One may note that $\boldsymbol{x}$, $\boldsymbol{d}$, $\boldsymbol{p}$ may be continuous, discrete, integer, binary, etc., depending on the scope of the model. Stochasticity in (12.2) comes from two set of variables which are $\boldsymbol{x}$ and $\boldsymbol{p}$, and for a better illustration of the same one should refer to Figure 12.5 which shows a hypothetical problem with two stochastic inequality constraints (i.e., function of x_1 and x_2), where, typically the deterministic optimal solution lies on a constraint boundary or at the intersection of more than one constraint, as shown by the red dot (Figure 12.5).

In the event of uncertainties in the design/decision variables (i.e., x_1 and x_2 in this example), there may be instances when the deterministic solution become infeasible, and to find a reliable solution (meaning that there is a very small probability of instances producing an infeasible solution), the true optimal solution is sacrificed and a solution (blue dot, Figure 12.5) in the interior of the feasible region is chosen, such that the level of confidence inside the region marked with blue outline is actually at the specified reliability level, say α_j. Thus, for a reliability measure α_j, it is desired to find the feasible solution that will ensure that the probability of having an infeasible solution is at most α_j. To incorporate this uncertainty and find the reliable solution, the following probabilistic modification of (12.2), is considered, where in the reliability concept is formulated as the probability of the constraint satisfaction, $g_j(d,p,x) \geq 0$ greater than or equal to the desired probability a_j. So the probabilistic counter part of (12.2) is shown in (12.3).

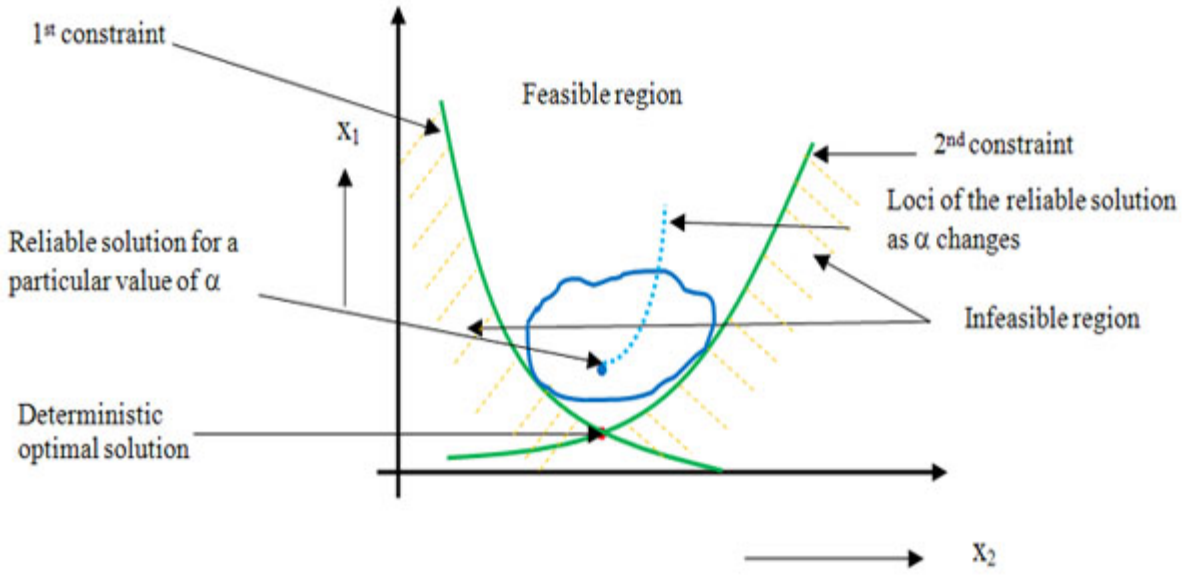


Figure 12.5: Illustration of the concept of Reliability Based Design Optimization (RBDO)

(12.3)

$$
\begin{array}{lll}
\underset{\forall(x,d)}{Optimize} & \sum_{i=1}^{I} f_i(x,d,p) & \\
s.t.: & P\left[g_j(\mathrm{x},\mathrm{d},\mathrm{p}) \geq 0\right] \geq \alpha_j & j = 1,\ldots,J \\
 & h_k(x,d,p) = 0 & k = 1,\ldots,K \\
 & x \in \Re^n & n = 1,\ldots,N \\
 & d \in \Re^m & m = 1,\ldots,M \\
 & p \in \Re^l & l = 1,\ldots,L
\end{array}
$$

Previous Next

An example in optimization and use of reliability based optimization

In optimization problems one tries to find the best possible allocation of resources amongst the probable set of alternatives (feasible sets), to find a unique combination of the control/decision variables, subject to some restrictions/constraints on the control/decision variables domain space, in order to optimize the given objective function(s). Mathematically a generic optimization problem can be stated as given in (12.2).

$$\left.\begin{array}{lll} \underset{v(x,d)}{Optimize} & \sum_{i=1}^{I} f_i(x,d,p) & \\ s.t.: & g_j(\mathrm{x},\mathrm{d},\mathrm{p}) \geq 0 & j=1,\ldots,J \\ & h_k(x,d,p)=0 & k=1,\ldots,K \\ & x \in \Re^n & n=1,\ldots,N \\ & d \in \Re^m & m=1,\ldots,M \\ & p \in \Re^l & l=1,\ldots,L \end{array}\right\} \qquad (12.2)$$

where (i) $\mathbf{x}'$ is the objective function which may be linear/non-linear, single/multi-objective, (ii) $g_j(x,d,p) \geq 0$, $j=1,\ldots,J$, are the inequality constraints, (iii) $h_k(x,d,p)=0$, $k=1,\ldots,K$, are the equality constraints, (iv) $x \in \Re^n$, $n=1,\ldots,N$, is the vector of probabilistic control/decision variables, (v) $d \in \Re^m$, $m=1,\ldots,M$, is the vector of deterministic control/decision variables, and (vi) $p \in \Re^l$, $l=1,\ldots,L$, is the vector of probabilistic exogenous parameters to the system.

Previous Next

Concept of Markov Chain Monte Carlo (MCMC) method

A class of algorithms which are used for sampling from a particular population distribution whether of parameter form or non-parametric form such that one can understand the population characteristics using the concept of Markov chain is caled Markov chain Monte Carlo (MCMC) methods. One must remember that MCMC methods include random walk Monte Carlo methods also. For this we use the state of the Markov chain as a sample from the desired distribution, and to achieve this one has to use a large number of steps/iterations to obtain a level of stability. So the quality of the sample improves as a function of the number of steps. Thus in many of the MCMC methods it is advisable that the first 1000 to 5000 iterative run values are disregarded till consistency is achieved. Hence the number of such iterative runs which are to be disregarded depends on the type of Markov chain one has, based on which the sampling plan is being built.

Generally it is easy to know the properties of the Markov Chain, based on which the MCMC works. As already mentioned the difficult part is to determine how many steps are needed to converge to the stationary distribution within an acceptable level of error. A clever way is to start the MCMC method, (based on some Markov chain concept) is to consider an arbitrary initial starting point at each run of the MCMC method, and then allow the MCMC method to continue for N number of times. Repeat the MCMC method with the same Markov chain, but now considering a different starting point, i.e., state. Repeat this large number of times such that the general properties of the MCMC methods are obtained in the long run. The importance of this can be judged from the fact that the states can have different probability, and in the long run it dictates the efficacy of starting at a partcilar state, *i*. Thus one should continue sampling from the stochastic random process such that one is able to ensure that $\lim_{n\to\infty} P^n$ is obtained. It is important to remember that we need to mix the chain in some manner so that stability is achieved such that the values obtained are taken from the actual theoretical distribution, which is very important to us. For the convenience of the reader a simple pseudo-code for the Markov chain Monte Carlo (MCMC) is illustrsted in Figure 12.4.

Pseudo code for MCMC Method

Step1. Take some initial guess of the parameter vector $\theta = (\theta_1, \ldots, \theta_k)$.

Step2. Suppose at the i^{th} step the parameter vector $\theta = (\theta_1, \ldots, \theta_k)$ take values $\theta_i = (\theta_{1,i}, \ldots, \theta_{k,i})$

Step 3 Generate $\theta_{i+1} = (\theta_{1,i+1}, \ldots, \theta_{k,i+1})$ from $f(\theta_{1,i+1} | X, \theta_{2,i+1}, \ldots, \theta_{k,i+1})$ and so on for each of the parameter values

Step3. Repeat steps 1-2 N number of times.

Step4. Calculate Bayes estimator (*here a prior distribution is very important*) of the function g(θ) depending on the estimate based on the fact how many reading you will use and howmany you will discard. Here the number of reading you disregard is the burn in period, usually denoted by M.

Figure 12.4: Pseudo-code for Markov chain Monte Carlo (MCMC)

Typical use of MCMC sampling can only approximate the target distribution, as there is always some residual effect of the starting point/state. More sophisticated MCMC-based algorithms such as coupling from the past can produce exact samples, at the cost of additional computation and an unbounded (though finite in expectation) running time.

We know that random walk methods are a kind of random simulation or Monte Carlo method. An important thing to note is the fact that the random samples generated during a conventional Monte Carlo integration are statistically independent, but those used in MCMC are correlated.

Previous Next

Simulated Annealing (based on Metropolis Algorithm)

Simulated annealing (SA) is a generalization of a Monte Carlo method for examining the equations of state and frozen states of n-body systems. The concept is based on the manner in which liquids freeze or metals recrystalize in the process of annealing. In an annealing process a metal, initially at high temperature and in a disordered state is slowly cooled so that the system at any time is approximately in thermodynamic equilibrium. As cooling proceeds, the system becomes more ordered and approaches a frozen ground state at $T = 0$.Hence the process can be thought of as an adiabatic approach to the lowest energy state. If the initial temperature of the system is too low or cooling is done insufficiently slowly then the system may become quenched forming defects or freezing out in metastable states (i.e., trapped in a local minimum energy state).

By analogy the generalization of this Monte Carlo approach to combinatorial problems is straight forward. The current state of the thermodynamic system is analogous to the current solution to the combinatorial problem, the energy equation for the thermodynamic system is analogous to at the objective function, and the ground state is analogous to the global minimum. The major difficulty in implementation of the algorithm is that there is no obvious analogy for the temperature with respect to a free parameter in the combinatorial problem. Furthermore, avoidance of entrainment in local minima (quenching) is dependent on the "annealing schedule", (i) the choice of initial temperature, (ii) how many iterations are performed at each temperature, and (iii) how much the temperature is decremented at each step as cooling proceeds.

Simulated annealing has been used in various combinatorial optimization problems and has been particularly successful in circuit design problems. Figure 12.3 gives the pseudo-code for the Simulated Annealing (SA) process, which the reader is advised to understand for a better understanding of the procedure.

Pseudo code for the Simulated Annealing (SA) method

```
Generate random valid solution x
Repeat
    Generate new solution x′ by randomly modifying the current solution x
        Evaluate new solution x′.
        If acceptance criterion is met Then;
        Replace x with x′;
End;
Adjust acceptance criterion;
Until halting criterion is met;
```

Figure 12.3: Pseudo-code for Simulated Annealing (SA)

Previous Next

The question which is important for us to answer is, what happens when the search is probabilistic or stochastic. Let us consider the example for which we have $P_{11}=1$ and $P_{ij}=\left(\frac{1}{i-1}\right)$, where $j=1,2,\ldots,i-1$ and $i>1$. Let T_i denote the number of transitions one needs to go from state i to state 1. So given the initial condition we have $E(T_i)=1+\left(\frac{1}{i-1}\right)\sum_{j=1}^{i-1}E(T_j)$ and with the initial condition that $E(T_1)=0$, we can obtain all the successive expected values.

Now if $T_N=\sum_{j=1}^{N-1}I_j$ where $I_j=\begin{cases}1 & if\ process\ ever\ enters\ j\\ 0 & otherwise\end{cases}$, then we can prove

(i) $E(T_N)=\sum_{j=1}^{N-1}\frac{1}{j}$

(ii) $V(T_N)=\sum_{j=1}^{N-1}\frac{1}{j}\left(1-\frac{1}{j}\right)$

(iii) For large N, T_N follows a Poisson process with a mean of $\log(N)$

Previous Next

Module 12:Application of stochastic processes in areas of engineering and management science

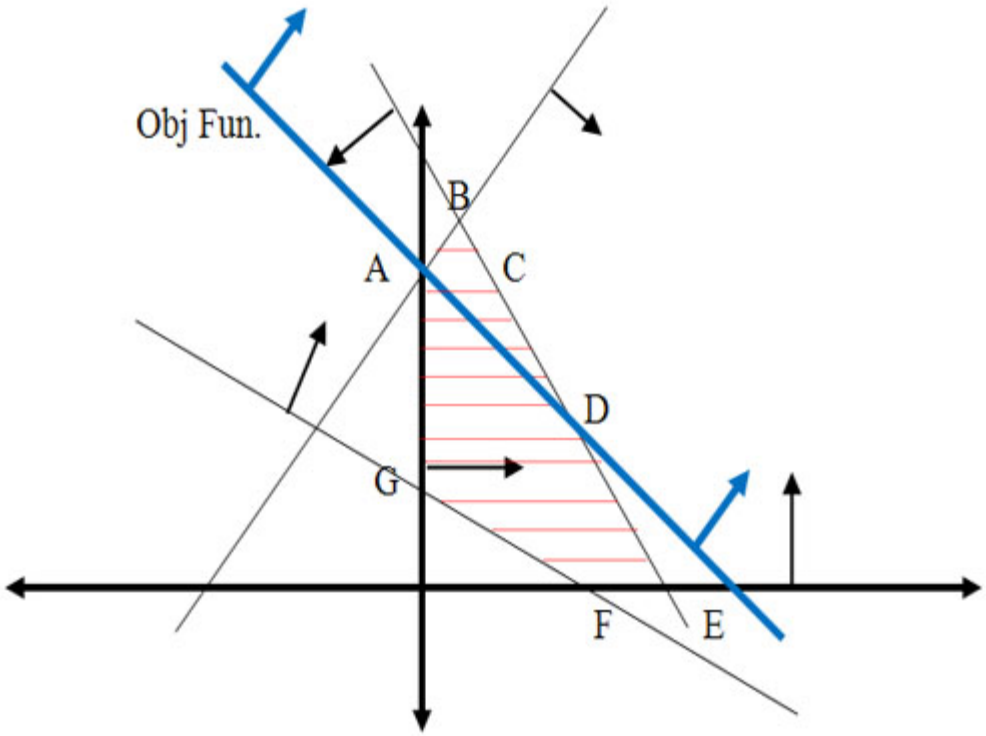
Lecture 39:Application of Markov chain in trying to judge the efficiency of algorithms in OR

It would be imperative to know what are the properties of LP and they are:

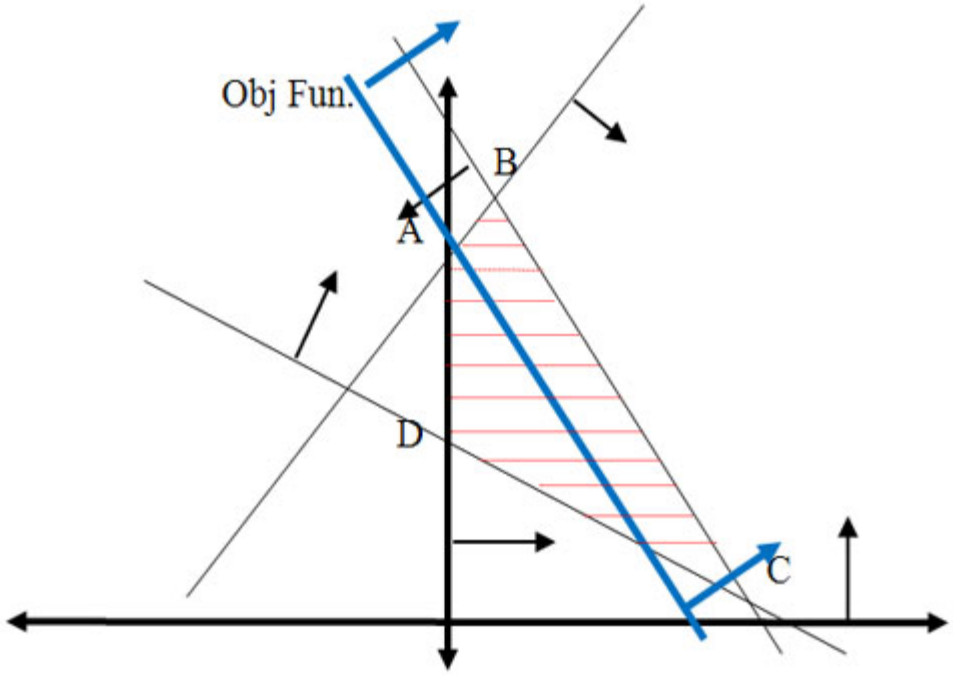
1) Proportionality
2) Additivity
3) Certainty

The main question which is important to understand is the feasible space which if known can enhance you search procedure to find the optimal point in the most efficient way. This feasible region/space (ABCDE, Figire 12.1) may of different types some of which are shown through case 12.1, case 12.2 and case 12.3. For this one can refer to Figure 12.2.

Case: 12.1



Case: 12.2



Case: 12.3

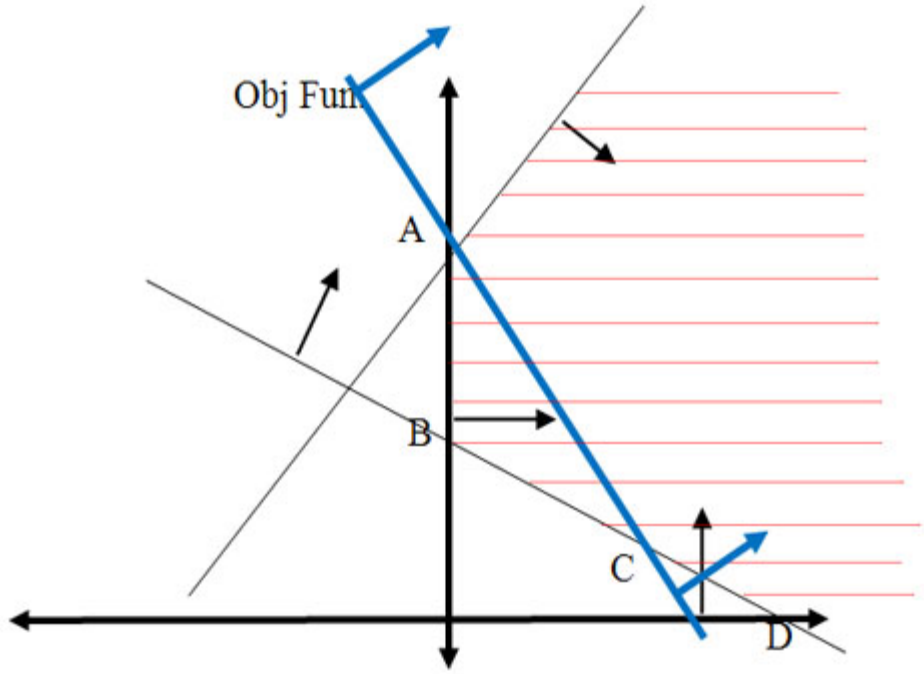


Figure 12.2: Different examples of feasible region/space

Previous Next

Thus we have from the table: $6*x_1 + 4*x_2 \leq 24$ and $1*x_1 + 2*x_2 \leq 6$. Here a colouring scheme has been employed to make the concepts clear when one refers to the diagrams given later. Finally the other constraints are: $x_2 - x_1 \leq 1$ and $x_2 \leq 2$. Thus the problem is as given below and for illustration the reader is requested to refer to Figure 12.1.

Maximize $z = 5*x_1 + 4*x_2$

s.t:

$$6*x_1 + 4*x_2 \leq 24 \quad (12.1a)$$

$$1*x_1 + 2*x_2 \leq 6 \quad (12.1b)$$

$$-x_1 + x_2 \leq 1 \quad (12.1c)$$

$$0*x_1 + x_2 \leq 2 \quad (12.1d)$$

$$x_1, x_2 \geq 0 \quad (12.1e)$$

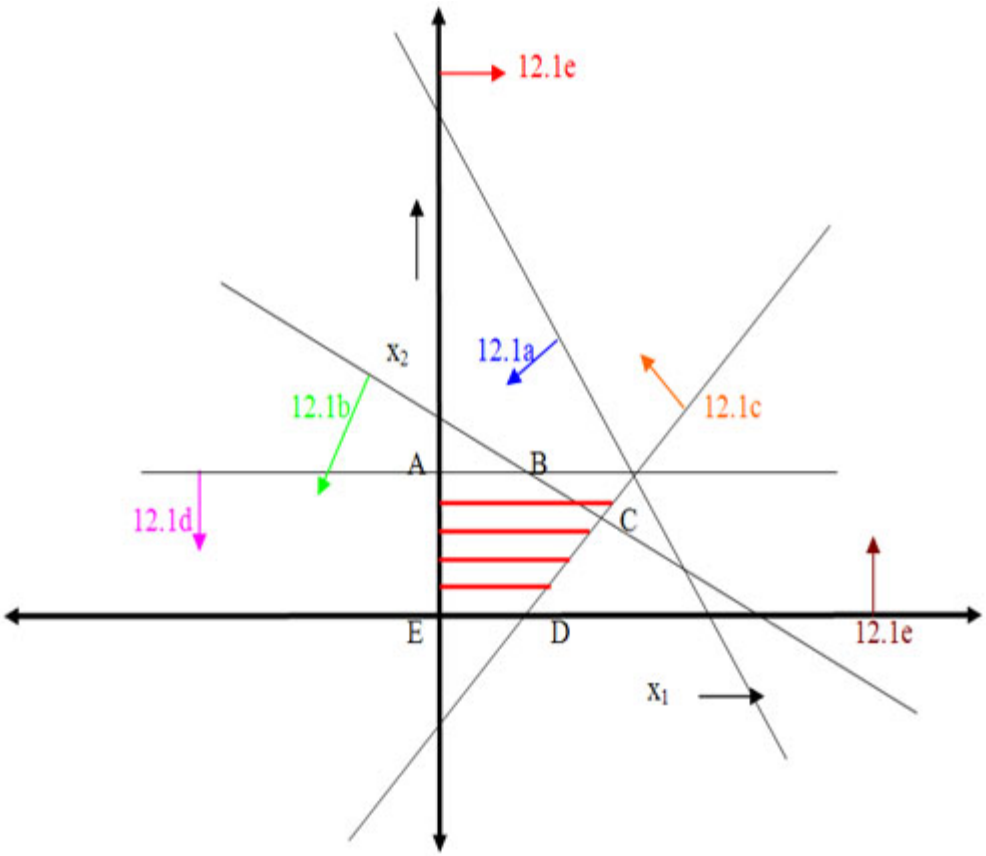


Figure 12.1: A simple example of linear program in two dimension

Previous Next

Application of Markov chain in trying to judge the efficiency of algorithms in OR

In certain OR problems the idea is to search and determine the best amongst a set of N number of search points so that at each point we find the objective function and try to ascertain whether we are able to reach our optimum (whether minimum or maximum that is a different question). Now consider that in a linear programming problem (LPP) if we have N number of corner points, then we have to search $(N-1)$ number of search points.

Let us illustrate that with a simple example.

Example 12.1

A paint company manufactures two different types of paints, P_1 and P_2, from the raw materials, M_1 and M_2. The following table provides the basic data for our example

	Tons of raw materials per ton of		Maximum daily availability (tons)
	P_1	P_2	
Raw material M_1	6	4	24
Raw material M_2	1	2	6
Profit per ton (Rs. 1000)	5	4	

A marketing survey indicates that the daily demand for P_2 cannot exceed that of P_1 by more than 1 ton. Also the maximum daily demand of P_2 is 2 tons. The company would like to determine the optimal mix of the two paints in order to maximize its daily profit.

Step1: To determine the amount to be produced of P_1 and P_2 we denote x_1 as the amount of P_1 paint and x_2 as the amount of P_2 paint.

Step 2: To construct the objective function the company wants to increase its profit as much as possible. If we denote the profit function as z, then we need to maximize $z = 5x_1 + 4x_2$.

Step 3: The constraints, that restricts the raw materials and demand, is related by the fundamental principle that $\{\text{usage of raw materials for both the paints}\} \leq \{\text{maximum raw materials available}\}$.

Previous Next

If $g_j^{\alpha} \geq 0$ indicates $P\left[g_j(\mathrm{d},\mathrm{p},\mathrm{x}) \geq 0\right] = \alpha_j$, then it means that the constraint is feasible. With this inverse reliability transformation, the original constraints that require reliability assessments are converted to equivalent constraints that evaluate the aj-percentile performance. Hence instead of checking the actual reliability, the location of g_j^{α} will now determine the feasibility of a constraint. Applying the above concept to a portfolio optimization (considering minimization) problem gives us the following two equations, namely (12.4) and (12.5).

$$\begin{aligned} &\underset{\forall \Omega}{Minimize} && Risk(\Omega, V, M) \\ &s.t.: && \mathrm{Re}\,turn(\Omega, V, M) \geq R^{*} \\ & && \sum_{i=1}^{N} w_i = 1 \\ & && \Omega = (w_1, \ldots, w_N) \end{aligned} \qquad (12.4)$$

$$\begin{aligned} &\underset{\forall \Omega}{Minimize} && Risk(\Omega, V, M) \\ &s.t.: && P\left[\mathrm{Re}\,turn(\Omega, V, M) \geq R^{*}\right] \geq \alpha \\ & && \sum_{i=1}^{N} w_i = 1 \\ & && \Omega = (w_1, \ldots, w_N) \end{aligned} \qquad (12.5)$$

Here the deterministic portfolio optimization problem, (12.4), and the reliable formulation, (12.5), correspond to the general equations (12.2) and (12.3) respectively. *N* is the universe of assets from which the portfolio is to be formed, **Ω**, the vector comprising the weights $(w_1, \ldots, w_N)$ for each of the corresponding asset in the optimal portfolio, M, the vector consisting of the expected returns for the *N* assets and finally, V, the variance-covariance matrix of the returns of the *N* assets. In (12.5) the constraint on the returns means that the probability of returns being greater than a certain desired value *R** satisfies a given confidence level α.

Previous Next

Reliability Analysis

To illustrate (12.3) further, let us consider Figure 12.6 (a), in which $P\left[g_j(\mathrm{x},\mathrm{d},\mathrm{p})\right]$ is plotted against $g_j(\mathrm{x},\mathrm{d},\mathrm{p})$ and the shaded area underneath the probability density function $P\left[g_j(\mathrm{x},\mathrm{d},\mathrm{p})\right]$ depicts the case when this area is greater than or equal to a_j, i.e., $P\left[g_j(\mathrm{x},\mathrm{d},\mathrm{p})\geq 0\right]\geq \alpha_j$, holds true. So logically it implies that given $F_{x,d,p}\left\{g_j(\mathrm{x},\mathrm{d},\mathrm{p})\right\}$ as the joint distribution function, the shaded area depicts the reliability corresponding to the *jth* constraint, $\forall\, j=1,\ldots,J$. To understand this further we discuss the concept of reliability analysis which offers tools for making reliable decisions with the consideration of uncertainty in design variables and/or parameters. Thus in reliability analysis, given a pre-specified performance level, one is interested to find the probability/reliability greater or less than that pre-specified performance. So in order to use (12.3) we need to evaluate the reliabilities of $P\left[g_j(\mathrm{x},\mathrm{d},\mathrm{p})\right]$, $\forall\, j=1,\ldots,J$ and in presence of these multiple constraints, some of them may never be active and consequently their reliabilities are extremely high (i.e., almost 1.0). Although these constraints are the least critical, the evaluations of these reliabilities will unfortunately dominate the computational effort in probabilistic optimization. The solution to this, is to perform the reliability assessment only up to the necessary level, hence, a formulation of percentile performance (inverse reliability) is used and formulation of the same is as follows, i.e., $g_j^{\alpha}\geq 0$, where, g_j^{α} is the aj-percentile performance of $g_j(\mathrm{x},\mathrm{d},\mathrm{p})$, namely $P\left[g_j(\mathrm{x},\mathrm{d},\mathrm{p})\geq g_j^{\alpha}\right]=\alpha_j$, where it indicates $P\left[g_j(\mathrm{x},\mathrm{d},\mathrm{p})\geq g_j^{\alpha}\right]\geq\alpha_j$ is exactly equal to the desired reliability a_j. This is illustrated in the Figure 12.6 (b), where the shaded area is again equal to the desired reliability a_j, and g_j^{α} point is called the aj-percentile of function g_j.

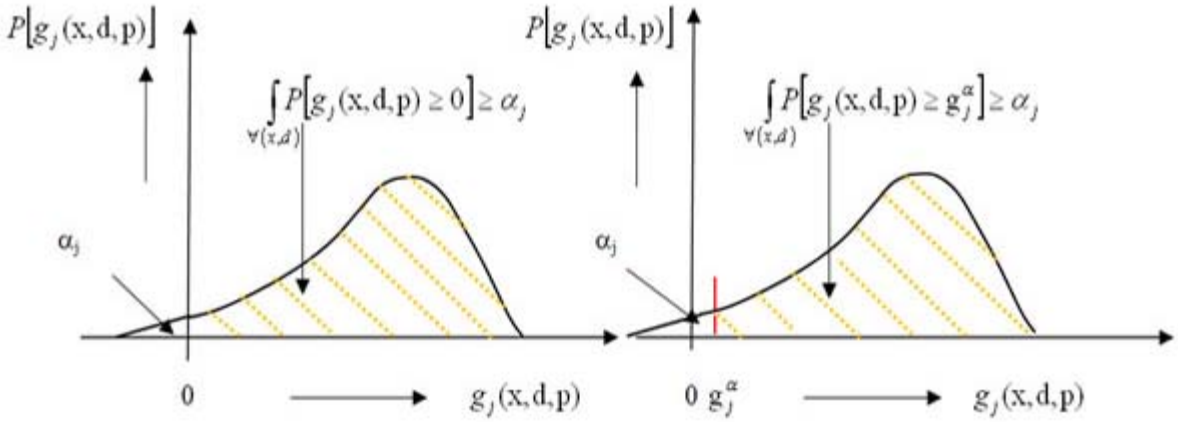


Figure 12.6 (a) & (b): General representation of reliability constraint & aj-percentile reliability constraint

Module 12:Application of stochastic processes in areas of engineering and management science

Lecture 39:Application of Markov chain in trying to judge the efficiency of algorithms in OR

The Lecture Contains :

Previous Next