCHING**

Let X_j represents one of the variables that violated the integrality conditions in the optimal solution to the problem that was solved most recently and let b_j represent its non-integer value. Let $INT(b_j)$ represent the largest integer that is less than b_j.

Create two new candidate problems:

- one by appending the constraint $X_j \leq INT(b_j)$ to the most recently solved LP problem,
- and the other by appending the constraint $X_j \geq INT(b_j)+1$ to the most recently solved LP problem.

Place both of these new LP problems in a list of candidate problems to be solved.

Step 3 **SOLVE THE RELAXED SUBPROBLEMS AND BOUNDING**

(a) If the list of candidate problems is empty, proceed to **step 4** (**STOP**). Otherwise, remove a candidate problem from the list, relax any integrality conditions in the problem, and solve it.

(b) If there is not a solution to the current candidate problem (that is, it is infeasible), proceed to **step 3(a)** Otherwise, let Z_{cp} denote the optimal objective function value for the current candidate problem.

(c) If Z_{cp} is not better than Z_{best} (for a maximization problem $Z_{cp} \leq Z_{best}$ or for a minimization problem $Z_{cp} \geq Z_{best}$), proceed to **step 3(a)**.

(d) If the solution to the current candidate problem does not satisfy the original integrality conditions, and Z_{cp} is better than Z_{best} proceed to **step 2** (**BRANCHING**).

(e) If the solution to the current candidate problem does satisfy the original integrality conditions, a better integer solution has been found. Thus, let $Z_{best} = Z_{cp}$ and save the solution obtained for this candidate problem. Then go back to **step 3**.

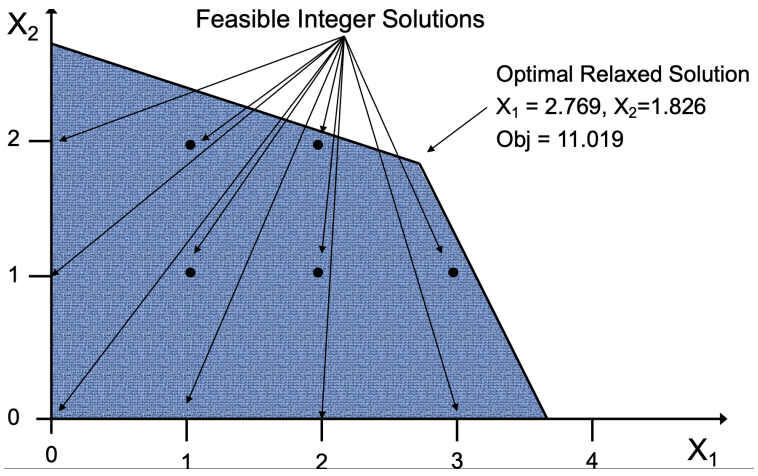
Step 4 **STOP**

The optimal solution has been found and has an objective function value given by the current value of Z_{best}.

Now we can proceed with B&B in order to find the optimal solution. As suggested in the previous sections, we first solve the relaxed linear problem, which excludes the integrality constraints on the decision variables.

- LP Relaxation

MAX: $2X_1 + 3X_2$
S.T.: $X_1 + 3X_2 <= 8.25$
$2.5X_1 + X_2 <= 8.75$
$X_1, X_2 >= 0$



Then, the algorithm is divided into two phases, the first being the Branch phase. A variable (X_1 or X_2) is selected, and the problem is decomposed into two subproblems.

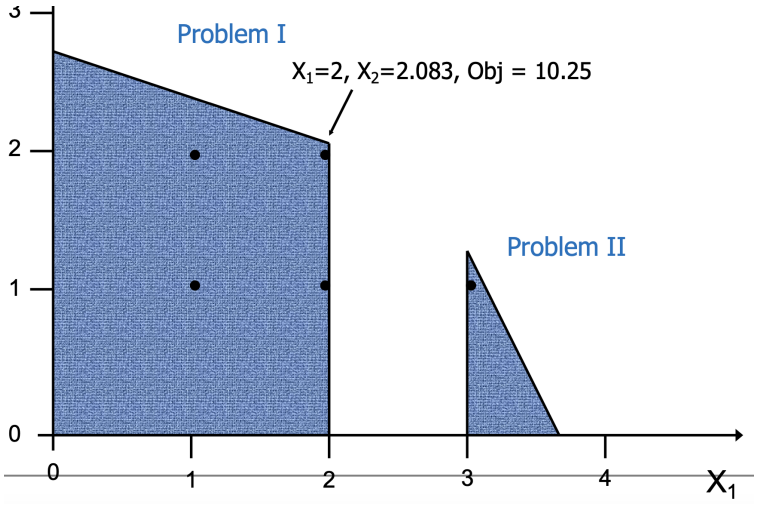
- Problem I

MAX: $2X_1 + 3X_2$
S.T.: $X_1 + 3X_2 <= 8.25$
$2.5X_1 + X_2 <= 8.75$
$X_1 <= 2$
$X_1, X_2 >= 0$
X_1, X_2 must be integers

- Problem II

MAX: $2X_1 + 3X_2$
S.T.: $X_1 + 3X_2 <= 8.25$
$2.5X_1 + X_2 <= 8.75$
$X_1 >= 3$
$X_1, X_2 >= 0$
X_1, X_2 must be integers

By selecting X_1 (since in the optimal solution, X_1 has a value of 2.7), we divide the problem by forcing X_1 to take a value $<= 2$ for one subproblem and $>= 3$ for the second subproblem.



This eliminates part of the feasible region but ensures that the optimal solution for the subproblem contains either $X_1 = 2$ or $X_1 = 3$.

Looking at the first subproblem, we can see that the solution is not integer, which is why we apply the branching phase again, generating two new subproblems by selecting X_2.

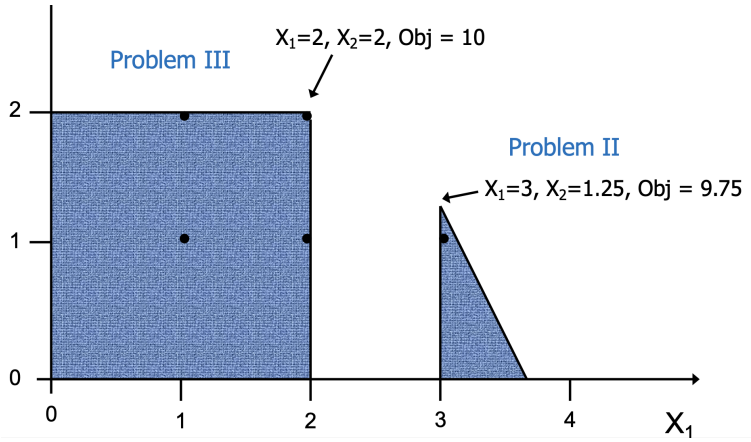
- Problem III

MAX: $2X_1 + 3X_2$
S.T.: $X_1 + 3X_2 <= 8.25$
$2.5X_1 + X_2 <= 8.75$
$X_1 <= 2$
$X_2 <= 2$
$X_1, X_2 >= 0$
X_1, X_2 must be integers

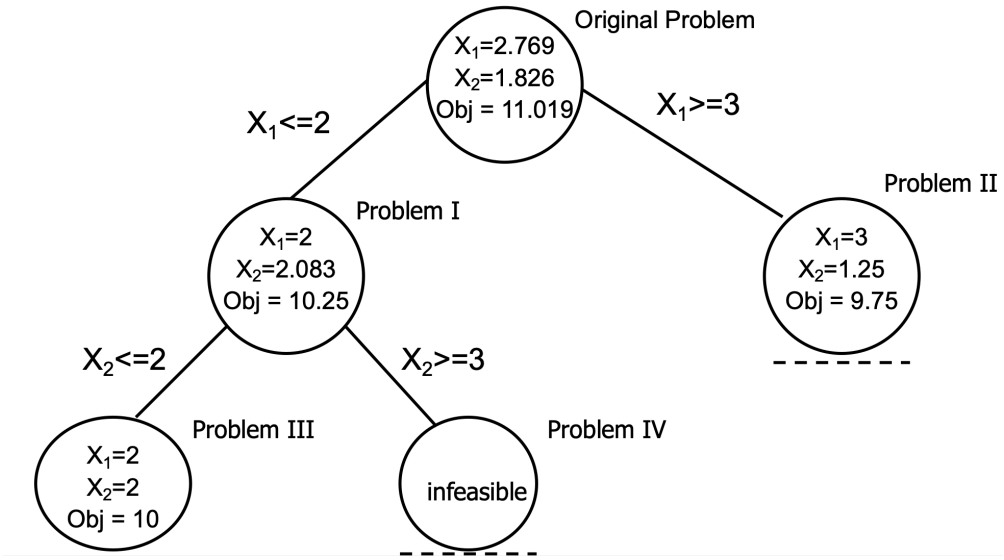
- Problem IV

MAX: $2X_1 + 3X_2$
S.T.: $X_1 + 3X_2 <= 8.25$
$2.5X_1 + X_2 <= 8.75$
$X_1 <= 2$
$X_2 >= 3$
$X_1, X_2 >= 0$
X_1, X_2 must be integers

Since problem 4 is unreasonable as there are no values of $X_2 >= 3$ that satisfy the given constraints, we take subproblem 3.



The relaxed solution for problem 3 becomes an integer optimal solution for problem 1. An optimization procedure can be applied to avoid exploring problem 2 since its subproblems can never generate an optimal function value greater than 9.75, which means that they can never exceed the current optimal solution found in problem 1.


Original Problem
X1=2.769
X2=1.826
Obj = 11.019
X1<=2
X1>=3
Problem I
X1=2
X2=2.083
Obj = 10.25
Problem II
X1=3
X2=1.25
Obj = 9.75
X2<=2
X2>=3
Problem III
X1=2
X2=2
Obj = 10
Problem IV
infeasible

Chapter 4

Network Models

4.1 Introduction

In many business scenarios, it is helpful to represent the problem graphically as a network consisting of nodes and arcs. A network is a collection of interconnected nodes, and in the context of business problems, nodes represent different entities such as factories, warehouses, customers, or distribution centers.

A number of business problems can be represented graphically as networks. This chapter focuses on several such problems:

- **Transshipment Problems**: Transshipment refers to the movement of goods from one location to another, usually via a third location. In a transshipment problem, the nodes represent sources, destinations, and intermediate points, and the arcs represent the possible routes that the goods can take. The goal of a transshipment problem is to find the optimal way to move the goods through the network while minimizing the cost.
- **Shortest Path Problems**: The shortest path problem involves finding the shortest path between two nodes in a network, where the length of each arc represents a cost or distance. The shortest path problem is often used in logistics and transportation to determine the most efficient route between two points.
- **Maximal Flow Problems**: Maximal flow problems involve finding the maximum amount of flow that can pass through a network from a source to a destination. The nodes in a maximal flow problem represent sources, destinations, and intermediate points, while the arcs represent the capacity of the network to carry flow. The goal of a maximal flow problem is to find the maximum amount of flow that can pass through the network while satisfying the capacity constraints.

- **Transportation/Assignment Problems**: Transportation and assignment problems are two related problems that can also be represented graphically as networks. In a transportation problem, the goal is to transport goods from sources to destinations while minimizing the transportation cost. In an assignment problem, the goal is to assign tasks to workers or resources while minimizing the total cost or time required to complete the tasks.
- **Generalized Network Flow Problems**: Generalized network flow problems are a more complex type of problem that involves finding the optimal flow through a network while satisfying a set of constraints. The constraints can include capacity constraints, flow conservation constraints, and other types of constraints.
- **The Minimum Spanning Tree Problem**: The minimum spanning tree problem involves finding the shortest path that connects all the nodes in a network. In this problem, the goal is to minimize the total distance or cost required to connect all the nodes in the network.

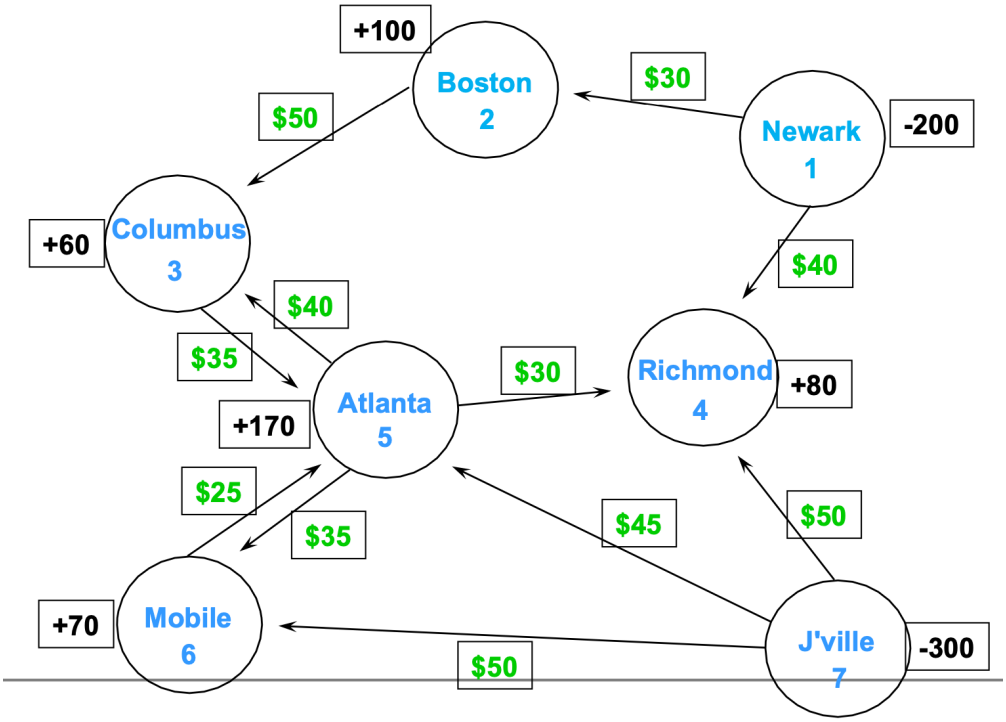
In a network, there are three types of nodes:

- **Supply**
- **Demand**
- **Transshipment**

The supply nodes represent sources of goods or materials, while the demand nodes represent destinations or customers. Transshipment nodes are intermediate points where goods can be transferred from one route to another. To represent supplies, *negative* numbers are used, and *positive* numbers represent demands.

4.2 Transshipment Problem

Let's imagine a company, The Bavarian Motor Company, which produces luxury cars to export them to the United States. The cars are exported to the ports of Newark and Jacksonville, and from these ports, they are then transported by distributors to the cities of Mobile, Atlanta, Columbus, Boston, and Richmond. The arcs of the graph indicate the possible routes for transportation, and in green, the cost of transportation following those routes.



There are 200 cars available at the port of Newark and 300 at the port of Jacksonville. The number of cars that the distributors need is shown in black (positive).

In a network, each arc has a direction. The nodes Newark and Jacksonville are the supply nodes, while the others are demand nodes.

The objective of a network problem is to determine the number of objects that flow through the arcs. By transforming it into a linear programming problem, the *arcs become the decision variables*. For each arc in a network flow model we define a decision variable as:

$$X_{ij} = \text{ the amount being shipped (or flowing) from node } i \text{ to node } j$$

For example here we have that:
X_{12} = the number of cars shipped from node 1 (Newark) to node 2 (Boston)
X_{56} = the number of cars shipped from node 5 (Atlanta) to node 6 (Mobile)
The number of arcs determines the number of variables.

Compared to the graph represented before, the objective function is the following:

$$\text{MIN:} \quad 30X_{12} + 40X_{14} + 50X_{23} + 35X_{35} + 40X_{53} + 30X_{54} + 35X_{56} + 25X_{65} + 50X_{74} + 45X_{75} + 50X_{76}$$

The objective is to minimize the total cost. The nodes, on the other hand, will affect the constraints. There must be a constraint for each node.

For Minimum Cost Network Flow Problems Where:	Apply This Balance-of-Flow Rule At Each Node:
Total Supply > Total Demand	Inflow−Outflow >= Supply or Demand
Total Supply < Total Demand	Inflow−Outflow <= Supply or Demand
Total Supply = Total Demand	Inflow−Outflow = Supply or Demand

Based on the correct inequality between supply and demand, the correct rule must be applied for each node.

In the BMC problem we have that:

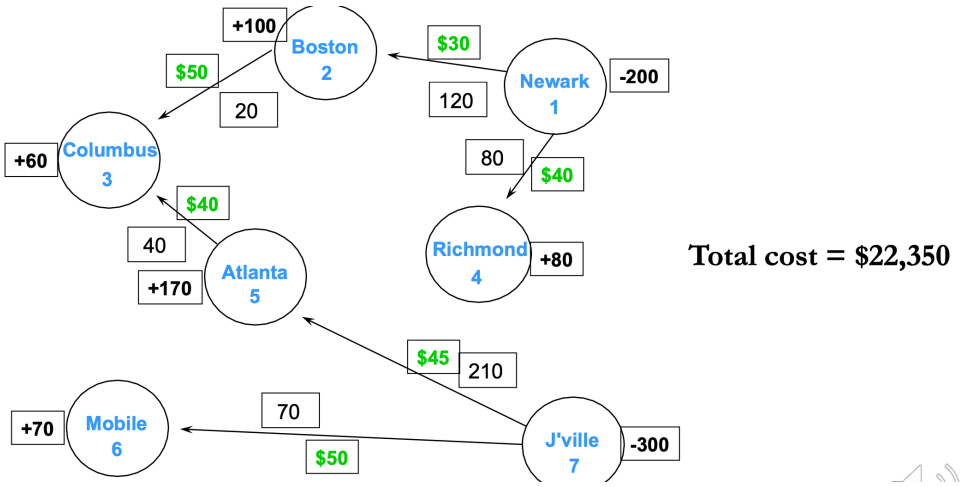
- Total Supply = 500 cars
- Total Demand = 480 cars

Where the supply is greater then the demand. For each node we need a constraint like the following one:
Inflow − Outflow >= Supply or Demand.
For example in the node 1 we have that $-X_{12} - X_{14} >= -200$, and it is equivalent to $+X_{12} + X_{14} <= 200$. Note that there is no inflow for node 1.

By following this approach we can define the constraint for each node.

Flow constraints	
$-X_{12}-X_{14} >= -200$	} node 1
$+X_{12}-X_{23} >= +100$	} node 2
$+X_{23} + X_{53}-X_{35} >= +60$	} node 3
$+X_{14} + X_{54} + X_{74} >= +80$	} node 4
$+X_{35} + X_{65} + X_{75}-X_{53}-X_{54}-X_{56} >= +170$	} node 5
$+X_{56} + X_{76}-X_{65} >= +70$	} node 6
$-X_{74}-X_{75}-X_{76} >= -300$	} node 7
Non-negativity conditions	
$X_{ij} >= 0$ for all i, j	

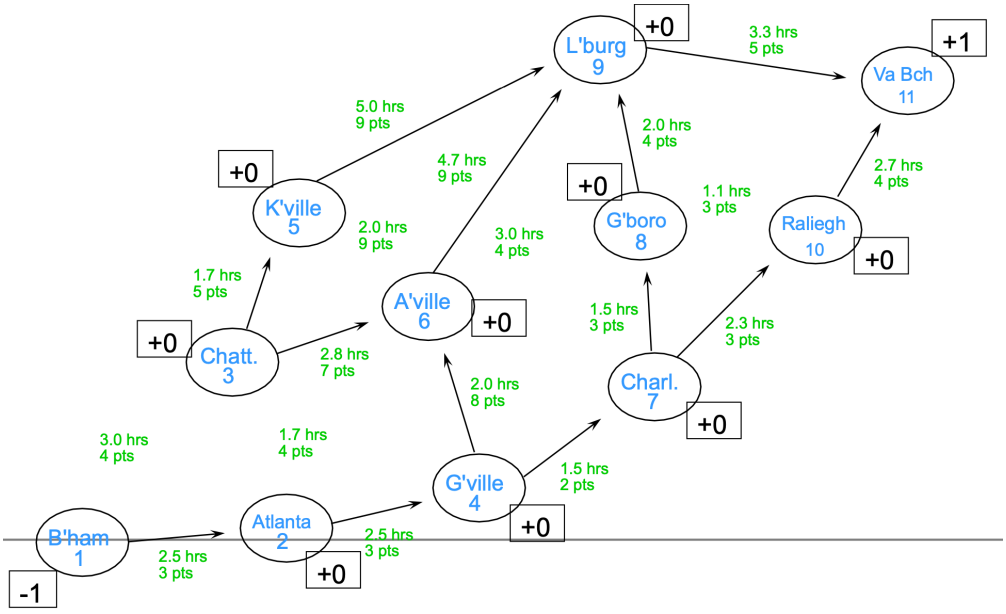
By this definition of the problem we can get the optimal solution.



4.3 Shortest Path Problem

Many decision problems boil down to determining the **shortest** (or least costly) route or path through a network. The Vehicle Routing Problem is a particular case of transshipment where:

- There is one supply node with a supply of -1
- There is one demand node with a demand of $+1$
- All other nodes have supply/demand of $+0$



In this graph, we can see the aforementioned characteristics. On the arcs, we have the cost in terms of time and a score that identifies the scenicness of the road.

There are two possible objectives for this problem:

- Finding the quickest route (*minimizing* travel time)
- Finding the most scenic route (*maximizing* the scenic rating points)

$$\begin{aligned}\text{MIN:}\quad & +2.5X_{12} + 3X_{13} + 1.7X_{23} + 2.5X_{24} + 1.7X_{35} + 2.8X_{36} + 2X_{46} + 1.5X_{47} + 2X_{56} \\ & +5X_{59} + 3X_{68} + 4.7X_{69} + 1.5X_{78} + 2.3X_{7,10} + 2X_{89} + 1.1X_{8,10} + 3.3X_{9,11} + 2.7X_{10,11}\end{aligned}$$

Subject to:

$-X_{12} - X_{13}$	$= -1$	} flow constraint for node 1
$+X_{12} - X_{23} - X_{24}$	$= 0$	} flow constraint for node 2
$+X_{13} + X_{23} - X_{35} - X_{36}$	$= 0$	} flow constraint for node 3
$+X_{24} - X_{46} - X_{17}$	$= 0$	} flow constraint for node 4
$+X_{35} - X_{56} - X_{59}$	$= 0$	} flow constraint for node 5
$+X_{36} + X_{46} + 756 - X_{68} - X_{69}$	$= 0$	} flow constraint for node 6
$+X_{47} - X_{78} - X_{7,10}$	$= 0$	} flow constraint for node 7
$+X_{68} + X_{78} - X_{89} - X_{8,10}$	$= 0$	} flow constraint for node 8
$+X_{59} + X_{69} + X_{89} - X_{9,11}$	$= 0$	} flow constraint for node 9
$+X_{7,10} + X_{8,10} - X_{10,11}$	$= 0$	} flow constraint for node 10
$+X_{9,11} + X_{10,11}$	$= +1$	} flow constraint for node 11
$Xi \geq O$ for all i and j	} nonnegativity conditions	

A peculiar aspect of this type of problems is that if the coefficients are integer, then also the optimal solution will be integer. Here we can see the optimal solution for the problem just specified.

Select Route?	From		To		Driving Time	Scenic Rating
1.0	1	Birmingham	2	Atlanta	2.5	3
0.0	1	Birmingham	3	Chattanooga	3.0	4
0.0	2	Atlanta	3	Chattanooga	1.7	4
1.0	2	Atlanta	4	Greenville	2.5	3
0.0	3	Chattanooga	5	Knoxville	1.7	5
0.0	3	Chattanooga	6	Asheville	2.8	7
0.0	4	Greenville	6	Asheville	2.0	8
1.0	4	Greenville	7	Charlotte	1.5	2
0.0	5	Knoxville	6	Asheville	2.0	9
0.0	5	Knoxville	9	Lynchburg	5.0	9
0.0	6	Asheville	8	Greensboro	3.0	4
0.0	6	Asheville	9	Lynchburg	4.7	9
0.0	7	Charlotte	8	Greensboro	1.5	3
1.0	7	Charlotte	10	Raleigh	2.3	3
0.0	8	Greensboro	9	Lynchburg	2.0	4
0.0	8	Greensboro	10	Raleigh	1.1	3
0.0	9	Lynchburg	11	Virginia Beach	3.3	5
1.0	10	Raleigh	11	Virginia Beach	2.7	4
				Total	11.5	15

In case we want instead optimize the scenic route the solution will be the following:

Select Route?	From		To		Driving Time	Scenic Rating
1.0	1	Birmingham	2	Atlanta	2.5	3
0.0	1	Birmingham	3	Chattanooga	3.0	4
1.0	2	Atlanta	3	Chattanooga	1.7	4
0.0	2	Atlanta	4	Greenville	2.5	3
1.0	3	Chattanooga	5	Knoxville	1.7	5
0.0	3	Chattanooga	6	Asheville	2.8	7
0.0	4	Greenville	6	Asheville	2.0	8
0.0	4	Greenville	7	Charlotte	1.5	2
1.0	5	Knoxville	6	Asheville	2.0	9
0.0	5	Knoxville	9	Lynchburg	5.0	9
0.0	6	Asheville	8	Greensboro	3.0	4
1.0	6	Asheville	9	Lynchburg	4.7	9
0.0	7	Charlotte	8	Greensboro	1.5	3
0.0	7	Charlotte	10	Raleigh	2.3	3
0.0	8	Greensboro	9	Lynchburg	2.0	4
0.0	8	Greensboro	10	Raleigh	1.1	3
1.0	9	Lynchburg	11	Virginia Beach	3.3	5
0.0	10	Raleigh	11	Virginia Beach	2.7	4
				Total	15.9	35

4.3.1 Equipment Replacement Problem

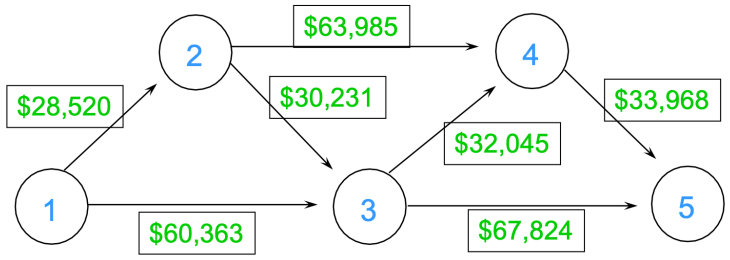
The problem of determining when to replace equipment is another common business problem. It can also be modeled as a shortest path problem.

Let's consider the case of a company, Compu-Train, which provides software tutoring. To fulfill this task, they lease the machines. The computers must be replaced at least every two years. Currently, two contracts are under consideration, each initially requiring an investment of $62,000:

- Contract 1:
  - Prices for purchasing new goods increase by 6% annually
  - 60% of one-year-old computers
  - 15% of two-year-old computers can be replaced.
- Contract 2:
  - Prices increase by 2% annually
  - 30% of one-year-old computers
  - 10% of two-year-old computers can be replaced.

We know that the initial cost is $62,000, but the goal is to minimize the remaining leasing cost. Let's imagine projecting and optimizing this cost for the next 5 years. We can model the problem with a network of 5 nodes, one for each year.

We have for contract one the following network.



Cost of trading after 1 year: $1.06 \cdot \$62,000 - 0.6 \cdot \$62,000 = \$28,520$
Cost of trading after 2 years: $1.062 \cdot \$62,000 - 0.15 \cdot \$62,000 = \$60,363$
etc, etc ...

Each arc represents a choice: the arc from node 1 to node 2 represents the owner's choice to keep the goods initially and replace them at the end of the first year, while the arc from node 1 to node 3 represents the choice to wait two years before replacement, for example. The arc from node 2 to node 3 represents the choice to replace one-year-old computers at the end of the second year, taking into account price increases: $1.062 \cdot \$62,000 - 0.6 \cdot 1.06 \cdot \$62,000 = \$30,231$. The arc from node 3 to node 4 represents the choice to replace one-year-old

computers at the end of the third year, with ongoing price increases: $1.063 \cdot \$62,000 - 0.6 \cdot 1.062 \cdot \$62,000 = \$32,045.072$. The arc from node 2 to node 4 represents the choice to replace the goods at the end of the third year: $1.063 \cdot \$62,000 - 0.15 \cdot 1.061 \cdot \$62,000 = \$63,984.992$.

Once the problem is modeled with a graph like this, we can solve it by applying the shortest path algorithm.

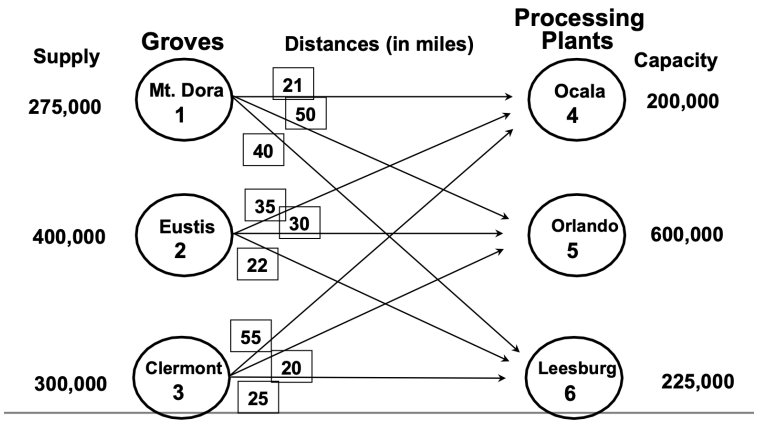
Select	From	To	Cost
1.0	1	2	$28,520
0.0	1	3	$60,363
1.0	2	3	$30,231
0.0	2	4	$63,985
1.0	3	4	$32,045
0.0	3	5	$67,824
1.0	4	5	$33,968
		Total Cost	$124,764

It is quite clear that in the first contract it is more convenient to change the equipment every year. With the second contract, however, we obtain a new optimal solution that involves replacement in the 3rd and 5th year.

Select	From	To	Cost
0.0	1	2	$44,640
1.0	1	3	$58,305
0.0	2	3	$45,533
0.0	2	4	$59,471
0.0	3	4	$46,443
1.0	3	5	$60,660
0.0	4	5	$47,372
		Total Cost	$118,965

4.4 Transportation & Assignment Problem

Some network flow problems don't have Transshipment nodes; only Supply and Demand nodes.



In this case, the Supply nodes are connected to all the Demand nodes, but this is not a general characteristic of this type of problem, and it may be necessary to model situations in which it is not the case. However, it is still possible to set it up in this way by inserting the missing arcs with a very high cost, so it will never be chosen for the final solution.

We have that: $X_{ij} = \#$ of bushels shipped from node i to node j
Specifically, the nine decision varibles are:

$X_{14} = \#$ of bushels shipped from Mt. Dora (node 1) to Ocala (node 4)
$X_{15} = \#$ of bushels shipped from Mt. Dora (node 1) to Orlando (node 5)
$X_{16} = \#$ of bushels shipped from Mt. Dora (node 1) to Leesburg (node 6)
$X_{24} = \#$ of bushels shipped from Eustis (node 2) to Ocala (node 4)
$X_{25} = \#$ of bushels shipped from Eustis (node 2) to Orlando (node 5)
$X_{26} = \#$ of bushels shipped from Eustis (node 2) to Leesburg (node 6)
$X_{34} = \#$ of bushels shipped from Clermont (node 3) to Ocala (node 4)
$X_{35} = \#$ of bushels shipped from Clermont (node 3) to Orlando (node 5)
$X_{36} = \#$ of bushels shipped from Clermont (node 3) to Leesburg (node 6)

Minimize the total number of bushel-miles.

$$\text{MIN:} \quad 21X_{14} + 50X_{15} + 40X_{16} + 35X_{24} + 30X_{25} + 22X_{26} + 55X_{34} + 20X_{35} + 25X_{36}$$

Now we can define the constraints.

- Capacity constraints

$X_{14} + X_{24} + X_{34} <= 200,000$ } Ocala
$X_{15} + X_{25} + X_{35} <= 600,000$ } Orlando
$X_{16} + X_{26} + X <= 225,000$ } Leesburg

- Supply constraints

$X_{14} + X_{15} + X_{16} = 275,000$ } Mt. Dora
$X_{24} + X_{25} + X_{26} = 400,000$ } Eustis
$X_{34} + X_{35} + X_{36} = 300,000$ } Clermont

- Nonnegativity conditions

$X_{ij} >= 0$ for all i and j

4.5 Generalized Network Flow Problem

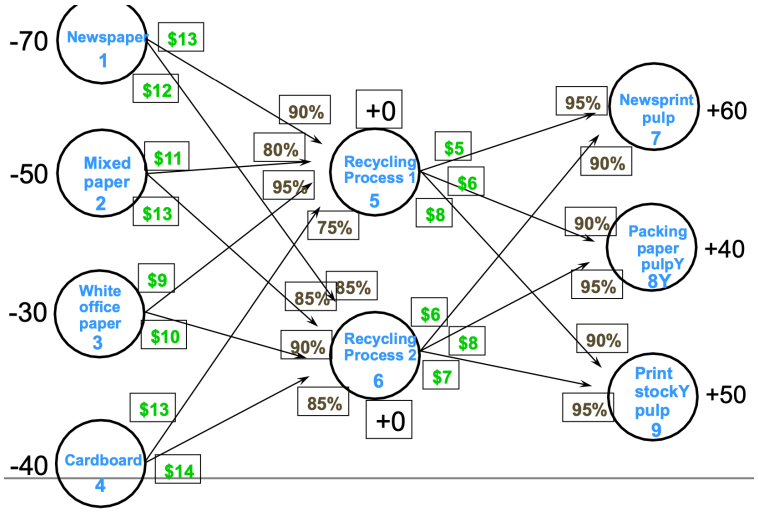
In some problems, a gain or loss occurs in flows over arcs. For examples this can happen: with oil or gas shipped through a leaky pipeline, for imperfections in raw materials entering a production process, for spoilage of food items during transit, for theft during transit, or for interest or dividends on investments. These type of problems require some modeling changes.

Suppose a company, Coal Bank Hollow Recycling, produces paperboard from four different types of recycled paper. Each ton of material involved in the first recycling process costs $5 and produces 95% of a ton of paperboard.

For each material involved in the processes, there is also a production cost and a utilization percentage. In fact, with each ton of newspaper, it is possible to produce 90% of a ton of paperboard at a cost of $13.

	Process 1		Process 2		
Material	Cost	Yield	Cost	Yield	Supply
Newspaper	$13	90%	$12	85%	70 tons
Mixed Paper	$11	80%	$13	85%	50 tons
White Office Paper	$9	95%	$10	90%	30 tons
Cardboard	$13	75%	$14	85%	40 tons

	Newsprint		Packaging Paper		Print Stock	
Pulp Source	Cost	Yield	Cost	Yield	Cost	Yield
Recycling Process 1	\$5	95%	\$6	90%	\$8	90%
Recycling Process 2	\$6	90%	\$8	95%	\$7	95%
Demand	60 tons		40 tons		50 tons	



In a generalized problem, each arc is represented by the cost and the reduction of flow (called the reduction factor). In this case too, we have a minimization problem, minimizing the total production cost.

MIN: $13X_{15} + 12X_{16} + 11X_{25} + 13X_{26} + 9X_{35} + 10X_{36} + 13X_{45}$
$+14X_{46} + 5X_{57} + 6X_{58} + 8X_{59} + 6X_{67} + 8X_{68} + 7X_{69}$

Therefore, we define the balance constraints, so since Tot. Supply > Tot. Demand
Input Flow $-$Output Flow >= Supply or Demand.

- Raw Materials

$-X_{15} - X_{16} >= -70$ } node 1
$-X_{25} - X_{26} >= -50$ } node 2
$-X_{35} - X_{36} >= -30$ } node 3
$-X_{45} - X_{46} >= -40$ } node 4

- Recycling Process

$0.9X_{15} + 0.8X_{25} + 0.95X_{35} + 0.75X_{45} - X_{57} - X_{58} - X_{59} >= 0$ } node 5
$0.85X_{16} + 0.85X_{26} + 0.9X_{36} + 0.85X_{46} - X_{67} - X_{68} - X_{69} >= 0$ } node 6

- Paper Pulp

$$\begin{aligned} 0.95X_{57} + 0.90X_{67} >= 60 \quad &\} \text{ node 7} \\ 0.90X_{57} + 0.95X_{67} >= 40 \quad &\} \text{ node 8} \\ 0.90X_{57} + 0.95X_{67} >= 50 \quad &\} \text{ node 9} \end{aligned}$$

Now let's see the problem's solution.

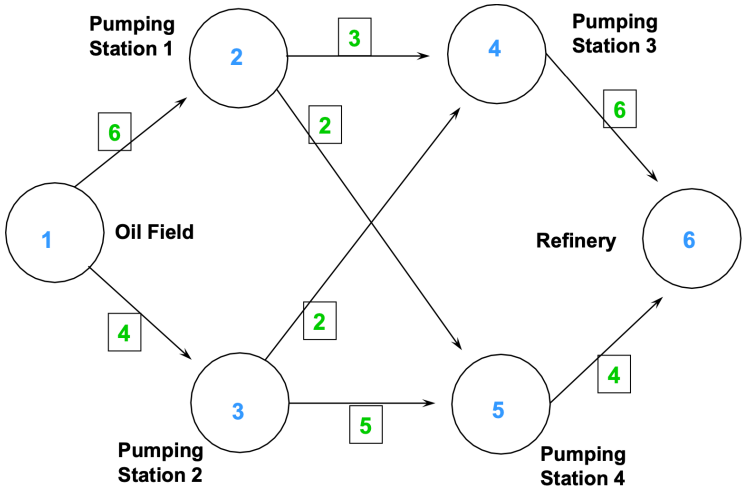
Flow From Node			Yield	Flow Into Node			Cost
43.4	1	Newspaper	0.90	39.1	5	Process 1	$13
26.6	1	Newspaper	0.85	22.6	6	Process 2	$12
50.0	2	Mixed Paper	0.80	40.0	5	Process 1	$11
0.0	2	Mixed Paper	0.85	0.0	6	Process 2	$13
30.0	3	White Office	0.95	28.5	5	Process 1	$9
0.0	3	White Office	0.90	0.0	6	Process 2	$10
0.0	4	Cardboard	0.75	0.0	5	Process 1	$13
35.4	4	Cardboard	0.85	30.1	6	Process 2	$14
63.2	5	Process 1	0.95	60.0	7	Newsprint	$5
44.4	5	Process 1	0.90	40.0	8	Packaging	$6
0.0	5	Process 1	0.90	0.0	9	Print Stock	$8
0.0	6	Process 2	0.90	0.0	7	Newsprint	$6
0.0	6	Process 2	0.95	0.0	8	Packaging	$8
52.6	6	Process 2	0.95	50.0	9	Print Stock	$7
						Total Cost	$3,149

4.6 Maximal Flow Problem

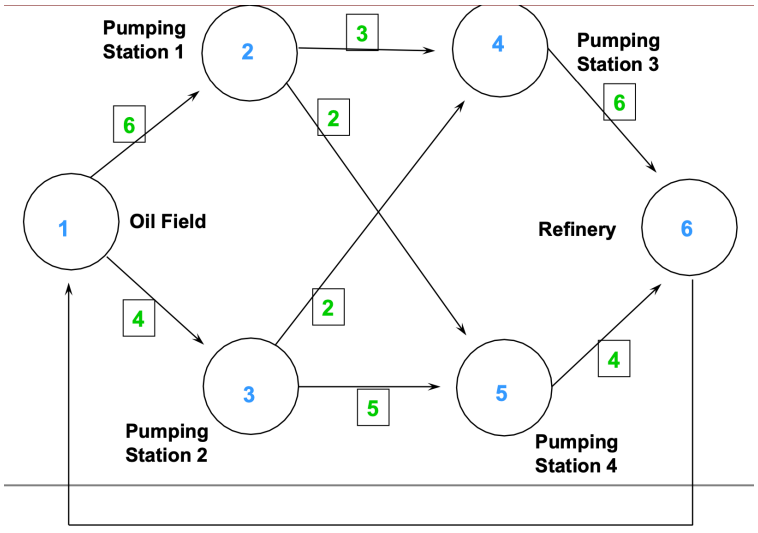
In some network problems, the objective is to determine the maximum amount of flow that can occur through a network. The arcs in these problems have two limits, an upper and a lower flow limit. For example:

- How much water can flow through a network of pipes?
- How many cars cant ravel through a network of streets?

Let's a graphical example.



The oil is shipped to the refinery from node 1 to node 6. Each arc contains a capacity, which is the amount of oil that can travel through that route (expressed, for example, in thousands of barrels per hour). This problem appears different from the network problems described so far, but it can be solved as a normal Transshipment problem. In fact, the network can be transformed into a Transshipment network by adding an arc from the final node to the initial node and assigning a demand of 0 to all nodes in the network, and finally maximizing the flow on the arcs. With this new arc from the refinery to the deposit, the final node, the refinery, can only send oil to the initial node, the deposit, if it receives it (not more because it has no supply).



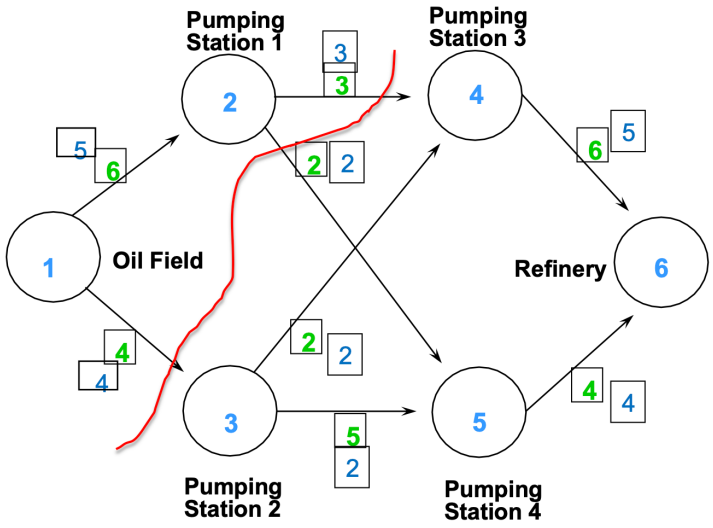
So the problem is the following:

$$
\begin{array}{ll}
\text{MAX:} & X_{61} \\
\text{Subject to:} & +X_{61} - X_{12} - X_{13} = 0 \\
& +X_{12} - X_{24} - X_{25} = 0 \\
& +X_{13} - X_{34} - X_{35} = 0 \\
& +X_{24} + X_{34} - X_{46} = 0 \\
& +X_{25} + X_{35} - X_{56} = 0 \\
& +X_{46} + X_{56} - X_{61} = 0
\end{array}
$$

with the following bounds on the decision variables:

$$
\begin{array}{lll}
0 <= X_{12} <= 6 & 0 <= X_{25} <= 2 & 0 <= X_{46} <= 6 \\
0 <= X_{13} <= 4 & 0 <= X_{34} <= 2 & 0 <= X_{56} <= 4 \\
0 <= X_{24} <= 3 & 0 <= X_{35} <= 5 & 0 <= X_{61} <= inf
\end{array}
$$

In the final solution, we have therefore made decisions on the amount of oil to be moved along the various routes in order to maximize this flow. By finding the solution to the maximum flow problem, we can also notice that there are two subsets of the solution in which all the units transported from one set of nodes to the other saturate the edges that connect them.

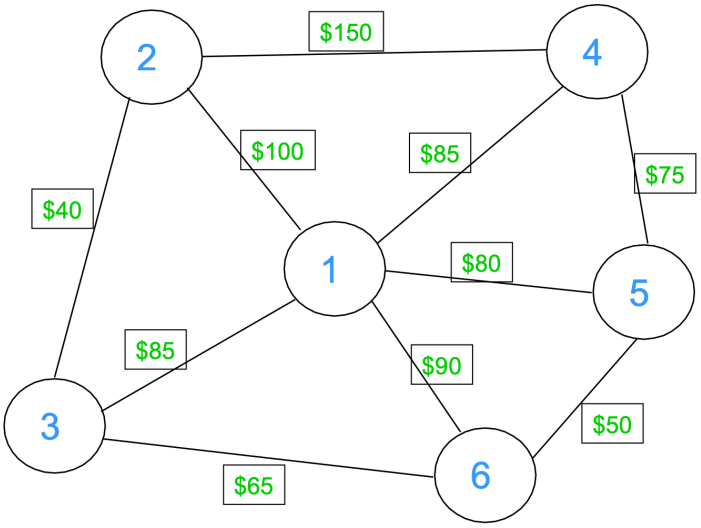


The edges 13, 25, and 24 saturate the flow capacity, as can be seen, and divide the sets {1,2} and {3,4,5,6}. This cut represents the bottleneck that prevents further barrels from traveling to node 6. The maximum flow problem is a dual problem, corresponding to the search for the minimum cut, that is the cut with the minimum cost in terms of the cumulative capacity among the various capacities assigned to the edges.

4.7 Minimal Spanning Tree Problem

Another problem with networks is the so-called Minimal Spanning Tree. For a network with n nodes, a **spanning tree** is a set of n-1 arcs that connects all the nodes and contains *no loops*. The minimal spanning tree problem involves determining the set of arcs that connects all the nodes at minimum cost. This problem can be solved with linear programming but also with a greedy algorithm.

Let's do another example. We are tasked with creating a series of connections between 6 different LANs, with a cost for each possible connection.



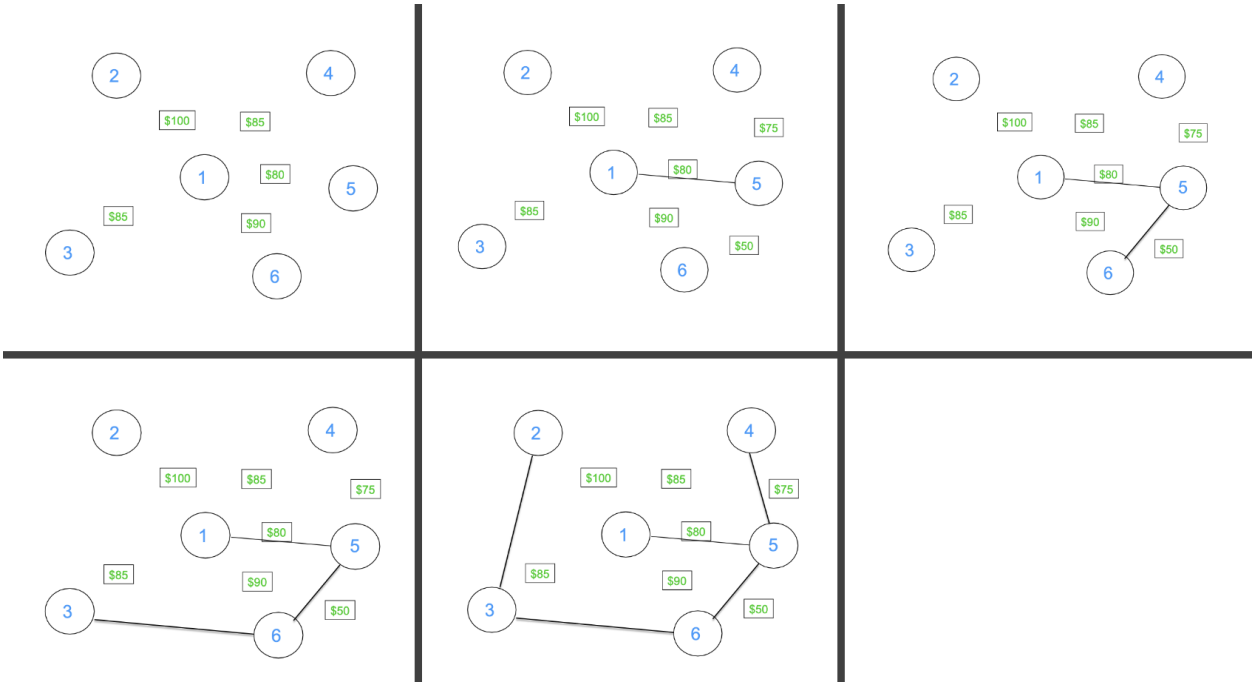
The network shown represents all possible connections between the LANs. Each node represents a LAN, a group of interconnected computers, and the various connections between the different LANs have costs. In this network, there are no directed edges, which means it does not have a single direction, and the objective is to determine which edge to activate and which to deactivate.

The minimal spanning tree **algorithm** is the following:

Step 1 Select any node. Call this the current subnetwork.

Step 2 Add to the current subnetwork the cheapest arc that connects any node within the current subnetwork to any node not in the current subnetwork. (Ties for the cheapest arc can be broken arbitrarily.) Call this the current subnetwork.

Step 3 If all the nodes are in the subnetwork, stop; this is the optimal solution. Otherwise, return to **step 2**.



A Greedy algorithm is an algorithm that finds a solution by accepting at each step a current suboptimal solution, and then iteratively improving it. The images show how the algorithm works, starting from node 1 and including the cheapest arc to form a new subnetwork that iteratively evolves until it contains all the nodes of the initial network.

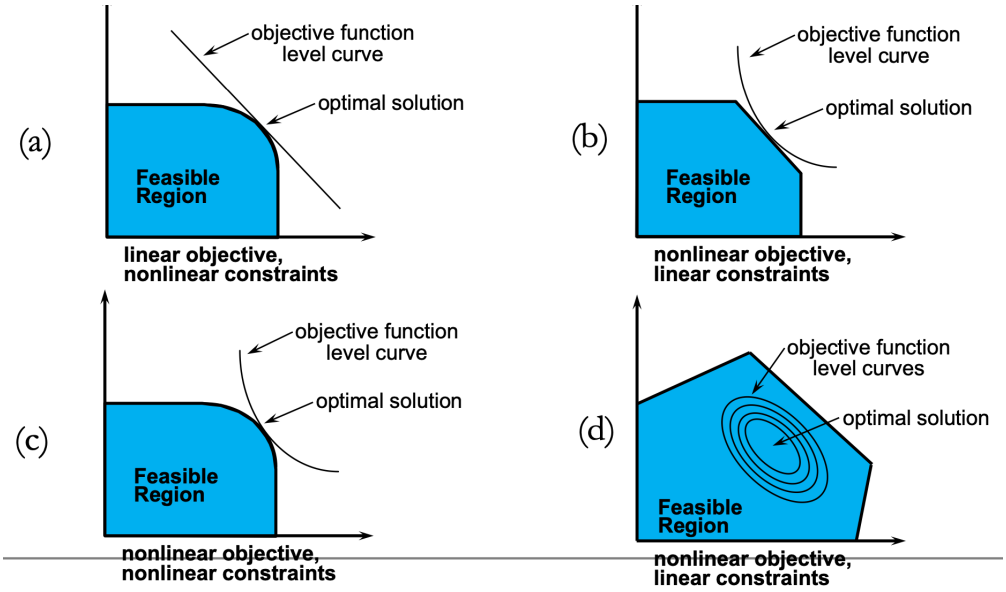
Chapter 5

Non Linear Programming

5.1 Introduction to NLP

So far in this course, we have dealt with mathematical problems that can be solved by optimizing linear functions subject to constraints expressed as linear functions. However, in many practical problems, this is not always the case, and we need to model them with nonlinear functions. This class of problems falls under the so-called nonlinear programming.

An NLP problem has a non linear objective function and/or one or more nonlinear constraints. NLP problems are formulated and implemented in virtually the same way as linear problems. The mathematics involved in solving NLPs is quite different than for LPs. Solver tends to mask this different but it is important to understand the difficulties that may be encountered when solving NLPs.

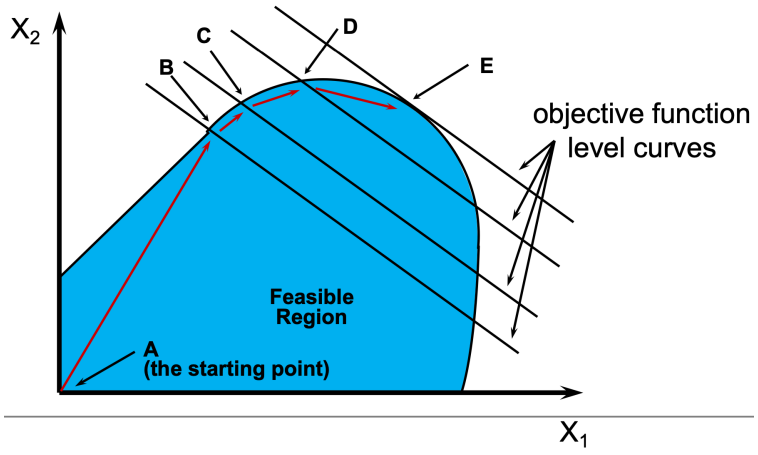


To understand nonlinearity, we present the four possible cases:

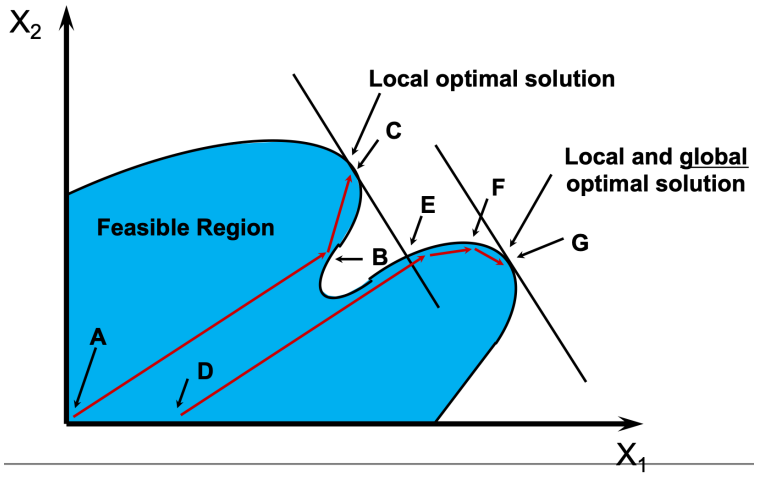
(a) In the first box we have a linear objective function (OF) and a nonlinear feasible region (FR)
(b) In the second case we have a nonlinear OF and a linear FR
(c) In the third case nonlinearity exists in both
(d) In the fourth case, there is a peculiarity not typical of linear programming problems: the optimal solution may lie within the feasible region.

With nonlinear problems, the optimal solution is not necessarily expressed at a corner of the feasible region, as can also be seen in the second case.

Since we cannot rely on the vertices of the feasible region, the **simplex method cannot be applied** to nonlinear programming problems. New strategies are therefore needed.



The strategies applied to NLP problems are **iterative**, i.e. algorithms that involve an initialization phase and iteratively improve the current solution until finding the optimal one, moving from point to point within the feasible region. These algorithms stop when they reach a point for which it is not possible to make a feasible move in any direction that produces an improvement. However, a problem can arise because the attainment of an optimal solution strongly depends on the starting point.



For this reason, we distinguish between **local optimal solutions** and **global optimal solutions**, as a local optimal solution is the optimal solution within a subset of the feasible region, while the global optimal solution is optimal over the entire feasible region. A global optimal solution is also a local optimal solution. We cannot know in advance from which point to start to reach the global optimal solution.

It is not always best to move in the direction producing the fastest rate of improvement in the objective. NLP algorithms can terminate at local optimal solutions. The starting point influences the local optimal solution obtained.

To choose the initial point, we should avoid the so-called **null starting point** (the simplex method use it), and, where possible, choose values that have approximately the same magnitude as the expected optimal value, which means they are "close" to the expected solution.

In conclusion when a sw NLP solver finds a solution it means that it found a local optimal solution, but does not guarantee that the solution is the global optimal solution.

5.2 Univariate NLP

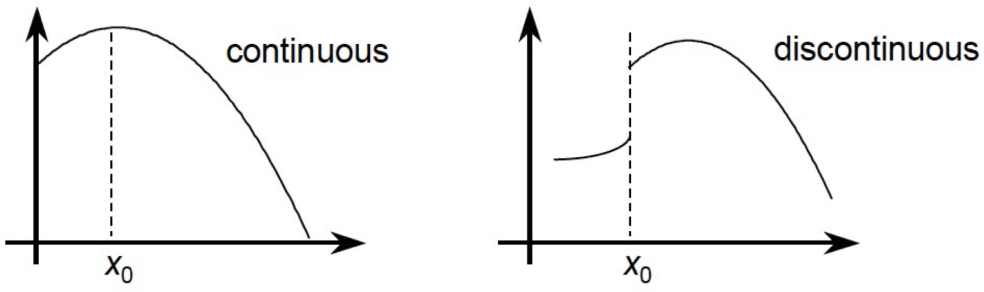
5.2.1 Mathematical Basis

We are going to introduce some methods to solve the optimization of an objective function without constraints. Before proceeding, let's review some basic concepts of mathematical analysis.

Continuity The continuity of a function at a point expresses a property according to which it exists at that given point, and the right and left limits coincide with that point.

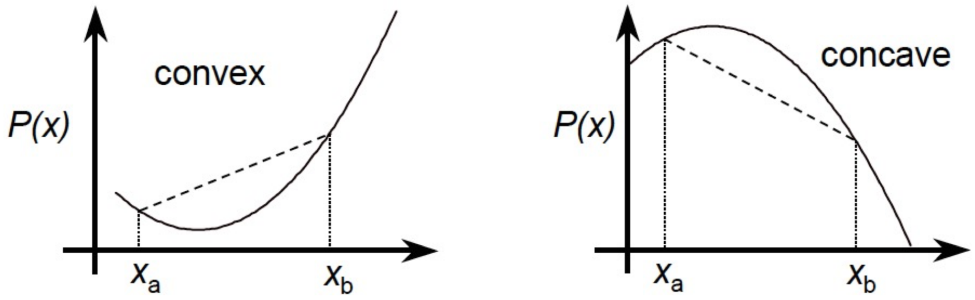
A function $P(x)$ is continuous at a point x_0 if:

$$P(x_0) \text{ exist and } \lim_{x \to +x_0} P(x_0) \equiv \lim_{x \to -x_0} P(x_0) \equiv P(x_0)$$



Functions can be continuous but their derivative not be.

Convexity Convexity and concavity are properties that express that, given two points of a function and the segment that connects them, in the first case the function's points will be below or on the segment, while in the second case, above or on the segment.



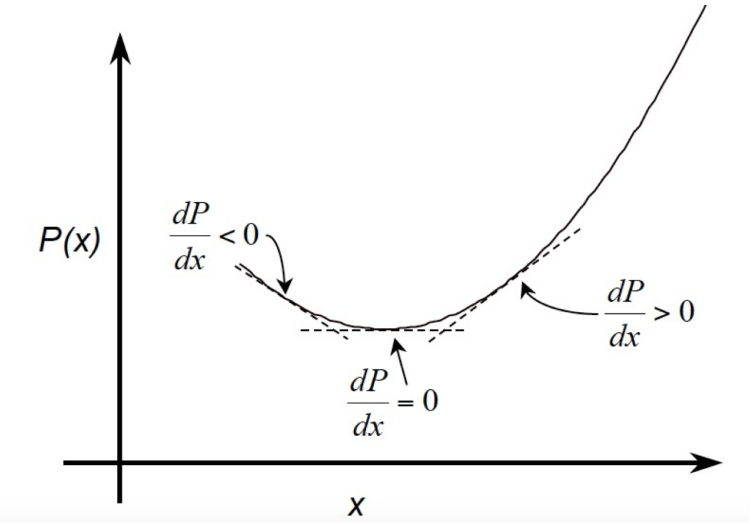
This can be expressed mathematically as:

1. for a convex function and $\forall\theta \in [0,1]$

$$P(\theta x_a + [1-\theta]x_b) \leq P(x_a) + [1-\theta]P(x_b)$$

2. for a convex function and $\forall\theta \in [0,1]$

$$P(\theta x_a + [1-\theta]x_b) \geq P(x_a) + [1-\theta]P(x_b)$$



The study of the derivative is particularly important because we can observe that it is negative when the function decreases and positive when it increases.

Convexity is an important property, we know that nonlinear solvers give us local optimal solutions, but they do not guarantee the global optimality of the solution. The possible convexity of a function also guarantees us that the local solution is also a global optimal solution.

A point x^* is a (local) minimum if:

$$P(x^*) \leq P(x) \forall x \in [x_a, x_b]$$

If $P(x)$ is convex then x^* is also a global minimum.

Necessary and sufficient contidions for an optimum Recall that for a twice continuously differentiable function P(x), the point x^* is an optimum if:

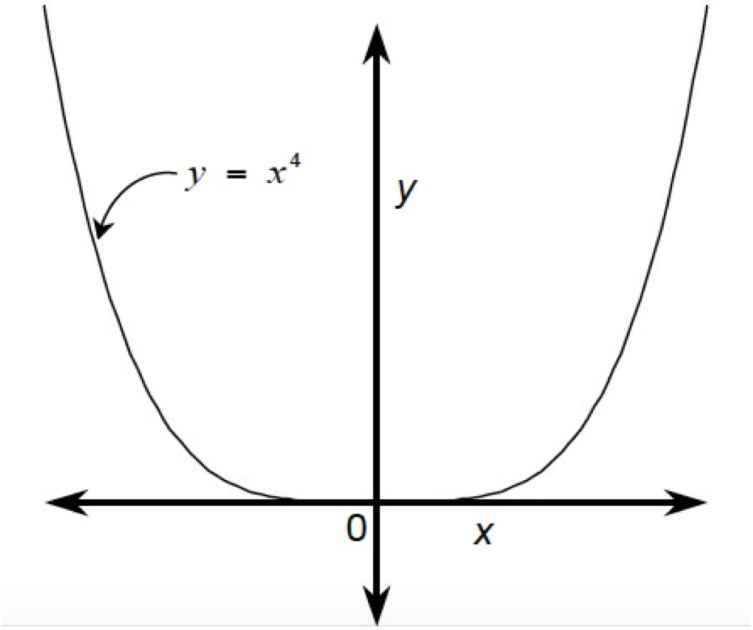
$$\frac{dP}{dx}|_{x^*} = 0 \rightarrow \text{stationarity}$$

and:

$$\frac{d^2P}{dx^2}|_{x^*} > 0 \rightarrow \text{a minimum}$$

$$\frac{d^2P}{dx^2}|_{x^*} < 0 \rightarrow \text{a maximum}$$

What happen when the secon derivative is zero? Consider the function: $y = x^4$. Which we know to have a minimum at $x = 0$



At the point $x = 0$:

$$\frac{dy}{dx}|_{x=0} = 4x^3 = 0 \rightarrow \text{stationarity}$$

$$\frac{d^2y}{dx^2}|_{x=0} = 12x^2 = 0 \rightarrow \text{a maximum?}$$

$$\frac{d^3y}{dx^3}|_{x=0} = 24x = 0$$

$$\frac{d^4y}{dx^4}|_{x=0} = 24 > 0$$

We need to add something to our optimality conditions. If at the stationary point x^* the second derivative is zero we must check the higher-order derivatives. Thus, for the first non-zero derivative is odd, that is:

$$\frac{d^nP}{dx^n}|_{x^*} \neq 0 \text{ where } n \in \{3, 5, 7, ...\}$$

Then x^* is an inflection point. Otherwise, if the first higher-order, non-zero derivative is even:

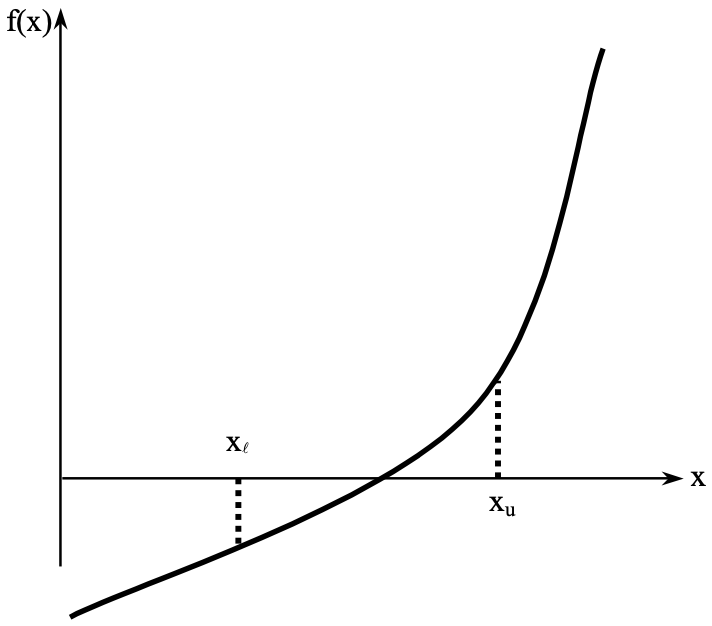
$$\frac{d^n P}{dx^n}|_{x^*} \geq 0 \text{ where } n \in \{4, 6, 8, ...\} \rightarrow \text{a minimum}$$

$$\frac{d^n P}{dx^n}|_{x^*} \leq 0 \text{ where } n \in \{4, 6, 8, ...\} \text{a maximum}$$

If the equation to find the point is analytically solvable, then the procedure is straightforward. However, when it is too complex to be solved analytically, we resort to specific methods. In this section, we will see the bisection method.

5.2.2 Bisection Method

An equation $f(x) = 0$, where $f(x)$ is a real continuous function, has at least one root between x_l and x_u if $f(x_l) \cdot f(x_u) < 0$.



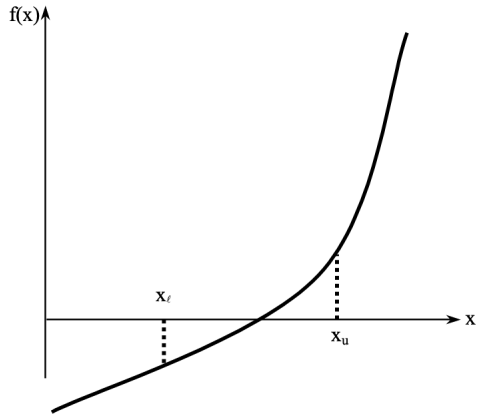
At least one root exists between the two points if the function is real, continuous, and changes sign.

In addition, the concept of **unimodality**: a function that has only one monotonically increasing part and only one monotonically decreasing part. If we have two points where the function is always positive or always negative, we can say that the function has no roots between those two points. In the case of a bimodal function, however, we cannot say the same thing, as more than one local minimum and maximum can occur for bimodal functions.

The Bisection Method is used to find the roots of a function. We can use it for optimization problems as well, as applied to the first derivative of a function it helps us find the points of minimum/maximum.

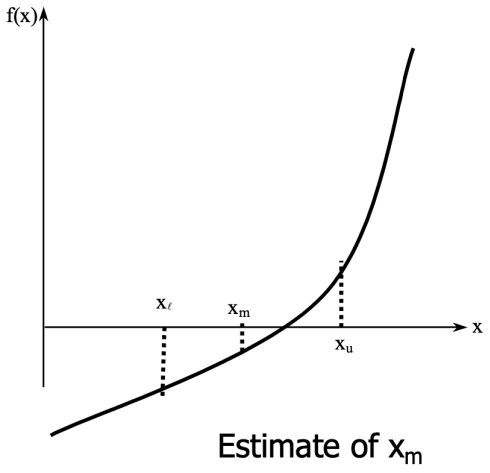
Here are the steps: we start with two points such that the product of the function applied to them is negative (thus the function changes sign at least once between them), we divide the range in two by finding a midpoint, if the midpoint evaluates to zero then it is a root, otherwise the root will be in one of the two divided parts. By recursively subdividing the ranges, we will find the points of minimum/maximum.

Step 1 Choose x_l and x_u as two guesses for the root such that $f(x_l) \cdot f(x_u) < 0$, or in other words, $f(x)$ changes sign between x_l and x_u.



Step 2 Estimate the root, x_m of the equation $f(x) = 0$ as the mid point between x_l and x_u as:

$$x_m \frac{x_l + x_u}{2}$$



Step 3 Now check the following:

a) If $f(x_l) \cdot f(x_m) < 0$, then the root lies between x_l and x_m; then $x_l = x_l; x_u = x_m$.

b) If $f(x_l) \cdot f(x_m) > 0$, then the root lies between x_m and x_u; then $x_l = x_m; x_u = x_u$.

c) If $f(x_l) \cdot f(x_m) = 0$; then the root is x_m. Stop the algorithm if this is true.

Step 4 Find the new estimate of the root

$$x_m = \frac{x_l + x_u}{2}$$

Find the absolute relative approximate error

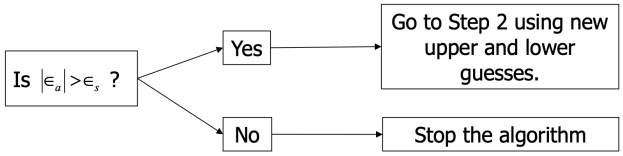
$$| \in_a | = |\frac{x_m^{new} - x_m^{old}}{x_m^{new}}| \times 100$$

where

$$x_m^{old} = \text{ previous estimate of root}$$

$$x_m^{new} = \text{ current estimate of root}$$

Step 5 Compare the absolute relative approximate error $| \in_a |$ with the pre-specified error tolerance $\in_s$.

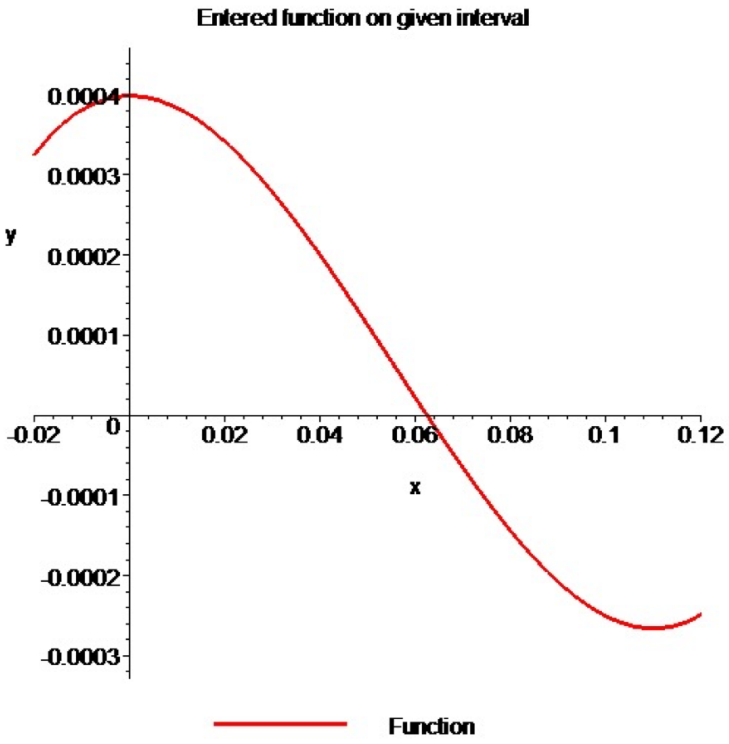


Note one should also check whether the number of iterations is more than the maximum number of iterations allowed. If so, one needs to terminate the algorithm and notify the user about it.

Let's now see an example of the algorithm at work.

$$f(x) = x^3 - 0.165x^2 + 3.993 \times 10^{-4}$$

To aid in the understanding of how this method works to find the root of an equation, the graph of $f(x)$ is shown now.



Let us assume: $x_l = 0.00$ and $x_u = 0.11$. Check if the function changes sign between x_l and x_u.

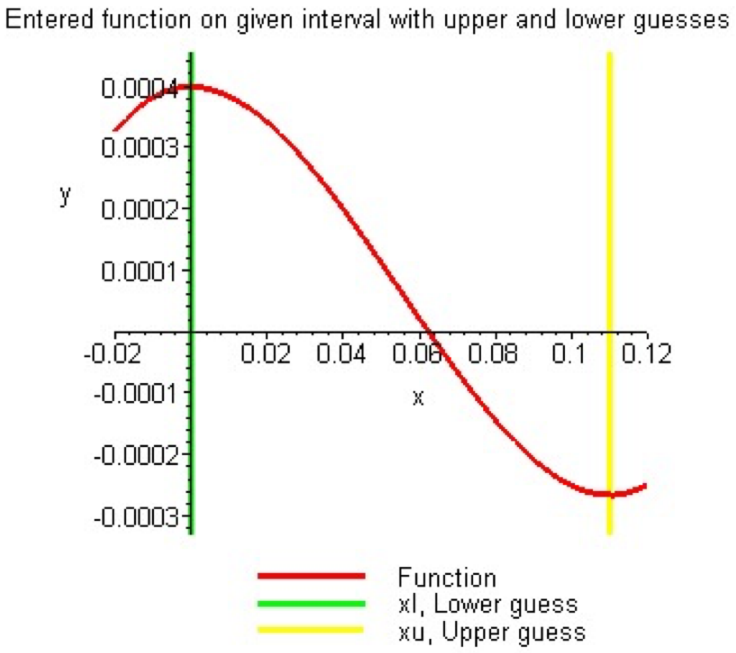
$$f(x_l) = f(0) = (0)^3 - 0.165(0)^2 + 3.993 \times 10^{-4} = 3.993 \times 10^{-4}$$

$$f(x_u) = f(0.11) = (0.11)^3 - 0.165(0.11)^2 + 3.993 \times 10^{-4} = -2.662 \times 10^{-4}$$

Hence

$$f(x_l)f(x_u) = f(0)f(0.11) = (3.993 \times 10^{-4})(-2.662 \times 10^{-4}) < 0$$

So there is at least on root between x_l and x_u, that is between 0 and 0.11.



This graph demonstrate the sign change between initial limits.

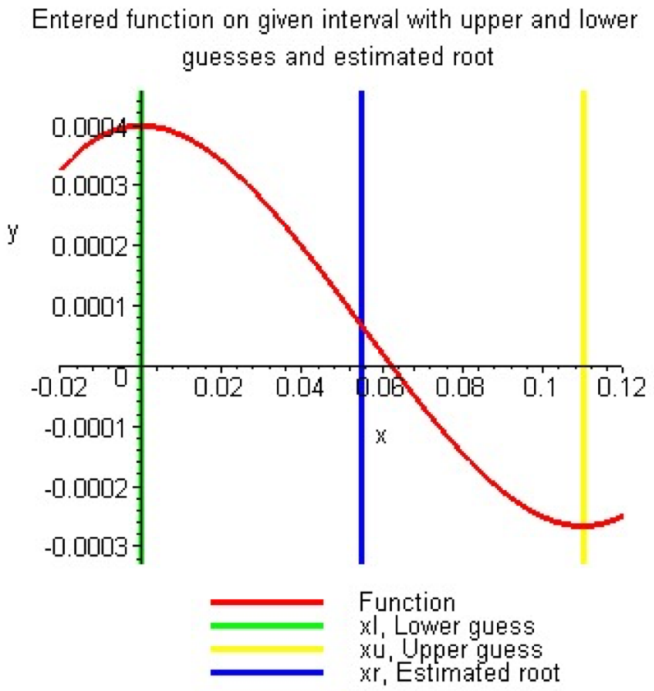
Iteration 1

The estimate of the root is $x_m = \frac{x_l + x_u}{2} = \frac{0+0.11}{2} = 0.055$

$$f(x_m) = f(0.055) = (0.055)^3 - 0.165(0.055)^2 + 3.993 \times 10^{-4} = 6.655 \times 10^{-5}$$

$$f(x_l)f(x_m) = f(0)f(0.055) = (3.993 \times 10^{-4})(5.655 \times 10^{-4}) > 0$$

Hence the root is bracketed between x_m and x_u, that is, between 0.055 and 0.11. So, the lower and upper limits of the new bracket are $x_l = 0.055$, $x_u = 0.11$. At this point, the absolute relative approximate error $| \in_a |$ cannot be calculated as we do not have a previous approximation. Here the estimate of the root for the iteration 1.



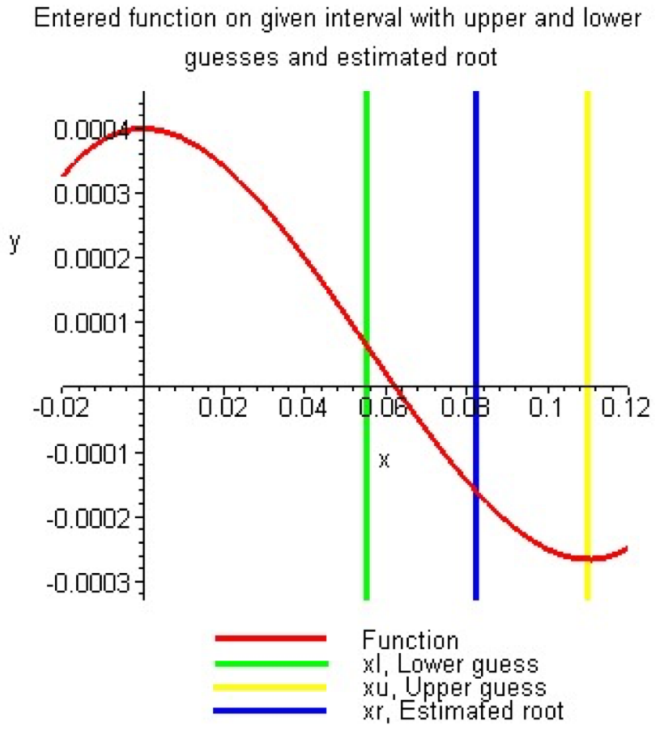
Iteration 2

The estimate of the root is $x_m = \frac{x_l + x_u}{2} = \frac{0.055+0.11}{2} = 0.0825$

$$f(x_m) = f(0.0825) = (0.0825)^3 - 0.165(0.0825)^2 + 3.993 \times 10^{-4} = 6.655 \times 10^{-4}$$

$$f(x_l)f(x_m) = f(0.055)f(0.0825) = (-1.622 \times 10^{-4})(6.655 \times 10^{-5}) < 0$$

Hence the root is bracketed between x_l and x_m, that is, between 0.055 and 0.0825. So, the lower and upper limits of the new bracket are $x_l = 0.055$, $x_u = 0.0825$. Here the estimate of the root for iteration 2.



The absolute relative approximate error $|\in_a|$ at the end of Iteration 2 is

$$|\in_a| = |\frac{x_m^{new} - x_m^{old}}{x_m^{new}}| \times 100 = |\frac{0.0825 - 0.055}{0.0825}| \times 100 = 33.333\%$$

Suppose now that we want to find the root with a tolerance error lower than 5%, then we have to continue with the next iteration.

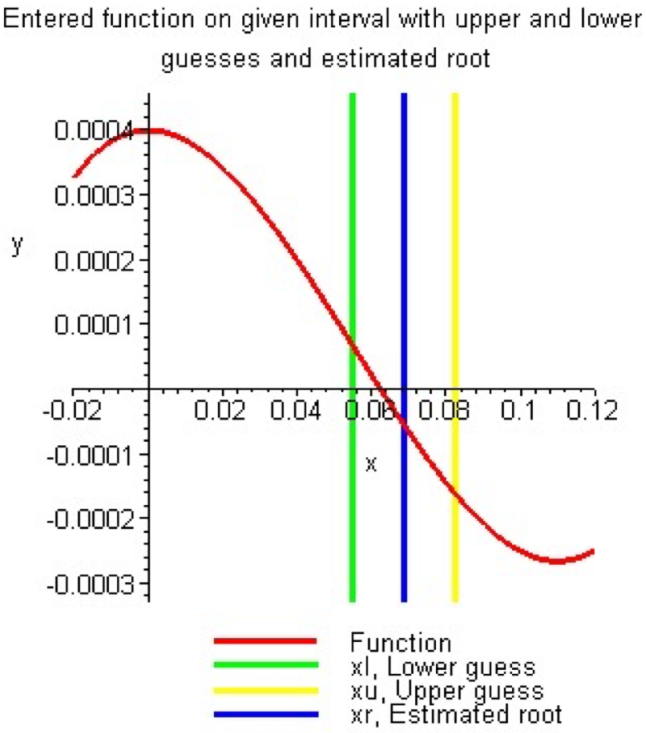
Iteration 3

The estimate of the root is $x_m = \frac{x_l + x_u}{2} = \frac{0.055 + 0.0825}{2} = 0.06875$

$$f(x_m) = f(0.06875) = (0.06875)^3 - 0.165(0.06875)^2 + 3.993 \times 10^{-4} = -5.563 \times 10^{-4}$$

$$f(x_l)f(x_m) = f(0.055)f(0.06875) = (6.655 \times 10^{-5})(-6.563 \times 10^{-5}) < 0$$

Hence the root is bracketed between x_l and x_m, that is, between 0.055 and 0.06875. So, the lower and upper limits of the new bracket are $x_l = 0.055$, $x_u = 0.06875$. Here the estimate for the root for iteration 3.



The absolute relative approximate error $|\in_a|$ at the end of Iteration 3 is

$$|\in_a| = |\frac{x_m^{new} - x_m^{old}}{x_m^{new}}| \times 100 = |\frac{0.0.6875 - 0.0825}{0.06875}| \times 100 = 20\%$$

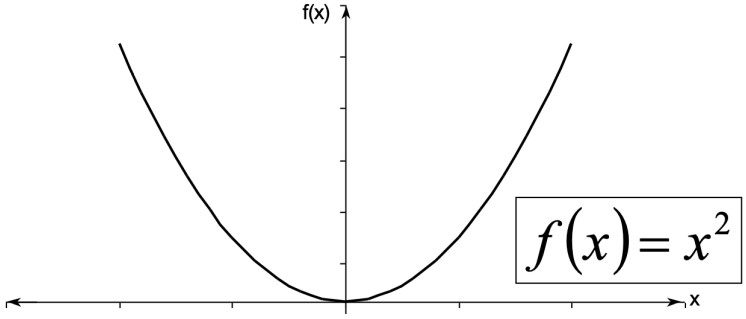
Still not enough as the absolute relative approximate error is greater than 5%.

Root of $f(x) = 0$ as function of number of iterations for bisection method.

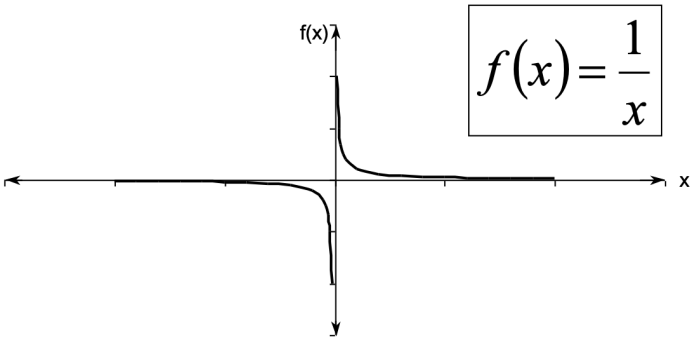
Iteration	x_ℓ	x_u	x_m	$\|\epsilon_a\|\%$	$f(x_m)$
1	0.00000	0.11	0.055	----------	6.655×10^{-5}
2	0.055	0.11	0.0825	33.33	-1.622×10^{-4}
3	0.055	0.0825	0.06875	20.00	-5.563×10^{-5}
4	0.055	0.06875	0.06188	11.11	4.484×10^{-6}
5	0.06188	0.06875	0.06531	5.263	-2.593×10^{-5}
6	0.06188	0.06531	0.06359	2.702	-1.0804×10^{-5}
7	0.06188	0.06359	0.06273	1.370	-3.176×10^{-6}
8	0.06188	0.06273	0.0623	0.6897	6.497×10^{-7}
9	0.0623	0.06273	0.06252	0.3436	-1.265×10^{-6}
10	0.0623	0.06252	0.06241	0.1721	-3.0768×10^{-7}

Advantages and Drawbacks The Bisection Method has some advantages but also some important drawback.

- **Advantages**:
  - Always convergent
  - The root bracket gets halved with each iteration (guaranteed).
- **Drawbacks**:
  - Slow convergence
  - If one of the initial guesses is close to the root, the convergence is slower
  - If a function $f(x)$ is such that it just touches the x-axis it will be unable to find the lower and upper guesses



  - Function changes sign but root does not exist



5.2.3 Newton Method

Another method for solving nonlinear problems is Newton's method. We present it for univariate cases and we will extend it for multivariate cases. Algorithm can be used for a numerical solution of the problem. These optimization algorithms generate a sequence of points converging to the optimal solution.

There are two types of optimization algorithms:

- **Dicotonomous**: find the root of the equation of the derivative equal to zero and at each iteration reduce the search interval. We have seen the bisection method, another algorithm of this type is the secant method (explained later on).
- **Approximation**: these algorithms use local approximations of the function to be optimised.

This type of algorithms need **termination criteria**, we can use one of the follwing depending on the situations;

- The solution is accurate with a given level of tolerance
- Very small improvement from one iteration to the next:

$$|x_k - x_{k-1}| \leq \varepsilon_x, \, |f(x_k) - f(x_{k-1)}| \leq \varepsilon_f$$

- The maximum number of iterations has been reached
- The solution diverges
- The solutions are in a loop

Basically the Newton's method consist in:

1. Fit a quadratic approximation to $f(x)$ using both gradient and curvature information at x.
2. Then use the Taylor approximation:

$$f(x+h) = f(x) + f'(x)h + \frac{1}{2}f''(x)h^2$$

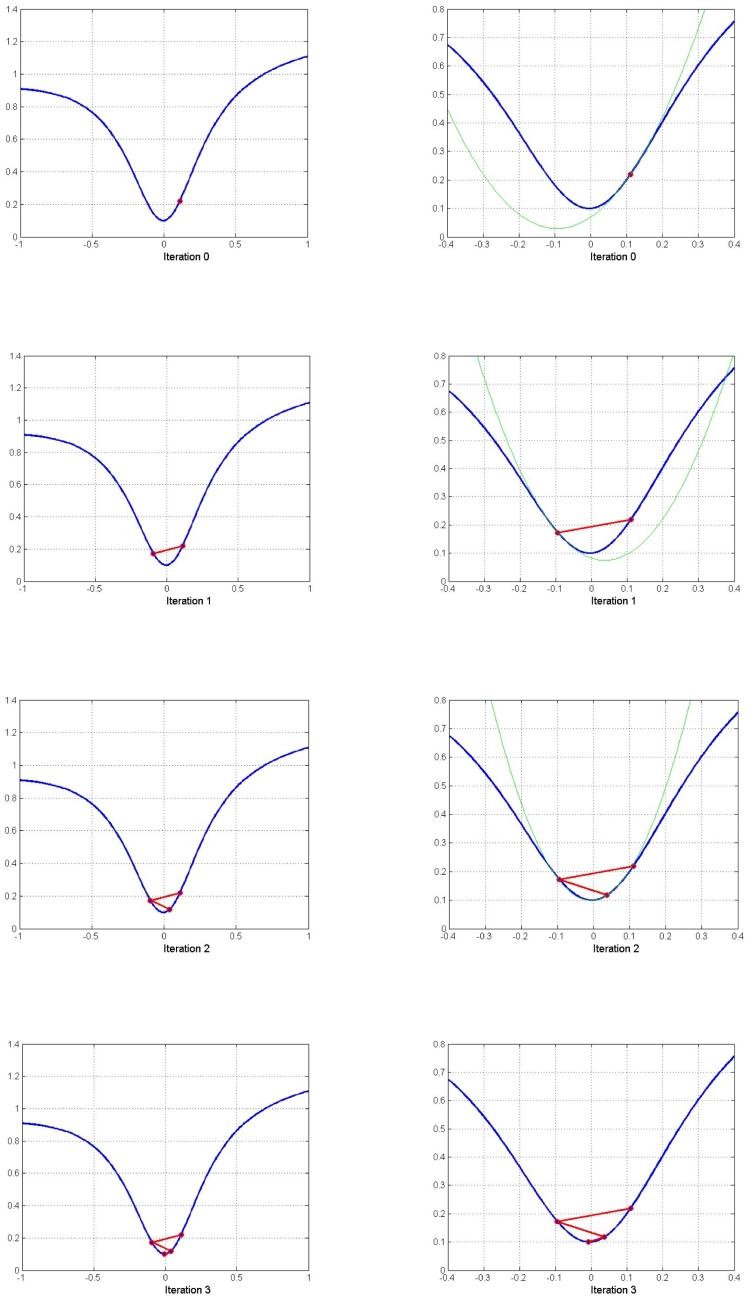
3. Root finding of f':

$$f'(x+h) = f'(x) + f''(x)h + o(h^2) \Rightarrow h = -\frac{f'(x)}{f''(x)}$$

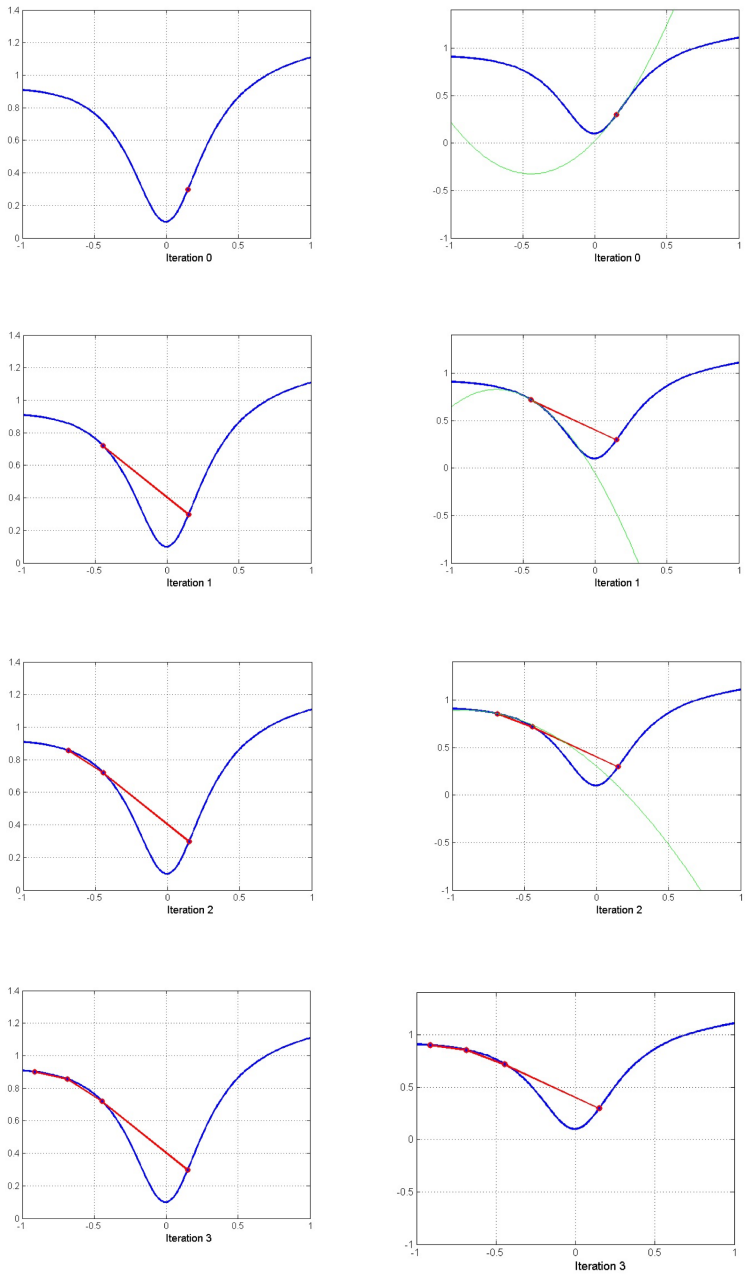
The basic idea of the approximation algorithms is to determine the type of approximation (for example the square one) and then optimize the approximation of f at a given point. Excluding $o(h^2)$, because it is insignificant for the approximation, we get that the minimum can be found by applying the ratio shown above. So we have that:

$$\text{New guess: } x_{k+1} = x_k + h_k = x_k - \frac{f'(x_k)}{f''(x_k)}$$

At each iteration the next point is defined by moving the x to the point of intersection between the x-axis and the tangent of the approximation (function in green) at the point x. In green we represent the approximation of the original function with the quadratic one. The algorithm converges quadratic (decimal accuracy doubles at each iteration).



Newton's method converges quickly but does **not guarantee convergence**. The global convergence of Newton's method is poor, and often fails if the starting point is too far from the minimum.



In practice, must be used with a globalization strategy which reduces the step length until function decrease is assured.

With Newton's method we have no control over the direction in which we are going to search because it depends on the curvature of the function approximated at the point x.

In machine learning, problems are composed of many variables in play and it is necessary to define a multivariate extension to N dimensions. Problem sizes can vary from a handful of parameters to many thousands.

5.3 Multivariate NLP

5.3.1 Gradient Method

Consider a function $f(x)$ where x is the n-vector $x = [x_1, x_2, ..., x_n]^T$. The gradient vector of this function is given by the partial derivatives with respect to each of the independent variables:

$$\nabla f(x) = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{bmatrix}$$

In the multivariate case, the gradient vector is perpendicular to the the hyperplane tangent to the contour surfaces of f.

Now an example of the computation of the vector gradient.

$$f = 15x_1 + 2(x_2)^3 - 3x_1(x_3)^2 \Longrightarrow \nabla f = \begin{bmatrix} 15 - 3(x_3)^2 & 6(x_2)^2 & -6x_1x_3 \end{bmatrix}$$

While the gradient of a function of n variables is an n-vector, the "second derivative" of an n-variable function is defined by n^2 partial derivatives (the derivatives of the n first partial derivatives with respect to the n variables):

$$\nabla^2 f = H_f = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}$$

If the partial derivatives are continuous and f is single valued then the second-order partial derivatives can be represented by a square symmetric matrix called the **Hessian matrix**.

For example:

$$\nabla f = \begin{bmatrix} 15 - 3(x_3)^2 & 6(x_2)^2 & -6x_1x_3 \end{bmatrix} \Longrightarrow \nabla^2 f = \begin{bmatrix} 0 & 0 & -6x_3 \\ 0 & 12x_2 & 0 \\ -6x_3 & 0 & -6x_1 \end{bmatrix}$$

Now that we have identified the fundamental conceptual elements, we can discuss methods of nonlinear optimization.

Similarly to each other, all nonlinear optimization algorithms carry out two fundamental steps:

1. Starting from x_k choose a *search direction* d_k
2. Minimize/maximize along that direction to find a new point:

$$x_{k+1} = x_k + \alpha_k d_k$$

where k is the current iteration number and α_k is a positive scalar called the step size.

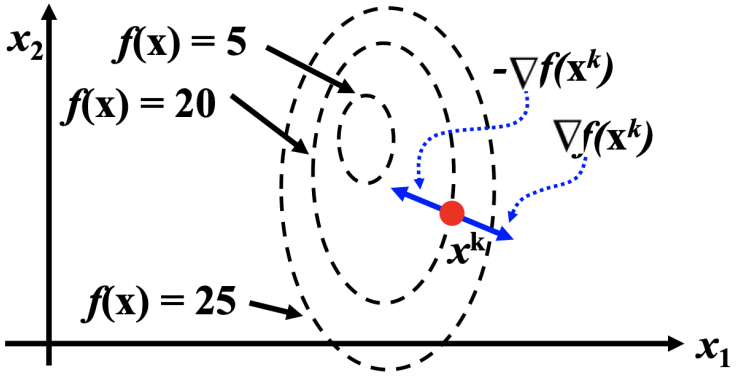
The just mentioned strategy is called **Steepest Descent**. It uses the gradient (for maximization) or the negative gradient (for minimization) as the search direction:

$$d_k = \left\{ \begin{array}{c} + \\ - \end{array} \right\}, \text{for} \left\{ \begin{array}{c} \max \\ \min \end{array} \right\} \rightarrow x_{k+1} = x_k \left\{ \begin{array}{c} + \\ - \end{array} \right\} \alpha_k \nabla f(x_k)$$

The step size, α_k, can be calculated in the following way:

- We want to minimize/maximize the function $f(x_{k+1}) = f(x_k + \alpha_k \nabla f(x_k))$ where the only variable is α_k because x_k and $\nabla f(x_k)$ are known.
- We set $\frac{\partial f(x_k + \alpha_k \nabla f(x_k))}{\partial \alpha_k}$
- and solve for α using a single-variable solution method such as the ones shown previously.

Because the gradient is the rate of change of the function at that point, using the gradient (or negative gradient) as the search direction helps reduce the number of iterations needed.



So the steps of the Steepest Descent Method are the following:

1. Choose an initial point x_0
2. Calculate the gradient $\nabla F(x_k)$ where k is the iteration number
3. Calculate the search vector: $d_k = \left\{ \begin{array}{c} + \\ - \end{array} \right\} \nabla f(x_k)$

4. Calculate the next x_{k+1}:

$$x_k + 1 = x_k \left\{ \begin{array}{c} + \\ - \end{array} \right\} \alpha_k \nabla f(x_k)$$

5. Use a single-variable optimization method to determine α_k

To determine convergence, either use some given tolerance ϵ_1 and evaluate:

$$|f(x_{k+1}) - f(x_k)| < \epsilon_1$$

for convergence. Or, use another tolerance ϵ_2 and evaluate:

$$||\nabla f(x_{k+1})|| < \epsilon_2$$

for convergence. These two criteria can be used for any of the multivariable optimization methods discussed here. We recall that the norm of a vector, $||x||$ is given by:

$$||x|| = \sqrt{x^T \cdot x} = \sqrt{(x_1)^2 + (x_2)^2 + ... + (x_n)^2}$$

Steepest Descent Example

Let's solve the following problem with the Steepest Descent Method: we want to minimize the function $f(x_1, x_2, x_3) = (x_1)^2 + x_1(1 - x_2) + (x_2)^2 - x_2x_3 + (x_3)^2 + x_3$.

Fist we calculate the gradient as it follow:

$$\nabla f(x) = \begin{bmatrix} 2x_1 + (1 - x_2), & -x_1 + 2x_2 - x_3, & -x_2 + 2x_3 + 1 \end{bmatrix}$$

We decide the initial point.

$$x_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\nabla f(x_0) = \begin{bmatrix} 2(0) + (1 - 0), & -0 + 2(0) - 0, & -0 + 2(0) + 1 \end{bmatrix} = \begin{bmatrix} 1, & 0, & 1 \end{bmatrix}$$

Let's check to see if the stop criteria is satisfied evaluating $||\nabla f(x_0)|| < \epsilon$, where ϵ is a small number close to 0, but for this exercise solved manually we can set $\epsilon = 0.1$. $||\nabla f(x_0)|| = \sqrt{1 + 0 + 1} = \sqrt{2} > \epsilon$ The stop criteria is not satisfied, then find the next point.

Then we find the direction, in this case since we are looking for the minimum we take the negative gradient.

$$d_0 = -\nabla f(x_0) = -\begin{bmatrix} 1, & 0, & 1 \end{bmatrix}$$

$$x_1 = \begin{bmatrix} 0, & 0, & 0 \end{bmatrix} + \alpha_0 \begin{bmatrix} -1, & 0, & -1 \end{bmatrix} = \begin{bmatrix} -\alpha_0, & 0, & -\alpha_0 \end{bmatrix}$$

Now, we need to determine α_0.

$$x_1 = \begin{bmatrix} -\alpha_0, & 0, & -\alpha_0 \end{bmatrix}$$

$$f(x_1, x_2, x_3) = (x_1)^2 + x_1(1 - x_2) + (x_2)^2 - x_2x_3 + (x_3)^2 + x_3$$

$$f(-\alpha_0, 0, -\alpha_0) = (-\alpha_0)^2 - \alpha_0(1 - 0) + (0)^2 - 0(-\alpha_0) + (-\alpha_0)^2 - \alpha_0 = 2(\alpha_0)^2 - 2\alpha_0$$

$$f'(-\alpha_0, 0, -\alpha_0) = 4(\alpha_0) - 2 = 0$$

$$\alpha_0 = 1/2$$

$$x_1 = \begin{bmatrix} -\frac{1}{2}, & 0, & -\frac{1}{2} \end{bmatrix}$$

Let's check to see if the stop criteria is satisfied.

$$\nabla f(x) = \begin{bmatrix} 2x_1 + (1 - x_2), & -x_1 + 2x_2 - x_3, & -x_2 + 2x_3 + 1 \end{bmatrix}$$

$$\nabla f(x_1) = \begin{bmatrix} 2(-\frac{1}{2}) + (1 - 0), & +\frac{1}{2} + 2(0) + \frac{1}{2}, & 0 + 2(-\frac{1}{2}) + 1 \end{bmatrix}$$

$$= \begin{bmatrix} -1 + 1 - 0, & +\frac{1}{2} + 0 + \frac{1}{2}, & 0 - 1 + 1 \end{bmatrix} = \begin{bmatrix} 0, & 1, & 0 \end{bmatrix}$$

$$||\nabla f(x_1)|| = \sqrt{0 + 1 + 0} = \sqrt{1} > \epsilon$$

The stop criteria is not satisfied, then find the next point.

Take the negative gradient in $x_1 = \begin{bmatrix} -\frac{1}{2}, & 0, & -\frac{1}{2} \end{bmatrix}$ to find the next search direction:

$$d_1 = -\nabla f(x_1) = -\begin{bmatrix} 0, & 1, & 0 \end{bmatrix}$$

Update the iteration formula: $x_{k+1} = x_k - \alpha_k \nabla f(x_k)$

$$x_2 = \begin{bmatrix} -\frac{1}{2}, & 0, & -\frac{1}{2} \end{bmatrix} - \alpha_1 \begin{bmatrix} 0, & 1, & 0 \end{bmatrix} = \begin{bmatrix} -\frac{1}{2}, & \alpha_1, & -\frac{1}{2} \end{bmatrix}$$

Now we use it in the initial function.

$$f(-\frac{1}{2}, -\alpha_1, -\frac{1}{2}) = (-\frac{1}{2})^2 - \frac{1}{2}(1 + \alpha_1) + (-\alpha_1)^2 - \alpha_1\frac{1}{2} + (-\frac{1}{2})^2 - \frac{1}{2}$$

$$f(-\frac{1}{2}, -\alpha_1, -\frac{1}{2}) = \frac{1}{4} - \frac{1}{2} - \frac{1}{2}\alpha_1 + \alpha - 1^2 - \frac{\alpha_1}{2} + \frac{1}{4} - \frac{1}{2}$$

$$f(-\frac{1}{2}, -\alpha_1, -\frac{1}{2}) = \alpha_1^2 - \alpha_1 - \frac{1}{2}$$

And we put the first derivative to zero.

$$f'(-\frac{1}{2}, -\alpha_1, -\frac{1}{2}) = 2\alpha_1 - 1 = 0$$

$$\alpha_1 = \frac{1}{2}$$

So we have $x_2 = \left[-\frac{1}{2}, \ -\frac{1}{2}, \ -\frac{1}{2}\right]$. Check the stop condition $\nabla f(x_2) = \left[-\frac{1}{2}, \ 0, \ -\frac{1}{2}\right]$ Evaluate

$$||\nabla f(x_2)|| = \sqrt{\frac{1}{4} + 0 + \frac{1}{4}} > \epsilon$$

The stop criteria is not satisfied, then find the next point.

Take the negative gradient in $x_2 = \left[-\frac{1}{2}, \ -\frac{1}{2}, \ -\frac{1}{2}\right]$ to find the next search direction:

$$d_2 = -\nabla f(x_2) = -\left[\frac{1}{2}, \ 0, \ \frac{1}{2}\right]$$

We update the point.

$$x_3 = \left[-\frac{1}{2}, \ -\frac{1}{2}, \ -\frac{1}{2}\right] - \alpha_2 \left[\frac{1}{2}, \ 0, \ \frac{1}{2}\right] = \left[\frac{1}{2} - \frac{\alpha_2}{2}, \ -\frac{1}{2}, \ -\frac{1}{2} + \frac{\alpha_2}{2}\right]$$

Find α_2:

$$f(x_3) = \frac{1}{2}(\alpha_2 + 1)^2 - \frac{3}{2}(\alpha_2 + 1) + \frac{1}{4}$$

Set the derivative equal to zero and solve:

$$f'(x_3) = (\alpha_2 + 1)^2 - \frac{3}{2} = 0 \Rightarrow \alpha_2 = \frac{1}{2}$$

So $x_3 = \left[-\frac{3}{4}, \ -\frac{1}{2}, \ -\frac{3}{4}\right]$. Check the stop condition:

$$\nabla f(x_3) = \left[0, \ \frac{1}{2}, \ 0\right]$$

$$||\nabla f(x_3)|| = \sqrt{\frac{1}{4}} > \epsilon$$

The stop criteria is not satisfied, then find the next point.

Take the negative gradient in x_3 to find the next search direction:

$$d_3 = -\nabla f(x_3) = -\left[0, \ \frac{1}{2}, \ 0\right]$$

We update the point.

$$x_3 = \left[-\frac{3}{4}, \ -\frac{1}{2}, \ -\frac{3}{4}\right] - \alpha_3 \left[0, \ \frac{1}{2}, \ 0\right] = \left[-\frac{3}{4}, \ -\frac{1}{2}(\alpha_3 + 1), \ -\frac{3}{4}\right]$$

Find α_3:

$$f(x_4) = \frac{1}{4}(\alpha_3 + 1)^2 - \frac{3}{2}(\alpha_3) - \frac{3}{2}$$

Set the derivative equal to zero and solve:

$$f'(x_4)\frac{1}{2}(\alpha_3 + 1) - \frac{9}{8} = 0 \Rightarrow \alpha_3 = \frac{5}{4}$$

So $x_4 = \left[-\frac{3}{4}, \quad -\frac{9}{8}, \quad -\frac{3}{4}\right]$. Check the stop condition:

$$\nabla f(x_4) = \left[\frac{5}{8}, \quad -\frac{3}{4}, \quad \frac{5}{8}\right]$$

$$||\nabla f(x_4)|| > \epsilon$$

The stop criteria is not satisfied, then find the next point.

Take the negative gradient in x_4 to find the next search direction:

$$d_4 = -\nabla f(x_4) = -\left[\frac{5}{8}, \quad -\frac{3}{4}, \quad \frac{5}{8}\right]$$

We update the point.

$$x_5 = \left[-\frac{3}{4}, \quad -\frac{9}{8}, \quad -\frac{3}{4}\right] - \alpha_4 \left[-\frac{5}{8}, \quad \frac{3}{4}, \quad -\frac{5}{8}\right]$$

Find α_4:

$$f(x_5) = \frac{73}{32}(\alpha_4)^2 - \frac{43}{32}(\alpha_4) - \frac{51}{64}$$

Set the derivative equal to zero and solve:

$$f'(x_5) = \frac{73}{16}\alpha_4 - \frac{43}{32} = 0 \Rightarrow \alpha_4 = \frac{43}{146}$$

So $x_5 = \left[-\frac{1091}{1168}, \quad -\frac{66}{73}, \quad -\frac{1091}{1168}\right]$. Let's check to see if the convergence criteria is satisfied:

$$\nabla f(x_5) = \left[-\frac{21}{584}, \quad -\frac{35}{584}, \quad -\frac{21}{584}\right]$$

$$||\nabla f(x_5)|| = 0.0786$$

So, $||\nabla f(x_5)|| = 0.0786$, which is very small and we can take it to be close enough to zero for our example. Notice that the answer of $x_5 = \left[-\frac{1091}{1168}, \quad -\frac{66}{73}, \quad -\frac{1091}{1168}\right]$ is very close to the value of $x^* = \left[-1, \quad -1, \quad -1\right]$ that we can obtain analytically.

5.3.2 Newton's Method

Newton's method approximates $f(x)$ with a quadratic function in the neighborhood of the current point using the Taylor-series expansion of f then optimizes the approximated quadratic function to obtain the new iterate point.

As in the single-variable case the optimality conditions can be derived from the Taylor-series expansion

$$f(x_k + \Delta x) \approx f(x_k) + \nabla f(x_k)\Delta x + \Delta x H(x_k)\Delta x$$

Note that x_k is the known current point, therefore, also $\nabla f(x_k)$ and $H(x_k)$ are known. The objective is now to determine Δx which optimizes $f(x_k + \Delta x)$. Then we solve:

$$\frac{\partial f(x_k + \Delta x)}{\partial \Delta x} = 0$$

From here we derive that $H(x_k)\Delta x = -\nabla f(x_k)$ so:

$$\Delta x = -H(x_k)^{-1}\nabla f(x_k)$$

Newton step, it moves to a stationary point of the second order approximation derived from the Taylor-series expansion:

$$x_{k+1} = x_k - H(x_k)^{-1}\nabla f(x_k)$$

If $H(x_k)$ is definite positive than only one iteration is required for a quadratic function to reach the optimum point, from any starting point. We remind that a matrix A is said to be positive definite if its quadratic form $x^T Ax$ is positive for any $x \neq 0$.

The **Newton's Method Steps** are the following:

Step 1 $K = 0$
Step 2 Choose a starting point x_k
Step 3 Calculate $\nabla f(x_k)$ and $H(x_k)$
Step 4 Calculate the next x_{k+1} using the equation

$$x_{k+1} = x_k - H(x_k)^{-1}\nabla f(x_k)$$

Step 5 Use either of the convergence criteria discussed earlier to determine convergence. If it hasn't converged, return to **Step 2**.

We can see that unlike the gradient descend, Newton's method uses both the gradient and the Hessian. This usually reduces the number of iterations needed, but increases the computation needed for each iteration. So, for very complex functions, a simpler method is usually faster.

Newton's Method Example

For an example, we will use the same problem as before: we want to minimize the function $f(x_1, x_2, x_3) = (x_1)^2 + x_1(1 - x_2) + (x_2)^2 - x_2x_3 + (x_3)^2 + x_3$.

So we calculate the gradient:

$$\nabla f(x) = \begin{bmatrix} 2x_1 + (1 - x_2), & -x_1 + 2x_2 - x_3, & -x_2 + 2x_3 + 1 \end{bmatrix}$$

The Hessian is:

$$H(x_k) = \begin{bmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{bmatrix}$$

And we will need the inverse of the Hessian:

$$H(x_k)^{-1} = \begin{bmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{bmatrix}^{-1} = \begin{bmatrix} \frac{3}{4} & \frac{1}{2} & \frac{1}{4} \\ \frac{1}{2} & 1 & \frac{1}{2} \\ \frac{1}{4} & \frac{1}{2} & \frac{3}{4} \end{bmatrix}$$

So, pick $x_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$ Calculate the gradient for the 1st iteration:

$$\nabla f(x_0) = \begin{bmatrix} 0-0+1, & -0+0-0 & -0+0+1 \end{bmatrix} = \begin{bmatrix} 1, & 0 & 1 \end{bmatrix}$$

So, the new x is:

$$x_1 = x_0 - H(x_0)^{-1}\nabla f(x_0) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} - \begin{bmatrix} \frac{3}{4} & \frac{1}{2} & \frac{1}{4} \\ \frac{1}{2} & 1 & \frac{1}{2} \\ \frac{1}{4} & \frac{1}{2} & \frac{3}{4} \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix}$$

Now calculate the new gradient:

$$\nabla f(x_1) = \begin{bmatrix} -2+1+1, & 1-2+1 & 1-2+1 \end{bmatrix} = \begin{bmatrix} 0, & 0 & 0 \end{bmatrix}$$

Since the gradient is zero, the method has converged.

Because it uses the 2nd derivative, Newton's Method models quadratic functions exactly and can find the optimum point in one iteration. If the function had been a higher order, the Hessian would not have been constant and it would have been much more work to calculate the Hessian and take the inverse for each iteration.

5.4 Constrained Nonlinear Programming and Lagrangian Duality

5.4.1 Constrained Nonlinear Programming

Consider a general nonlinear programming problem

$$\begin{cases} \min f(x) \\ g_i(x) \leq 0 \quad i = 1, ..., m \\ h_j(x) = 0 \quad j = 1, ..., p \end{cases}$$

where v(P) denotes the optimal value of (P). The Lagrangian function $L : R^n \times R^m \times R^p \to R$ is defined as

$$L(x, \lambda, \mu) := f(x) + \sum_{i=1}^{m} \lambda_i g_i(x) + \sum_{j=1}^{p} \mu_j h_j(x)$$

The necessary optimality conditions is given by the **Karush-Kuhn-Tucker Theorem**.

Theorem 1 (Karush-Kuhn-Tucker Theorem)**.** *Assume that f , g , h are continuously differentiable functions. If x^* is a local optimum and a suitable constraint qualification holds at x^*, then there exist $\lambda^* \in R^m$ and $\mu^* \in R^p$ s.t. (x^*, λ^*, μ^*) satisfies the Karush-Kuhn-Tucker (KKT) system called (P):*

$$\begin{cases} \nabla f(x) + \sum_{i=1}^{m} \lambda_i^* \nabla g_i(x^*) + \sum_{j=1}^{p} \mu_j^* \nabla h_j(x^*) = 0 & \\ \lambda_i^* g_i(x) = 0 & \forall i = 1, ..., m \\ \lambda^* \geq 0 & \\ g(x^*) \leq 0 & \\ h(x^*) = 0 & \end{cases}$$

We have the following constraint qualifications.

Theorem 2.

- *(Affine constraints): If g_i and h_j are affine for all $i = 1, ..., m$ and $j = 1, ..., p$, then a constraint qualification holds at any feasible point x.*
- *(Slater condition): If g_i are convex for any $i = 1, ..., m$, h_j are affine for any $j = 1, ..., p$ and there exists $\overline{x}$ s.t. $g_i(\overline{x}) < 0$ for any $i = 1, ..., m$ and $h_j(\overline{x}) = 0$ for any $j = 1, ..., p$, then a constraint qualification holds at any feasible point x.*

KKT Theorem gives necessary optimality conditions, but not sufficient ones.

For example if we have

$$\begin{cases} \min x_1 + x_2 \\ -x_1^2 - x_2^2 + 2 \leq 0 \end{cases}$$

$x^* = (1, 1)$, $\lambda^* = \frac{1}{2}$ solves KKT system, but x^* is not a local optimum.

Theorem 3 (KKT Theorem for convex problems)**.** *If we assume that problem is convex, i.e., f is convex, g_i is convex for any $i = 1, ..., m$, h_j is affine for any $j = 1, ..., p$. If (x^*, λ^*, μ^*) solves the KKT system, then x^* is a global optimum.*

5.4.2 Lagrangian relaxation and dual function

Lagrangian relaxation

By definition given $\lambda > 0$ and $\mu \in R^p$, the problem

$$\begin{cases} \min L(x, \lambda, \mu) \\ x \in R^n \end{cases}$$

is called **Lagrangian relaxation** of the (P) and $\varphi(\lambda, \mu) = \inf_{x \in R^n} L(x, \lambda, \mu)$ is the **Lagrangian dual function**.

Dual function φ

- is concave because inf of affine functions w.r.t. (λ, μ)
- may be equal to $-\infty$ at some point
- may be not differentiable at some point

Lagrangian relaxation of (P) provides a lower bound to v(P).

Theorem 4. *For any $\lambda > 0$ and $\mu \in R^p$, we have $\varphi(\lambda, \mu) \leq v(P)$*

Lagrangian Dual Problem

The problem

$$\begin{cases} \max \varphi(\lambda, \mu) \\ \lambda \geq 0 \end{cases}$$

is called **Lagrangian dual problem** of (P), and (P) is called **primal problem**. The dual problem (D) consists in finding the best lower bound of v(P). (D) is always a convex problem, even if (P) is a non-convex problem.

Theorem 5 (Weak duality)**.** *For any optimization problem (P), we have $v(D) \leq v(P)$. Strong duality, i.e., $v(D) = v(P)$, does not hold in general.*

Theorem 6 (Strong duality)**.** *Suppose f,g, h are continuously differentiable, the primal problem*

$$\begin{cases} \min f(x) \\ g(x) \leq 0 \\ h(x) = 0 \end{cases}$$

is convex, there exists an optimal solution x^* of (P) and a constraint qualification holds at x^*. Then

- *the strong duality holds: $v(D) = v(P)$*

- (λ^*, μ^*) *is optimal for (D) if and only if* (λ^*, μ^*) *is a KKT multipliers vector associated to* x^*.

Strong duality may hold also for some non-convex problems.

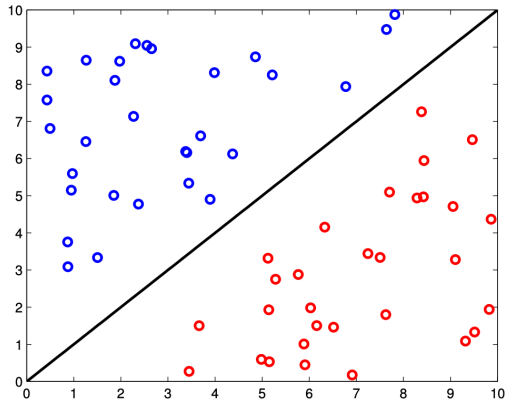
5.5 Support Vector Machines for Supervised Classification Problems

5.5.1 Linear SVM

Given a set of objects partitioned in several classes with known labels, we want to predict the class of any new future object with unknown label.

Consider binary classification. We have two finite sets $A, B \subset R^n$ with known labels (1 for points in A, -1 for points in B). R^n is the input space, $A \cup B$ is the training set. Assume that A and B are linearly separable, i.e., there is an hyperplane $H = x \in R^n : w^T x + b = 0$ such that

$$w^T x_i + b > 0 \quad \forall x_i \in A$$
$$w^T x_j + b < 0 \quad \forall x_j \in B$$



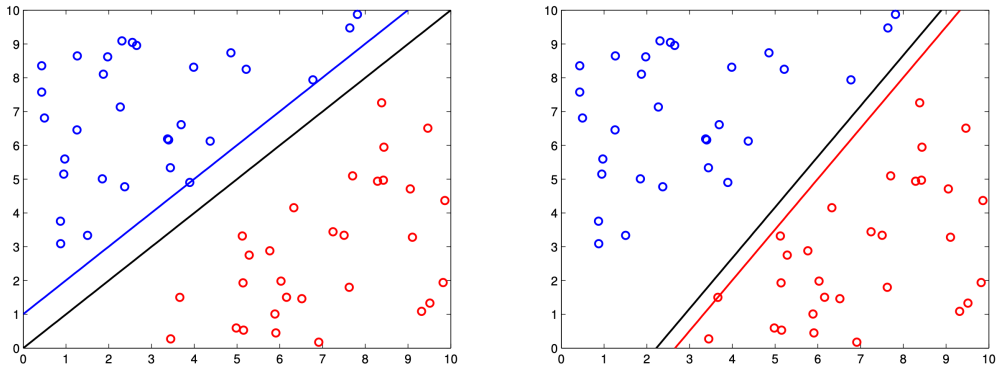
We have a new test data x: use the decision function

$$f(x) = \text{sign}(w^T x + b) = \begin{cases} 1 & \text{if } w^T x + b > 0 \\ -1 & \text{if } w^T x + b < 0 \end{cases}$$

There are many possible separating hyperplanes.

If H is a separating hyperplane, then the margin of separation of H is defined as the minimum distance between H and $A \cup B$, i.e.

$$\rho(H) = \min_{x \in A} \frac{|w^T x + b|}{||w||}$$



We look for the separating hyperplane with the maximum margin of separation.

Theorem 7. *Finding the separating hyperplane with the maximum margin of separation is equivalent to solve the following convex quadratic programming problem:*

$$\begin{cases} \min_{w,b} \frac{1}{2}||w||^2 \\ w^T x_i + b \geq 1 \quad \forall x_i \in A \\ w^T x_i + b \leq 1 \quad \forall x_j \in B \end{cases}$$

Let $l = |A \cup B|$. For any point $x_i \in A \cup B$, define a label

$$y_i = \begin{cases} 1 & \text{if } x_i \in A \\ -1 & \text{if } x_j \in B \end{cases}$$

Then the problem

$$\begin{cases} \min_{w,b} \frac{1}{2}||w||^2 \\ w^T x_i + b \geq 1 \quad \forall x_i \in A \\ w^T x_i + b \leq 1 \quad \forall x_j \in B \end{cases}$$

is equivalent to

$$\begin{cases} \min_{w,b} \frac{1}{2}||w||^2 \\ 1 - y_i(w^T x_i + b) \leq 0 \quad \forall x_i = 1, ..., l \end{cases}$$

It is useful to consider the Lagrangian dual of this problem.

The Lagrangian function is

$$L(w, b, \lambda) = \frac{1}{2}||w||^2 + \sum_{i=1}^{l} \lambda_i [1 - y_i(w^T x_i + b)] = \frac{1}{2}||w||^2 - \sum_{i=1}^{l} \lambda_i y_i w^T x_i - b \sum_{i=1}^{l} \lambda_i y_i + \sum_{i=1}^{l} \lambda_i$$

- If $\sum_{i=1}^{l} \lambda_i y_i \neq 0$ then $\min_{n,b} L(w, b, \lambda) = -\infty$
- If $\sum_{i=1}^{l} \lambda_i y_i = 0$ then Lvdoes not depend on b, L is strongly convex wrt w and $\arg\min_w L(w, b, \lambda)$ is given by the (unique) stationary point

$$\nabla_w L(w, b, \lambda) = w - \sum_{i=1}^{l} \lambda_i y_i x_i = 0$$

Therefore, the dual function is

$$\varphi(\lambda) \begin{cases} -\infty & \text{if } \sum_{i=1}^{l} \lambda_i y_i \neq 0 \\ -\frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} y_i y_j (x_i)^T x_j \lambda_i \lambda_j + \sum_{i=1}^{l} \lambda_i & \text{if } \sum_{i=1}^{l} \lambda_i y_i = 0 \end{cases}$$

So the dual problem of the problem of

$$\begin{cases} \min_{w,b} \frac{1}{2} ||w||^2 \\ 1 - y_i(w^T x_i + b) \leq 0 \quad \forall x_i = 1, ..., l \end{cases}$$

is

$$\begin{cases} \max -\frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} y_i y_j (x_i)^T x_j \lambda_i \lambda_j + \sum_{i=1}^{l} \lambda_i \\ \sum_{i=1}^{l} \lambda_i y_i = 0 \\ \lambda \geq 0 \end{cases}$$

or

$$\begin{cases} \max -\frac{1}{2} \lambda^T X^T X \lambda + e^T \lambda \\ \sum_{i=1}^{l} \lambda_i y_i = 0 \\ \lambda \geq 0 \end{cases}$$

where the n matrix $X = (y^1 x^1, y^2 x^2, ..., y^l x^l)$ and the vector $e^T = (1, ..., 1)$.

Dual problem is a convex quadratic programming problem. Dual constraints are simpler than primal constraints. Dual problem has optimal solutions: each KKT multiplier λ^* associated to the primal optimum (w^*, b^*) is a dual optimum

- If $\lambda^* > 0$, then x_i is said support vector
- If λ^* is a dual optimum, then

$$w^* = \sum_{i=1}^{l} \lambda_i^* y_i x_i$$

b^* is obtained using the complementarity conditions:

$$\lambda^* [1 - y^* ((w^*)^T x_i + b^*)] = 0$$

in fact, if i is such that $\lambda_i^* > 0$, then $b^* = \frac{1}{y_i} - (w^*)^T x_i$. Finally, the decision function is

$$f(x) = \text{sign}((w^*)^T x + b^*)$$

Linear SVM with soft margin

If sets A and B are not linearly separable the linear system

$$1 - y_i((w^T x_i + b) \leq 0 \text{ for } i = 1, ..., l$$

has no solutions.

We introduce slack variables $\xi_i \geq 0$ and consider the (relaxed) system:

$$\begin{array}{ll} 1 - y_i((w^T x_i + b) \leq \xi_i & i = 1, ..., l \\ x_i \geq 0 & i = 1, ..., l \end{array}$$

If x_i is misclassified, then $\xi_i > 1$, thus $\sum_{i=1}^{l} \xi_i$ is an upper bound of the number of misclassified points. We add to the objective function the term $C \sum_{i=1}^{l} \xi_i$, where $C > 0$ is a parameter:

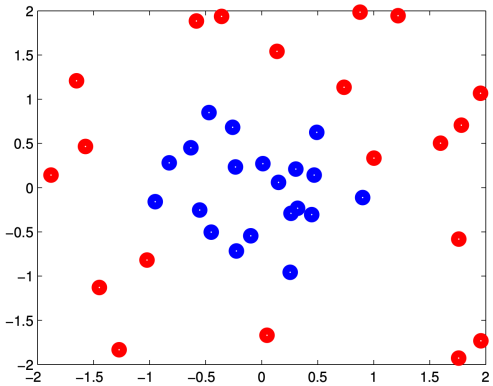
$$\begin{cases} \min_{w,b,\xi} -\frac{1}{2}||w||^2 + C \sum_{i=1}^{l} \xi_i & \\ 1 - y_i((w^T x_i + b) \leq \xi_i & i = 1, ..., l \\ x_i \geq 0 & i = 1, ..., l \end{cases}$$

And its dual problem is:

$$\begin{cases} \max_{\lambda} -\frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} y_i y_j (x_i)^T x_j \lambda_i \lambda_j + \sum_{i=1}^{l} \lambda_i & \\ \sum_{i=1}^{l} \lambda_i y_i = 0 & \\ 0 \leq \lambda_i \leq C & i = 1, ..., l \end{cases}$$

5.5.2 Nonlinear SVM

Consider now two sets A and B which are not linearly separable.



To answer if they're they linearly separable in other spaces we use a map $\phi : R^n \to H$, where H is an higher dimensional (maybe infinite) space. H is called the **features space**. We try to linearly separate the images $\phi(x_i)$,$i = 1, ..., l$ in the feature space.

The primal problem is the following:

$$\begin{cases} \min_{w,b,\xi} -\frac{1}{2}||w||^2 + C\sum_{i=1}^{l} \xi_i & \\ 1 - y_i((w^T\phi(x_i) + b) \leq \xi_i & i = 1, ..., l \\ x_i \geq 0 & i = 1, ..., l \end{cases}$$

w is a vector in a high dimensional space (maybe infinite variables).

And its dual problem is:

$$\begin{cases} \max_{\lambda} -\frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} y_i y_j \phi(x_i)^T\phi(x_j)\lambda_i\lambda_j + \sum_{i=1}^{l}\lambda_i & \\ \sum_{i=1}^{l}\lambda_i y_i = 0 & \\ 0 \leq \lambda_i \leq C & i = 1, ..., l \end{cases}$$

The number of variables is equal to the number of training data.

The steps for this type of problem are the following:

Step 1 Solve dual problem λ^*
Step 2 Compute $w^* = \sum_{i=1}^{l} \lambda_i y_i \phi(x_i)$
Step 3 Use any λ_i^* s.t. $0 < \lambda_i^* < C$ for finding b^*:

$$y_i[\sum_{i=1}^{l} \lambda_j^* y_j \phi(x_j)^T\phi(x_i) + b^*] - 1 = 0$$

Step 4 Find the decision function

$$f(x) = \text{sign}((w^*)^T\phi(x) + b^*) = \text{sign}(\sum_{i=1}^{l} \lambda_i^* y_i \phi(x_i)^T\phi(x) + b^*)$$

that depends on

- λ^*: know $\phi(x_i)^T\phi(x_j)$
- $\phi(x_i)^T\phi(x)$
- b^*: know $\phi(x_i)^T\phi(x_j)$

No need to explicitly know $\phi(x)$, but only $\phi(x)^T\phi(y)$

We use kernel functions. A function $k : R^n \times R^n \to R$ is called kernel if there exists a map $\phi : R^n \to H$ such that $k(x, y) = <\phi(x), \phi(y)>$ where $<\cdot, \cdot>$ is a scalar product in H.

Some examples are:

- $k(x, y) = x^T y$

- Polynomial: $k(x,y) = (x^T y + 1)^p$, whit $p \geq 1$
- Gaussian: $k(x,y) = e^{-\gamma||x-y||^2}$
- $k(x,y) = \tanh(\beta x^T + \gamma)$, with suitable β and γ

Theorem 8. *If $k : R^n \times R^n \to R$ is a kernel and $x^1, ..., x^l \in R^n$, then the matrix K defined as follows*

$$K_{ij} = k(x_i, x_j)$$

is positive semidefinite.

The dual problem depends on the kernel k:

$$\begin{cases} \max_\lambda -\frac{1}{2}\sum_{i=1}^l \sum_{j=1}^l y_i y_j k(x_i, x_j)\lambda_i \lambda_j + \sum_{i=1}^l \lambda_i \\ \sum_{i=1}^l \lambda_i y_i = 0 \\ 0 \leq \lambda_i \leq C \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad i = 1, ..., l \end{cases}$$

In practice:

Step 1 Chose a kernel k
Step 2 Find an optimal solution λ^* of the dual
Step 3 Choose i s.t. $0 < \lambda_i^* < C$ and find b^*:

$$b^* = \frac{1}{y_i} \sum_{j=1}^l \lambda_j^* y_j k(x_i, x_j)$$

Step 4 Decision function

$$f(x) = \text{sign}(\sum_{j=1}^l \lambda_i^* y_i k(x_i, x) + b^*)$$

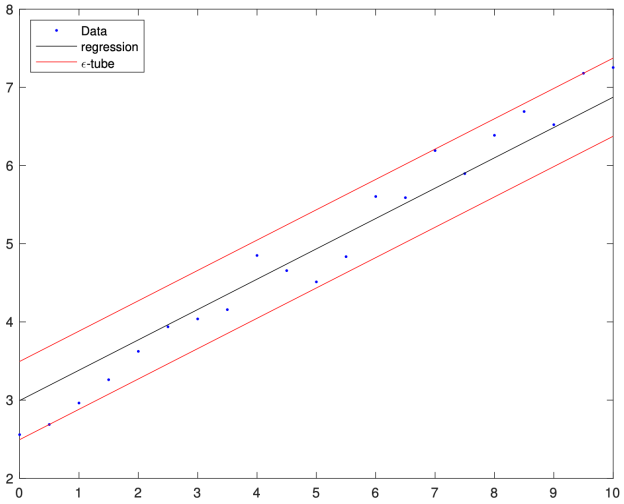
Separating surface $f(x) = 0$ is linear in the features space and nonlinear in the input space.

5.6 ε-SV Regression

5.6.1 Linear ε-SV regression

We have a set of training data $\{(x_1, y_1), ..., (x_l, y_l)\}$, where $x_i \in R^n$ and $y_i \in R$. In ε-SV regression we aim to find a function f that

- has at most ε deviation from the targets y_i for all the training data
- is as flat as possible



Start with linear regression. Consider an affine function $f(x) = w^T x + b$ and set a tolerance parameter ε. Flatness means that one seek a small w, that is we aim to solve the convex quadratic optimization problem

$$\begin{cases} \min_{w,b} -\frac{1}{2}||w||^2 \\ y_i \leq w^T x_i + b + \varepsilon \quad i = 1, ..., l \\ y_i \geq w^T x_i + b - \varepsilon \quad i = 1, ..., l \end{cases}$$

Lienar ε-SV regression with slack variables

If ε is too small, the model could not be feasible.

The linear ε-SV regression model can be extended by introducing slack variables ξ^+ and ξ^- to relax the constraints of problem:

$$\begin{cases} \min_{w,b,\xi^+,\xi^-} -\frac{1}{2}||w||^2 + C\sum_{i=1}^{l}(\xi_i^+ + \xi_i^-) & \\ y_i \leq w^T x_i + b + \varepsilon + \xi_i^+ & i = 1, ..., l \\ y_i \geq w^T x_i + b - \varepsilon - \xi_i^- & i = 1, ..., l \\ \xi_i^+ \geq 0 & \\ \xi_i^- \geq 0 & \end{cases}$$

where parameter C gives the trade-off between the flatness of f and tolerance to deviations larger than ε.

The relative dual problem consider the Lagrangian function that is:

$$L(w,b,\xi^+,\xi^-,\lambda^+,\lambda^-,\eta^+,\eta^-) = \frac{1}{2}||w||^2 - w^T[\sum_{i=1}^{l}(\lambda_i^+ - \lambda_i^-)x_i] - b\sum_{i=1}^{l}(\lambda_i^+ - \lambda_i^-) + \sum_{i=1}^{l}\xi_i^+(C - \lambda_i^+ - \eta_i^+) + \sum_{i=1}^{l}\xi_i^-(C - \lambda_i^- - \eta_i^-)$$

If $\sum_{i=1}^{l}(\lambda_i^+ - \lambda_i^-) \neq 0$ or $C - \lambda_i^+ - \eta_i^+ \neq 0$ for some i or $C - \lambda_i^- - \eta_i^- \neq 0$ for some i, then $\min_{w,b,\xi^+,\xi^-} L = -\infty$. Otherwise

$$\nabla_w L = w - \sum_{i=1}^{l}(\lambda_i^+ - \lambda_i^-)x_i = 0$$

.

The dual problem is the following:

$$\begin{cases} \max_{\lambda^+,\lambda^-} -\frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}(\lambda_i^+ - \lambda_i^-)(\lambda_j^+ - \lambda_j^-)(x_i)^T x_j - \varepsilon\sum_{i=1}^{l}(\lambda_i^+ - \lambda_i^-) + \sum_{i=1}^{l} y_i(\lambda_i^+ - \lambda_i^-) \\ \sum_{i=1}^{l}(\lambda_i^+ - \lambda_i^-) = 0 \\ \lambda_i^+ \in [0, C] \\ \lambda_i^- \in [0, C] \end{cases}$$

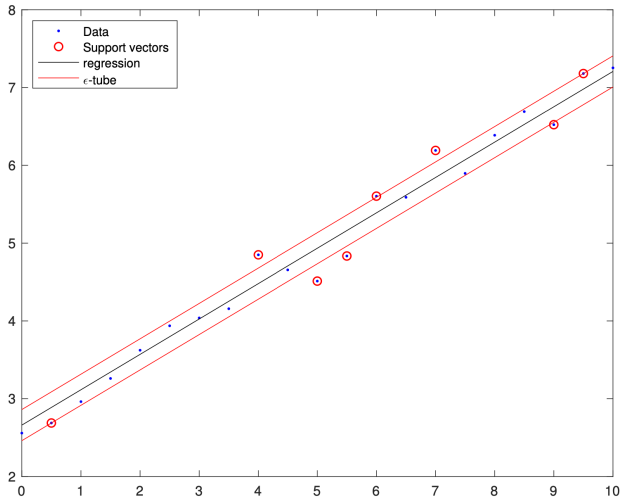
Dual problem is a convex quadratic programming problem. Dual constraints are simpler than primal constraints. If $\lambda_i^+ > 0$ or $\lambda_i^- > 0$, then x_i is said **support vector**. If $(\lambda_i^+, \lambda_i^-)$ is a dual optimum, then

$$w = \sum_{i=1}^{l}(\lambda_i^+ - \lambda_i^-)x_i$$

b is obtained using the complementary conditions:

$$\lambda_i^+[\varepsilon + \xi_i^+ - y_i + w^T x_i + b] = 0$$
$$\lambda_i^-[\varepsilon + \xi_i^- - y_i + w^T x_i - b] = 0$$
$$\xi_i^+(C - \lambda_i^+) = 0$$
$$\xi_i^-(C - \lambda_i^-) = 0$$

Hence, if there is some i s.t. $0 < \lambda_i^+ < C$, then $b = y_i - w^T x_i - \varepsilon$; if there is some i s.t. $0 < \lambda_i^- < C$, then $b = y_i - w^T x_i + \varepsilon$.



5.6.2 Nonlinear ε-SV regression

In order to generate a nonlinear regression function f we use the kernel. Define a map $\phi : R^n \to H$, where H (features space) is an higher dimensional (maybe infinite) space and find the linear regression for the points $\{(\phi(x_i), y_i)\}$ in the feature space H.

The primal problem is the following one:

$$\begin{cases} \min \frac{1}{2}||w||^2 + C\sum_{i=1}^{l}(\xi_i^+ + \xi_i^-) & \\ y_i \leq w^T\phi(x_i) + b + \varepsilon + \xi_i^+ & \forall i = 1, ..., l \\ y_i \geq w^T\phi(x_i) + b - \varepsilon - \xi_i^- & \forall i = 1, ..., l \end{cases}$$

w is a vector in a high dimensional space (maybe infinite variables).

The relative dual problem is:

$$\begin{cases} \max_{\lambda^+,\lambda^-} -\frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}(\lambda_i^+ - \lambda_i^-)(\lambda_j^+ - \lambda_j^-)\phi((x_i))^T\phi(x_j) - \varepsilon\sum_{i=1}^{l}(\lambda_i^+ - \lambda_i^-) + \sum_{i=1}^{l} y_i(\lambda_i^+ - \lambda_i^-) \\ \sum_{i=1}^{l}(\lambda_i^+ - \lambda_i^-) = 0 \\ \lambda_i^+, \lambda_i^- \in [0, C] \end{cases}$$

The number of variables is equal to $2l$.

So we have:

$$\begin{cases} \max_{\lambda^+,\lambda^-} -\frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}(\lambda_i^+ - \lambda_i^-)(\lambda_j^+ - \lambda_j^-)k(x_i, x_j) - \varepsilon\sum_{i=1}^{l}(\lambda_i^+ - \lambda_i^-) + \sum_{i=1}^{l} y_i(\lambda_i^+ - \lambda_i^-) \\ \sum_{i=1}^{l}(\lambda_i^+ - \lambda_i^-) = 0 \\ \lambda_i^+, \lambda_i^- \in [0, C] \end{cases}$$

Therefore the steps to solve the problem are the followings.

Step 1 Chose a kernel k

Step 2 Solve the dual: find (λ^+, λ^-)

Step 3 Find b:

$$b = y_i - \varepsilon - \sum_{j=1}^{l} (\lambda_j^+ - \lambda_j^-) k(x_i, x_j) \text{ for some } i \text{ s.t. } 0 < \lambda^+ < C$$

or

$$b = y_i + \varepsilon - \sum_{j=1}^{l} (\lambda_j^+ - \lambda_j^-) k(x_i, x_j) \text{ for some } i \text{ s.t. } 0 < \lambda^- < C$$

Step 4 Calculate the recession function:

$$f(x) = \sum_{i=1}^{l} (\lambda_i^+ - \lambda_i^-) k(x_i, x) + b$$

Recession function is linear in the features space and nonlinear in the input space.

Chapter 6

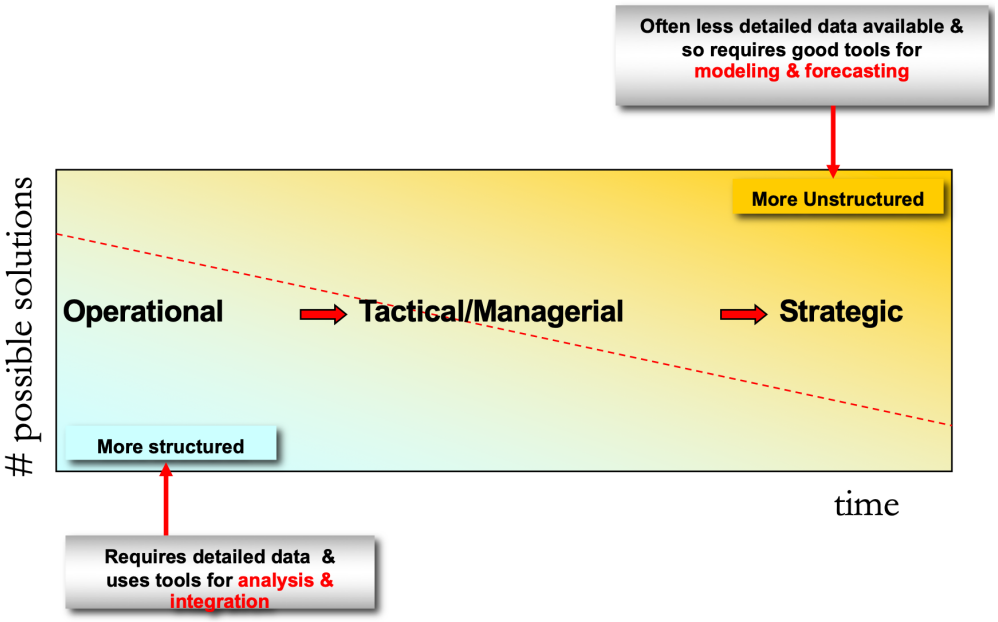
Decision Theory

6.1 Decision Making Under Uncertainty

6.1.1 Decision Analysis

Before continuing, let us recall some concepts about decision problems.

- **Structured problems**: Their goals are clear, are familiar (have occurred before), are easily and completely defined (information about the problem is available and complete).
- **Programmed decision**: A repetitive decision that can be handled by a routine approach.
- **Unstructured problems**: Problems that are new or unusual and for which information is ambiguous or incomplete, problems that will require custom-made solutions.
- **Non-programmed decisions**: Decision that are unique and nonrecurring, decision that generate unique responses.

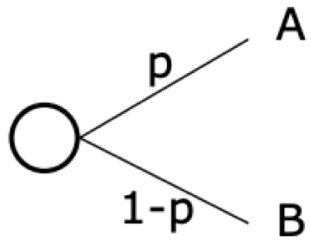


Certainty refers to a circumstance where all potential consequences of a given choice are clearly understood, allowing for precise and reliable decision-making. It's a state where one can be confident in the outcome of their actions because they have complete knowledge of all possible outcomes. On the other hand, **uncertainty** describes a situation where the outcomes of various options are unknown, making it difficult to make a definitive choice. Without knowing the result of each option, one cannot make an informed decision.

However, if there is some information available about the probability of different outcomes, the situation is referred to as *risk*. In this case, while the outcome cannot be known with absolute certainty, it is possible to estimate the likelihood of each possible outcome and make a decision based on that estimation.

In deterministic problems, as we have already mentioned, the results of our decisions are known and certain. However, in situations of uncertainty, the difficulties in making decisions arise even when there are only a few possible options, precisely because the challenge lies in the intrinsic evaluation of the choices.

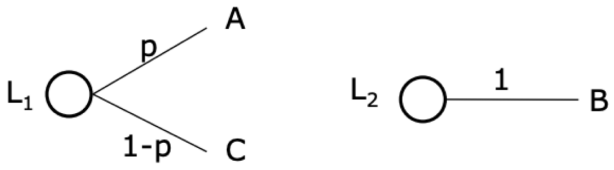
In decision theory, we talk about **lotteries** by identifying uncertain outcomes and representing them with probability theory.



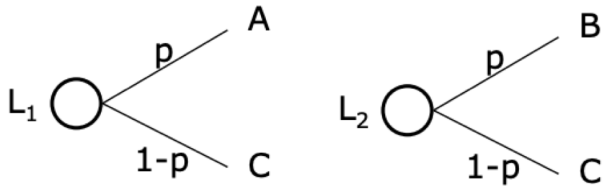
Whit probability p, outcome A occurs; with probability $1 - p$; outcome B occurs. The circle is also called a chance node.

We identify the following axioms:

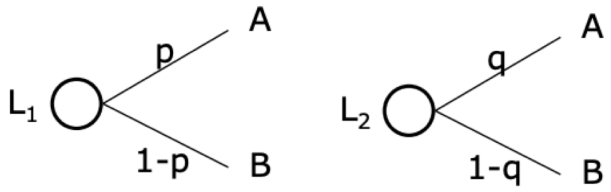
- **Orderability**: for every pair of primitive outcomes, either you prefer A to B, you prefer B to A, or you think A and B are equally preferable.
- **Transitivity**: if you like A better than B, and you like B better than C, then you like A better than C.
- **Continuity**: if you prefer A to B and you prefer B to C, then there's some probability that makes the following lotteries equivalently preferable for you.



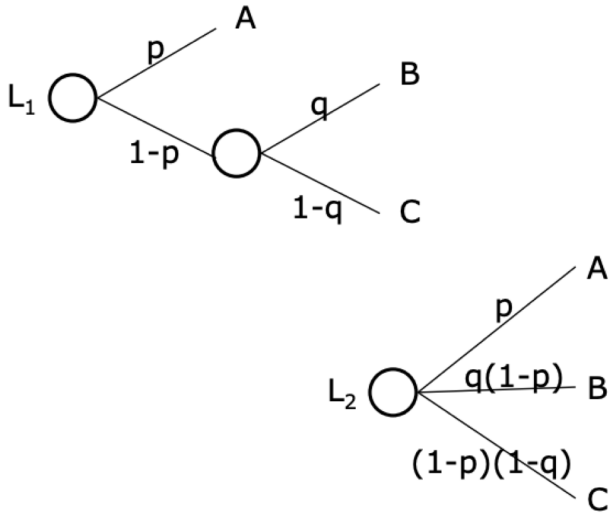
- **Substitutability**: if you prefer A to B, then given two lotteries that are exactly the same, except that one has A in a particular position and the other has B, you should prefer the lottery that contains A.



- **Monotonicity**: if you prefer A to B, and if p is greater than q, then you should prefer a lottery that gives A over B with higher probability.



- **Decomposability**: a two-stage lottery, where in the first stage you get A with probability p, and in the second stage you get B with probability q and C with probability $1 - q$ is equivalent to a single-stage lottery with three possible outcomes: A with probability p, B with probability $(1 - p)q$, and C with probability $(1 - p)(1 - q)$.



If preferences satisfy these six assumptions, then there exists U (a real valued function) such that:

- If A is preferred to be then $U(A) > U(B)$ and
- If A and B are equally perferable then $U(A) = U(B)$

Utility of a lottery is equal to expected utility of the outcomes.

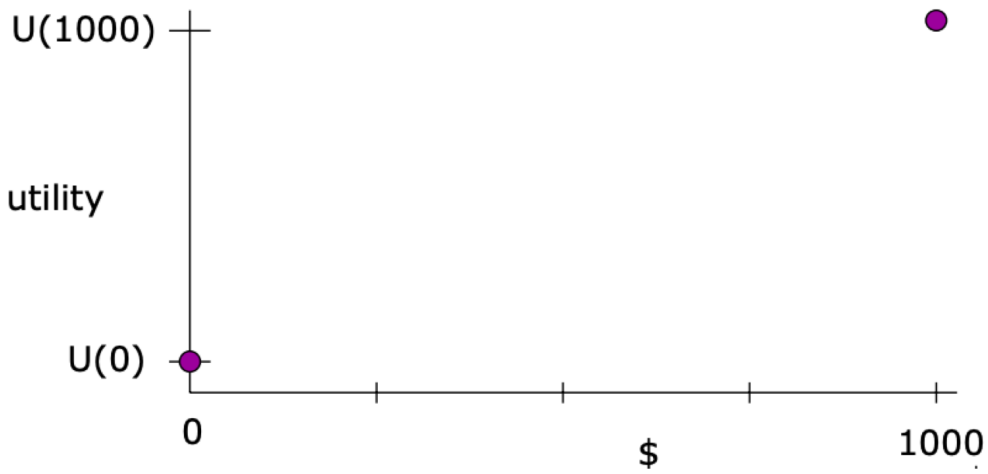
$$U(L) = p \cdot U(A) + (1 - p) \cdot U(B)$$

Let's now see an example

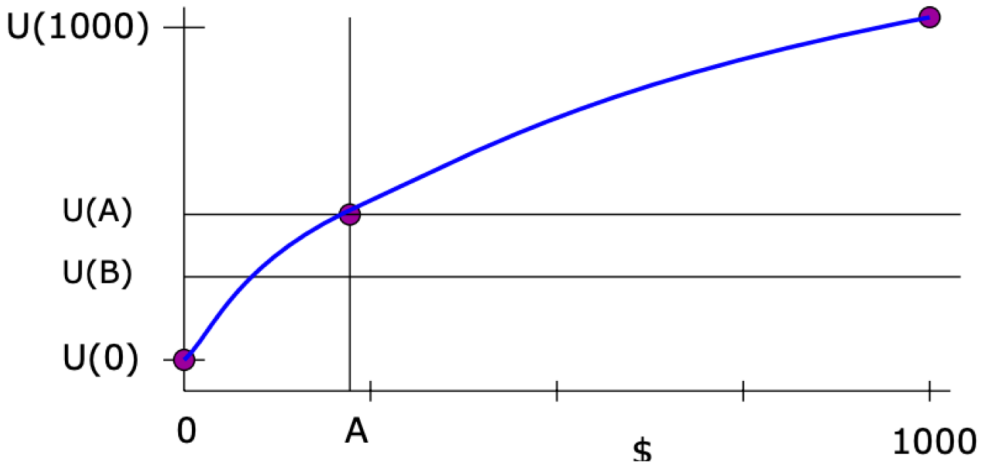
Which alternative would you prefer:

A. A sure gain of \$240
B. A 25% chance of winning \$1000 and a 75% chance of winning nothing.

$U(B) = 0.25U(\$1000) + 0.75U(\$0)$
$U(A) = U(\$240)$
$U(A) > U(B)$



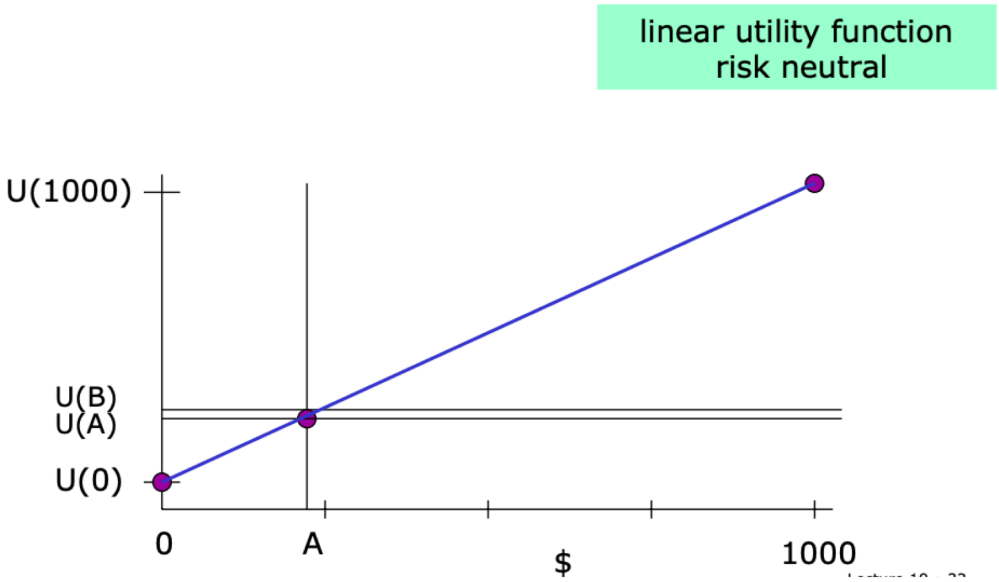
$U(B) = 0.25U(\$1000) + 0.75U(\$0)$
$U(A) = U(\$240)$
$U(A) > U(B)$



The shape of the interpolation curve is concave, and this type of curve is often referred to as **Risk Averse** because people often prefer a certain small amount of money rather than a higher expected value, due to aversion to risk.

This is just one form of curve, as we can also encounter another form: **Risk Neutrality**.

$U(B) = 0.25U(\$1000) + 0.75U(\$0) = U(\$250)$
$U(A) = U(\$240)$



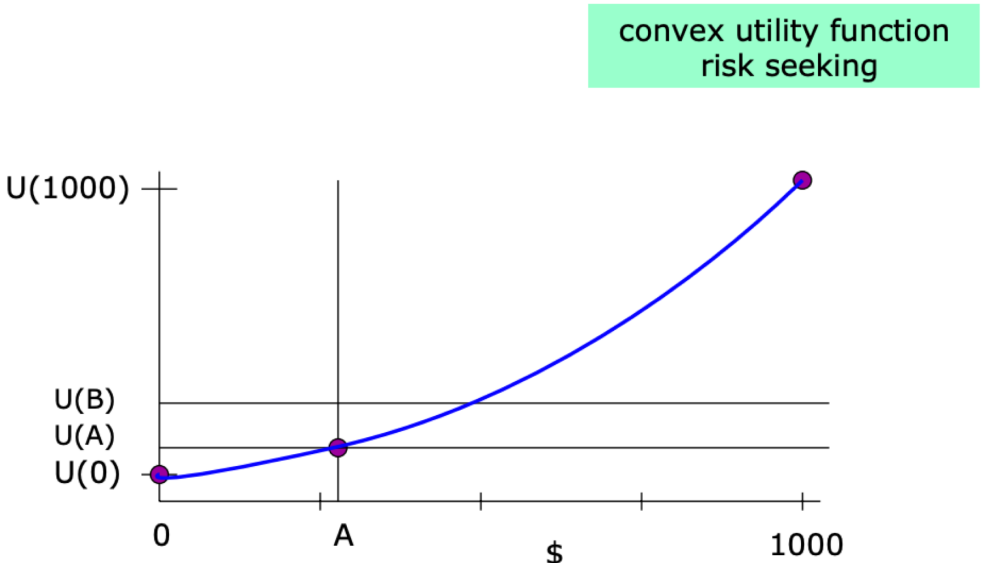
In this case, the expected utility value is proportional to the value that we could obtain. In this case, option B would be preferable.

Another form is that of the so-called **Risk-Seeking**, where the utility function is convex.

It's kind of like preferring lottery B to A, below:

A. A sure gain of \$260
B. A 25% chance of winning \$1000 and a 75% chance of winning nothing

$U(B) = 0.25U(\$1000) + 0.75U(\$0)$
$U(A) = U(\$260)$
$U(A) < U(B)$

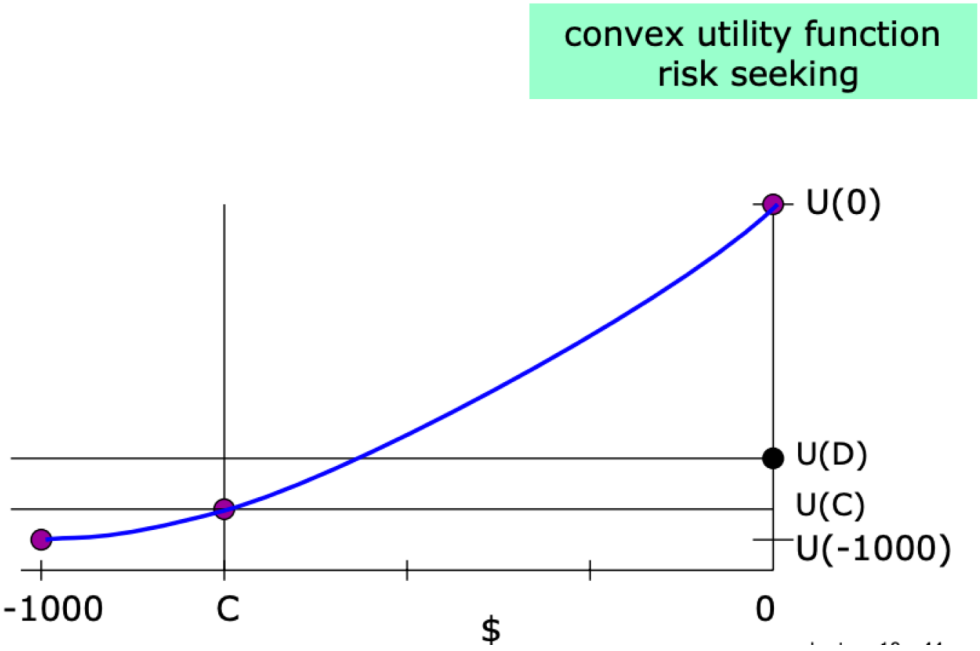


Which alternative would you prefer:

C. A sure loss of \$750
D. A 75% chance of losing \$1000 and a 25% chance of losing nothing

If we prefer option D to option C, then we could represent the utility function as follows:

$U(D) = 0.75U(-\$1000) + 0.25U(\$0)$
$U(C) = U(-\$750)$
$U(D) > U(C)$

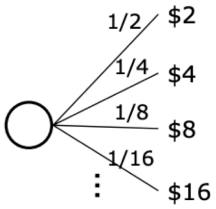


That is, we accept the risk of losing \$1000 in order to have a chance of not losing anything.

Finally, let's analyze a new question. It is a gambling game that requires an entry fee.

How much would you pay to play the following game? We flip a coin.

- If it comes up heads, I'll pay you \$2.
- If it comes up tails, we'll flip again, and if it comes up heads, I'll pay you \$4.
- If it comes up tails, we'll flip again, and if it comes up heads, I'll pay you \$8.
- And so on, out to infinity.



Expected value = $1 + 1 + 1 + \ldots = \infty$

This is called the St. Petersburg paradox. According to traditional decision theory, when observing the expected value, one could afford to pay any amount A to participate, since the expected value is infinite. In fact, even if the entry fee may be enormous, according to probability theory, there will be a sequence of throws sooner or later that can compensate

for all the losses that occurred before. However, we would not want to pay more than a few dollars to participate, and this is the paradox.

6.2 Decision Trees

6.2.1 DT Introduction

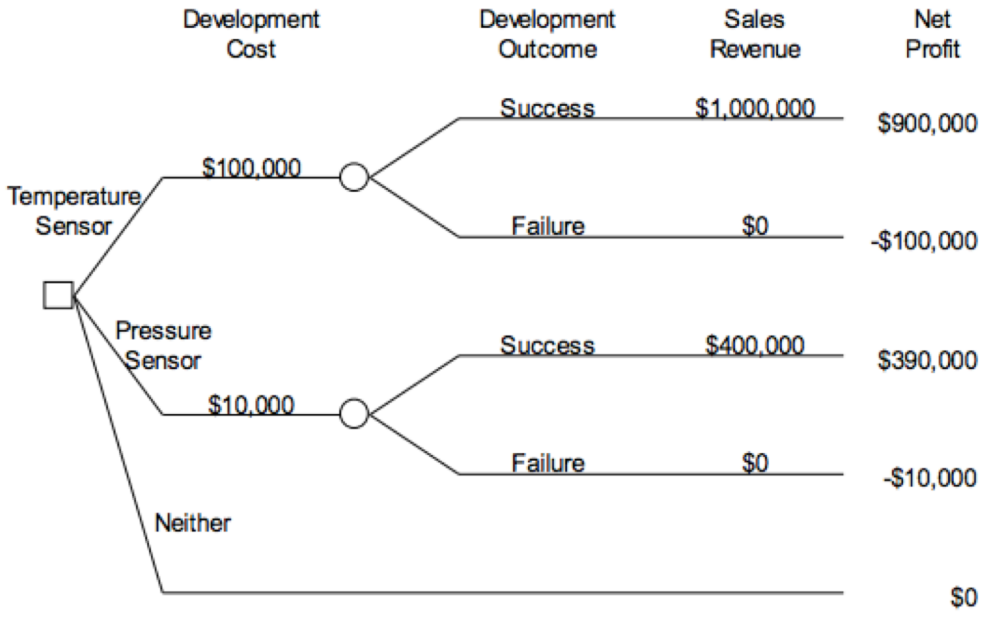
Undoubtedly, one of the most crucial and challenging tasks that a manager encounters is to make decisions in an environment that is fraught with uncertainty. This is particularly true when faced with questions such as: how much capital should be invested in new plant capacity, especially when the future demand for products is uncertain? Which marketing strategy should be implemented for a new product, given that consumer response to different marketing strategies is uncertain? And, should one invest in a new venture or merge with another firm in a foreign country, given the uncertain economic and political environment?

The analysis of complex decisions that involve significant uncertainty can be perplexing. This is because the outcome cannot be predicted with certainty, and there are a multitude of different factors that must be taken into account. Moreover, collecting additional information can be crucial in reducing the level of uncertainty. Lastly, the decision taken can be heavily influenced by the manager's attitude towards risk.

We introduce a very important method for structuring and analyzing managerial decision problems in the face of uncertainty, in a systematic and rational manner. The method goes by the name **decision analysis**. The analytical model that is used in decision analysis is called a **decision tree**.

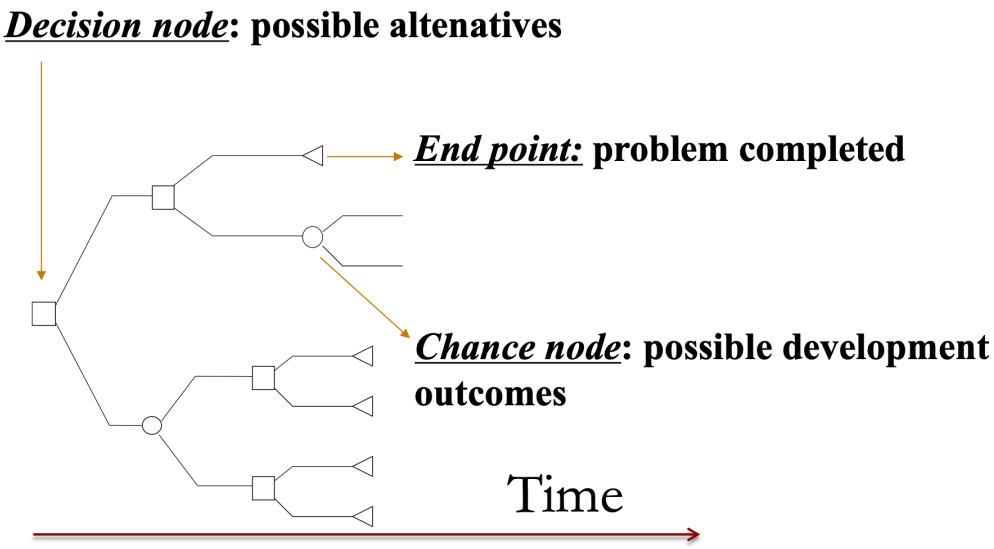
For example we can face a product decision like the following. *To absorb some short-term excess production capacity at its Arizona plant, Special Instrument Products is considering a short manufacturing run for either of two new products, a temperature sensor or a pressure sensor. The market for each product is known if the products can be successfully developed. However, there is some chance that it will not be possible to successfully develop them. Revenue of $1,000,000 would be realized from selling the temperature sensor and revenue of $400,000 would be realized from selling the pressure sensor. Both of these amounts are net of production cost but do not include development cost. If development is unsuccessful for a product, then there will be no sales, and the development cost will be totally lost. Development cost would be $100,000 for the temperature sensor and $10,000 for the pressure sensor.*

Which, if either, of these products should Special Instrument Products attempt to develop?



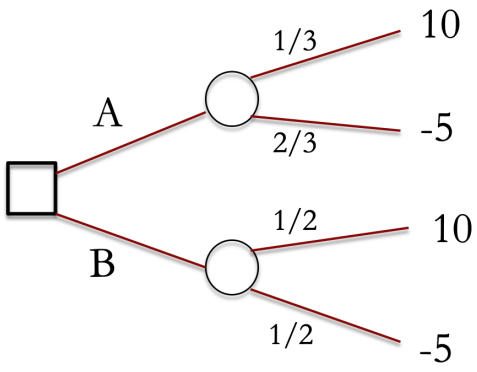
The decision tree is made of several key elements:

- **Decision node**: that identify possible alternatives
- **Chance node**: that provide possible development of outcomes
- **End point**: and of a root, identify the problem completed
- **Time**: dimension on which the tree is developed



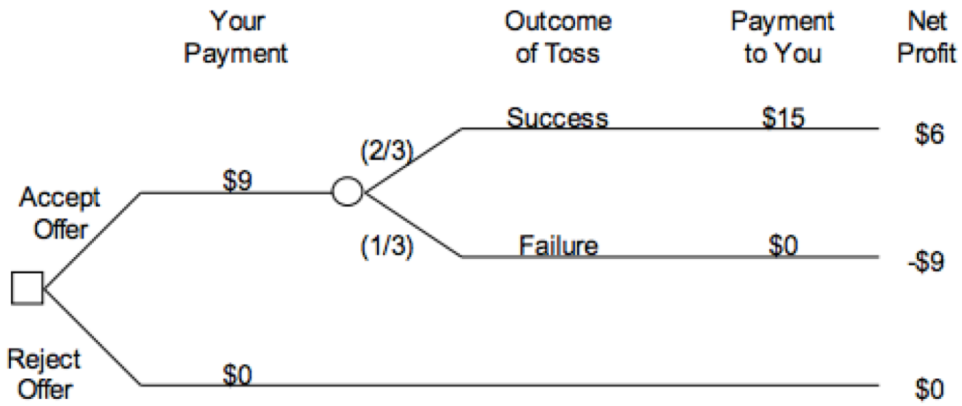
Another example can be about tossing a die. *Suppose you are offered two alternatives, each of which consists of a single toss of a fair die. With the first alternative, you will win $10*

if any number greater than 4 is thrown, and lose \$5 otherwise. With the second alternative, you will win \$10 if any number greater than 3 is thrown, and lose \$5 otherwise. In this case, since there are 6 faces on a die, the probability of winning is $2/6 = 1/3$ for the first alternative and $3/6 = 1/2$ for the second alternative. Since the consequences are the same for both alternatives and the probability of winning is greater for the second alternative, you should select the second alternative.



In order to decide which alternative to select we need a decision criteria. **Expected value** (EV) is often a good measure of the value of an alternative since over the long alternative run this is the average amount that you expect to make from selecting the alternative.

For example in a dice roll. *A friend proposes a wager: You will pay her \$9.00, and then a fair die will be rolled. If the die comes up a 3, 4, 5, or 6, then your friend will pay you \$15.00. If the die comes up 1 or 2, she will pay you nothing. Furthermore, your friend agrees to repeat this game as many times as you wish to play.*

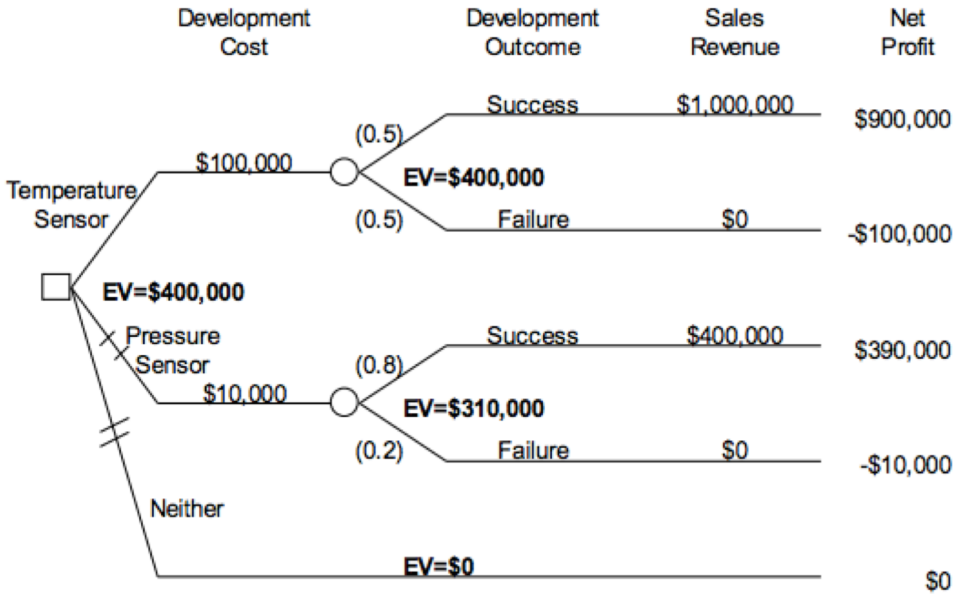


We have:

$$EV = 2/3 * 6 + 1/3 * (-9) = 1\$$$

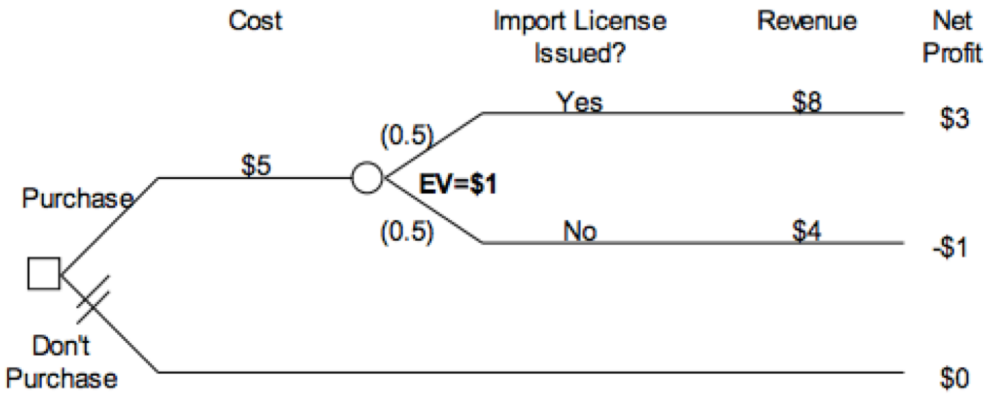
$$EV(1500plays) = 2/3 * 1500 * 6 + 1/3 * 1500 * (-9) = 1000 * 6 + 500(-9) = 1500\$$$

Another example is the following product decision. *Suppose that in the first example the probability of development success is 0.5 for the temperature sensor and 0.8 for the pressure sensor.*



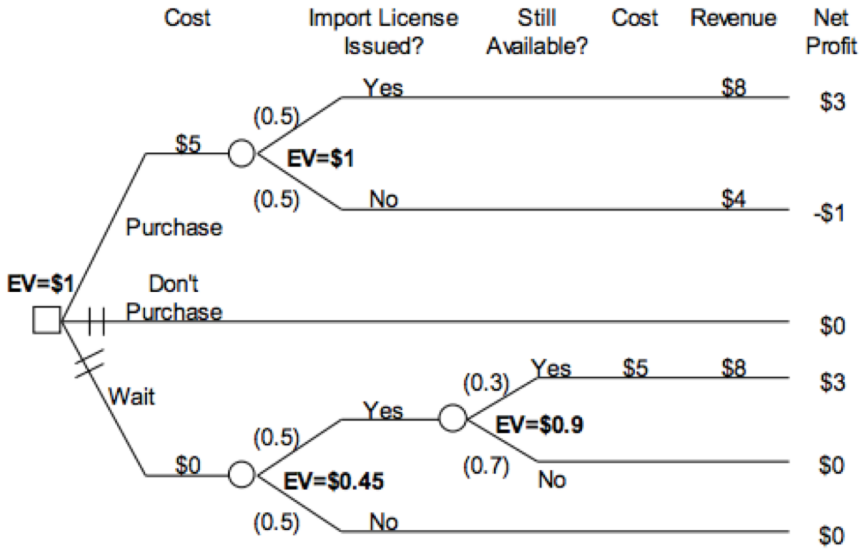
6.2.2 Expected Value Solution

Xanadu Traders (U.S. metals broker) has acquired an option to purchase one million Kgs of partially refined molyzirconium ore from the Zeldavian goverment for $5.00 per kg. Molyzirconium can be processed into several products used in semiconductor manufacturing and Xanadu Traders estimates would be able to sell the ore for $8.00 per kg after importing it. However the US goverment is currently negotiating with Zeldavia over alleged "dumping" of certain manufactured goods which the country export to the U.S. As part of this negotiation, The U.S. goverment has threatened to ban the import from Zeldavia of a class of materials that includes molyzirconium. If the U.S. goverment refuses to issue an import licence for the molyzirconium after Xanadu has purchased it, then Xanadu will have to pay a penalty of $1.00 per kg to the Zaldavian goverment to annul the purchase. Xanadu estimates that there are 50% of chances to obtain the import licence.
Which alternative should Xanadu select ?



Xanadu Traders ask support to an Analyst that suggests that in order to reduce the risk they can apply for the import licence and wait until they know if it is approved before closing the deal with Zeldavia.
In this case there is a 0.70 probability that someone else will take Zeldavia's offer.

In this new scenario, in addition to the various possibilities already explored, a new choice is presented that is divided into further sub-choices. This choice is whether to wait for the outcome of the request and make the decision to purchase or not based on that outcome.



Decision Tree Rollback

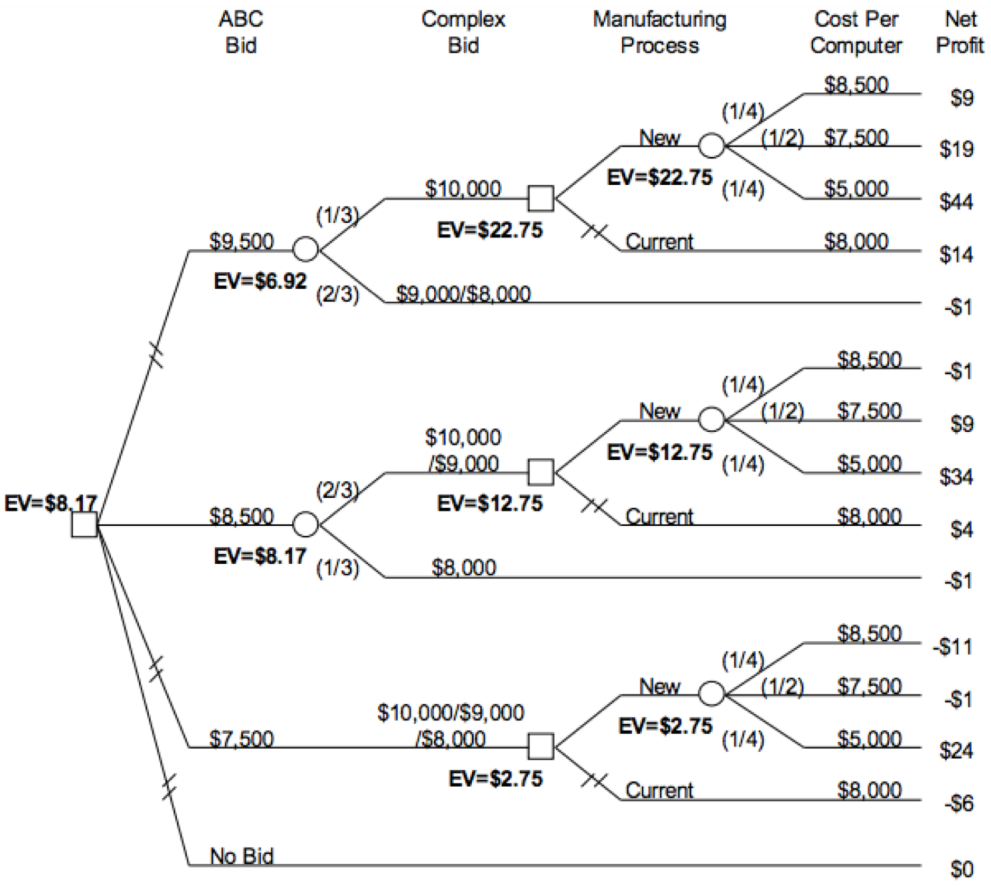
The uncertainty of this problem was given by the lack of prior knowledge of the acceptance or rejection of the license.

Now, we will analyze a case where uncertainty is instead linked to previous actions, meaning that the decisions we make in the future will depend on the decisions we make now.

ABC Computer Company is considering submission of a bid for a government contract to provide 10.000 specialized computers There is only one other potential bidder for this contract and low bidder will receive the contract. ABC is currently working on a new process to manufacture the computers that it should substantially lower the cost of making the computers. However, it may not work and in this case the process will be more expensive than the current process. If ABC decides to bid, it will make one of the three bids: $9500, $8500 or $7500 per computer The competitor will certainly bid either $10000, $9000 or $8000 per computer If ABC decides to bid than it will cost $1.000.000 to prepare the bid With current process the cost of building a computer is $8000 With the new process this cost will be $5000 with prob 0,25, $7500 with prob 0,5 and $8500 otherwise

It is clear that if the company does not issue any invoices, it will certainly not make any profit. However, if it were to submit a bid, the probability of winning each bid (the lowest bid wins) must be evaluated. Once the tender is won, the decision about the production process will finally be made, namely whether to stay in the current process or invest in the new one. In this latter branch of the decision-making process, there is a probabilistic dependence on the event of unit cost for computers.

Therefore, let us examine the new decision tree, with all the expected values included.



The net profit is calculated, in case of winning the bid, by subtracting one million and the cost incurred from the bid made. In case of defeat, however, one million is still charged for preparing the bid. The Net Profit column is expressed in millions.

Therefore, the most convenient decision is the one that gives us the highest expected value, always expressed in millions. Thus, making a bid of $8500 per computer has an expected profit of 8.17 million.

6.2.3 Utility Functions

Using expected value as a decision criterion is reasonable as long as the stakes involved are small enough to rely on the long-term averages. It's worth noting that the value of a risky alternative can differ from its expected value.

To address this, the concept of certainty equivalent comes into play. The certainty equivalent is the guaranteed amount that you would be equally happy to receive instead of a risky alternative. Selling price is another term for certainty equivalent.

As an example, let's say you have obtained a risky alternative with a 50-50 chance of yielding either a \$10,000 profit or a \$5,000 loss, through a previous business deal. The expected value of this alternative is \$2,500. However, you might decide that you're willing to sell this alternative for \$500. In this case, your certainty equivalent for the alternative is \$500.

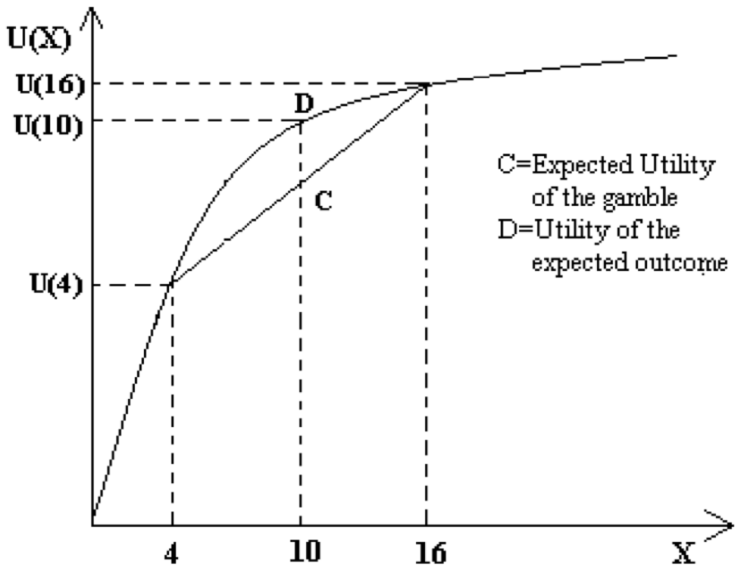
Depending on how your certainty equivalent compares to the expected value, you can be categorized as risk averse, risk neutral, or risk seeking.

- **Risk Averse**: If your Certainty Equivalent $<$ Expected Value
- **Risk Neutral**: If your Certainty Equivalent $=$ Expected Value
- **Risk Seeking**: If your Certainty Equivalent $>$ Expected Value

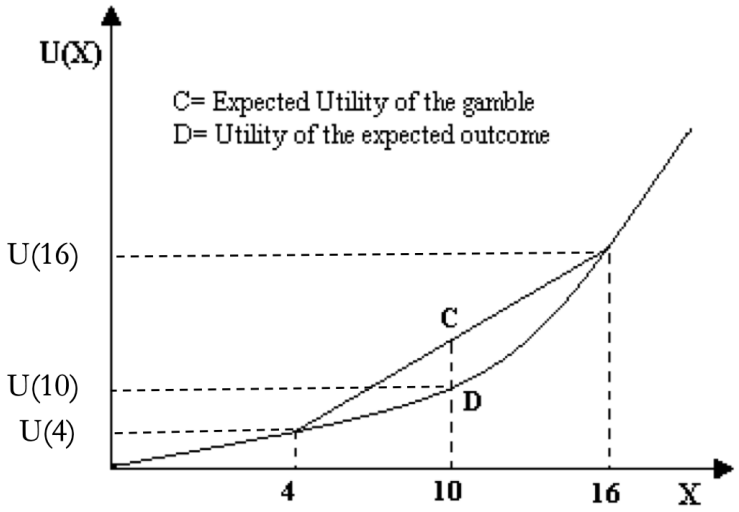
A **utility function** translates outcomes into numbers such that the expected value of the utility numbers can be used to calculate certainty equivalents for alternatives in a manner that is consistent with a decision maker's attitude toward risk taking.

Suppose a game with two possible outcomes with equal probability and a given utility function.

If $U(10) - U(4) > U(16) - U(10)$ and $U(10) > (1/2)U(16) + (1/2)U(4)$.



If $U(10) - U(4) < U(16) - U(10)$ and $U(10) < (1/2)U(16) + (1/2)U(4)$.



We can transform the Utility function in the Exponential Utility function as follow:

$$U(x) = 1 - e^{-x/R}$$

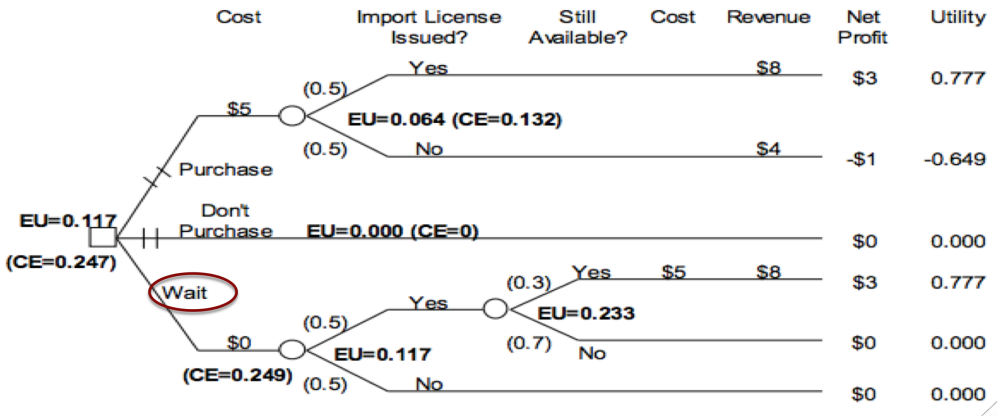
For $R > 0$, it is a constant called risk tolerance and can depend on the asset position of the decision making entity. As R becomes larger the utility function display less risk aversion

Ask the decision maker to consider a hypotetical alternative that has equal chances of yielding a profit of r or a loss of $r/2$. *For example if Xanadu is just willing to accept a deal with fifty-fifty chance of making \$2.000.000 or losing \$1.000.000 then* $r = 2$.

For each utility function there is a specific **certainty equivalent**. In particular, for and exponential utility function:

$$CE = -R \cdot \ln(1 - E[U])$$

So we can have both expected values analysis and expected utility analysis. Now lets see expected utility.



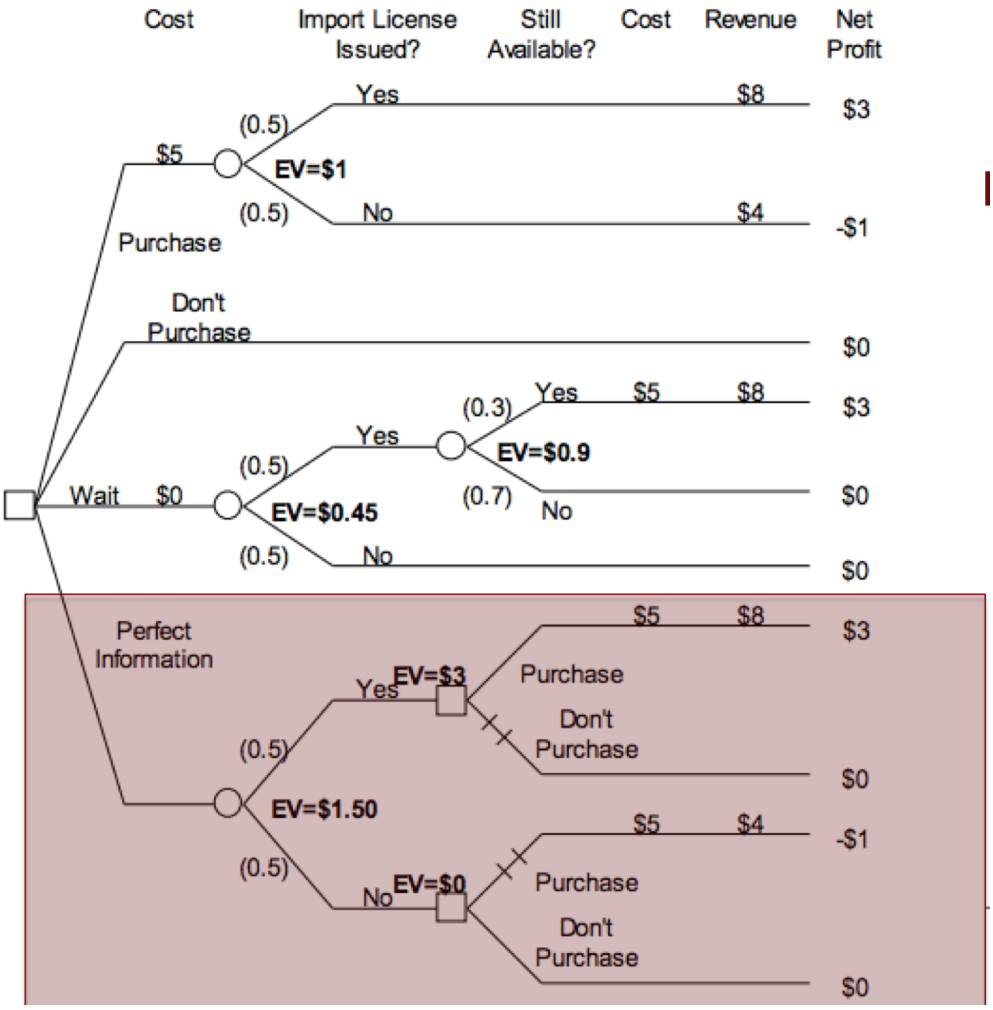
We can see the comparison of expected values and certainty equivalents.

Alternative	Expected Value	Certanty Equivalent	Difference
Purchase	1.000	0.132	0.868
Don't Purchase	0.000	0.000	0.000
Wait	0.450	0.249	0.201

6.2.4 The Value of Information

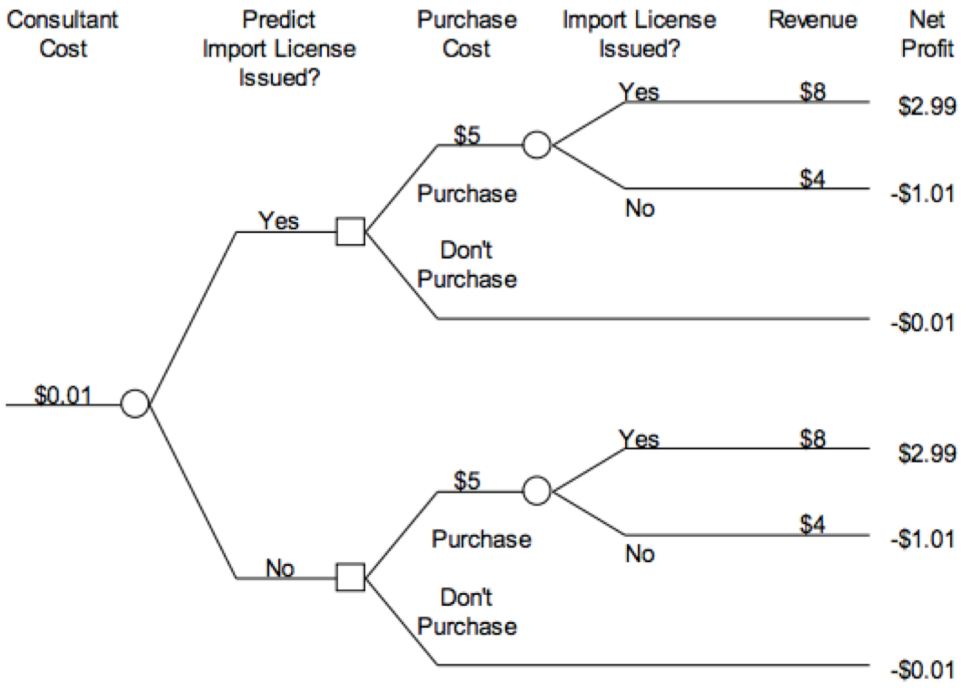
The concept of the **value of perfect information** revolves around the idea that if we have all the necessary information to make a decision, we can eliminate any uncertainty surrounding the potential outcomes of the available alternatives. It follows that no amount of information that is less than perfect can be more valuable than perfect information itself. This means that imperfect information, which is always a part of any decision-making process, can never be more valuable than perfect information. In essence, the value of perfect information serves as the maximum value that any source of information can provide.

Suppose a source of perfect information existed that would let Xanadu know if the import license would be issued: *How much money would it be worth to pay in order to obtain perfect information about issuance of the import license?*

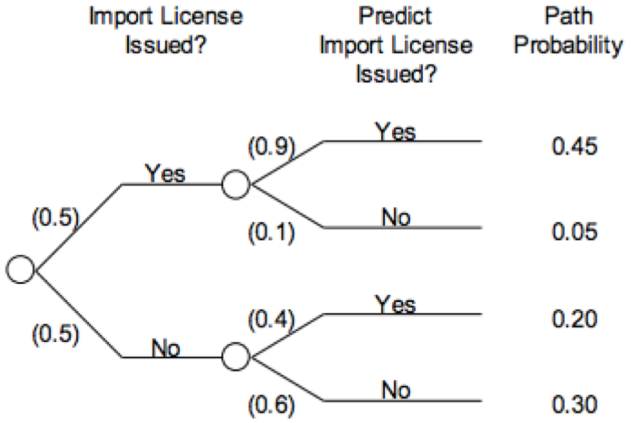


Let us now suppose that Xanadu can pay a consultant with good connections in the import licensing bureaucracy. For a fee, he will consult his contacts and see if they think the license will be granted. This is a pretty standard job for him. His fee for this type of service is \$10,000. Of course, his assessment that the license will come through is no guarantee. Xanadu knows the consultant and says that in cases where the import license was ultimately granted, he called it right 90% of the time. However, he hasn't been so good on the license requests that were turned down. In those cases, he only called it right 60% of the time. Should Xanadu hire the consultant, and if so, what is the maximum amount that he should pay him for his services?

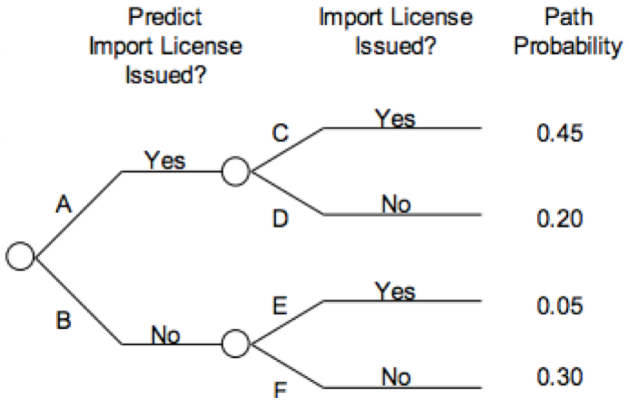
We know that the maximum amount that it is worth to purchase the services is \$0.5 million. Since he would only charge \$10,000 it is possible that it would be worth purchasing his services. However, it is clear from the discussion above that the consultant often makes mistakes, and perhaps Xanadu would not learn enough to warrant paying him the \$10,000.

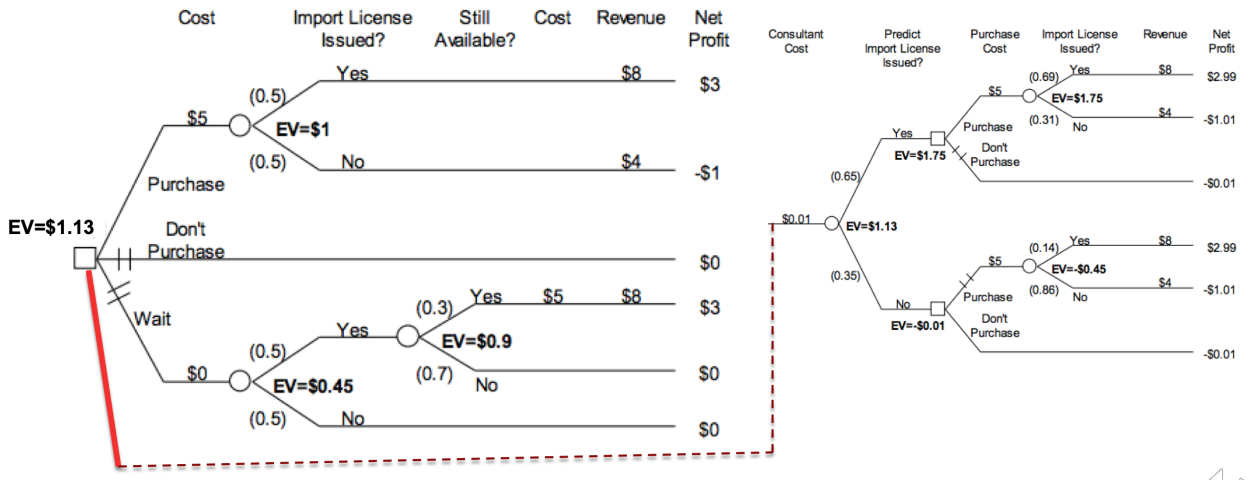


We need to know the probability of the branches "Import licence issued" given the "predicted import licence issued" and we do not have the information in a form that directly provides the required probabilities.



Apply Bayes Rule….





So we have at the end:

- Max Expected Net Profit without information = $\$1(x1.000.000)$
- Max Expected Net Profit with information = $\$1,13(x1.000.000)$
- Cost of Information = $\$0,01(x1.000.000)$
- Expected value of Information = $\$1,13 + \$0,01 - \$1 = \$0,14(x1.000.000)$

It is worth to hire the consultant to acquire more information and the increase in Expected Net Profit is $\$0,13(x1.000.000)$.
Expected value of Information (EVI) $\leq$ Expected Value of Perfect Information (EVPI)