Optimization (Linear Programming)

Computer Applications in Civil Engineering

Drs. Trani and Rakha
Civil and Environmental Engineering
Virginia Polytechnic Institute and State University

Spring 2000

Resource Allocation

Principles of **Mathematical Programming**

Mathematical programming is a general technique to solve resource allocation problems using optimization. Types of problems:

- Linear programming
- Integer programming
- Dynamic programming
- Decision analysis
- Network analysis and CPM

Mathematical Programming

Operations research was born with the increasing need to solve optimal resource allocation during WWII.

- Air Battle of Britain
- North Atlantic supply routing problems
- Optimal allocation of military convoys in Europe

Dantzig (1947) is credited with the first solutions to linear programming problems using the Simplex Method

Resource Allocation

Linear Programming Applications

- Allocation of products in the market
- Mixing problems
- Allocation of mobile resources in infrastructure construction (e.g., trucks, loaders, etc.)
- Crew scheduling problems
- Network flow models
- Pollution control and removal
- Estimation techniques

Linear Programming

General Formulation

Maximize $\sum_{j=1}^{n} c_j x_j$

subject to: $\sum_{j=1}^{n} a_{ij} x_j \le b_i$ for $i = 1, 2, \ldots, m$

$x_j \ge 0$ for $j = 1, 2, \ldots, n$

Linear Programming

Maximize $Z = c_1x_1 + c_2x_2 + \ldots + c_nx_n$

Subject to:

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

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

...

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

and $x_1 \geq 0,\ x_2 \geq 0,\ \ldots,\ x_n \geq 0$

Linear Programming

$$\sum_{j=1}^{n} c_j x_j$$ Objective Function (OF)

$$\sum_{j=1}^{n} a_{ij} x_j \le b_i$$ Functional Constraints (*m* of them)

$x_j \ge 0$ Nonnegativity Conditions (*n* of these)

x_j are decision variables to be optimized (min or max)

c_j are costs associated with each decision variable

Linear Programming

a_{ij} are the coefficients of the functional constraints

b_i are the amounts of the resources available (RHS)

Some definitions

Feasible Solution (FS) - A solution that satisfies all functional constraints of the problem

Basic Feasible Solution (BFS)- A solution that needs to be further investigated to determine if optimal

Initial Basic Feasible Solution - a BFS used as starting point to solve the problem

LP Example (Construction)

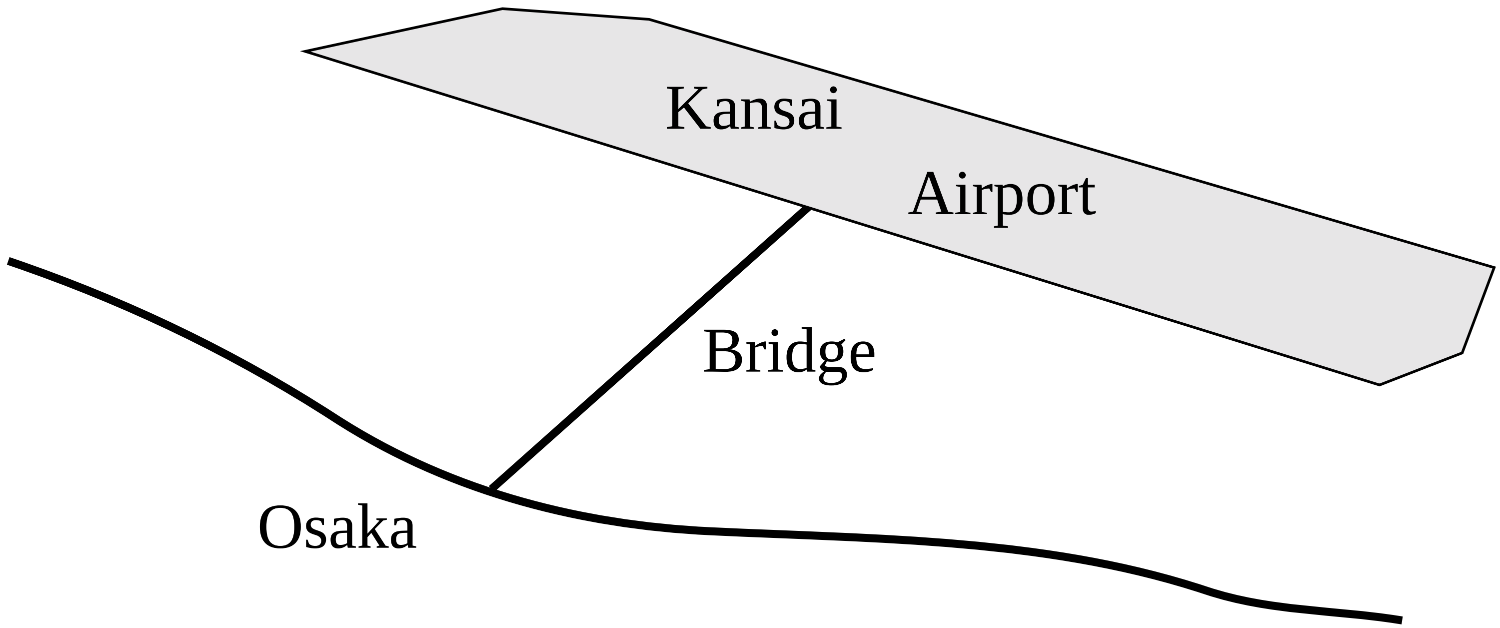
During the construction of an off-shore airport in Japan the main contractor used two types of cargo barges to transport materials from a fill collection site to the artificial island built to accommodate the airport.

The types of cargo vessels have different cargo capacities and crew member requirements as shown in the table:

Vessel Type	Capacity (m-ton)	Crew required	Number available
Fuji	300	3	40
Haneda	500	2	60

Osaka Bay Model

According to company records there are 180 crew members in the payroll and all crew members are trained to either manage the “Haneda” or “Fuji” vessels.



Osaka Bay Model

Mathematical Formulation

Maximize $Z = 300x_1 + 500x_2$

subject to: $3x_1 + 2x_2 \leq 180$

$x_1 \leq 40$

$x_2 \leq 60$

$x_1 \geq 0$ and $x_2 \geq 0$

Note: let x_1 and x_2 be the no. “Fuji” and “Haneda” vessels

Osaka Bay LP Model

Maximize $Z = 300x_1 + 500x_2$

Solution:

a) Covert the problem in standard (canonical) form

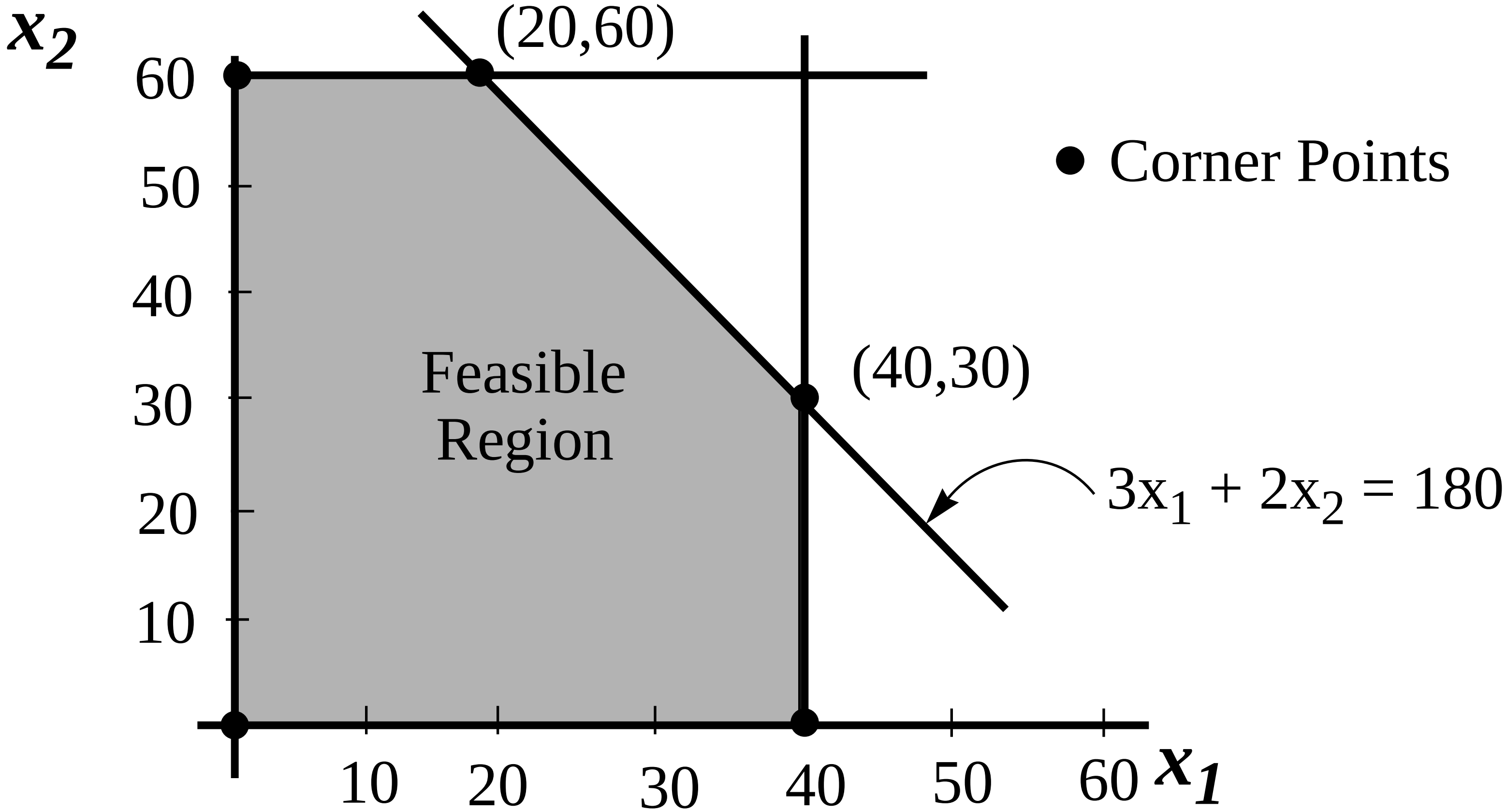
subject to: $3x_1 + 2x_2 + x_3 = 180$

$$x_1 + x_4 = 40$$

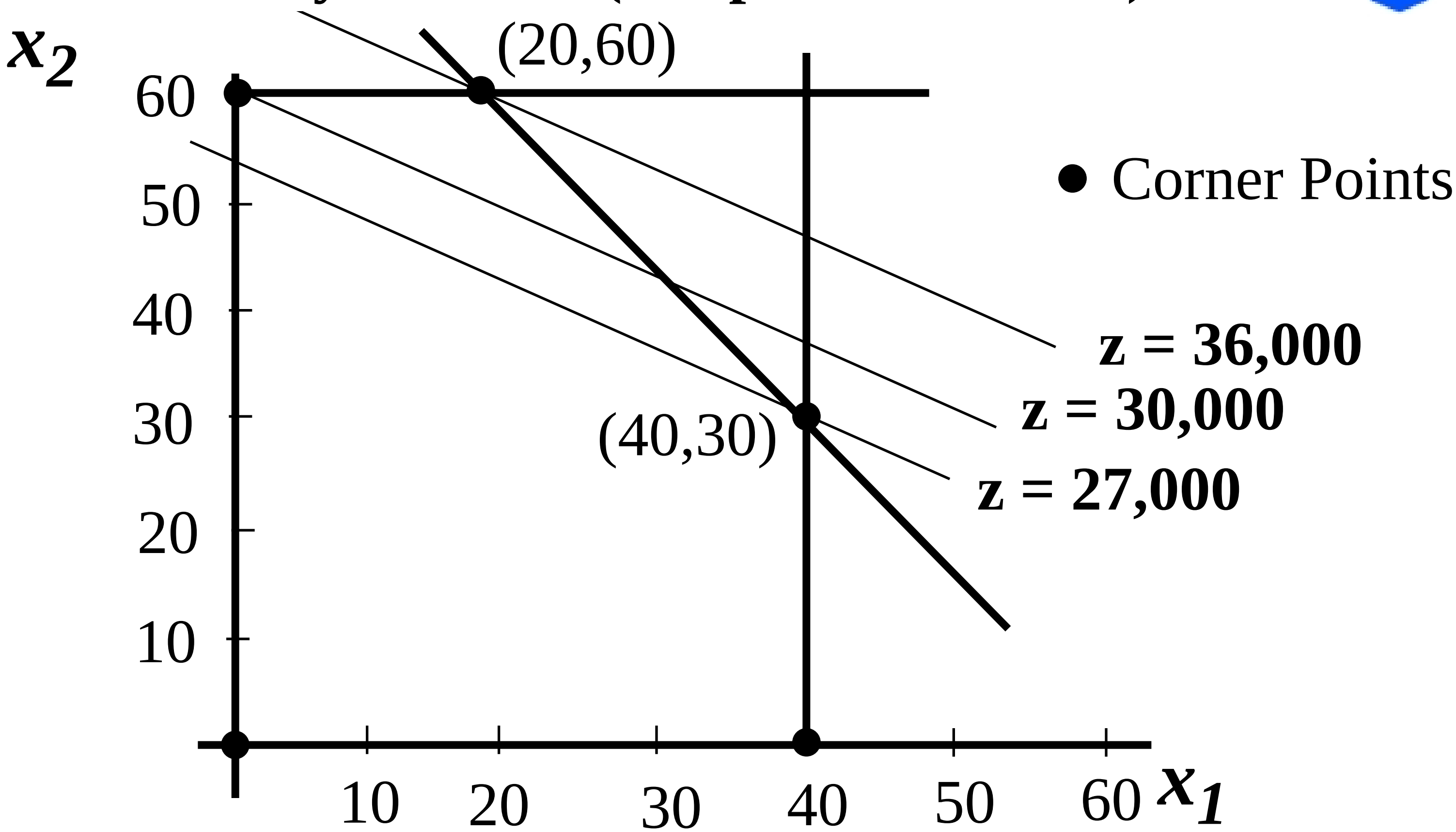
$$x_2 + x_5 = 60$$

$x_1 \geq 0$ and $x_2 \geq 0$

Osaka Bay Problem (Graphical Solution)



Osaka Bay Problem (Graphical Solution)



Note: Optimal Solution (x_1, x_2) = (20,60) vessels

Osaka Bay Problem (Simplex)

Arrange objective function in standard form to perform Simplex tableaus

$$Z - 300x_1 - 500x_2 = 0$$

$$3x_1 + 2x_2 + x_3 = 180$$

$$x_1 + x_4 = 40$$

$$x_2 + x_5 = 60$$

$x_1 \geq 0$, $x_2 \geq 0$, $x_3 \geq 0$, $x_4 \geq 0$ and $x_5 \geq 0$

Note: x_3, x_4, x_5 are **slack variables**

Osaka Bay Example (Initial Tableau)

BV	**z**	$\boldsymbol{x_1}$	$\boldsymbol{x_2}$	$\boldsymbol{x_3}$	$\boldsymbol{x_4}$	$\boldsymbol{x_5}$	**RHS**
z	**1**	**-300**	**-500**	**0**	**0**	**0**	**0**
$\boldsymbol{x_3}$	0	3	2	1	0	0	180
$\boldsymbol{x_4}$	0	1	0	0	1	0	40
$\boldsymbol{x_5}$	0	0	1	0	0	1	60

BV = x_3, x_4, x_5 and NBV = x_1, x_2

Solution: $(x_1, x_2, x_3, x_4, x_5) = (0,0,180,40,60)$

Osaka Bay Example (Initial Tableau)

BV	z	x_1	x_2	x_3	x_4	x_5	RHS	ratio
z	**1**	**-300**	**-500**	**0**	**0**	**0**	**0**	
x_3	0	3	2	1	0	0	180	90
x_4	0	1	0	0	1	0	40	inf
x_5	0	0	1	0	0	1	60	60

x_2 improves the objective function more than x_1

Leaving BV = x_5 : New BV = x_2

Osaka Bay Example (Second Tableau)

BV	**z**	x_1	x_2	x_3	x_4	x_5	**RHS**	**ratio**
z	**1**	**-300**	**0**	**0**	**0**	**500**	**30,000**	
x_3	0	3	0	1	0	0	60	20
x_4	0	1	0	0	1	0	40	40
x_2	0	0	1	0	0	1	60	inf

x_1 improves the objective function the maximum

Leaving BV = x_3 : New BV = x_1

Osaka Bay Example (Final Tableau)

BV	**z**	x_1	x_2	x_3	x_4	x_5	**RHS**
z	**1**	**0**	**0**	**100**	**0**	**300**	**36,000**
x_1	0	1	0	1/3	0	0	20
x_4	0	0	0	-1/3	1	2/3	20
x_2	0	0	1	0	0	1	60

Note: All NVB coefficients are positive or zero in tableau

Optimal Solution: $(\boldsymbol{x_1}, \boldsymbol{x_2}, x_3, x_4, x_5) = (\mathbf{20}, \mathbf{60}, 0, 20, 0)$

Osaka Bay Model (Revised)

Mathematical Formulation

Maximize $Z = 300x_1 + 500x_2$

subject to: $3x_1 + 2x_2 = 180$ Revised Constraint

$x_1 \leq 40$

$x_2 \leq 60$

$x_1 \geq 0$ and $x_2 \geq 0$

Note: let x_1 and x_2 be the no. "Fuji" and "Haneda" vessels

Osaka Bay Model (Revised)

Maximize $Z = 300x_1 + 500x_2$

a) Covert the problem in standard form

subject to: $3x_1 + 2x_2 = 180$

$$x_1 + x_3 = 40$$

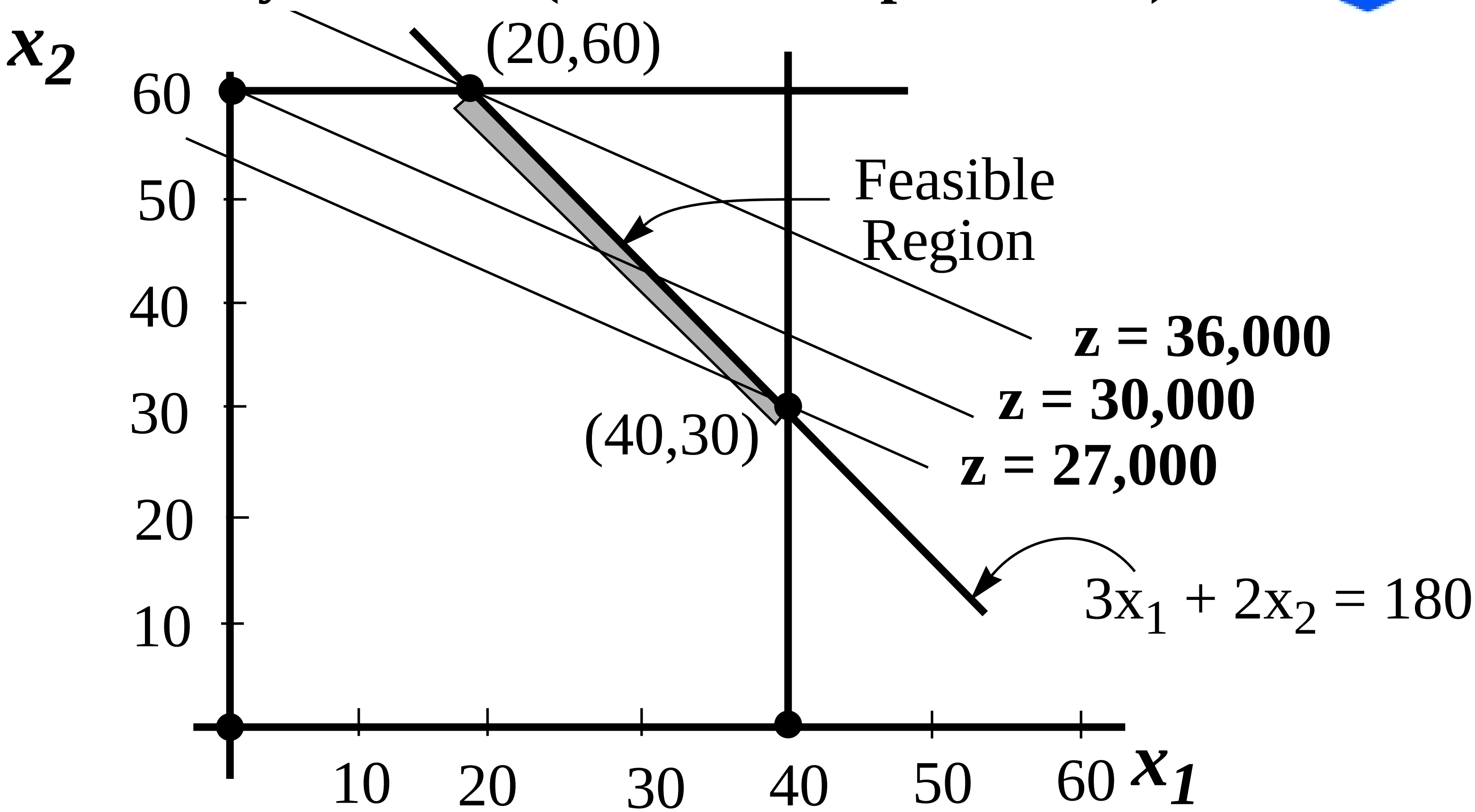
$$x_2 + x_4 = 60$$

$x_1 \geq 0$, $x_2 \geq 0$, $x_3 \geq 0$ and $x_4 \geq 0$

- Note: Problem lacks an intuitive IBFS (see first constraint)

- Note that setting $x_1 = 0$ and $x_2 = 0$ produces finite integer values for x_3 and x_4 (40 and 60, respectively) but fails to provide and adequate solution for constraint (1).
- This requires a reformulation step where another variable is added to the problem to identify an IBFS
- Add an artificial variable to the first constraint to solve the problem
- Adding an artificial variable in the constraint equation requires the addition of a large penalty to the objective function (z) to avoid this artificial variable being part of the solution

Osaka Bay Problem (Revised Graphical Sol.)



Osaka Bay Model (Revised)

Maximize $Z = 300x_1 + 500x_2$

a) Add an artificial variable to the initial "equal to" constraint

subject to: $3x_1 + 2x_2 + \bar{x}_5 = 180$

$$x_1 + x_3 = 40$$

$$x_2 + x_4 = 60$$

$x_1 \geq 0$, $x_2 \geq 0$, $x_3 \geq 0$, $x_4 \geq 0$ and $\bar{x}_5 \geq 0$

IBFS is now evident with x_1 and x_2 being zero (NVB).

Revised Solution (Big-M Method)

Revise the **objective function** to drive artificial variable to zero in the optimal solution. M is a large positive number.

Maximize $Z = 300x_1 + 500x_2 - M\bar{x}_5$

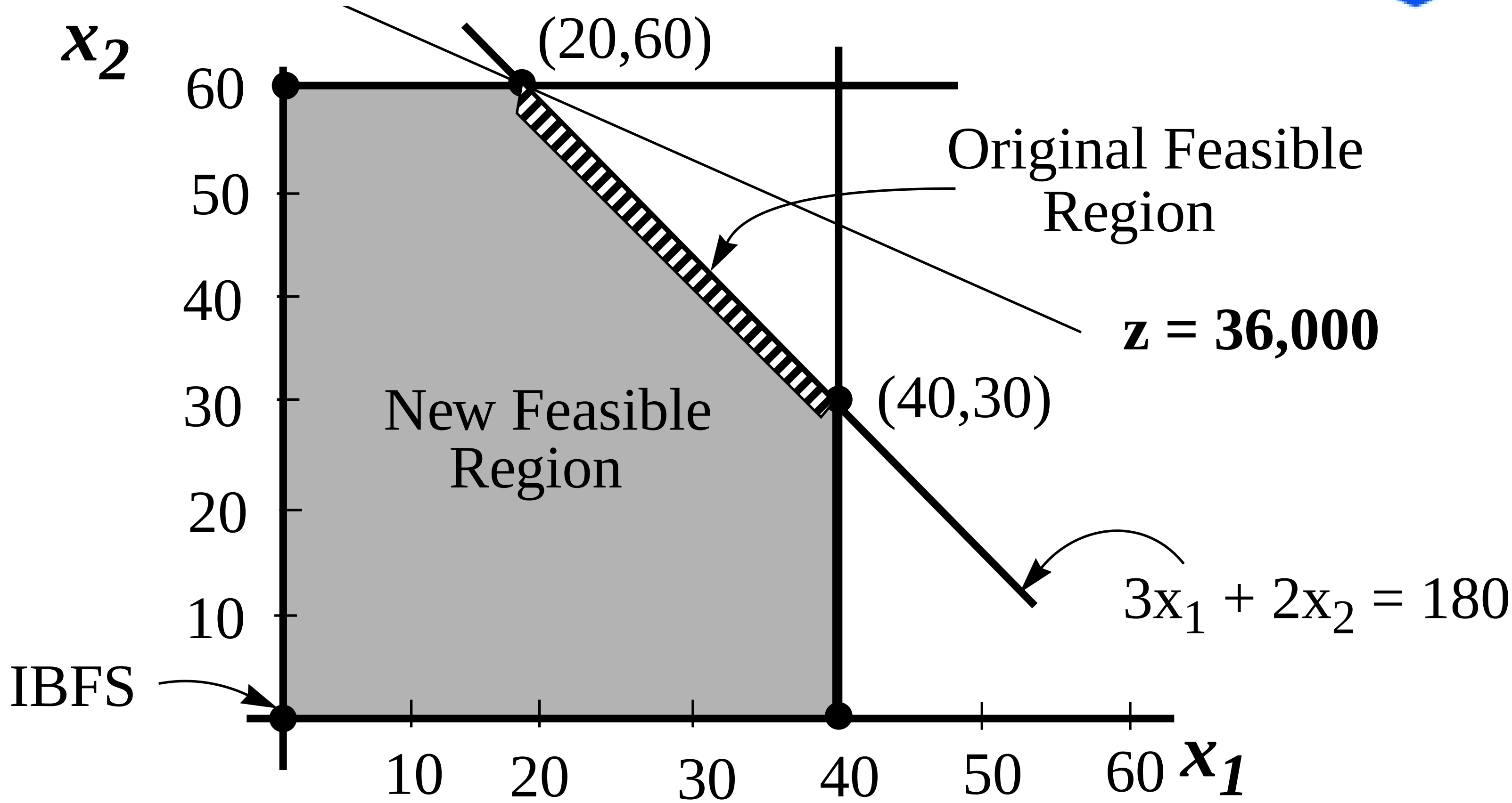
subject to: $3x_1 + 2x_2 + \bar{x}_5 = 180$

$x_1 + x_3 = 40$

$x_2 + x_4 = 60$

$x_1 \geq 0$, $x_2 \geq 0$, $x_3 \geq 0$, $x_4 \geq 0$ and $\bar{x}_5 \geq 0$

Osaka Bay LP (Expanded Feasible Region)



Revised Solution (Big-M Method)

Rearrange the OF and constraints before solving

Maximize $Z - 300x_1 - 500x_2 + M\bar{x}_5 = 0$

subject to: $x_1 + x_3 = 40$

$x_2 + x_4 = 60$

$3x_1 + 2x_2 + \bar{x}_5 = 180$

$x_1 \geq 0$, $x_2 \geq 0$, $x_3 \geq 0$, $x_4 \geq 0$ and $\bar{x}_5 \geq 0$

Note: the “Big M” (or a large penalty) is added to each artificial variable in OF. x_3 and x_4 are slack variables, $\bar{x}_5$ is an artificial variable.

Revised Osaka Bay LP (Initial Tableau)

BV	z	x_1	x_2	x_3	x_4	$\bar{x}_5$	RHS
z	**1**	**-300**	**-500**	**0**	**0**	**M**	**0**
x_3	0	1	0	1	0	0	40
x_4	0	0	1	0	1	0	60
x_5	0	3	2	0	0	1	180

BV = $x_3, x_4, \bar{x}_5$ and NBV = x_1, x_2

Solution: $(x_1, x_2, x_3, x_4, \bar{x}_5) = (0,0,40,60,180)$

Revised Osaka Bay LP (Initial Tableau)

BV	z	x_1	x_2	x_3	x_4	$\bar{x}_5$	RHS	
z	**1**	**-3M-300**	**-2M-500**	**0**	**0**	**0**	**-180M**	
x_3	0	1	0	1	0	0	40	40
x_4	0	0	1	0	1	0	60	inf
x_5	0	3	2	0	0	1	180	60

x_1 improves the objective function the maximum

Leaving BV = x_3 : New BV = x_1

Revised Osaka Bay LP (2nd Tableau)

BV	z	x_1	x_2	x_3	x_4	$\bar{x}_5$	RHS	
z	**1**	**0**	**-2M-500**	**3M+300**	**0**	**0**	**-60M+ 12000**	
x_1	0	1	0	1	0	0	40	inf
x_4	0	0	1	0	1	0	60	60
x_5	0	0	2	-3	0	1	60	30

x_2 improves the objective function the maximum. Leaving BV = $\bar{x}_5$: New BV = x_2

Revised Osaka Bay LP (3rd Tableau)

BV	**z**	x_1	x_2	x_3	x_4	$\bar{x}_5$	**RHS**	
z	**1**	**0**	**0**	**-450**	**M+250**	**0**	**27000**	
x_1	0	1	0	1	0	0	40	40
x_4	0	0	0	3/2	1	-1/2	30	20
x_2	0	0	1	-3/2	0	1/2	30	no

x_3 improves the objective function the maximum. Leaving BV = x_4 : New BV = x_3

Revised Osaka Bay LP (Final Tableau)

BV	**z**	x_1	x_2	x_3	x_4	$\bar{x}_5$	**RHS**
z	**1**	**0**	**0**	**0**	**300**	**M+100**	**36000**
x_1	0	1	0	0	-2/3	1/3	20
x_3	0	0	0	1	2/3	-1/3	20
x_2	0	0	1	0	-1/2	1/2	60

Note: All NVB coefficients are positive or zero in tableau

Optimal Solution: (x_1, x_2, x_3, x_4, $\bar{x}_5$) = (**20,60**,20,0,0)

Simplex Method Anomalies

a) Ties for leaving BV - break without arbitration

b) Ties for entering BV - break without arbitration

c) Zero coefficient of NBV in OF (final tableau) - Implies multiple optimal solutions

d) No leaving BV - implies unbounded solution

Steps in the Simplex Method

I) Initialization Step

- Introduce slack variables
- Select original variables of the problems as part of the NBV
- Select slacks as BV

II) Stopping Rule

- The solution is optimal if every coefficient in the OF is nonnegative

- Coefficients of OF measure the rates of change of the OF as any other variable increases from zero

III) Iterative Step

- Determine the entering NBV (pivot column)
- Determine the leaving BV (from BV set) as the first variable to go to zero without violating constraints
- Perform row operations to make coefficients of BV unity in their respective rows
- Eliminate new BV coefficients (from pivot column) from other equations performing row operations

Linear Programming Strategies Using the Simplex Method

- Identify the problem
- Formulate the problem using LP
- Solve the problem using LP
- Test the model (correlation and sensitivity analysis)
- Establish controls over the model
- Implementation
- Model re-evaluation

LP Formulations

Type of Constraint	**How to handle**
$3x_1 + 2x_2 \leq 180$	Add a slack variable
$3x_1 + 2x_2 = 180$	Add an artificial variable Add a penalty to OF (BigM)
$3x_1 + 2x_2 \geq 180$	Add a negative slack and a positive artificial variable

LP (Handling Constraints)

Type of Constraint	**Equivalent Form**
$3x_1 + 2x_2 \leq 180$	$3x_1 + 2x_2 + x_3 = 180$
$3x_1 + 2x_2 = 180$	$3x_1 + 2x_2 + \bar{x}_3 = 180$
	$z = c_1x_1 + c_2x_2 - Mx_3$
$3x_1 + 2x_2 \geq 180$	$3x_1 + 2x_2 - x_3 + \bar{x}_4 = 180$
	$z = c_1x_1 + c_2x_2 - \overline{M}\bar{x}_4$

Note: M is a large positive number

Theory Behind Linear Programming

General Formulation

Maximize $Z = \sum_{j=1}^{n} c_j x_j$

subject to: $\sum_{j=1}^{n} a_{ij} x_j \leq b_i$ for $i=1,2, \ldots, m$

$x_j \geq 0$ for $j=1,2,\ldots,n$

General LP Formulation (Matrix Form)

Maximize $Z = \boldsymbol{cx}$

subject to: $\boldsymbol{Ax} = \boldsymbol{b}$

$\boldsymbol{x} \geq \boldsymbol{0}$ where:

$\boldsymbol{c}$ is the vector containing the coefficients of the O.F.,

$\boldsymbol{A}$ is the matrix containing all coefficients of the functional constraints,

$\boldsymbol{b}$ is the column vector for RHS coefficients,

$\boldsymbol{x}$ is the vector of decision variables

note that: $\boldsymbol{c} = \begin{bmatrix} c_1 & c_2 \dots & c_n \end{bmatrix}$

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_n \end{bmatrix}, \ \boldsymbol{b} = \begin{bmatrix} b_1 \\ b_2 \\ b_n \end{bmatrix}, \ \boldsymbol{0} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$ and matrix $\boldsymbol{A}$

$$A = \begin{bmatrix} a_{11} & a_{12} \dots & a_{1n} \\ a_{21} & a_{22} \dots & a_{2n} \\ a_{m1} & a_{m2 \dots} & a_{mn} \end{bmatrix}$$

Theory Behind the Simplex Method

Addition of slack variables to the problem yields:

$$\boldsymbol{x}_s = \begin{bmatrix} x_{n+1} \\ x_{n+2} \\ x_{n+m} \end{bmatrix}$$ where $\boldsymbol{x}_s$ is a vector of slack variables (m)

New augmented constraints become,

$$\begin{bmatrix} A & I \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{x}_s \end{bmatrix} = \boldsymbol{b} \text{ and } \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{x}_s \end{bmatrix} \geq \boldsymbol{0}$$

Note: I is an $m \times m$ identity matrix.

Theory Behind the Simplex Method

Basic Feasible Solution. From the system,

$\begin{bmatrix} A & I \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{x}_s \end{bmatrix} = \boldsymbol{b}$ n Nonbasic Variables (NBV) from the set,

$\begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{x}_s \end{bmatrix}$ are set to be equal to zero.

This leaves a set of m equations and m unknowns.

These unknowns correspond to the set of <u>basic variables</u>

Theory Behind the Simplex Method

Let the set of basic variables be called $\boldsymbol{x}_B$ and the matrix containing the coefficients of the functional constraints be called $\bar{\boldsymbol{A}}$ (basis matrix) so that,

$$\bar{\boldsymbol{A}}\boldsymbol{x}_B = \boldsymbol{b}$$

$$\boldsymbol{x}_B = \begin{bmatrix} x_{B1} \\ x_{B2} \\ x_{Bm} \end{bmatrix}$$

The vector $\boldsymbol{x}_B$ is called vector of basic variables.

Theory Behind the Simplex Method

The idea behind each basic feasible solution in the Simplex Algorithm is to eliminate NBV from the set,

$$\begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{x}_s \end{bmatrix}$$

and

$$\bar{\boldsymbol{A}} = \begin{bmatrix} \bar{a}_{11} & \bar{a}_{12} \ldots & \bar{a}_{1m} \\ \bar{a}_{21} & \bar{a}_{22} \ldots & \bar{a}_{2m} \\ \bar{a}_{m1} & \bar{a}_{m2 \ldots} & \bar{a}_{mm} \end{bmatrix}$$ the basis matrix (a square matrix).

Theory Behind the Simplex Method

From simple matrix algebra (solve for $\boldsymbol{x}_B$) from,

$$\bar{\boldsymbol{A}}\boldsymbol{x}_B = \boldsymbol{b}$$

$$(\bar{\boldsymbol{A}})^{-1}\bar{\boldsymbol{A}}\boldsymbol{x}_B = (\bar{\boldsymbol{A}})^{-1}\boldsymbol{b}$$

$$\boldsymbol{x}_B = (\bar{\boldsymbol{A}})^{-1}\boldsymbol{b}$$

if $\boldsymbol{c}_B$ is the vector of the coefficients of the objective function this brings us to the following value of the objective function:

$$Z = \boldsymbol{c}_B\boldsymbol{x}_B = (\bar{\boldsymbol{A}})^{-1}\boldsymbol{b}$$

Theory Behind the Simplex Method

The original set of equations to start the Simplex Method is,

$$\begin{bmatrix} 1 & -\boldsymbol{c} & \boldsymbol{o} \\ \boldsymbol{o} & \boldsymbol{A} & I \end{bmatrix} \begin{bmatrix} Z \\ \boldsymbol{x} \\ \boldsymbol{x}_s \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{b} \end{bmatrix}$$

after each iteration in the Simplex Method,

$$\boldsymbol{x}_B = (\bar{\boldsymbol{A}})^{-1}\boldsymbol{b}$$

and $Z = \boldsymbol{c}_B \boldsymbol{x}_B = (\bar{\boldsymbol{A}})^{-1}\boldsymbol{b}$

The RHS of the new set of equations becomes,

Theory Behind the Simplex Method

$$\begin{bmatrix} Z \\ \boldsymbol{x}_B \end{bmatrix} = \begin{bmatrix} 1 & \boldsymbol{c}_B(\bar{A})^{-1} \\ \mathbf{0} & (\bar{A})^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{0} \\ \boldsymbol{b} \end{bmatrix} = \begin{bmatrix} \boldsymbol{c}_B(\bar{A})^{-1} b \\ (\bar{A})^{-1} b \end{bmatrix}$$

$$\begin{bmatrix} 1 & \boldsymbol{c}_B(\bar{A})^{-1} \\ \mathbf{0} & (\bar{A})^{-1} \end{bmatrix} \begin{bmatrix} 1 & -\boldsymbol{c} & \boldsymbol{o} \\ \boldsymbol{o} & \boldsymbol{A} & I \end{bmatrix} = \begin{bmatrix} 1 & \boldsymbol{c}_B(\bar{A})^{-1} - \boldsymbol{c} & \boldsymbol{c}_B(\bar{A})^{-1} \\ \boldsymbol{o} & (\bar{A})^{-1}\boldsymbol{A} & (\bar{A})^{-1} \end{bmatrix}$$

After any iteration,

$$\begin{bmatrix} 1 & \boldsymbol{c}_B(\bar{A})^{-1} - \boldsymbol{c} & \boldsymbol{c}_B(\bar{A})^{-1} \\ \boldsymbol{o} & (\bar{A})^{-1}\boldsymbol{A} & (\bar{A})^{-1} \end{bmatrix} \begin{bmatrix} Z \\ \boldsymbol{x} \\ \boldsymbol{x}_s \end{bmatrix} = \begin{bmatrix} \boldsymbol{c}_B(\bar{A})^{-1} b \\ (\bar{A})^{-1} b \end{bmatrix}$$

In tableau format this becomes,

Theory of the Simplex Method

Iteration	**BV**	**Z**	**Original Variables**	**Slack Variables**	**RHS**
0	Z	1	$-\boldsymbol{c}$	$\mathbf{0}$	0
	$\boldsymbol{x}_B$	$\mathbf{0}$	$\boldsymbol{A}$	$\boldsymbol{I}$	$\boldsymbol{b}$
[redacted]					
Any	Z	1	$\boldsymbol{c}_B(\bar{A})^{-1}-\boldsymbol{c}$	$\boldsymbol{c}_B(\bar{A})^{-1}$	$\boldsymbol{c}_B(\bar{A})^{-1}b$
	$\boldsymbol{x}_B$	$\mathbf{0}$	$(\bar{\boldsymbol{A}})^{-1}\boldsymbol{A}$	$(\bar{A})^{-1}$	$(\bar{A})^{-1}b$

Numerical Example

To illustrate the use of the revised simplex method consider the Osaka Bay example:

Maximize $Z = 300x_1 + 500x_2$

subject to: $3x_1 + 2x_2 \leq 180$

$x_1 \leq 40$

$x_2 \leq 60$

$x_1 \geq 0$ and $x_2 \geq 0$

Note: let x_1 and x_2 be the no. "Fuji" and "Haneda" vessels

note that: $\boldsymbol{c} = \left[\begin{bmatrix} 300 & 500 \end{bmatrix}\right]$ coefficients of real variables

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \ \boldsymbol{b} = \begin{bmatrix} 180 \\ 40 \\ 60 \end{bmatrix}, \ \boldsymbol{0} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$ and matrix $\boldsymbol{A}$

$$A = \begin{bmatrix} 3 & 2 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \end{bmatrix}$$

Theory Behind the Simplex Method

Addition of slack variables to the problem yields:

$$\boldsymbol{x}_s = \begin{bmatrix} x_3 \\ x_4 \\ x_5 \end{bmatrix} \text{ where } \boldsymbol{x}_s \text{ is a vector of slack variables}$$

Executing the procedure for the Simplex Method

Iteration 0:

$$\boldsymbol{x}_B = \begin{bmatrix} x_3 \\ x_4 \\ x_5 \end{bmatrix}, \ (\bar{\boldsymbol{A}})^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \text{ and } \begin{bmatrix} x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 180 \\ 40 \\ 60 \end{bmatrix} = \begin{bmatrix} 180 \\ 40 \\ 60 \end{bmatrix}$$

also known,

$\boldsymbol{c}_B = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}$ and hence $Z = \boldsymbol{c}_B \boldsymbol{x}_B = (\bar{\boldsymbol{A}})^{-1} \boldsymbol{b}$ or

$$Z = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 180 \\ 40 \\ 60 \end{bmatrix} = 0$$

Iteration 1: (refer to 2nd tableau in Simplex)

Note: substitute values for $\bar{\boldsymbol{A}}$ using columns for x_3, x_4 and x_2 in the original A matrix.

$$\boldsymbol{x}_B = \begin{bmatrix} x_3 \\ x_4 \\ x_2 \end{bmatrix}, \bar{\boldsymbol{A}} = \begin{bmatrix} 1 & 0 & 2 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \bar{\boldsymbol{A}}^{-1} = \begin{bmatrix} 1 & 0 & -2 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$ and

$$\begin{bmatrix} x_3 \\ x_4 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & -2 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 180 \\ 40 \\ 60 \end{bmatrix} = \begin{bmatrix} 60 \\ 40 \\ 60 \end{bmatrix}$$

also known,

$\boldsymbol{c}_B = \begin{bmatrix} 0 & 0 & 500 \end{bmatrix}$ and hence $Z = \boldsymbol{c}_B \boldsymbol{x}_B = (\bar{\boldsymbol{A}})^{-1} \boldsymbol{b}$ or

$$Z = \begin{bmatrix} 0 & 0 & 500 \end{bmatrix} \begin{bmatrix} 60 \\ 40 \\ 60 \end{bmatrix} = 30000$$

Iteration 2: (refer to 3rd tableau in Simplex)

Note: substitute values for $\bar{\boldsymbol{A}}$ using columns for x_1, x_4 and x_2 in the original A matrix.

$$\boldsymbol{x}_B = \begin{bmatrix} x_1 \\ x_4 \\ x_2 \end{bmatrix}, \bar{\boldsymbol{A}} = \begin{bmatrix} 3 & 0 & 2 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \bar{\boldsymbol{A}}^{-1} = \begin{bmatrix} \frac{1}{3} & 0 & -\frac{2}{3} \\ -\frac{1}{3} & 1 & \frac{2}{3} \\ 0 & 0 & 1 \end{bmatrix} \text{ and}$$

$$\begin{bmatrix} x_1 \\ x_4 \\ x_2 \end{bmatrix} = \begin{bmatrix} \frac{1}{3} & 0 & -\frac{2}{3} \\ -\frac{1}{3} & 1 & \frac{2}{3} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 60 \\ 40 \\ 60 \end{bmatrix} = \begin{bmatrix} 20 \\ 20 \\ 60 \end{bmatrix}$$

also known,

$c_B = \begin{bmatrix} 300 & 0 & 500 \end{bmatrix}$ and hence $Z = c_B x_B = (\bar{A})^{-1} b$ or

$$Z = \begin{bmatrix} 300 & 0 & 500 \end{bmatrix} \begin{bmatrix} 20 \\ 20 \\ 60 \end{bmatrix} = 36000$$ **Optimal Solution**

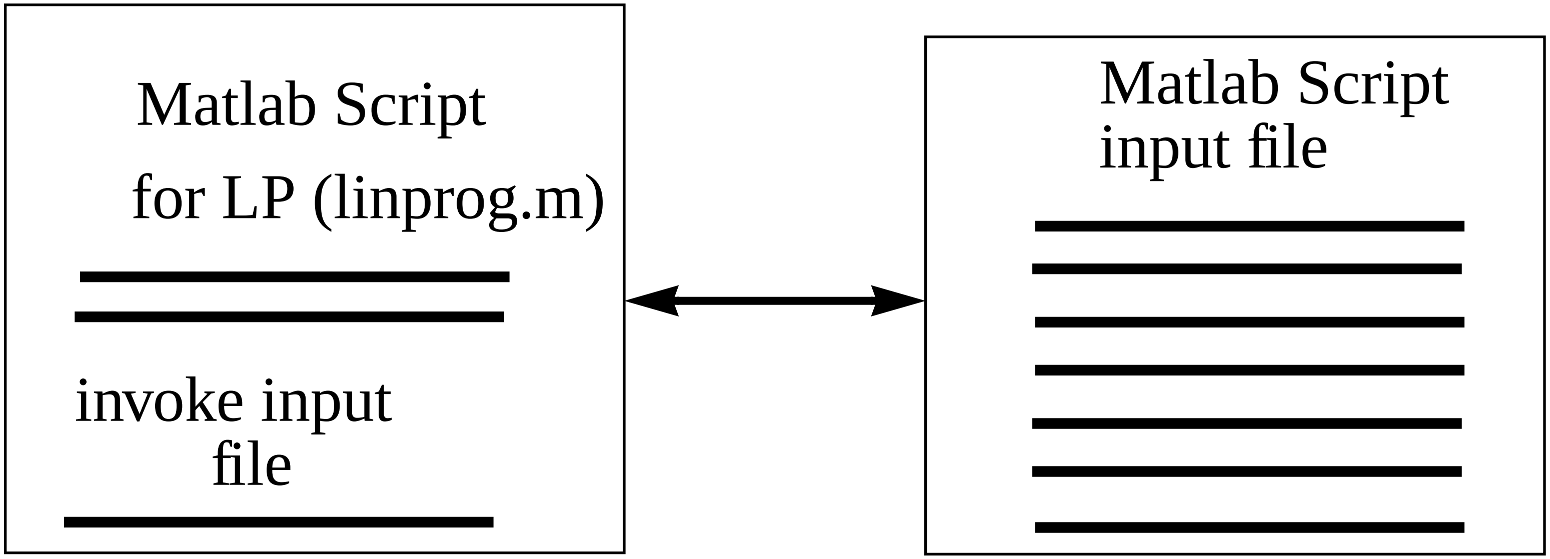
Linear Programming Programs

Several computer programs are available to solve LP problems:

- LINDO - Linear INteractive Discrete Optimizer
- GAMS - also solves non linear problems
- MINUS
- Matlab Toolbox - Optimization toolbox (from Mathworks)
- QSB - LP, DP, IP and other routines available (good for students)

LP Routine for Matlab

- Developed by Prof. Henry Wolkowicz (Univ. of Waterloo)
- Adapted by H. Baik, A. Trani, and D.R. Drew.
- Create two M files (linprog.m and input.m)



Input File

% Example: Enter the data:

```
minmax=0;    % minmax = 0  (maximizes a function)
a=[3 4 1 0 0
   1 0 0 1 0
   0 1 0 0 1 ]
b=[520 80 70]'
c=[-300 -500 0 0 0]
bas=[3 4 5]
```

LP Standard Form

Note that this input file should include also the large penalty (Big M) as part of the coefficients in OF in order to work. The problem needs to be stated in standard (canonical) form.

$$\sum_{j=1}^{n} c_j x_j \qquad \text{Objective Function (OF)}$$

$$\sum_{j=1}^{n} a_{ij} x_j \le b_i \qquad \text{Functional Constraints (m of them)}$$

Note: Nonnegativity constraints are always enforced

LP Routine for Matlab

File linprog.m to execute LP by the Simplex Method

```
% Title      : Linear Programming
% Author : Prof. Henry Wolkowicz  (Univ. of Waterloo)
% Modified: By H. Baik, A. Trani and D.R. Drew
% Date       : Nov. 29, 1996

%  The Simplex Method (data file is hw*.m)
%  Solves 'small' Linear Programming Problems (in canonical form)
%  (LP)  max cx s.t. ax=b, x>=0
%
%  Data input by user or calling routine: a,b,c,bas,pt where
%    minmax=1, if minimizing problem
```

```
%               0, if maximizing problem
%   a   - m by (m+n) matrix (containing identity)
%   b   - m-column vector assumed >=0
%   c   - m+n-row vector of costs which is placed left hand side(LHS)
%   bas - m-row vector with column indices corresp. to the
%         identity matrix in a, i.e. a(bas,:)=identity.
%   pt  - pause time after each iteration, 0 denotes no pause, while
%               any number > 20 denotes infinite time, i.e. you
%               are prompted to hit return to continue.
%  The user can change the upper bound of 100 iterations - see iterm
%     below.
%
%  The matrix a is assumed to contain an m by m identity matrix
%     corresponding to the basic columns.
%  A relative accuracy of approximately 15 significant decimal
```

```
%      digits is assumed. This affects the values of 4 accuracy
%      parameters, epsi, i=0,1,2,3.
%  The pivot step is done using 'Gauss-Jordan' elimination.
%      No special factorizations to ensure stability are used.
%      We do not use the revised simplex method.
%      A final check on roundoff error is made.
%      We use a threshold value when finding the pivot element.
%  Problem (LP) is assumed to be in canonical form, i.e. slacks
%    have been added and/or phase 1 has finished. However, we still
%    price out the cost vector c.
%
% datafile (read input file):
osaka2

disp(['If the input data is correct, hit return to continue'])
```

```
        pause
if minmax==1,
   c=-c;
end

echo off
%
pt=1
% Now solve the LP by executing the file reg.m:
% The Program:
%
%
rnderr=0;
iterm=100;
stop=1; % use to overcome the bug in the return statement
```

```
% Error tolerances (from 'Advanced Linear Progr.' by
             B.A.Murtaugh, pg 34.)

eps0=10^(-10);  %  numerical zero

eps1=10^(-5);   %  accuracy parameter for optimality check

eps2=10^(-8);   %  accuracy parameter pivot element (threshold test)

eps3=10^(-6);   %  accuracy parameter for final roundoff error check

a0=a;           %  save the matrix a for the final roundoff error test

b0=b;           %  save the vector b for the final roundoff error test

c0=c;

bas0=bas;

[m,mn]=size(a); %  row and column size of a

z=0;    %  initial value for z

          clc

          home,disp([blanks(30)]),

          disp(['Initial tableau ' blanks(10)])
```

```
if minmax==1
   tc=-c;
   [tc z
    a b]
else
   [c z
     a b]
end

 if pt > 20,
      disp(['Hit return to continue'])
      pause
 else
      pause(pt)
```

```
        end
% make cannonical form

if minmax==1
  z=-c(bas)*b;
else
  z=c(bas)*b;
end

clc
for i=1:m,
      c=c-c(bas(i))*a(i,:);
end
        clc
            home,disp([blanks(30)]),
```

```
%disp(['Price out the cost vector ' blanks(10) ])
disp(['standard or canonical form of this problem' ])

if minmax==1
   tc=-c;
   tz=-z;
   [tc tz
    a b]
 else
   [c z
     a b]
 end



 if pt > 20,
```

```
            disp(['Hit return to continue'])
            pause
        else
            pause(pt)
        end
iter=0;
       %  initialize the iteration count
n=mn-m;
       %  number of nonbasic variables
%  nbas - indices of the nonbasic variables
nbas=[];
for j=1:mn,
      if all(j~=bas),
            nbas=[nbas j];
      end
```

```
end
%  Perform simplex iterations as long as there is a neg cost
while iter<iterm,
%  Find a negative reduced cost.
    ctemp=c;
    % temporary work vector
    neg=[];
    for j=1:n,
        if ctemp(nbas(j))<-eps1,
            neg=[neg nbas(j)];
        end
    end
    ct=-1;
  if length(neg)==0,
        disp(['This phase is completed   - current basis is: '])
```

```
bas=bas
disp(['The current basic variable values are : '])
b
disp(['The current objective value is:'])
        if minmax==1
                c0(bas)*b
        else
                -c0(bas)*b
        end


disp(['The number of iterations is ' int2str(iter) ])
if norm(a0(:,bas)*b-b0,inf)>eps3, % check solution
   disp(['^WARNING^ roundoff error is significant'])
end
```

```
if any(b<-eps0), % check positive final solution
        disp(['^WARNING^ final b not nonnegative'])
end
pause(3)
clc
home,disp([blanks(30)]),
disp(['Final tableau in this phase' blanks(10)])
          disp(['          ' blanks(10)])
          if minmax==1
                    tc=-c;
                    tz=-z;
                    [tc tz a b]
          else
             [c z a b]
         end
```

```
            stop=0;
            return
  else
        while ct<-eps1,   % continue till we find a suitable pivot
         [ct,i]=min(ctemp(neg));
         if ct>=-eps1,   % no suitable pivot columns are left
              disp(['a suitable pivot element cannot be found'])
          disp(['probable cause: roundoff error or ill-cond prob'])
              disp(['equilibrate problem before solving'])
              stop=0;
              return
         end
         t=neg(i);       % index of the most neg reduced cost
%  Now, let x sub t enter the basis
```

```
%
%  First, we need to find the variable which leaves the basis
        pos=[];
        ind=[];
        for i=1:m,
            if a(i,t)>eps0,
                ind=[ind i]; % suitable rows
            end
        end
        if length(ind)==0,
            disp(['The problem is unbounded '])
            stop=0;
            return
        end
        [alpha,i]=min(b(ind)./a(ind,t));
```

```
        i=ind(i); % pivot row
        if a(i,t)>eps2, % a suitable pivot element is found
            ct=0;
        else
            ctemp(t)=0;    % column t is unsuitable pivot col.
        end
       end
       if stop==0,
        return  % Ensure that we return
       end
%  Update the basic and nonbasic vectors.
        nbas(nbas==t)=bas(i);
        bas(i)=t;
        alpha=a(i,t);   % pivot element
%  Store the data in ap,bp
```

```
        ap=a;
        bp=b;
%  Now pivot by row
      iter=iter+1;

      if pt > 20,
         disp(['Hit return to continue'])
         pause
      else
         pause(pt)
      end
%
        for k=1:m,
            ratio=ap(k,t)/ap(i,t);
            a(k,:)=ap(k,:)-ap(i,:)*ratio;
```

```
            b(k)=bp(k)-bp(i)*ratio;

        end
%  Now for the objective row update
        ratio=c(t)/ap(i,t);
        c=c-ap(i,:)*ratio;
        z=z-bp(i)*ratio;

%  Now for the pivot row update
        a(i,:)=ap(i,:)/ap(i,t);
        b(i)=bp(i)/ap(i,t);
         clc
         home,disp([blanks(30)]),
        disp(['pivot= a(' int2str(i)+1 ',' int2str(t) ')' blanks(10)])
       disp(['after pivoting                '])
```

```
        if minmax==1
                tc=-c;
                tz=-z;
                [tc tz a b]
        else
                [c z a b]
        end

if pt <= 20,
pause(pt)
else
disp(['Hit return to continue'])
pause
end
```

```
    end
%
end
if iter>=100,
        text='Iteration bound has been exceeded ^^^ '
end
```

LP Example Using Matlab Program (sim.m)

Mathematical Formulation

Maximize $Z = 300x_1 + 500x_2$

subject to: $3x_1 + 2x_2 \leq 180$

$x_1 \leq 40$

$x_2 \leq 60$

$x_1 \geq 0$ and $x_2 \geq 0$

Mathematical Formulation (adding slacks)

Maximize $Z = 300x_1 + 500x_2$

subject to: $3x_1 + 2x_2 + x_3 = 180$

$x_1 + x_4 = 40$

$x_2 + x_5 = 60$

$x_1 \geq 0$ and $x_2 \geq 0$

Osaka Bay Example Using Matlab

```
% Example: Enter the data:
  minmax=0;
  a=[3 2 1 0 0
      1 0 0 1 0
      0 1 0 0 1 ]
  b=[180 40 60]'
  c=[-300 -500 0 0 0]
  bas=[3 4 5]
```

Note: 3 slack variables added (minmax = 0 denotes maximization)

Osaka Bay Example (Matlab Output)

```
a =
     3     2     1     0     0
     1     0     0     1     0
     0     1     0     0     1

b =
   180
    40
    60
c =
  -300  -500     0     0     0

bas =
     3     4     5
If the input data is correct, hit return to continue
```

Matlab (Output)

standard or canonical form of this problem

```
ans =
  -300  -500     0     0     0     0
     3     2     1     0     0   180
     1     0     0     1     0    40
     0     1     0     0     1    60
pivot= a(4,2)
after pivoting

ans =
        -300           0           0           0         500       30000
           3           0           1           0          -2          60
           1           0           0           1           0          40
           0           1           0           0           1          60
```

Osaka Bay Problem - Matlab (Output)

```
pivot= a(2,1)
after pivoting
ans =
   1.0e+04 *

        0         0    0.0100         0    0.0300    3.6000
   0.0001         0    0.0000         0   -0.0001    0.0020
        0         0   -0.0000    0.0001    0.0001    0.0020
        0    0.0001         0         0    0.0001    0.0060

This phase is completed   - current basis is:
bas =
     1     4     2
```

Osaka Bay Problem - Matlab (Output)

The current basic variable values are :

b =

20
20
60

The current objective value is:

ans =

36000

The number of iterations is 2

Osaka Bay Problem - Matlab (Output)

Final tableau in this phase

```
ans =
   1.0e+04 *

        0         0    0.0100         0    0.0300    3.6000
   0.0001         0    0.0000         0   -0.0001    0.0020
        0         0   -0.0000    0.0001    0.0001    0.0020
        0    0.0001         0         0    0.0001    0.0060
»
```