



ENVIRONMENTAL SYSTEMS ANALYSIS

CENG 6652

Chapter 2

Optimization models and methods

Contents

1. Overview of optimization models
2. Dynamic programming
3. Linear programming

2.1 Overview of optimization models

- Optimization: Science of choosing the best amongst a number of possible alternatives. It Identifies the best through evaluation from a number of possible solutions.
- Driving force in the optimization is the objective function/s
- Optimal solution is the one which gives the best (either maximum or minimum) solution under all assumptions and constraints
- An optimization model can be stated as:

Objective function: “Maximize (or Minimize) $f(X)$ ”, Subject to

the **constraints**,

$$g_j(X) \geq 0, \quad j = 1, 2, \dots, m$$
$$h_j(X) = 0, \quad j = m+1, m+2, \dots, p$$

- Where X is the vector of **decision variables**; $g(X)$ are the **inequality constraints**; $h(X)$ are the **equality constraints**.

Classification of optimization problems

Optimization problems can be classified based on the

- Type of constraints
- Nature of design variables
- Physical structure of the problem
- Nature of the equations involved
- Permissible value of the design variables
- Deterministic/Stochastic nature of the variables
- Separability of the functions and number of objective functions

Classification based on the nature of the equations

- **Linear programming:** Objective function and all the constraints are 'linear' functions of the design variables
- **Nonlinear programming :** Any of the functions among the objectives and constraint functions is nonlinear
 - **Geometric programming :** Objective function and constraints are expressed as polynomials
 - **Quadratic programming:** Best behaved nonlinear programming problem with a quadratic objective function and linear constraints and is concave (for maximization problems)

Separability of the functions and no. of objective functions

- Objective functions can be classified as **single-objective** and **multi-objective** programming problems.
 - i. **Single-objective programming:** There is only one objective function.
 - ii. **Multi-objective programming:** A multi-objective programming problem can be stated as follows:
 - Find \mathbf{X} which maximizes/minimizes $f_1(X), f_2(X), \dots, f_k(X)$
Subject to: $g_j(\mathbf{X}) \leq 0, \quad j = 1, 2, \dots, m$
 - where f_1, f_2, \dots, f_k denote the objective functions to be maximized/ minimized simultaneously

Example to think

- Consider a reservoir from which diversions are made to three irrigation farms at downstream. The water allocations to these farms is x_j ; where $j = 1, 2, \text{ and } 3$. The problem is to determine x_j of water to each farms that maximize the total net benefits, $\sum_j NB_j(x_j)$, obtained.
- The total amount of water available is constrained to a quantity of 10 million m^3 (MCM), of which at least 2 MCM shall be released to downstream.
- The net benefits, $NB_j(x_j)$, derived from water x_j allocated to each farm j , are for example can be defined by:
 - $NB_1(x_1) = 6x_1 - x_1^2$,
 - $NB_2(x_2) = 7x_2 - 1.5x_2^2$ and
 - $NB_3(x_3) = 8x_3 - 0.5x_3^2$

Example to think

- The function $p_j(x_j)$ represent the maximum amount of product that can be produced by farm j from an allocation of water x_j .
 - $p_1(x_1) = 0.4(x_1)^{0.9}$
 - $p_2(x_2) = 0.5(x_2)^{0.8}$
 - $p_3(x_3) = 0.6(x_3)^{0.7}$
- The associated cost of production is expressed by:
 - $c_1(x_1) = 3(p_1(x_1))^{1.3}$
 - $c_2(x_2) = 5(p_2(x_2))^{1.2}$
 - $c_3(x_3) = 6(p_3(x_3))^{1.15}$
- The relationship between the unit price and the amount that will be demanded and sold.
 - $up_1 = 12 - p_1$
 - $up_2 = 20 - 1.5p_2$
 - $up_3 = 28 - 2.5p_3$
- Where the p_j 's are the amounts of each product produced.

Example to think

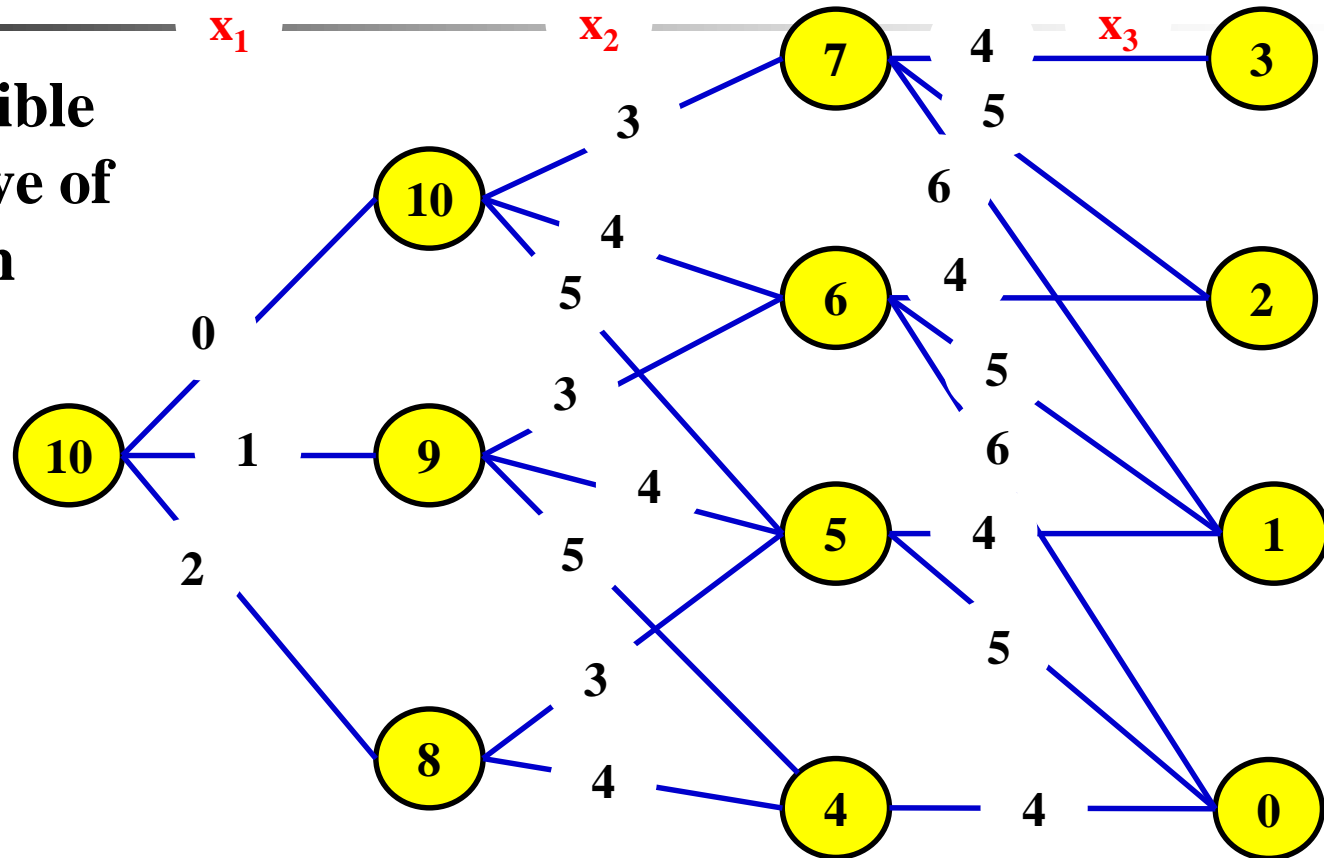
- The optimization problem is to find the water allocations, the production levels, and the unit prices that together maximize the total net benefit obtained from all three farms. The water allocations plus the amount that must remain in the river, R , cannot exceed the total amount of water Q available.
- The model: Maximize Net benefit Subject to constraints:
 - Net benefit = Total return - Total cost
 - Total return = $p_1(12 - p_1) + p_2(20 - 1.5p_2) + p_3(28 - 2.5p_3)$
 - Total cost = $3(p_1)^{1.3} + 5(p_2)^{1.2} + 6(p_3)^{1.15}$
 - Where $p_1 \leq 0.4(x_1)^{0.9}$, $p_2 \leq 0.5(x_2)^{0.8}$ and $p_3 \leq 0.6(x_3)^{0.7}$
 - Water-allocation restriction: $R + x_1 + x_2 + x_3 = Q$

2.2 Dynamic programming

- Dynamic programming (DP) is an approach that divides the original optimization problem, with all of its variables, into a set of smaller optimization problems, each of which are solved before the overall optimum solution.
- The water allocation problem, for example, needs to be solved for a range of water available to each irrigation farm. After which the particular allocations that maximize the total net benefit can be determined.
- A network of *nodes and links* can represent each discrete dynamic programming problem.
- The nodes represent possible discrete states of the system that can exist
- The links represent the decisions one could make to get from one state (node) to another.

The network

One possible alternative of allocation



A network representing some of the possible allocations of water to three irrigation farms j assuming 10 units of water are available. The circles or nodes represent the discrete quantities of water available to users not yet allocated, and the links represent feasible allocation decisions x_j to the next farm j

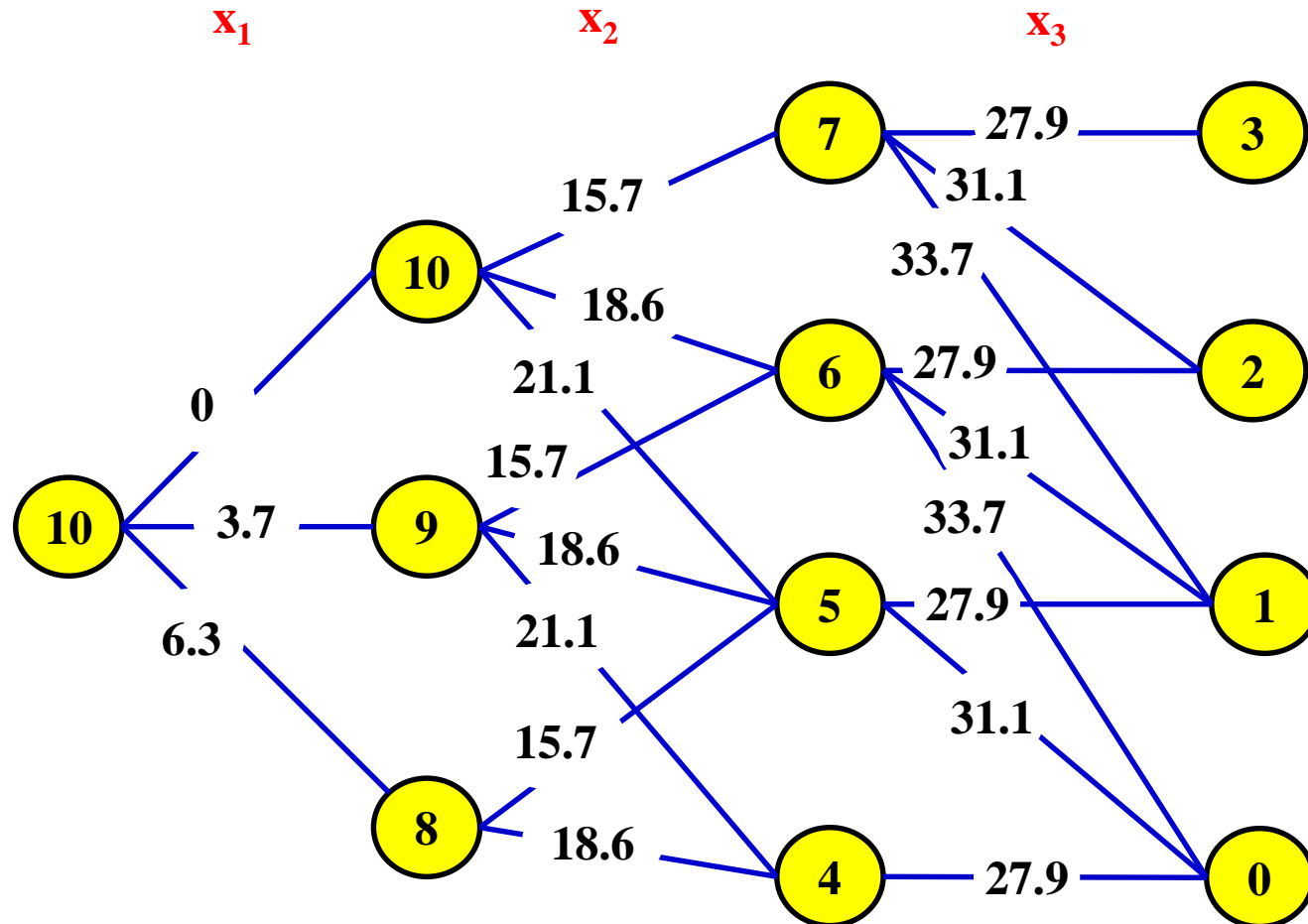
Note on The network

- Each link connects two nodes, the left node value indicating the state of a system before a decision is made, and the right node value indicating the state of a system after a decision is made. In this case, the state of the system is the amount of water available to allocate to the remaining farms.
- Note that the link allocations, the numbers on the links, cannot exceed the amount of water available, that is, the number in the left node of the link.
- The value in the right node, state S_{j+1} , at the beginning of stage $j + 1$, is equal to the value in the left node, S_j , less the amount of water, x_j , allocated to farm j .
- Hence,

$$\blacksquare S_1 - x_1 = S_2, \quad S_2 - x_2 = S_3 \quad \text{and} \quad S_3 - x_3 = S_4$$

The network

- This figure shows the same network as in above slide; however the numbers on the links represent the net benefits obtained from the associated water allocations (see next slide).



The network

- The net benefits, $NB_j(x_j)$, associated with allocations x_j are
 - $NB_1(x_1) = p_1(12 - p_1) - 3(p_1)^{1.3}$, where $p_1 \leq 0.4(x_1)^{0.9}$
 - $NB_2(x_2) = p_2(20 - 1.5p_2) - 5(p_2)^{1.2}$, where $p_2 \leq 0.5(x_2)^{0.8}$
 - $NB_3(x_3) = p_3(28 - 2.5p_3) - 6(p_3)^{1.15}$, where $p_3 \leq 0.6(x_3)^{0.7}$
- The discrete dynamic programming algorithm or procedure is a systematic way to find the best path through this network, or any other suitable network.
- What makes a network suitable for dynamic programming is the fact that all the nodes can be lined up in a sequence of vertical columns and each link connects a node in one column to another node in the next column of nodes. No link passes over or through any other column(s) of nodes. Links also do not connect nodes in the same column.

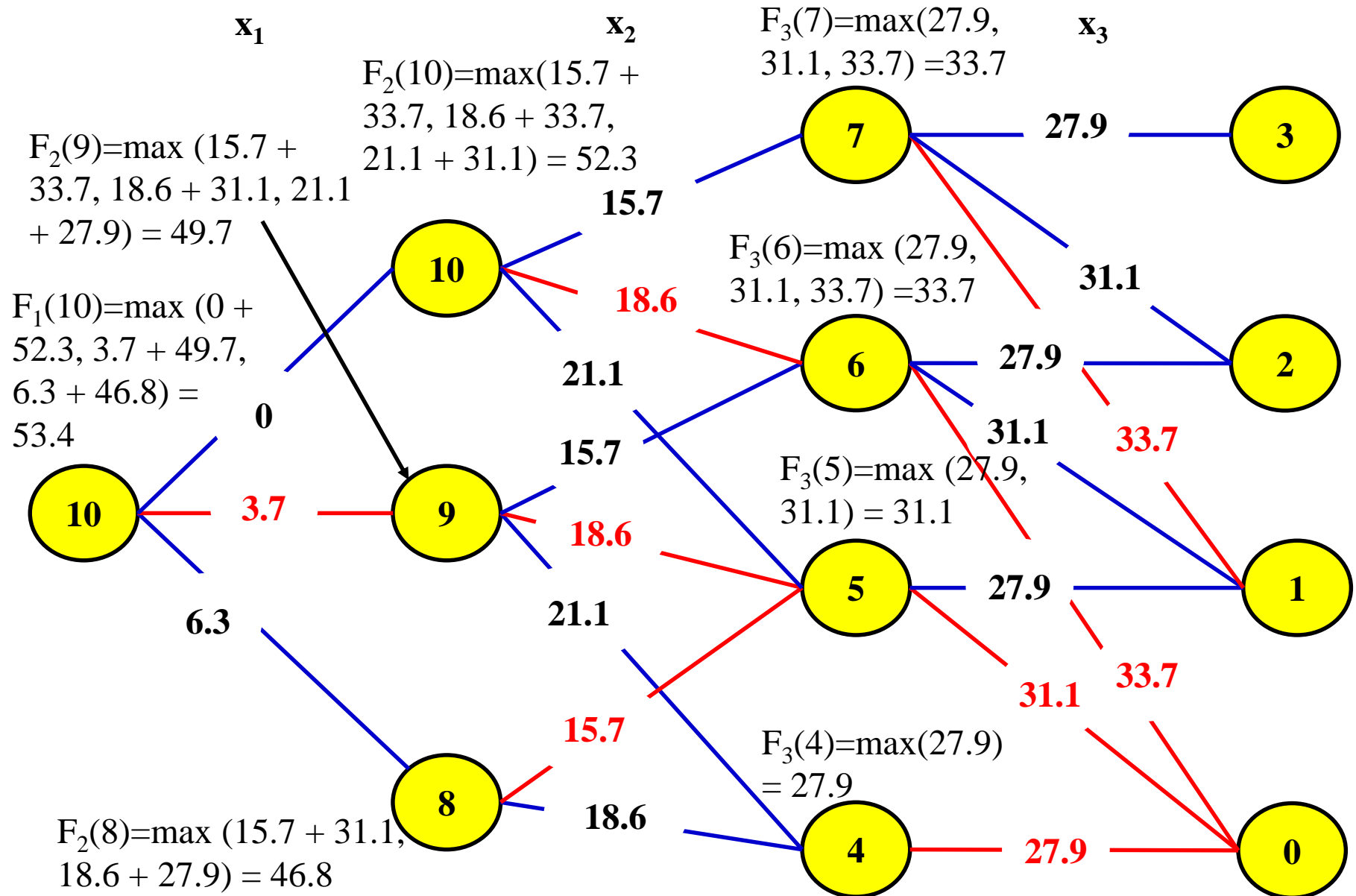
The network

- NOTE:
 - The main challenge in using discrete dynamic programming to solve an optimization problem is to structure the problem so that it fits this dynamic programming network format.
 - Perhaps surprisingly, many water environment planning and management problems do.
 - But it takes practice to become good at converting optimization problems to networks of states, stages, and decisions suitable for solution by discrete dynamic programming algorithms.

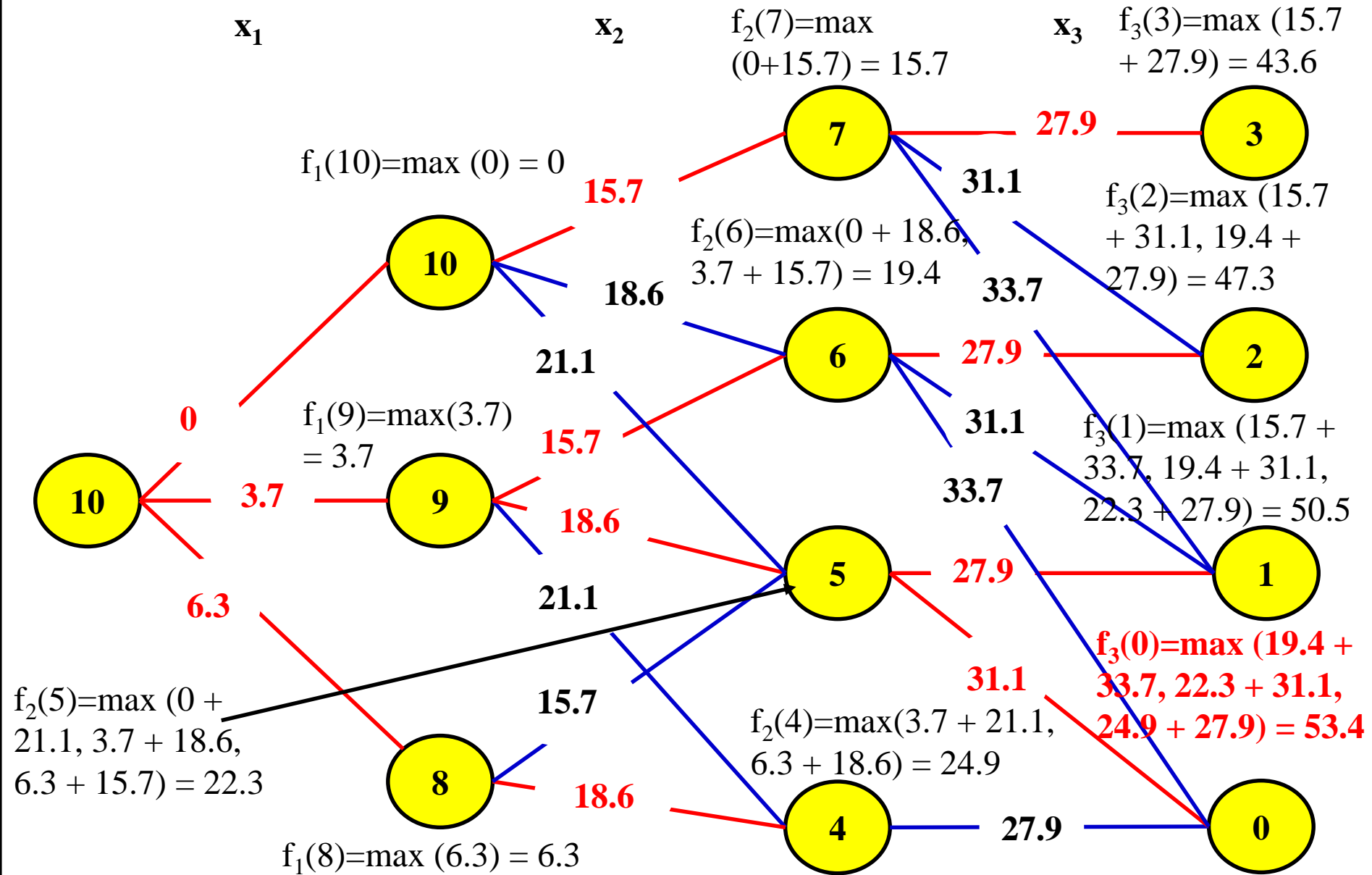
Solution approaches to DP

- DP can be solved in two ways—
 - Backward-moving (but forward-looking) algorithm:
Beginning at the most right column of nodes or states and moving from right to left,,
 - Forward-moving (but backward-looking) algorithm:
beginning at the leftmost node and moving from left to right.
- Both methods will find the best path through the network.
- In some problems, however, only the backward-moving algorithm produces a useful solution.

Backward-Moving Solution Procedure



Forward-Moving Solution Procedure



Note: on the two methods

- The backward-moving solution algorithm is based on the principal that no matter what the state and stage (i.e., the particular node you are at), an optimal policy is one that proceeds forward from that node or state and stage optimally.
- The forward-moving solution algorithm is based on the principal that no matter what the state and stage (i.e., the particular node you are at), an optimal policy is one that has arrived at that node or state and stage in an optimal manner.

DP Limitations

- The application of discrete DP to most practical problems will usually require writing some program code.
- There are no general DP computer programs available that will solve all DP problems.
- Thus, any user of DP will need to write a computer program/code to solve a particular problem unless they do it by hand/excel spreadsheet.
- DP has limitation in handling multiple state variables. In our water-allocation example, we had only one state variable: the total amount of water available.
- What if this problem include other types of resources the farms require (Labor, Energy, etc.) to make their products? Each of these state variables would need to be discretized.

DP Limitations

- If, for example, only m discrete values of each state variable are considered, for n different state variables there are m^n different combinations of state variable values to consider at each stage.
- As the number of state variables increases, the number of discrete combinations of state variable values increases exponentially. This is called dynamic programming's “curse of dimensionality”.

Example 2: Capacity extension

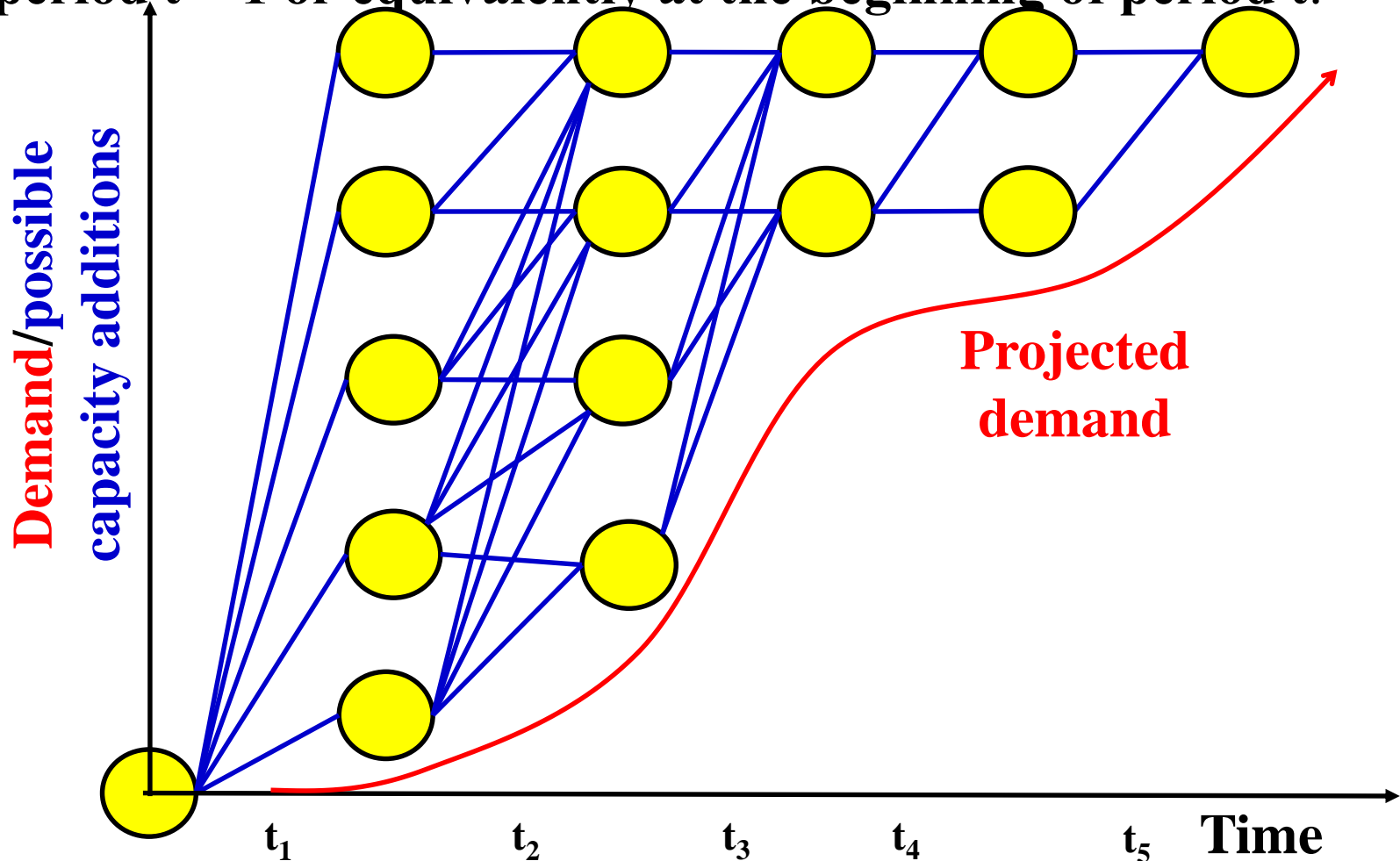
- Consider a municipality that must plan for the future expansion of its water supply system or some component of that system, such as a reservoir or treatment plant.
- The capacity needed at the end of each future period t has been estimated to be D_t .
- The cost, $C_t(s_t, x_t)$ of adding capacity x_t in each period t is a function of that added capacity as well as of the existing capacity s_t at the beginning of the period.
- The planning problem is to find that time sequence of capacity expansions that minimizes the present value of total future costs while meeting the predicted capacity demand requirements. This is the usual capacity expansion problem.

Example 2 ...

- This problem can be written as an optimization model: The objective is to minimize the present value of the total cost of capacity expansion.
 - *Minimize* $\sum_t C_t(s_t, x_t)$
 - where $C_t(s_t, x_t)$ is the present value of the cost of capacity expansion x_t in period t given an initial capacity of s_t .
- Constrained by $s_{t+1} = s_t + x_t, \text{ for } t = 1, 2, 3, \dots, T$
- The actual capacity s_{t+1} at the end of each future period t is no less than the capacity required D_t at the end of that period:
 $s_{t+1} \geq D_t, \text{ for } t = 1, 2, 3, \dots, T$
- The possible expansions in each period defined by a set Ω_t of feasible capacity additions in each period t : $x_t \in \Omega_t$

The network for five time

- The circles represent possible discrete states, S_t , of the system, the amounts of additional capacity existing at the end of each period $t - 1$ or equivalently at the beginning of period t .



Forward-moving DP

- To implement this, define $f_t(s_{t+1})$ as the minimum cost of achieving a capacity s_{t+1} , at the end of period t . Since at the beginning of the first period $t = 1$, the accumulated least cost is 0, $f_0(s_1) = 0$.
- Hence, for each final discrete state s_2 in stage $t = 1$ ranging from D_1 to the maximum demand D_T , define
 - $f_1(s_2) = \min\{C_1(s_1, x_1)\}$ in which $x_1 = s_2$ and $s_1 = 0$
- Moving to stage $t = 2$, for the final discrete states s_3 ranging from D_2 to D_T ,
 - $f_2(s_3) = \min\{C_2(s_2, x_2) + f_1(s_2)\}$
 - in which $0 \leq x_2 \leq (s_3 - D_1)$ and $s_2 = s_3 - x_2$

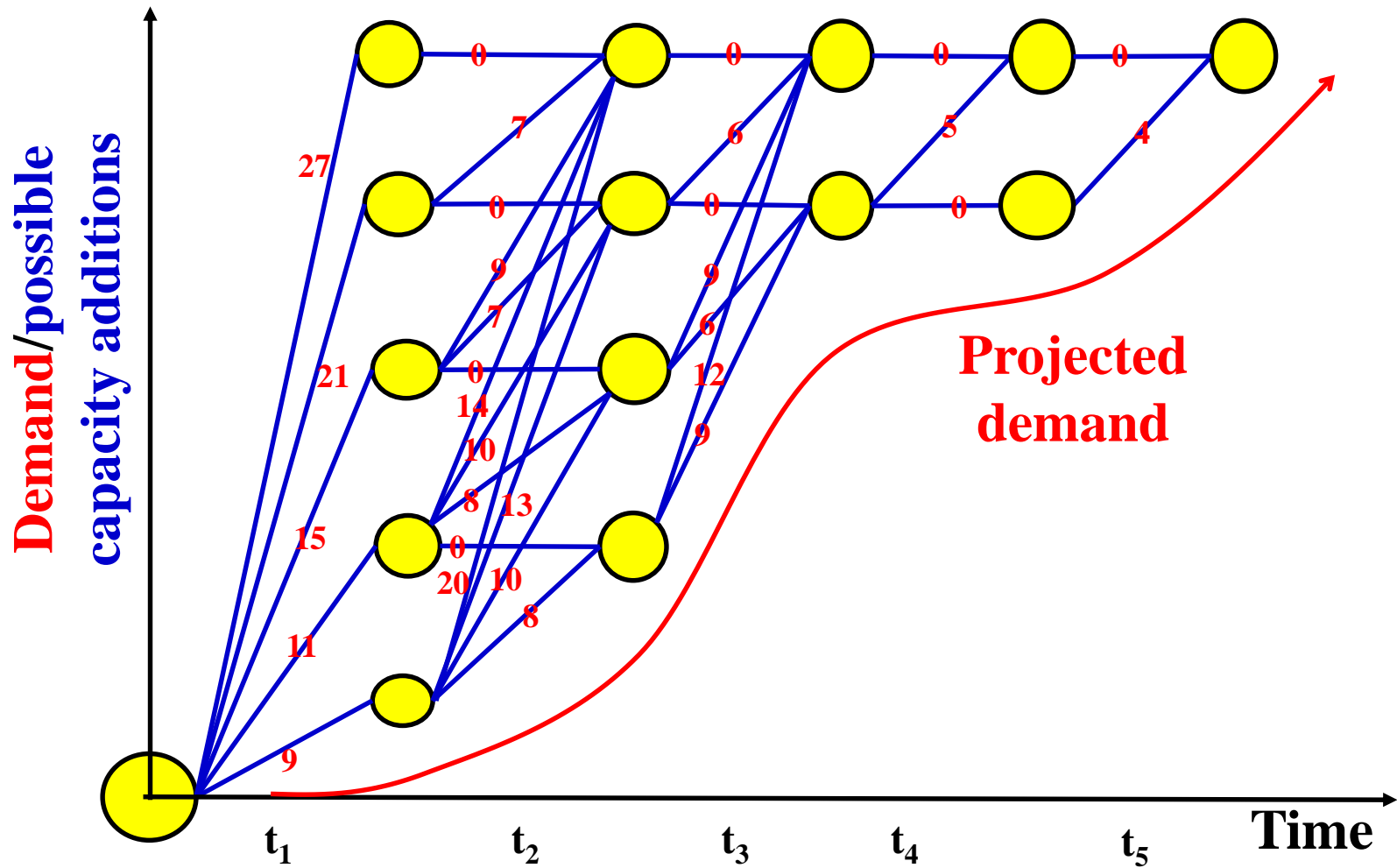
Forward-moving DP

- Moving to stage $t = 3$, for the final discrete states s_4 ranging from D_3 to D_T ,
 - $f_3(s_4) = \min\{C_3(s_3, x_3) + f_2(s_3)\}$
 - *in which $0 \leq x_3 \leq (s_4 - D_2)$ and $s_3 = s_4 - x_3$*
- In general for all stages t between the first and last:
 - $f_t(s_{t+1}) = \min\{C_t(s_t, x_t) + f_{t-1}(s_t)\}$
 - *in which $0 \leq x_t \leq (s_{t+1} - D_{t-1})$ and $s_t = s_{t+1} - x_t$*
- For the last stage $t = T$ and for the final discrete state $s_{T+1} = D_T$
 - $f_T(s_{T+1}) = \min\{C_T(s_T, x_T) + f_{T-1}(s_T)\}$
 - *in which $0 \leq x_T \leq (s_{T+1} - D_{T-1})$ and $s_T = s_{T+1} - x_T$*

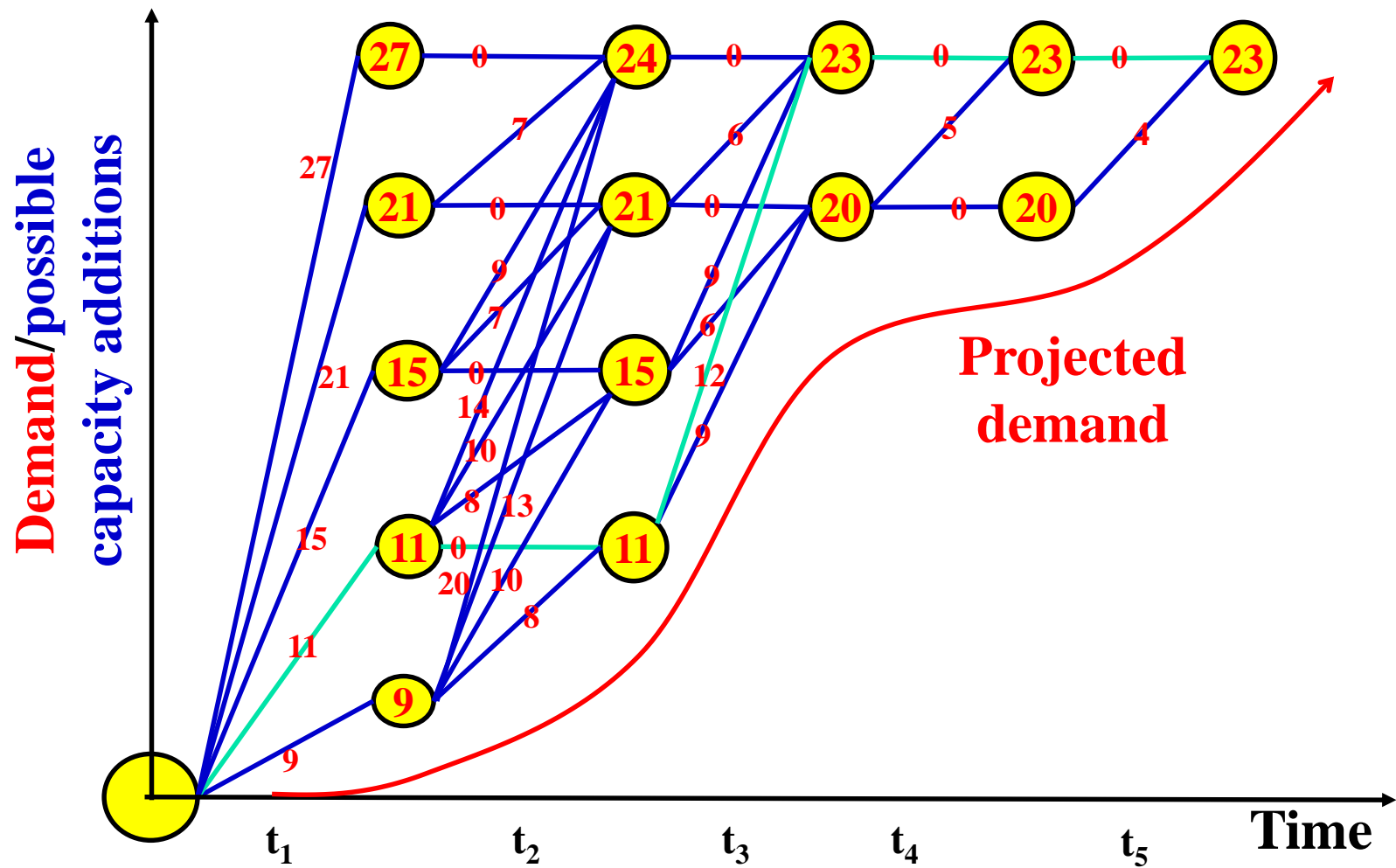
Forward-moving DP

- The value of $f_T(s_{T+1})$ is the minimum present value of the total cost of meeting the demand for T time periods. To identify the sequence of capacity expansion decisions that results in this minimum present value of the total cost requires backtracking to collect the set of best decisions x_t for all stages t.
- Figure in the next slide is the same network with the present value of the expansion costs on each link. The values of the states, the existing capacities, represented by the nodes, are shown on the left vertical axis.
- The capacity expansion problem is solved on Fig. below using the forward-moving algorithm.

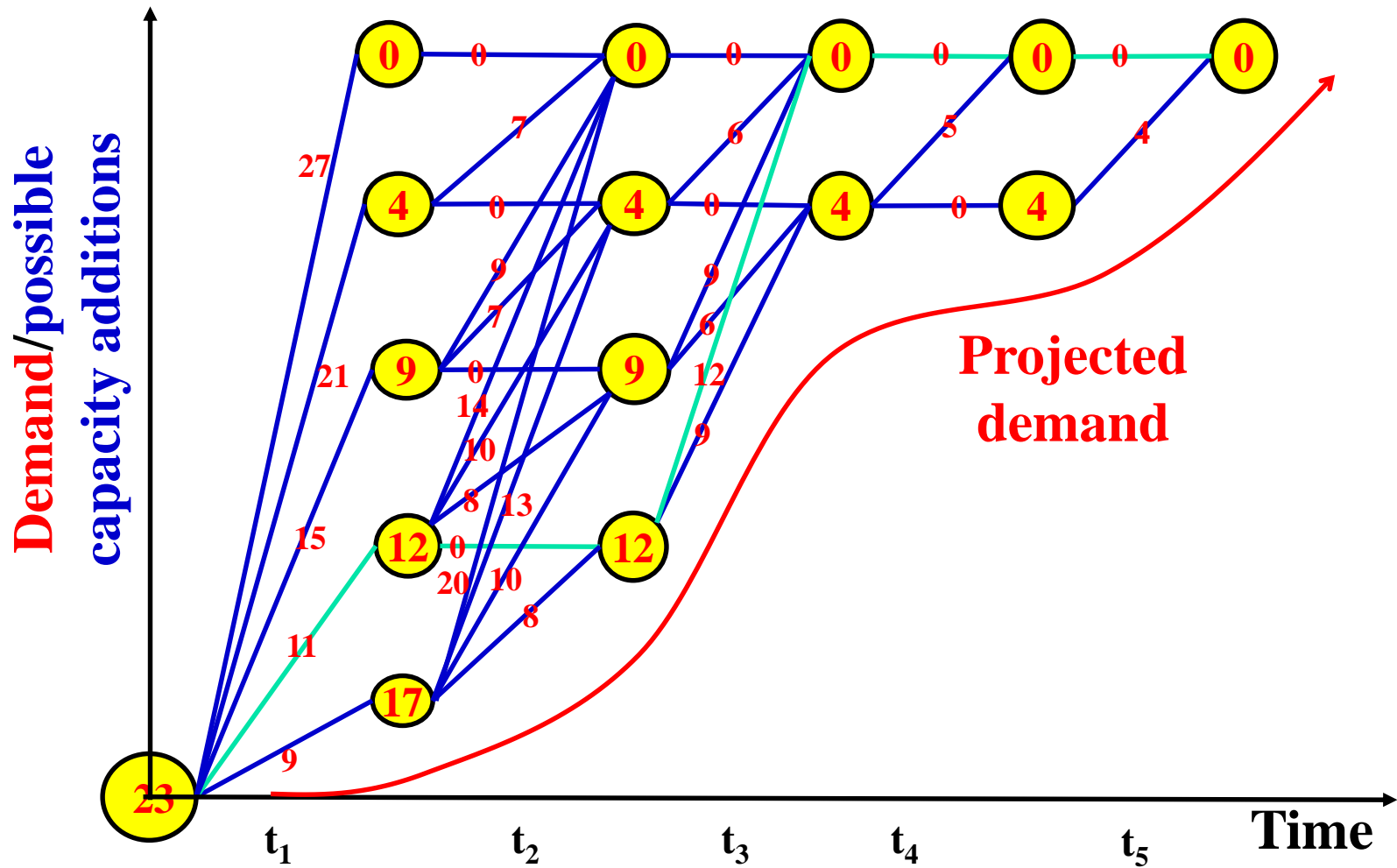
The network



Forward-moving solution



Backward-moving solution



2.3 Linear Programming

- If the objective function $F(\mathbf{X})$ is linear and
- If all the constraints $g_i(\mathbf{X})$ are linear
- Then the model becomes a linear programming model
- The general structure of a linear programming model is:

Maximize (or minimize) $\sum_j^n P_j x_j$

Subject to:

$$\sum_j^n a_{ij} x_j \leq b_i \quad \text{for } i = 1, 2, 3, \dots, m$$

$$x_j \geq 0 \quad \text{for all } j = 1, 2, 3, \dots, n$$

Reservoir Production Problem – A Prototype Example

- Reservoir is used for two purposes: Irrigation and Hydropower.
- Resources are limited to
 - 1000 million cubic meters (MMC) of water.
 - 40 hours of operation time per week (2080 hrs./year).
- Marketing requirement
 - Total irrigation and HP benefit cannot exceed 700 million birr per year.
 - Yearly Benefit from irrigation shall not exceed the HP by more than 30 million birr.
- Technological input
 - Irrigation requires 2 MCM water and 3 hours of labor and Hydropower requires 1 MCM of water and 1 hour of labor per million-birr worth of total benefit.

Reservoir Production Problem – A Prototype Example

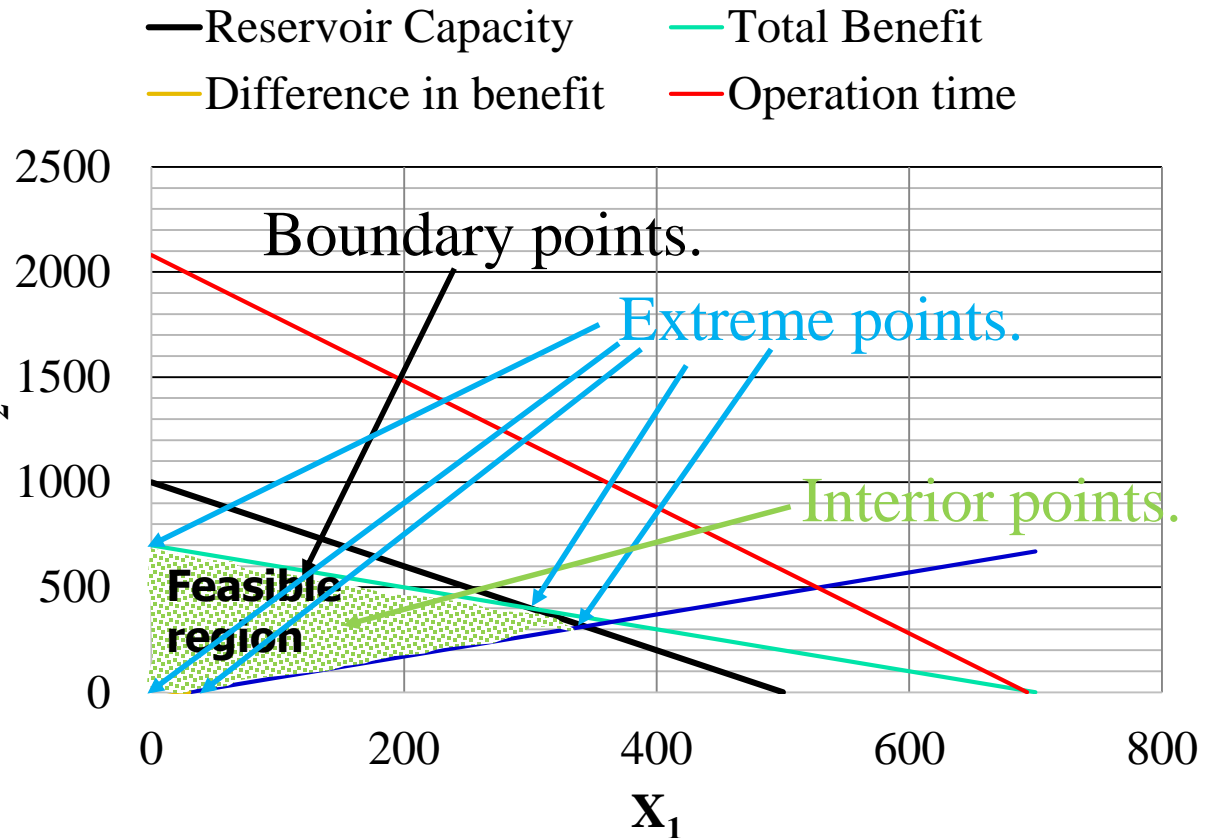
- The current production plan calls for:
 - Producing as much irrigation as possible (80,000 profit per million birr of total benefit).
 - Use the water for HP (50,000 profit per million birr of total benefit).
- The current production plan consists of:
 - Irrigation = 450-million-birr total benefit
 - Hydropower = 100-million-birr total benefit
 - Profit = 41 million birr ($80000 \times 450 + 50000 \times 100$)
- Management is seeking a production schedule that will increase the profit.
- A linear programming model can provide an insight and an intelligent solution to this problem.

Reservoir Production Problem – A Prototype Example

- Decisions variables:
 - X_1 = yearly benefit level of Irrigation (in million birrs)
 - X_2 = yearly benefit level of hydropower (in millions berr).
- Objective Function:
 - Yearly profit, to be maximized
 - $\text{Max } 80000X_1 + 50000X_2$ (Yearly profit)
- Subject to
 - $2X_1 + 1X_2 \leq 1000$ (Reservoir capacity)
 - $3X_1 + 1X_2 \leq 2080$ (Operation Time)
 - $X_1 + X_2 \leq 700$ (Total Benefit)
 - $X_1 - X_2 \leq 30$ (Difference in Benefit)
 - $X_j \geq 0, j = 1,2$ (Nonnegativity)

The Graphical Analysis of Linear Programming

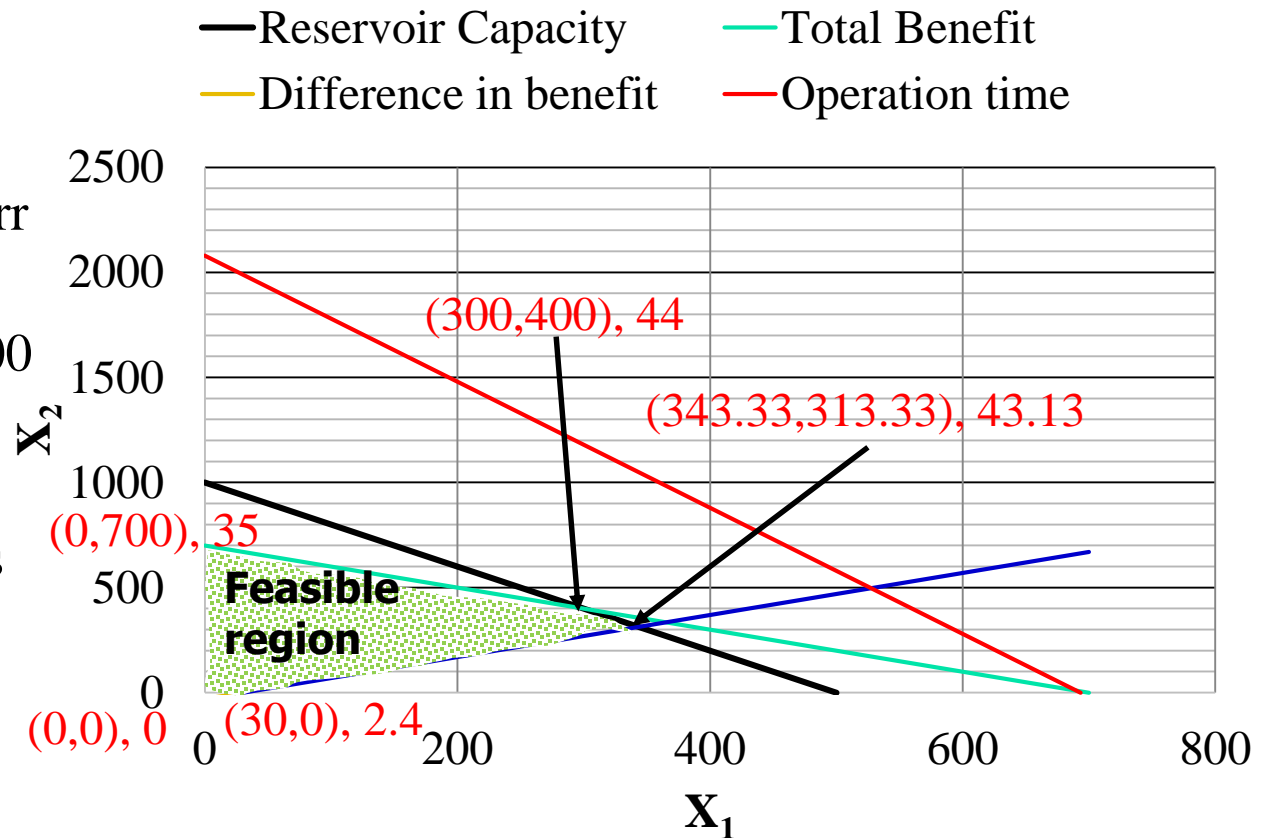
Using a graphical presentation we can represent all the constraints, the objective function, and the three types of feasible points.



- The set of all points that satisfy all the constraints of the model is called a Feasible region

The Graphical Analysis of Linear Programming

- Irrigation = 300 - million-birr benefit
- HP = 400 million Birr benefit
- Profit = $80000 \cdot 300 + 50000 \cdot 400 = 44$ million birr
- This solution utilizes all the water and the allowable maximum difference.



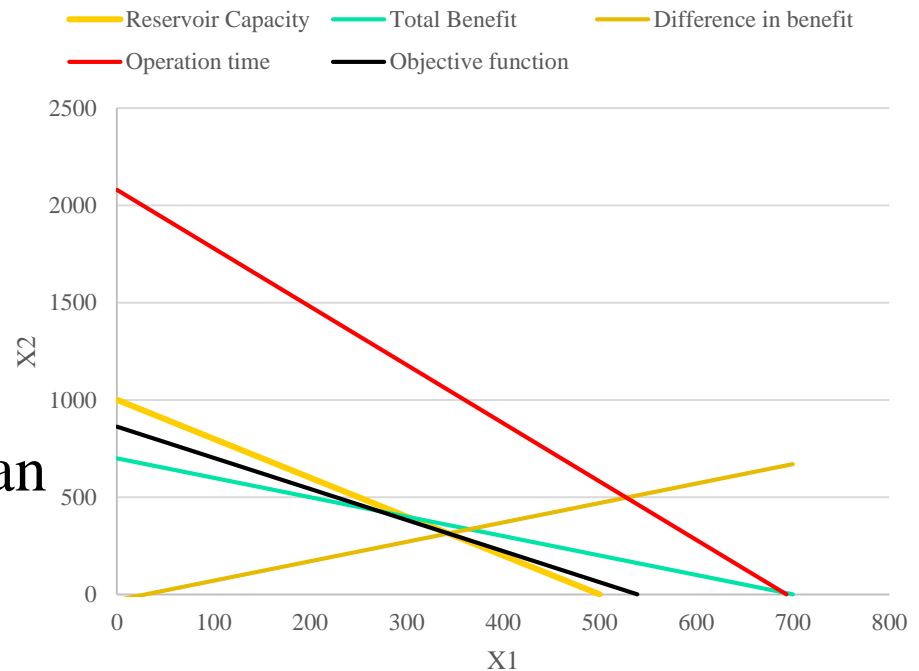
- Total production is only 700.
- The reservoir operation time will only be 1300 hours.

Extreme points and optimal solutions

- If a linear programming problem has an optimal solution, an extreme point is optimal.
- For multiple optimal solutions to exist, the objective function must be parallel to one of the constraints
- Any weighted average of optimal solutions is also an optimal solution.

■ NOTE

- Multiple solutions may exist
- What if there are more than two decision variables



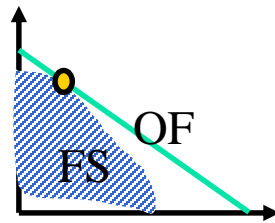
Note on graphical method

- Any point within or on the boundary of the feasible region is a feasible solution
- **An optimal solution** is a feasible solution that has the most favorable value of the objective function. (largest value for maximization and the smallest value for minimization problems).
- Plot the objective function, Z , on the same graph.
- Determine the direction for moving Z within the feasible range
- Shift the objective function line in the direction of improvement until it last intersected the feasible region

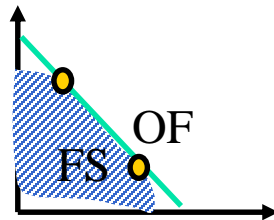
Note on graphical method

- Whenever a linear programming model is formulated and solved, the result will be one of four characteristic solution types:

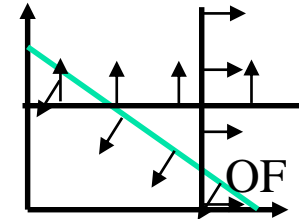
1. Unique optimal solution,



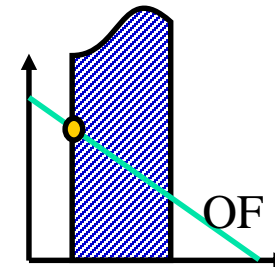
2. Alternate optimal solutions,



3. No-feasible solution,



4. Unbounded solutions.



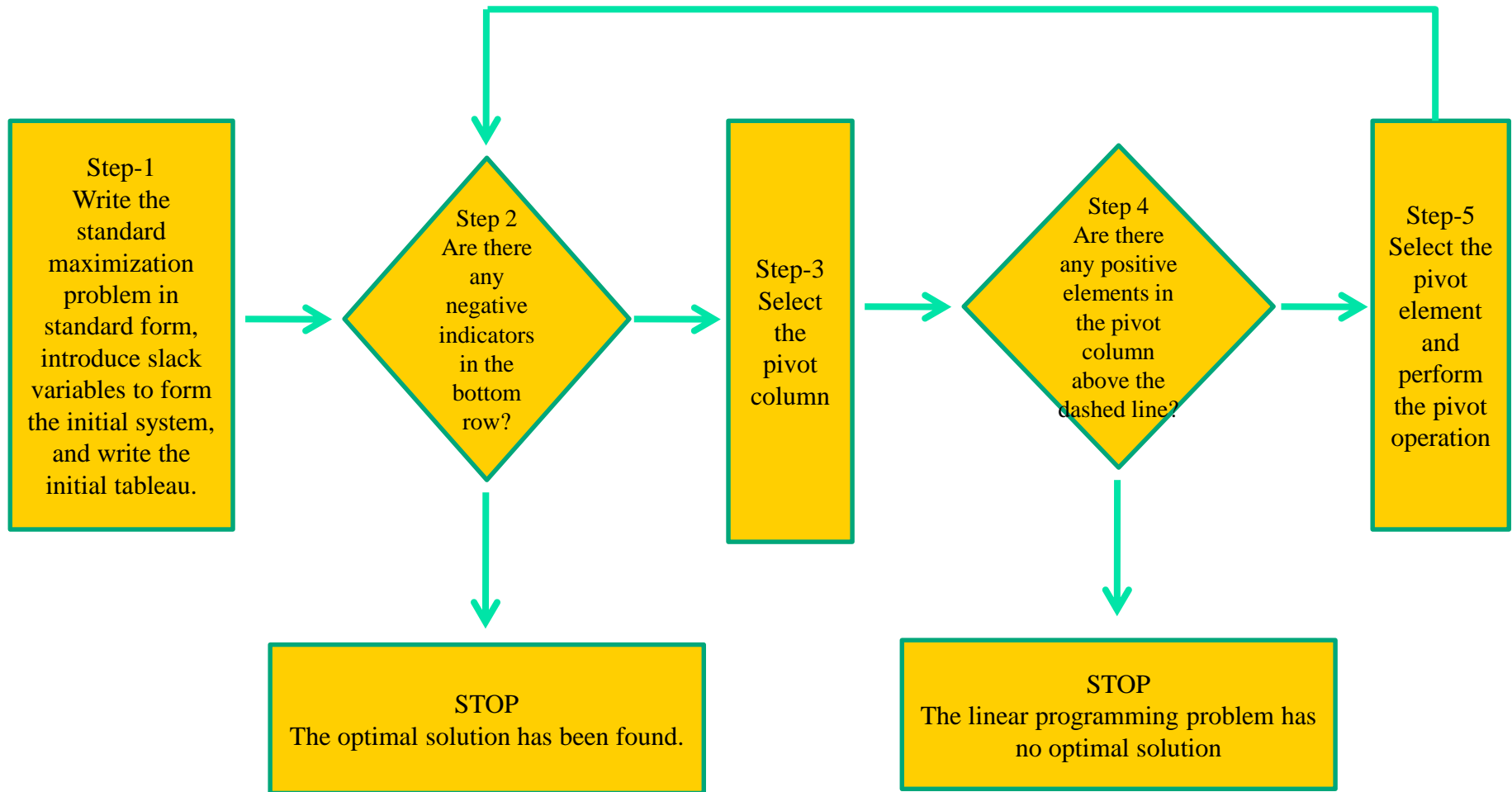
The Simplex Method

- The graphical method is easier to use only for problems involving two decision variables and relatively few problem constraints.
- What happens when we need more decision variables and more problem constraints?
- We use an algebraic method called the simplex method, which was developed by George B. DANTZIG (1914-2005).
- “A mathematical representation of surplus resources.” In real life problems, it’s unlikely that all resources will be used completely, so there usually are unused resources.
- Slack variables represent the unused resources between the left-hand side and right-hand side of each inequality.

Basic and Nonbasic Variables

- Basic variables are selected arbitrarily with the restriction that there be as many basic variables as there are equations. The remaining variables are non-basic variables.
 - $2x_1 + x_2 + s_1 = 1000$
 - $3x_1 + x_2 + s_2 = 2080$
 - $x_1 + x_2 + s_3 = 700$
 - $x_1 - x_2 + s_4 = 30$
- This system has four equations, we can select any four of the six variables as basic variables. The remaining two variables are then non-basic variables.
- A solution found by setting the two non-basic variables equal to 0 and solving for the two basic variables is a basic solution. If a basic solution has no negative values, it is a basic feasible solution.

Simplex method



Simplex algorithm for standard maximization problems

Simplex Method

1. Convert each inequality in the set of constraints to an equation by adding slack variables.
2. Create the initial simplex tableau.
3. Select the pivot column. (The column with the “most negative value” element in the last row.)
4. Select the pivot row. (The row with the smallest non-negative result when the last element in the row is divided by the corresponding in the pivot column.)
5. Use elementary row operations calculate new values for the pivot row so that the pivot is 1 (Divide every number in the row by the pivot number.)
6. Use elementary row operations to make all numbers in the pivot column equal to 0 except for the pivot number. If all entries in the bottom row are zero or positive, this the final tableau. If not, go back to step 3.
7. If you obtain a final tableau, then the linear programming problem has a maximum solution, which is given by the entry in the lower-right corner of the tableau.

Simplex Tableau and Pivot

| | All variables | Solution |
|-----------------|---------------|----------|
| Basic variables | coefficients | |
| | | 0 |

- A simplex tableau is a way to systematically evaluate variable mixes in order to find the best one.
- **Pivot Column:** The column of the tableau representing the variable to be entered into the solution mix.
- **Pivot Row:** The row of the tableau representing the variable to be replaced in the solution mix.
- **Pivot Number:** The element in both the pivot column and the pivot row.

Our previous example

- **Decisions variables:**

- X_1 = yearly benefit level of Irrigation (in million birrs)

- X_2 = yearly benefit level of hydropower (in millions birr).

- **Objective Function:**

- Yearly profit, to be maximized

- Max $80000X_1 + 50000X_2$ (Yearly profit)

- **Subject to**

- $2X_1 + 1X_2 \leq 1000$ (Reservoir capacity)

- $3X_1 + 1X_2 \leq 2080$ (Operation Time)

- $X_1 + X_2 \leq 700$ (Total Benefit)

- $X_1 - X_2 \leq 30$ (Difference in Benefit)

- $X_j \geq 0, j = 1,2$ (Nonnegativity)

-
- The first step of the simplex method requires that each inequality be converted into an equation. “ \leq ” inequalities are converted to equations by including slack variables.
 - $2x_1 + x_2 + s_1 = 2x_1 + x_2 + s_1 + 0s_2 + 0s_3 + 0s_4 = 1000$
 - $3x_1 + x_2 + s_2 = 3x_1 + x_2 + 0s_1 + s_2 + 0s_3 + 0s_4 = 2080$
 - $x_1 + x_2 + s_3 = x_1 + x_2 + 0s_1 + 0s_2 + s_3 + 0s_4 = 700$
 - $x_1 - x_2 + s_4 = x_1 - x_2 + 0s_1 + 0s_2 + 0s_3 + s_4 = 30$
 - The slack variables can be included in the objective function with zero coefficients:
 - $P = 80000x_1 + 50000x_2 + 0s_1 + 0s_2 + 0s_3 + 0s_4$
 - $P - 80000x_1 + 50000x_2 + 0s_1 + 0s_2 + 0s_3 + 0s_4 = 0$

Step 1

- The problem can now be considered as solving a system of 5 linear equations involving the 7 variables x_1 , x_2 , s_1 , s_2 , s_3 , s_4 and P in such a way that P has the maximum value;
 - $2x_1 + x_2 + s_1 + 0s_2 + 0s_3 + 0s_4 = 1000$
 - $3x_1 + x_2 + 0s_1 + s_2 + 0s_3 + 0s_4 = 2080$
 - $x_1 + x_2 + 0s_1 + 0s_2 + s_3 + 0s_4 = 700$
 - $x_1 - x_2 + 0s_1 + 0s_2 + 0s_3 + s_4 = 30$
 - $P - 80000x_1 - 50000x_2 + 0s_1 + 0s_2 + 0s_3 + 0s_4 = 0$
- Now, the system of linear equations can be written in matrix form or as a 5×8 augmented matrix. The initial tableau is;

Step 2

| Basic Variables | x_1 | x_2 | s_1 | s_2 | s_3 | s_4 | P | RHS |
|-----------------|--------|--------|-------|-------|-------|-------|---|------|
| s_1 | 2 | 1 | 1 | 0 | 0 | 0 | 0 | 1000 |
| s_2 | 3 | 1 | 0 | 1 | 0 | 0 | 0 | 2080 |
| s_3 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 700 |
| s_4 | 1 | -1 | 0 | 0 | 0 | 1 | 0 | 30 |
| P | -80000 | -50000 | 0 | 0 | 0 | 0 | 1 | 0 |

- The tableau represents the initial basic solution (by keeping the two non basic variables (x_1 and x_2 equal to zero);
- $x_1=0$, $x_2=0$, $s_1=1000$, $s_2=2080$, $s_3=700$ $s_4 = 30$ and $P = 0$
- The slack variables form the initial solution mix. The initial solution assumes the slack variables take the largest possible values

Step 3

- Select the pivot column (determine which variable to enter into the solution mix). Choose the column with the “most negative” element in the objective function row.

| Basic Variables | x_1 | x_2 | s_1 | s_2 | s_3 | s_4 | P | RHS |
|-----------------|--------|--------|-------|-------|-------|-------|---|------|
| s_1 | 2 | 1 | 1 | 0 | 0 | 0 | 0 | 1000 |
| s_2 | 3 | 1 | 0 | 1 | 0 | 0 | 0 | 2080 |
| s_3 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 700 |
| s_4 | 1 | -1 | 0 | 0 | 0 | 1 | 0 | 30 |
| P | -80000 | -50000 | 0 | 0 | 0 | 0 | 1 | 0 |

- x_1 should enter into the solution mix because each unit of x_1 (a table) contributes a profit of 80,000 compared with only 50,000 for each unit of x_2 (a chair)

Step 4

No, There aren't any positive elements in the pivot column.


We can go on step 5

Step 5

- Select the pivot row (determine which variable to replace in the solution mix). Divide the last element in each row by the corresponding element in the pivot column. The pivot row is the row with the smallest non-negative result.

| Basic Variables | x_1 | x_2 | s_1 | s_2 | s_3 | s_4 | P | RHS |
|-----------------|--------|--------|-------|-------|-------|-------|---|-------------------|
| s_1 | 2 | 1 | 1 | 0 | 0 | 0 | 0 | $1000/2 = 500$ |
| s_2 | 3 | 1 | 0 | 1 | 0 | 0 | 0 | $2080/3 = 693.33$ |
| s_3 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | $700/1 = 700$ |
| s_4 | 1 | -1 | 0 | 0 | 0 | 1 | 0 | $30/1 = 30$ |
| P | -80000 | -50000 | 0 | 0 | 0 | 0 | 1 | $0/-80000 = 0$ |

Pivot number




Step 5

- s_4 Should be replaced by x_1 in the solution mix.
- Now calculate new values for the pivot row. Divide every number in the row by the pivot number.

| Basic Variables | x_1 | x_2 | s_1 | s_2 | s_3 | s_4 | P | RHS |
|-----------------|--------|--------|-------|-------|-------|-------|---|-------------------|
| s_1 | 2 | 1 | 1 | 0 | 0 | 0 | 0 | $1000/2 = 500$ |
| s_2 | 3 | 1 | 0 | 1 | 0 | 0 | 0 | $2080/3 = 693.33$ |
| s_3 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | $700/1 = 700$ |
| x_1 | 1 | -1/1 | 0 | 0 | 0 | 1/1 | 0 | $30/1 = 30$ |
| P | -80000 | -50000 | 0 | 0 | 0 | 0 | 1 | 0 |

Pivot number



Step 5

- Use row operations to make all numbers in the pivot column equal to 0 except for the pivot number which remains as 1.

| Basic Variables | x_1 | x_2 | s_1 | s_2 | s_3 | s_4 | P | RHS |
|-----------------|-------|---------|-------|-------|-------|-------|---|----------------------|
| s_1 | 0 | 3 | 1 | 0 | 0 | -2 | 0 | $1000-2*30=940$ |
| s_2 | 0 | 4 | 0 | 1 | 0 | -3 | 0 | $2080-3*30=1990$ |
| s_3 | 0 | 2 | 0 | 0 | 1 | -1 | 0 | $700-1*30=670$ |
| x_1 | 1 | -1 | 0 | 0 | 0 | 1 | 0 | 30 |
| P | 0 | -130000 | 0 | 0 | 0 | 80000 | 1 | $0+80000*30=2400000$ |

- In this case, $x_1=30$, $x_2=0$, $s_1=940$, $s_2=1990$, $s_3=670$, $s_4 = 0$ and $P = 2,400,000$
- Now repeat the steps until there are no negative numbers in the last row. Select the new pivot column. x_2 should enter into the solution mix.

Step 5

- Select the new pivot column. Select the new pivot row. s_3 should be replaced by x_2 in the solution mix.

| Basic Variables | x_1 | x_2 | s_1 | s_2 | s_3 | s_4 | P | RHS |
|-----------------|-------|---------|-------|-------|-------|-------|---|-------------------------|
| s_1 | 0 | 3 | 1 | 0 | 0 | -2 | 0 | $940/3=313.33$ |
| s_2 | 0 | 4 | 0 | 1 | 0 | -3 | 0 | $1990/4=497.5$ |
| s_3 | 0 | 2 | 0 | 0 | 1 | -1 | 0 | $670/2=335$ |
| x_1 | 1 | -1 | 0 | 0 | 0 | 1 | 0 | $30/-1=-30$ |
| P | 0 | -130000 | 0 | 0 | 0 | 80000 | 1 | $2400000/-130000=-18.5$ |

Step 5

| Basic Variables | x_1 | x_2 | s_1 | s_2 | s_3 | s_4 | P | RHS |
|-----------------|-------|-------|-------|-------|-------|-------|---|-----------------------------------|
| x_2 | 0 | 1 | 1/3 | 0 | 0 | -2/3 | 0 | 313.33 |
| s_2 | 0 | 0 | -4/3 | 1 | 0 | -1/3 | 0 | 1990-4*313.33=736.67 |
| s_3 | 0 | 0 | -2/3 | 0 | 1 | 1/3 | 0 | 670-2*313.33=43.33 |
| x_1 | 1 | 0 | 1/3 | 0 | 0 | 1/3 | 0 | 30+313.33=343.33 |
| P | 0 | 0 | 0 | 0 | 0 | - | 1 | 2400000+130000*313.33=43133333.33 |

- In this case, $x_1=343.3$, $x_2=313.3$, $s_1=0$, $s_2=1990$, $s_3=670$, $s_4 = 0$ and $P = 43,133,333.33$
- Now repeat the steps until there are no negative numbers in the last row. Select the new pivot column. s_4 should enter into the solution mix.

Step 5

| Basic Variables | x_1 | x_2 | s_1 | s_2 | s_3 | s_4 | P | RHS |
|-----------------|-------|-------|-------|-------|-------|---------|---|----------------------------|
| x_2 | 0 | 1 | 1/3 | 0 | 0 | -2/3 | 0 | $313.33/(-2/3) = -470$ |
| s_2 | 0 | 0 | -4/3 | 1 | 0 | -1/3 | 0 | $736.67/(-1/3) = -2210$ |
| s_3 | 0 | 0 | -2/3 | 0 | 1 | 1/3 | 0 | $43.33/(1/3)=130$ |
| x_1 | 1 | 0 | 1/3 | 0 | 0 | 1/3 | 0 | $343.33/(1/3)=1030$ |
| P | 0 | 0 | 0 | 0 | 0 | -6666.7 | 1 | $43132900/(-6666.7)=-6470$ |

- Select the new pivot column. Select the new pivot row. s_3 should be replaced by s_4 in the solution mix.

Step 5

- Select the new pivot column. Select the new pivot row. s_3 should be replaced by s_4 in the solution mix.
- Now calculate new values for the pivot row. Divide every number in the row by the pivot number.

| Basic Variables | x_1 | x_2 | s_1 | s_2 | s_3 | s_4 | P | RHS |
|-----------------|-------|-------|-------|-------|-------|---------|---|-------------------------------|
| x_2 | 0 | 1 | 1/3 | 0 | 0 | -2/3 | 0 | $313.33/(-2/3) = -470$ |
| s_2 | 0 | 0 | -4/3 | 1 | 0 | -1/3 | 0 | $736.67/(-1/3) = -2210$ |
| s_4 | 0 | 0 | -2/3 | 0 | 1 | 1/3 | 0 | $43.33/(1/3)=130$ |
| x_1 | 1 | 0 | 1/3 | 0 | 0 | 1/3 | 0 | $343.33/(1/3)=1030$ |
| P | 0 | 0 | 0 | 0 | 0 | -6666.7 | 1 | $43133333.33/(-6666.7)=-6470$ |

Step 5: Final Result

- Use row operations to make all numbers in the pivot column equal to 0 except for the pivot number which remains as 1.

| | x_1 | x_2 | s_1 | s_2 | s_3 | s_4 | P | RHS |
|-------|--------|-------|-------|-------|-------|-------|---|---|
| x_2 | 0 | 1 | -1 | 0 | 2 | 0 | 0 | $313.33 + 2 * 43.33 = 400$ |
| s_2 | 0 | 0 | -2 | 1 | 1 | 0 | 0 | $736.667 + 1 * 43.33 = 780$ |
| s_4 | 0 | 0 | -2 | 0 | 3 | 1 | 0 | $43.33 / (1/3) = 130$ |
| x_1 | 1 | 0 | 1 | 0 | -1 | 0 | 0 | $343.33 - 1 * 43.33 = 300$ |
| P | 130000 | 0 | 30000 | 0 | 20000 | 0 | 1 | $43133333.33 + 6666.7 * 43.33 * 3 = 44000000$ |

- As the last row contains no negative numbers, this solution gives the maximum value of P. This simplex tableau represents the optimal solution to the LP problem and is interpreted as:
- $x_1 = 300$, $x_2 = 400$, $s_1 = 0$, $s_2 = 780$, $s_3 = 43.33$, $s_4 = 0$ and $P = 44,000,000$

Simplex method: the MATLAB code

- MATLAB works for minimization, thus any maximization problem to solve shall be converted into minimization
- Objective Function: **Min** $-80000X_1 - 50000X_2$,
- Subject to
 - $2X_1 + 1X_2 \leq 1000$
 - $3X_1 + 1X_2 \leq 2080$
 - $X_1 + X_2 \leq 700$
 - $X_1 - X_2 \leq 30$
 - $X_j \geq 0, j = 1,2$

```
clc, clear
%objective function: input coefficients
f=[-80000;-50000]; %note the negative
coefficients for minimization of OF
%Inequality constraints. They should be in the
form [A]{x}={b}
A=[2 1; 3 1; 1 1;1 -1;-1 0; 0 -1];
b=[1000;2080;700;30; 0; 0];
%Lower and upper bounds of variables
lb=zeros(2,1); ub=[1e4;1e4];
%Add empty matrices for coefficients of equality
constraints and initial guess
Aeq=[]; beq=[]; x0=[];
%Specify search options:
%Use the dual-simplex algorithm since simplex will
be removed after this version
%Display the results of all iterations
options=optimoptions('linprog','Algorithm','Simple
x','Display','iter');
[x,fval,exitflag,output] =
linprog(f,A,b,Aeq,beq,lb,ub,x0,options);
```

Example –Water Quality Management

- Waste load allocation for water quality management in a river system can be defined as
 - Determination of optimal treatment level of waste, which is discharged to a river
 - By maintaining the water quality standards set by Pollution Control Agency (PCA), through out the river
- Conventional waste load allocation involves minimization of treatment cost subject to the constraint that the water quality standards are not violated
- Consider a simple problem of M dischargers, who discharge waste into the river, and I checkpoints, where the water quality is measured by PCA

Example –

- Let x_j be the treatment level and a_j be the unit treatment cost for j^{th} discharger ($j = 1, 2, \dots, M$)
- c_i be the dissolved oxygen (DO) concentration at check point i ($i = 1, 2, \dots, I$), which is to be controlled
- Decision variables for the waste load allocation model are x_j ($j = 1, 2, \dots, M$).
- Objective function can be expressed as: *Minimize* $f = \sum_1^M a_j x_j$
- Relationship between the water quality indicator, c_i (DO) and the treatment level upstream to that checkpoint is linear. Let $g(x)$ denotes the linear relationship between c_i and x_j .
- Then, $c_i = g(x_j)$

Example –

- Let c_p be the permissible DO level set by PCA, which is to be maintained through out the river; Therefore, $c_i \geq c_p$
- Model can be solved using simplex algorithm which will give the optimal fractional removal levels required to maintain the water quality of the river
- Let there be two dischargers of wastes and two monitoring points along the river.
- The treatment cost by the first discharger is 2000 birr while the second be 3000 birr per unit (mg/lit) waste discharged.
- The DO level required at the monitoring 1 be 1.2 mg/lit.
- $DO_1 = 0.886x_1$
- The DO level required at the monitoring 2 be 1.6 mg/lit.
- $DO_2 = 0.586x_1 + 0.716x_2$

Example –

- Decide on the waste allocated to be discharged by each discharger
- Decision variables for the waste load allocation model are x_j ($j = 1, 2$).

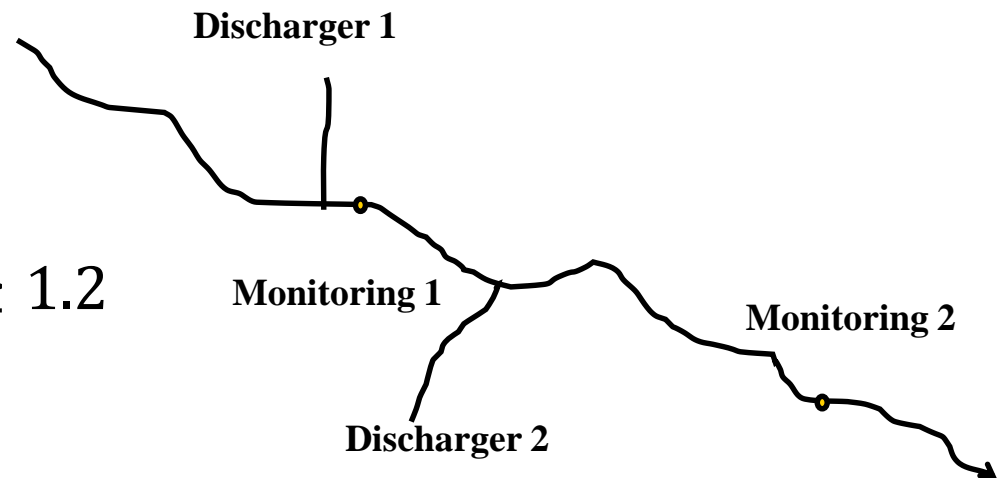
■ Objective function can be expressed as: *Minimize* $f =$

$$a_1x_1 + a_2x_2 = 2000x_1 + 3000x_2$$

■ **Subject to**

■ $0.886x_1 \geq 1.2$

■ $0.586x_1 + 0.716x_2 \geq 1.2$



File Home Insert Page Layout Formulas **Data** Review View Help Power Pivot Tell me what you want to do

| | | | | | | | | | | | | | | | | | | | | |
|-------------------|--------------------|----------------------|-----------------|-----------|----------------|-------------|-------------|------------|---------------|------|--------|------------|---------|----------|------------|-------------------|-----------------|-------------|---------------|-------------------|
| From Access | From Other Sources | Existing Connections | Show Queries | New Query | Recent Sources | Refresh All | Connections | Properties | Edit Links | Sort | Filter | Clear | Reapply | Advanced | Flash Fill | Remove Duplicates | Data Validation | Consolidate | Relationships | Manage Data Model |
| Get External Data | | | Get & Transform | | | Connections | | | Sort & Filter | | | Data Tools | | | | | | | | |

C12

A B C D E F G H I J K L M N O P Q

the following linear programming solves an optimization problem of minimizing the Cost (C) for waste treatment by the two dischargers.

Objective function Minimize (Cost)=2000x1 + 3000x2

Constraints $0.886x1 \geq 1.2$ (Constraint 1)

$0.586x1 + 0.716x2 \geq 1.2$ (Constraint 2)

| | | | | | |
|--|---------|----------|-------|-------|-----|
| | 3000 x1 | 1.354402 | 0.886 | 0.586 | |
| | 2000 x2 | 0.567487 | | 0.716 | |
| | | 5198.179 | 1.2 | 1.2 | 1.2 |

Variables

Solver Parameters

Set Objective:

SCS12

To:

 Max Min Value Of:

0

By Changing Variable Cells:

SCS10:SCS11

Subject to the Constraints:

SDS12 >= SFS12
SES12 >= SFS12

Add

Change

Delete

Reset All

Load/Save

 Make Unconstrained Variables Non-Negative

Select a Solving Method:

Simplex LP

Options

Solving Method

Select the GRG Nonlinear engine for Solver Problems that are smooth nonlinear. Select the LP Simplex engine for linear Solver Problems, and select the Evolutionary engine for Solver problems that are non-smooth.

Help

Solve

Close

DP

LP

LP2

LP3

ANN

MF

WQ

WQ2

PDF

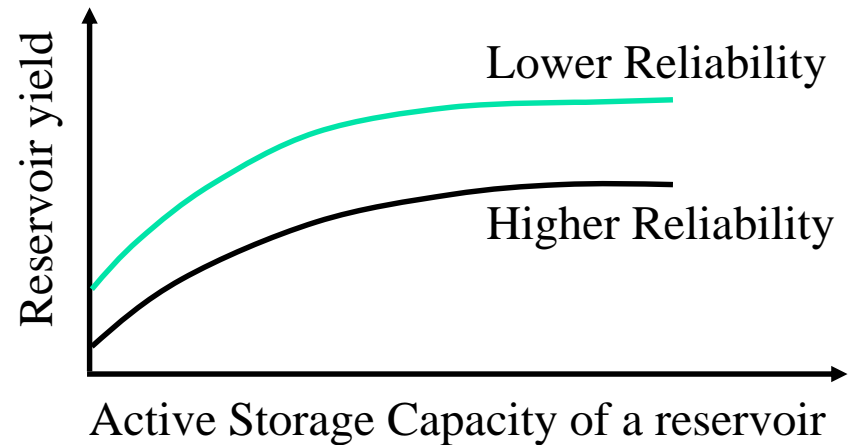
SI

Example: Regulation and Storage

- **Yield** - amount of water that can be supplied during some time interval
- **Firm yield** - amount of water that can be supplied in a *critical period*
 - Without storage: firm yield is lowest streamflow on record,
 - With storage: firm yield can be increased to approximately the mean annual flow of stream
- **Critical period** - period of lowest flow on record
- **Storage** must be provided to deliver additional water over total streamflow record
- Given target yield, required capacity depends on risk that yield will not be delivered, i.e., the **reliability** of the system

Storage Capacity – Yield Function

- Maximum constant ‘dependable’ reservoir release or yield available during each period of operation, as a function of the active storage volume capacity
- The yield from any reservoir depends on the active storage capacity and the inflow



Two storage – yield functions for a single reservoir defining the maximum and minimum dependable release. These functions can be defined for varying levels of yield reliability.

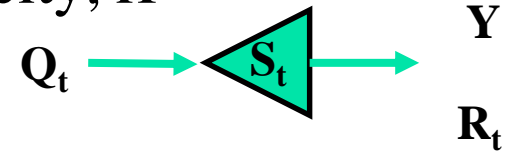
Increasing Yield – Add Storage

- Consider a sequence of 5 annual flows, say 2, 4, 1, 5 and 3, at a site in an ungauged stream.
- The minimum ‘dependable’ annual flow yield of the stream at this site is 1, the minimum observed flow. A discharge of 1 can be ‘guaranteed’ in each of the five periods of record.
- If a reservoir with an active storage capacity of 1 is built, it could store 1 unit of flow when the flow ≥ 2 , and then release it when the flow is 1, increasing the minimum dependable flow to 2 units
- Similarly, a yield of 3 could be attained in each time period with 2 units of active capacity
- The maximum annual yield cannot exceed the mean annual flow, which in this example is 3.

LP for Deriving Storage – Yield Functions

- Consider a single reservoir that must provide a minimum release or yield Y in each period t .
- Assume a record of known stream flows Q_t at the reservoir site is available.
- The problem here is to find the maximum uniform yield Y obtainable from a given active storage capacity, K

- Maximize Y , Subject to



- $S_t + Q_t - Y - R_t = S_{t+1} \quad t = 1, 2, 3, \dots, T; \quad T+1 = 1$

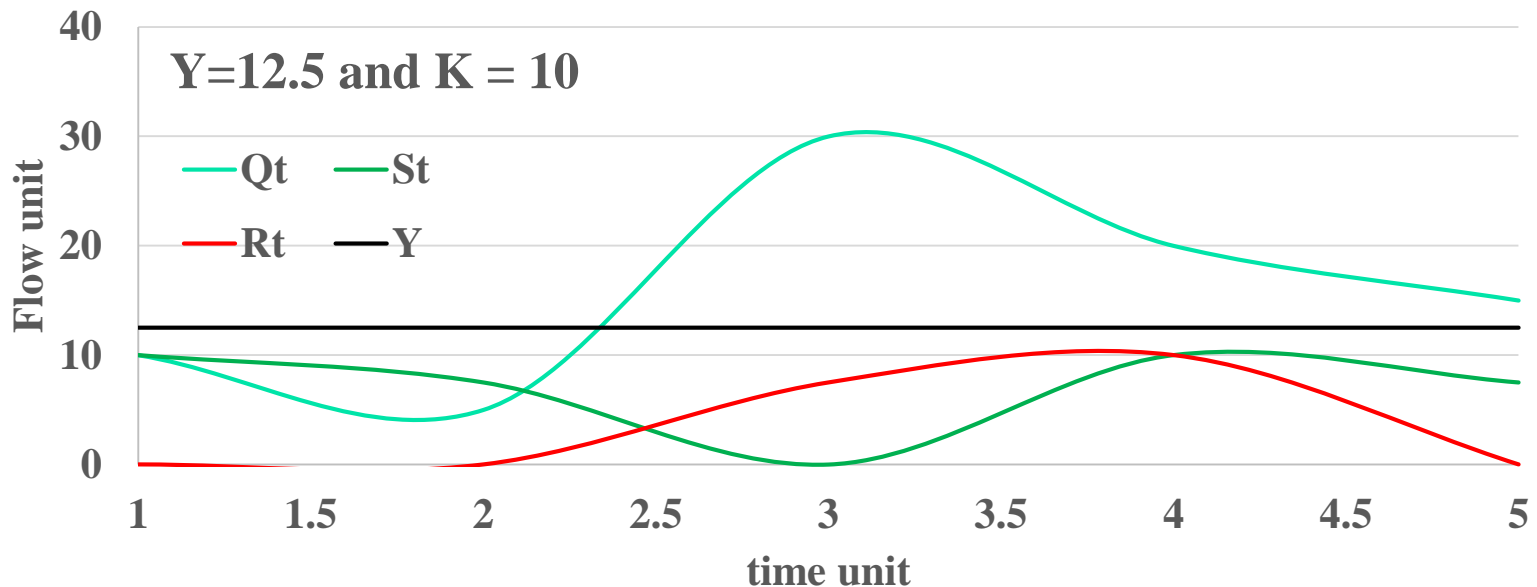
- $S_t \leq K \quad t = 1, 2, 3, \dots, T$

- The model must be solved for various a given value of capacity K .
- The upper bound on the yield, regardless of reservoir capacity, will be the mean inflow

Excel solver: solution

- The following linear programming solves an optimization problem of maximizing the uniform yield (Y) given an active storage capacity of K ($=10$).

| t | Q_t | S_t | R_t | Y |
|------|-------|-------|-------|------|
| 1 | 10 | 10 | 0 | 12.5 |
| 2 | 5 | 7.5 | 0 | 12.5 |
| 3 | 30 | 0 | 7.5 | 12.5 |
| 4 | 20 | 10 | 10 | 12.5 |
| 5 | 15 | 7.5 | 0 | 12.5 |
| mean | 16 | | | |



File Home Insert Page Layout Formulas Data Review View Help Power Pivot Tell me what you want to do

Get External Data: From Access, From Web, From Text, From Other Sources, Existing Connections

Get & Transform: Show Queries, From Table, Recent Sources, New Query

Connections: Refresh All, Properties, Edit Links

Sort & Filter: Sort, Filter

Data Tools: Flash Fill, Consolidate, Remove Duplicates, Relationships, Text to Columns, Data Validation, Manage Data Model

E8

the following linear programming solves an optimization problem of maximizing the uniform yield (Y) given an active storage capacity of K (=10). the flow recorded in each of the five time periods of an average year are 10, 5,30,20 and 15

Objective function Maximize (Y)

Constraints

$$S_t + Q_t - Y - R_t = S_{t+1} \quad t = 1, 2, 3, \dots, T;$$

$$T+1 = I \text{ (Constraint 1)}$$

$$S_t \leq K \quad t = 1, 2, 3, \dots, T \text{ (Constraint 2)}$$

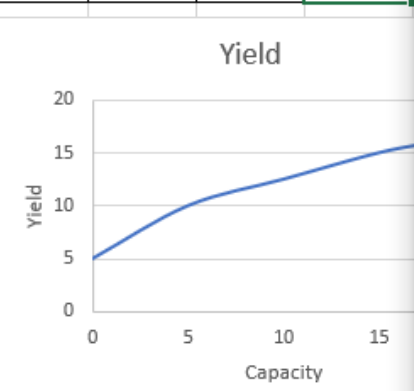
$$Yt = Y \text{ (Constraint 3)}$$

| t | Qt | St | Rt | Y | St+1 | K |
|------|----|-----|-----|------|------|----|
| 1 | 10 | 10 | 0 | 12.5 | 10 | 10 |
| 2 | 5 | 7.5 | 0 | 12.5 | 7.5 | |
| 3 | 30 | 0 | 7.5 | 12.5 | 0 | |
| 4 | 20 | 10 | 10 | 12.5 | 10 | |
| 5 | 15 | 7.5 | 0 | 12.5 | 7.5 | |
| mean | 16 | | | | | |

Given

Constrained Variable:

| Capacity | Yield |
|----------|-------|
| 0 | 5 |
| 5 | 10 |
| 10 | 12.5 |
| 15 | 15 |
| 18 | 16 |



Solver Parameters

Set Objective: ↑

To: Max Min Value Of:

By Changing Variable Cells: ↑

Subject to the Constraints:

- Add
- Change
- Delete
- Reset All
- Load/Save
- Add
- Change

Make Unconstrained Variables Non-Negative

Select a Solving Method: Options

Solving Method

Select the GRG Nonlinear engine for Solver Problems that are smooth nonlinear. Select the LP Simplex engine for linear Solver Problems, and select the Evolutionary engine for Solver problems that are non-smooth.

Help Solve Close

Exercise:

- Alternatively, one can solve a number of linear programming models that minimize an unknown storage capacity K needed to achieve various specified yields Y .
- The problem is to find the minimum storage capacity K obtainable from a specified yield, Y
- Minimize K , Subject to
 - $S_t + Q_t - Y - R_t = S_{t+1} \quad t = 1, 2, 3, \dots, T; \quad T+1 = 1$
 - $S_t \leq K \quad t = 1, 2, 3, \dots, T$
- Exercise: find the storage needed to achieve a uniform specific yield of 12.5

Excel Solver solution

- The following linear programming solves an optimization problem of minimizing the Storage (Y) given an average uniform Yield of $Y (=12.5)$.

| t | Q_t | S_t | R_t | Y_t |
|-----|-------|-------|-------|-------|
| 1 | 10 | 10 | 0 | 12.5 |
| 2 | 5 | 7.5 | 0 | 12.5 |
| 3 | 30 | 0 | 17.5 | 12.5 |
| 4 | 20 | 0 | 0 | 12.5 |
| 5 | 15 | 7.5 | 0 | 12.5 |

