**Introduction**

In the previous lectures we discussed about problems with a single state variable or input variable $S_t$ which takes only some range of values. In this lecture, we will be discussing about problems with state variable taking continuous values and also problems with multiple state variables.

**Discrete versus Continuous Dynamic Programming**

In a dynamic programming problem, when the number of stages tends to infinity then it is called a continuous dynamic programming problem. It is also called an infinite-stage problem. Continuous dynamic programming model is used to solve continuous decision problems. The classical method of solving continuous decision problems is by the calculus of variations. However, the analytical solutions, using calculus of variations, cannot be generally obtained, except for very simple problems. The infinite-stage dynamic programming approach, on the other hand provides a very efficient numerical approximation procedure for solving continuous decision problems.

The objective function of a conventional discrete dynamic programming model is the sum of individual stage outputs. If the number of stages tends to infinity, then summation of the outputs from individual stages can be replaced by integrals. Such models are useful when infinite number of decisions have to be made in finite time interval.

**Multiple State Variables**

In the problems previously discussed, there was only one state variable $S_t$. However there will be problems in which one need to handle more than one state variable. For example, consider a water allocation problem to $n$ irrigated crops. Let $S_t$ be the units of water available to the remaining $n-i$ crops. If we are concerned only about the allocation of water, then this problem can be solved as a single state problem, with $S_t$ as the state variable. Now, assume that $L$ units of land are available for all these $n$ crops. We want to allocate the land also to each crop after
considering the units of water required for each unit of irrigated land containing each crop. Let \( R_i \) be the amount of land available for \( n-i \) crops. Here, an additional state variable \( R_i \) is to be included while suboptimizing different stages. Thus, in this problem two allocations need to be made: water and land.

The figure below shows a single stage problem consisting of two state variables, \( S_t \) & \( R_t \).

![Stage diagram](image)

In general, for a multistage decision problem of \( T \) stages containing two state variables \( S_t \) and \( R_t \), the objective function can be written as

\[
f = \sum_{t=1}^{T} NB_t = \sum_{t=1}^{T} h(X_t, S_t, R_t)
\]

where the transformation equations are given as

\[
S_{t+1} = g(X_t, S_t) \quad \text{for } t = 1, 2, \ldots, T
\]

\& \quad R_{t+1} = g'(X_t, R_t) \quad \text{for } t = 1, 2, \ldots, T

**Curse of Dimensionality**

Dynamic programming has a serious limitation due to dimensionality restriction. As the number of variables and stages increase, the number of calculations needed increases rapidly thereby increasing the computational effort. If the number of stage variables is increased, then more combinations of discrete states should be examined at each stage. For a problem
consisting of 100 state variables and each variable having 100 discrete values, the suboptimization table will contain $100^{100}$ entries. The computation of this one table may take $100^{96}$ seconds (about $100^{92}$ years) even on a high speed computer. Like this 100 tables have to be prepared, which explains the difficulty in analyzing such a big problem using dynamic programming. This phenomenon is known as “curse of dimensionality” or “Problem of dimensionality” of multiple state variable dynamic programming problems as termed by Bellman.

**References/ Further Reading:**