SIT718 Real World Analytics

Lecturer: Dr Ye Zhu

School of Information Technology
Deakin University

Week 7: Introduction to linear programming

Recommended Textbooks

Recommended Textbooks

1. *Operations Research: Applications and Algorithms* by **Wayne L. Winston**
2. *Operations Research: An Introduction* by **Hamdy A. Taha**

Linear Programming

Linear programming is a method to achieve the best outcome (such as maximum profit or lowest cost) in a mathematical model whose requirements are represented by linear relationships.

- Can be used to support and improve managerial decision making.
- Maximize or minimize some function, called the objective function, and have a set of restrictions known as constraints.
- Can be linear or nonlinear

Typical Applications

- A manufacturer wants to develop a production schedule and an inventory policy that will satisfy demand in future periods and at the same time minimize the total production and inventory costs.
- A financial analyst would like to establish an investment portfolio from a variety of stock and bond investment alternatives that maximizes the return on investment.
- A marketing manager wants to determine how best to allocate a fixed advertising budget among alternative advertising media such as web, radio, television, newspaper, and magazine that maximizes advertising effectiveness.
- A company had warehouses in a number of locations. Given specific customer demands, the company would like to determine how much each warehouse should ship to each customer so that total transportation costs are minimized.
- Federal Emergency Management Agency used a stochastic optimisation for ventilator allocation to combat COVID-19

Linear Programming – Modelling with 2 variables

A Linear Programming model typically contains:

- **Decision Variables** (x, y) that are either ≥ 0 or unrestricted in sign (i.e., they can be negative as well);
  - (N.B. it is important that you define your variables clearly and carefully).

LINEAR PROGRAMMING – MODELLING WITH 2 VARIABLES

A Linear Programming model typically contains:

- **Decision Variables** (x, y) that are either ≥ 0 or unrestricted in sign (i.e., they can be negative as well);
  - (N.B. it is important that you define your variables clearly and carefully).
- An **objective function** (that is linear in terms of x and y), e.g., $\min 2x + 3y$ or $\max 4x - y$;
  - to minimize cost, or to maximize profit.

Linear Programming – Modelling with 2 variables

A Linear Programming model typically contains:

- **Decision Variables** (x, y) that are either ≥ 0 or unrestricted in sign (i.e., they can be negative as well);
  - (N.B. it is important that you define your variables clearly and carefully).
- An **objective function** (that is linear in terms of x and y), e.g., $\min 2x + 3y$ or $\max 4x - y$;
  - to minimize cost, or to maximize profit.
- Subject to: **constraints** (for modelling restrictions), e.g., $2x + y \leq 10$ or $-x + 3y \geq 6$,
  - resource/budget constraints, demand requirements, etc.

An Example: Mak & Hau Toy Company

Mak & Hau produces two types of toys: soldiers and trains.
[Example modified from Winston, Sec 3.1]
Each **soldier** built:

- Sells for $27 and uses $10 worth of raw materials.
- It costs $14 for labor.
- Requires 2 hours of finishing labour.
- Requires 1 hour of carpentry labour.

An Example: Mak & Hau Toy Company

Mak & Hau produces two types of toys: soldiers and trains.
[Example modified from Winston, Sec 3.1]
Each **soldier** built:

- Sells for $27 and uses $10 worth of raw materials.
- It costs $14 for labor.
- Requires 2 hours of finishing labour.
- Requires 1 hour of carpentry labour.

Each **train** built:

- Sells for $21 and uses $9 worth of raw materials.
- It costs $10 for labor.
- Requires 1 hours of finishing labour.
- Requires 1 hour of carpentry labour.

Resource and Demand Constraints

- Weekly resources available
  - All needed raw material
  - 100 hour of available finishing labour
  - 80 hours of available carpentry labour

Resource and Demand Constraints

- Weekly resources available
  - All needed raw material
  - 100 hour of available finishing labour
  - 80 hours of available carpentry labour
- Demand constraints
  - Unlimited demand for trains
  - At most 40 soldiers are bought each week

(NB: The example on CloudDeakin 7.5 is “at least 40 soldiers”)

An LP model for Mak & Hau's-The objective function

Work out a weekly plan to determine how many soldiers and trains Mak & Hau should produce so as to maximize profit.

An LP model for Mak & Hau's-The objective function

Work out a weekly plan to determine how many soldiers and trains Mak & Hau should produce so as to maximize profit.

- Let x_1 be the number of soldiers to be produced each week

An LP model for Mak & Hau's-The objective function

Work out a weekly plan to determine how many soldiers and trains Mak & Hau should produce so as to maximize profit.

- Let x_1 be the number of soldiers to be produced each week
- Let x_2 be the number of trains to be produced each week

An LP model for Mak & Hau's-The objective function

Work out a weekly plan to determine how many soldiers and trains Mak & Hau should produce so as to maximize profit.

- Let x_1 be the number of soldiers to be produced each week
- Let x_2 be the number of trains to be produced each week

Weekly profit $=$ weekly revenue $-$ weekly raw material costs $-$ labour cost

An LP model for Mak & Hau's-The objective function

Work out a weekly plan to determine how many soldiers and trains Mak & Hau should produce so as to maximize profit.

- Let x_1 be the number of soldiers to be produced each week
- Let x_2 be the number of trains to be produced each week

Weekly profit = weekly revenue − weekly raw material costs − labour cost

$$\text{Weekly profit} = (27x_1 + 21x_2) - (10x_1 + 9x_2) - (14x_1 + 10x_2)$$

An LP model for Mak & Hau's-The objective function

Work out a weekly plan to determine how many soldiers and trains Mak & Hau should produce so as to maximize profit.

- Let x_1 be the number of soldiers to be produced each week
- Let x_2 be the number of trains to be produced each week

Weekly profit = weekly revenue − weekly raw material costs − labour cost

Weekly profit $= (27x_1 + 21x_2) - (10x_1 + 9x_2) - (14x_1 + 10x_2)$

Objective function: $z = \max(27 - 10 - 14)x_1 + (21 - 9 - 10)x_2$
$= \max\ 3x_1 + 2x_2$

An LP model for Mak & Hau's-The objective function

- Each week, no more than 100 hours of finishing time may be used.

An LP model for Mak & Hau's-The objective function

- Each week, no more than 100 hours of finishing time may be used.

$$2x_1 + x_2 \leq 100$$

An LP model for Mak & Hau's-The objective function

- Each week, no more than 100 hours of finishing time may be used.

$$2x_1 + x_2 \leq 100$$

- Each week, no more than 80 hours of **carpentry** time may be used.

An LP model for Mak & Hau's-The objective function

- Each week, no more than 100 hours of finishing time may be used.

$$2x_1 + x_2 \leq 100$$

- Each week, no more than 80 hours of **carpentry** time may be used.

$$x_1 + x_2 \leq 80$$

An LP model for Mak & Hau's-The objective function

- Each week, no more than 100 hours of finishing time may be used.

$$2x_1 + x_2 \leq 100$$

- Each week, no more than 80 hours of **carpentry** time may be used.

$$x_1 + x_2 \leq 80$$

- A maximum weekly demand of 40 soldiers

An LP model for Mak & Hau's-The objective function

- Each week, no more than 100 hours of finishing time may be used.

$$2x_1 + x_2 \leq 100$$

- Each week, no more than 80 hours of **carpentry** time may be used.

$$x_1 + x_2 \leq 80$$

- A maximum weekly demand of 40 soldiers

$$x_1 \leq 40$$

An LP model for Mak & Hau's-The objective function

- Each week, no more than 100 hours of finishing time may be used.

$$2x_1 + x_2 \leq 100$$

- Each week, no more than 80 hours of **carpentry** time may be used.

$$x_1 + x_2 \leq 80$$

- A maximum weekly demand of 40 soldiers

$$x_1 \leq 40$$

- Non-negativity constraints

An LP model for Mak & Hau's-The objective function

- Each week, no more than 100 hours of finishing time may be used.

$$2x_1 + x_2 \leq 100$$

- Each week, no more than 80 hours of carpentry time may be used.

$$x_1 + x_2 \leq 80$$

- A maximum weekly demand of 40 soldiers

$$x_1 \leq 40$$

- Non-negativity constraints

$$x_1, x_2 \geq 0$$

The Full Model

$$\begin{aligned} \max\ z = 3x_1 + 2x_2 & \\ \text{s.t. } 2x_1 + x_2 & \leq 100 \\ x_1 + x_2 & \leq 80 \\ x_1 & \leq 40 \\ x_1 & \geq 0 \\ x_2 & \geq 0 \end{aligned}$$

Notice that we have $x_1,\ x_2 \geq 0$ as you can't exactly produce negative 3 soldiers or negative 2 trains, can you :-)?

Feasible Solutions Vs. Optimal Solution(s)

A **feasible** solution is one that satisfy all constraints.
So, $x_1 = x_2 = 0$ is a feasible solution (though it is a very silly solution); and

Feasible Solutions Vs. Optimal Solution(s)

A **feasible** solution is one that satisfy all constraints.
So, $x_1 = x_2 = 0$ is a feasible solution (though it is a very silly solution); and
$x_1 = 10$, $x_2 = 20$ is also a feasible solution (and this one is better).

Feasible Solutions Vs. Optimal Solution(s)

A **feasible** solution is one that satisfy all constraints.
So, $x_1 = x_2 = 0$ is a feasible solution (though it is a very silly solution); and
$x_1 = 10$, $x_2 = 20$ is also a feasible solution (and this one is better).

An **infeasible** solution is one that violates one or more of the constraints. E.g., $x_1 = 60$ and $x_2 = 100$ is an infeasible solution.

Feasible Solutions Vs. Optimal Solution(s)

A **feasible** solution is one that satisfy all constraints.
So, $x_1 = x_2 = 0$ is a feasible solution (though it is a very silly solution); and
$x_1 = 10$, $x_2 = 20$ is also a feasible solution (and this one is better).

An **infeasible** solution is one that violates one or more of the constraints. E.g., $x_1 = 60$ and $x_2 = 100$ is an infeasible solution.

An **optimal** solution is a feasible solution that maximizes the profit.

Feasible Solutions Vs. Optimal Solution(s)

A **feasible** solution is one that satisfy all constraints.
So, $x_1 = x_2 = 0$ is a feasible solution (though it is a very silly solution); and
$x_1 = 10$, $x_2 = 20$ is also a feasible solution (and this one is better).

An **infeasible** solution is one that violates one or more of the constraints. E.g., $x_1 = 60$ and $x_2 = 100$ is an infeasible solution.

An **optimal** solution is a feasible solution that maximizes the profit.

N.B.: There may be multiple optimal solutions in a linear program.

Graphical Method

Graphing Linear Equations

The graph of a linear equation in two variables is a line (that's why they call it **linear**).

If you know an equation is linear, you can graph it by finding any two solutions

(x_1, y_1) and (x_2, y_2) ,

plotting these two points, and drawing the line connecting them.

Example 1:

Graph the equation $x + 2y = 7$.

You can find two solutions, corresponding to the x -intercepts and y -intercepts of the graph, by setting first $x = 0$ and then $y = 0$.

https://www.varsitytutors.com/hotmath/hotmath_help/topics/graphing-linear-equations

Graphical Method (cont.)

Plot these two points and draw the line connecting them.

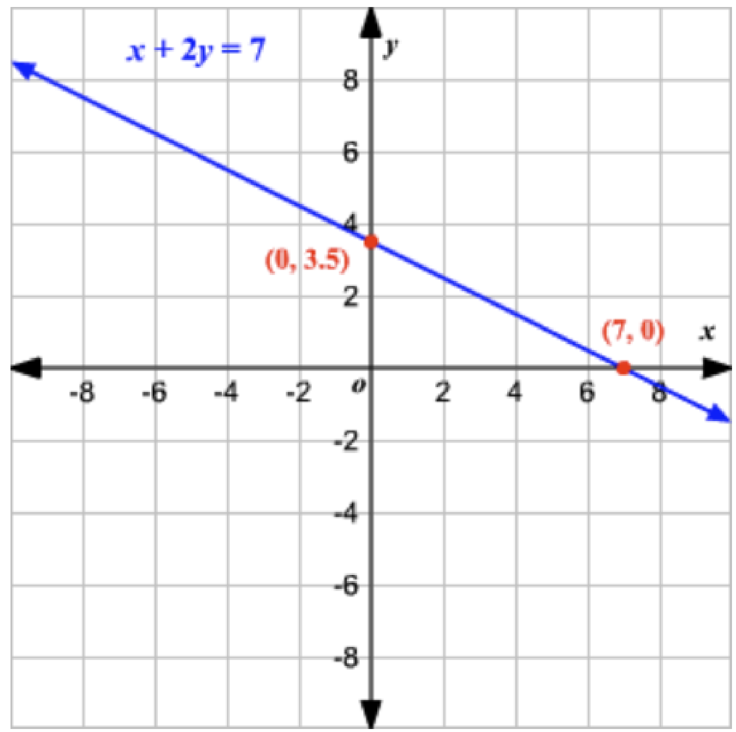
When $x = 0$, we get:

$$0 + 2y = 7$$
$$y = 3.5$$

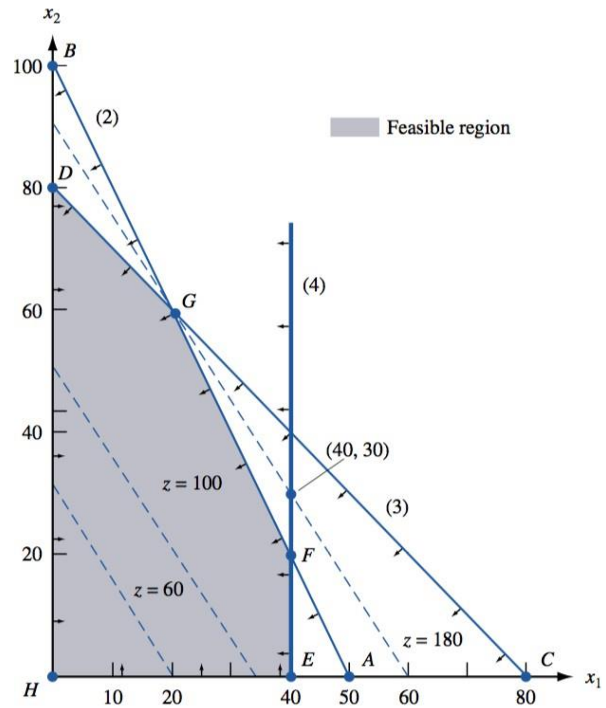
When $y = 0$, we get:

$$x + 2(0) = 7$$
$$x = 7$$

So the two points are $(0, 3.5)$ and $(7, 0)$.

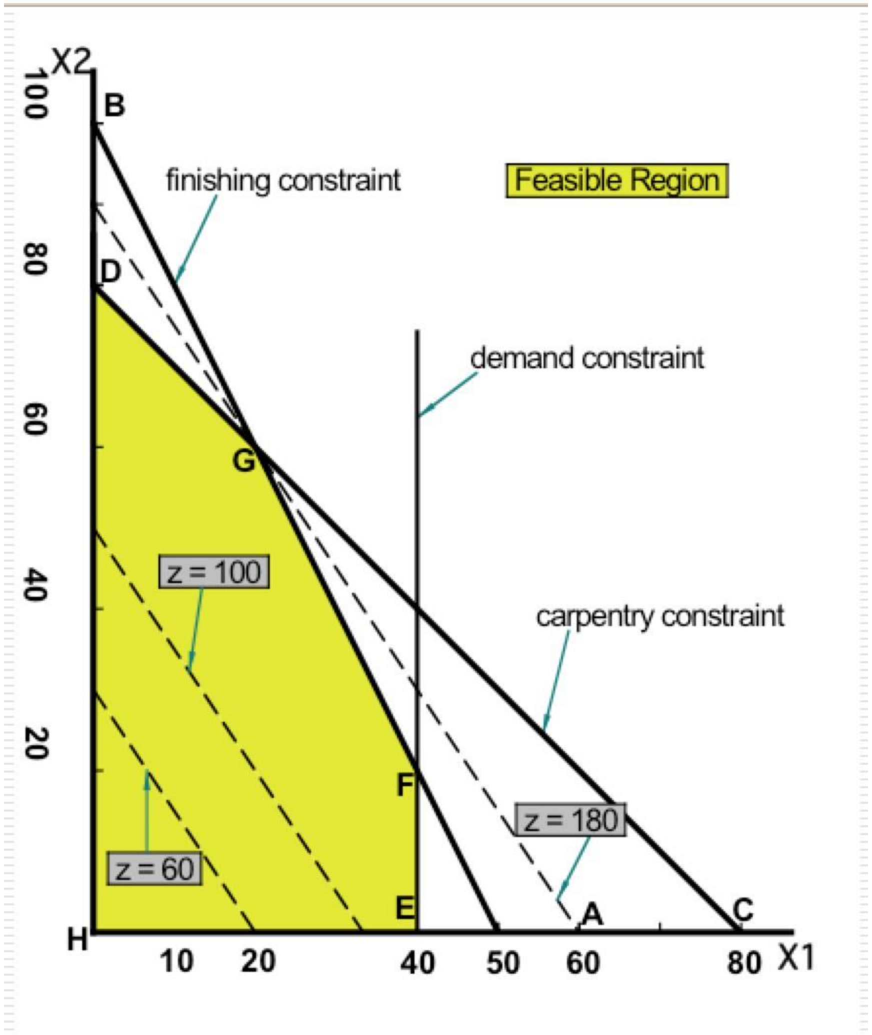


Graphical Method (cont.)



Refer to lecture recordings for the construction of the feasible region and the methodology for deducing the **iso-cost** or **iso-profit** lines, and the **optimal solution**. (An iso-cost or iso-profit line is a line with which the points give the same objective value).

Graphical Solution Toy Problem



$$\max z = 3x_1 + 2x_2$$

$2x_1 + x_2 \leq 100$ (finishing constraint)

$x_1 + x_2 \leq 80$ (carpentry constraint)

$x_1 \leq 40$ (demand constraint)

$x_1, x_2 \geq 0$ (sign restriction)

A Minimization Problem

[Modified from **Taha**] An assembly line consisting of three consecutive stations produces two smart phones. Smart-1 and Smart-2. The assembly times of the three workstations are listed below.

	Time (mins) required per unit at workstation	
Workstation	Smart-1	Smart-2
1	3	6
2	5	5
3	4	8

A Minimization Problem

[Modified from **Taha**] An assembly line consisting of three consecutive stations produces two smart phones. Smart-1 and Smart-2. The assembly times of the three workstations are listed below.

	Time (mins) required per unit at workstation	
Workstation	Smart-1	Smart-2
1	3	6
2	5	5
3	4	8

Each station can operate up to 600 minutes per day. The estimated daily maintenance times for Stations 1, 2, and 3 are 10%, 25%, and 20%, respectively.

A Minimization Problem

[Modified from **Taha**] An assembly line consisting of three consecutive stations produces two smart phones. Smart-1 and Smart-2. The assembly times of the three workstations are listed below.

	Time (mins) required per unit at workstation	
Workstation	Smart-1	Smart-2
1	3	6
2	5	5
3	4	8

Each station can operate up to 600 minutes per day. The estimated daily maintenance times for Stations 1, 2, and 3 are 10%, 25%, and 20%, respectively.
Optimize the production plan (i.e. find the numbers of each product to be produced) such that the total idle time in the 3 workstations is minimized.

A Minimization Problem–The LP model

Let x_1 be the number of Smart-1 to be produced, and x_2 be the number of Smart-2 to be produced.

A Minimization Problem–The LP model

Let x_1 be the number of Smart-1 to be produced, and x_2 be the number of Smart-2 to be produced.

The maximum number of hours available for Workstations 1, 2, and 3 are:

$600 \times (1 - 10\%) = 540$ mins,

$600 \times (1 - 25\%) = 450$ mins, and

$600 \times (1 - 20\%) = 480$ mins; (Total: 1470 mins for 3 stations).

A Minimization Problem–The LP model

Let x_1 be the number of Smart-1 to be produced, and x_2 be the number of Smart-2 to be produced.
The maximum number of hours available for Workstations 1, 2, and 3 are:
$600 \times (1 - 10\%) = 540$ mins,
$600 \times (1 - 25\%) = 450$ mins, and
$600 \times (1 - 20\%) = 480$ mins; (Total: 1470 mins for 3 stations).

$$
\begin{aligned}
\min z = 1470 - 12x_1 - 19x_2 \quad & \equiv (\max 12x_1 + 19x_2) \\
\text{s.t.} \quad 3x_1 + 6x_2 \quad & \leq 540 \\
5x_1 + 5x_2 \quad & \leq 450 \\
4x_1 + 8x_2 \quad & \leq 480 \\
x_1, \ x_2 \quad & \geq 0
\end{aligned}
$$

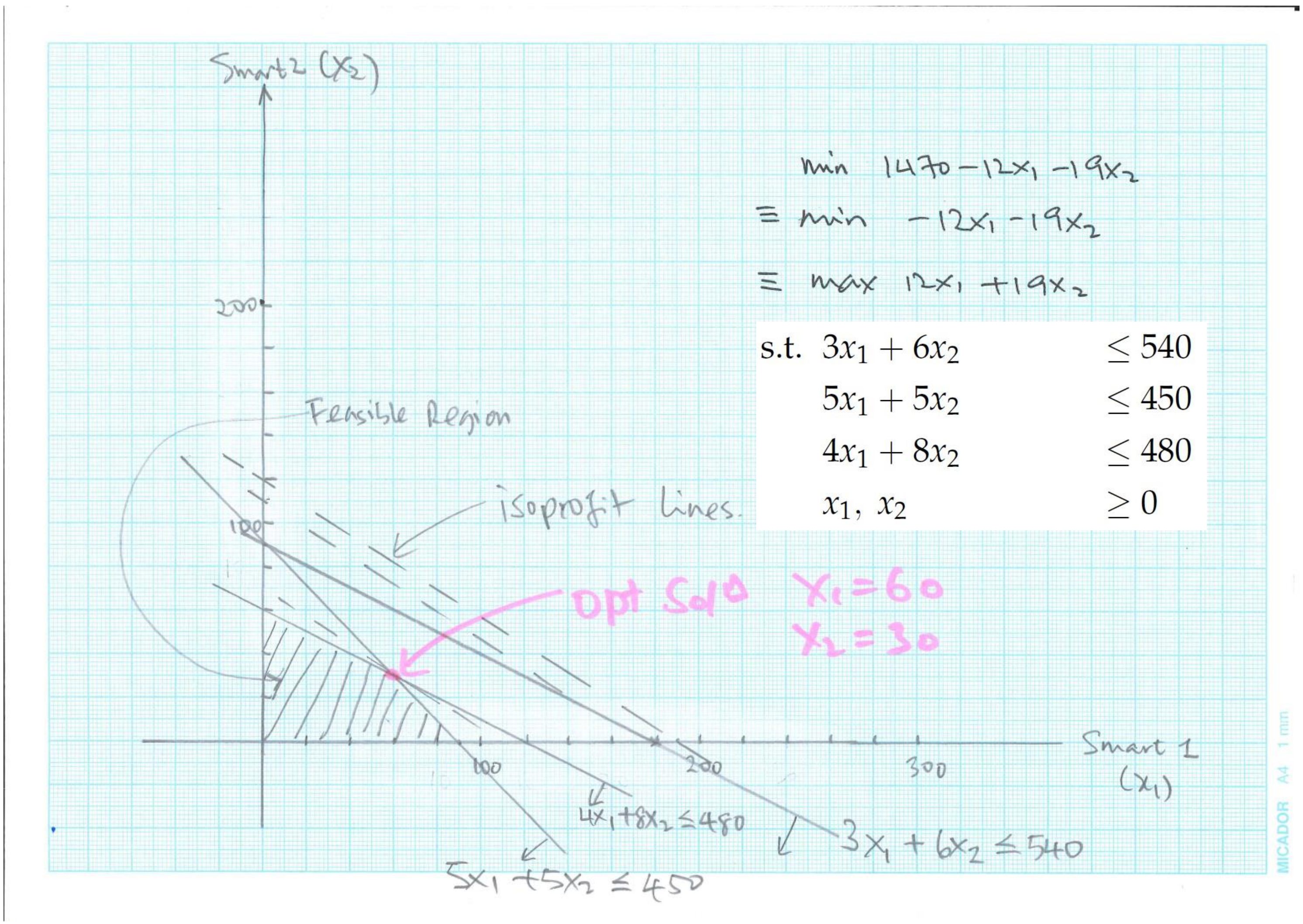
A Minimization Problem–The LP model

Let x_1 be the number of Smart-1 to be produced, and x_2 be the number of Smart-2 to be produced.
The maximum number of hours available for Workstations 1, 2, and 3 are:
$600 \times (1 - 10\%) = 540$ mins,
$600 \times (1 - 25\%) = 450$ mins, and
$600 \times (1 - 20\%) = 480$ mins; (Total: 1470 mins for 3 stations).

$$
\begin{aligned}
\min z = 1470 - 12x_1 - 19x_2 & \qquad \equiv (\max 12x_1 + 19x_2) \\
\text{s.t.}\quad 3x_1 + 6x_2 & \leq 540 \\
5x_1 + 5x_2 & \leq 450 \\
4x_1 + 8x_2 & \leq 480 \\
x_1,\ x_2 & \geq 0
\end{aligned}
$$

The optimal solution is: $x_1 = 60$, $x_2 = 30$, and $z = 180$ idle time.

A MINIMIZATION PROBLEM–GRAPHICAL SOLUTION



Sensitivity Analysis

- **Sensitivity analysis**: The study of how the changes in the input parameters of an optimization model affect the optimal solution.
- It helps in answering the questions:
- How will a change in a coefficient of the objective function affect the optimal solution?
- How will a change in the right-hand-side value for a constraint affect the optimal solution?
- Because sensitivity analysis (often referred to as postoptimality analysis) is concerned with how these changes affect the optimal solution, the analysis does not begin until the optimal solution to the original linear programming problem has been obtained.

Sensitivity Analysis (cont.)

Classical sensitivity analysis:

- Based on the assumption that only one piece of input data has changed.
- It is assumed that all other parameters remain as stated in the original problem.
- When interested in what would happen if two or more pieces of input data are changed simultaneously:
- The easiest way to examine the effect of simultaneous changes is to make the changes and rerun the model.

An Example: Mak & Hau Toy Company...

- Let us first study this graphically by revisiting the Mak & Hau's problem.

$$\max z = 3x_1 + 2x_2$$

$$2x_1 + x_2 \leq 100 \quad \text{(finishing constraint)}$$

$$x_1 + x_2 \leq 80 \quad \text{(carpentry constraint)}$$

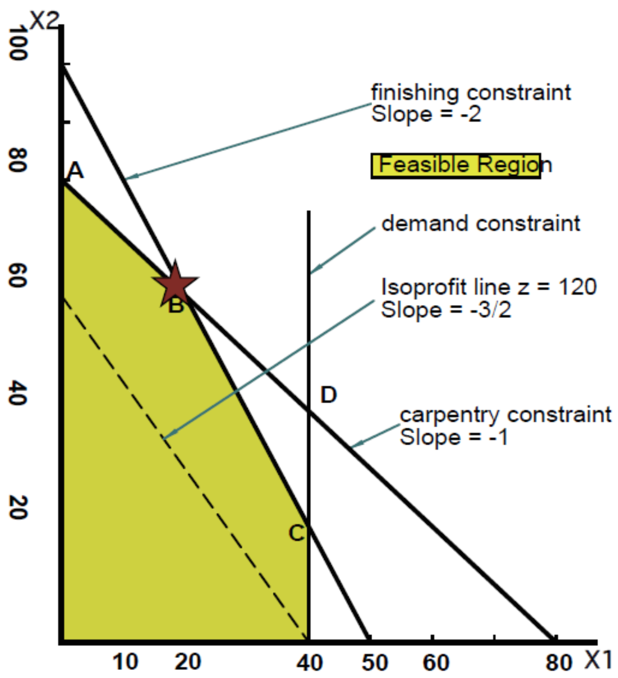
$$x_1 \leq 40 \quad \text{(demand constraint)}$$

$$x_1, x_2 \geq 0 \quad \text{(sign restriction)}$$

Where:

- x_1 = number of soldiers to be produced each week
- x_2 = number of trains to be produced each week.

Changes in objective function

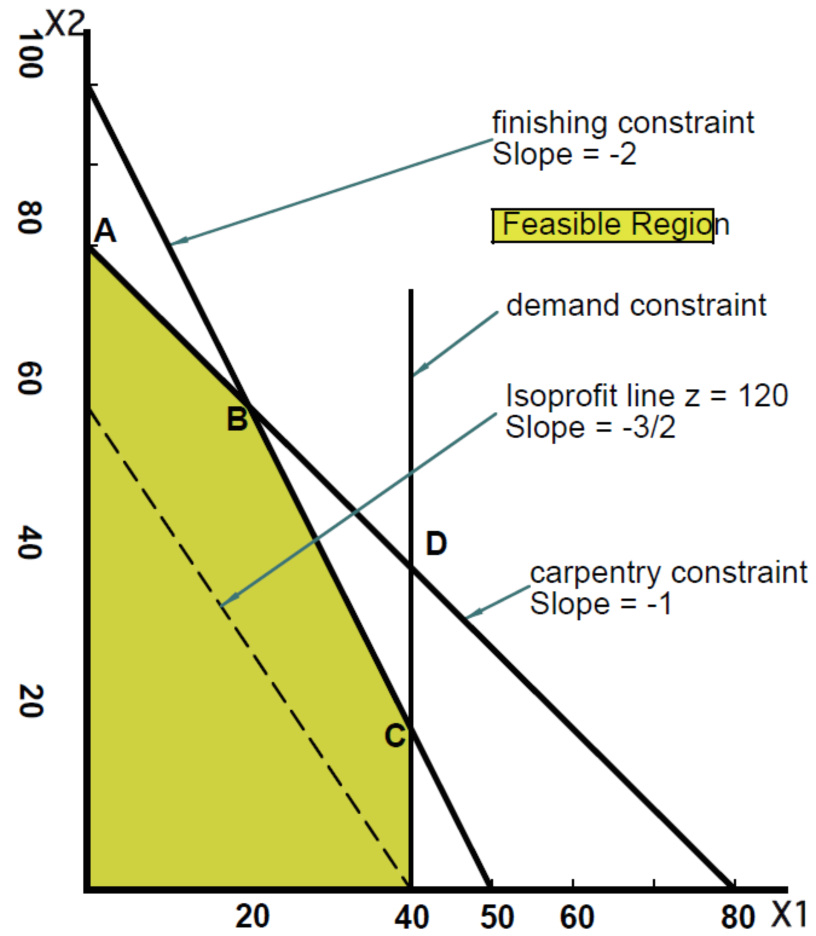


- The optimal solution for this LP was z = 180, x_1=20, x_2= 60 (at Point B) and it has x_1, x_2, and s_3 (the slack variable for the demand constraint) as basic variables.
- How would changes in the problem's objective function coefficients affect the optimal solution?

- By inspection, we can see that the finishing constraint has a slope of -2. If we make the slope of the isoprofit line <-2 (i.e. more negative than -2), the optimal point will move from B (20,60) to C (40,20).

- Similarly, the carpentry constraint has a slope of -1. If we make the slope of the isoprofit line > -1 (i.e.less negative than -1), then the optimal point will move from B (20,60) to A (0,80).

- So, if the slope of the isoprofit line is kept between -2 and -1 (i.e. -2 ≤ slope ≤ -1), the current optimal solution remains optimal.

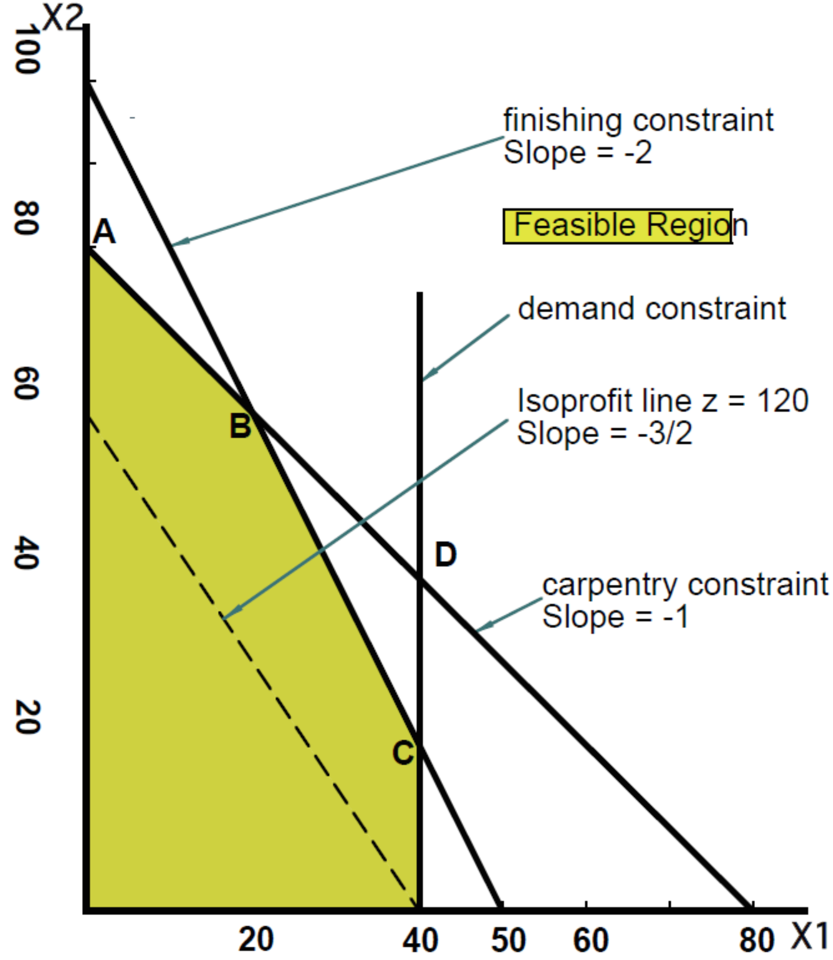


Let's analyze the effect of a change in c_1.
As: $c_1x_1 + 2x_2 = z$, i.e., $x_2 = (z - c_1x_1)/2$
the slope of the isoprofit line is $-c_1/2$.

So, as long as $-c_1/2$ is not more negative than -2 and
$-c_1/2$ is not less negative than -1, i.e.,
$-2 \leq -c_1/2 \leq -1$
the current solution is optimal.

In other words, as long as $2 \leq c_1 \leq 4$, the current solution is optimal.

Okay, so the optimal point won't change, but will the actual profit change? (Of course it will -- substitute the new c_1 into the objective function and you will find the new optimal objective value).



R for LP

7.5 - Toy Company Problem

```
# install the package in the first time
install.packages("lpSolveAPI")
library(lpSolveAPI)

# initialise 0 constaint and two variables
toyCompanyModel <- make.lp(0, 2) # two variables

# Set control parameters: "minimize" or "maximize"
lp.control(toyCompanyModel, sense= "maximize")

set.objfn(toyCompanyModel, c(3,2))  # max z=3s+2t

add.constraint(toyCompanyModel, c(2,1), "<=", 100)  # 2s + t<= 100
add.constraint(toyCompanyModel, c(1,1), "<=", 80)   # s + t <= 80

set.bounds(toyCompanyModel, lower = c(40,0), columns = c(1, 2))       # s >= 40
set.bounds(toyCompanyModel, upper = c(Inf,Inf), columns = c(1, 2))  # s , t >= 0

# Rename the rows and columns in the model
RowNames <- c("Constraint 1", "Constraint 2")
ColNames <- c("Soldiers", "Trains")
dimnames(toyCompanyModel) <- list(RowNames, ColNames)
```

7.5 - Toy Company Problem

```
> # Display the model
> toyCompanyModel
Model name:
                  Soldiers    Trains
Maximize                 3         2
Constraint 1             2         1   <=   100
Constraint 2             1         1   <=    80
Kind                   Std       Std
Type                  Real      Real
Upper                  Inf       Inf
Lower                   40         0
>
> # Solve the model
> solve(toyCompanyModel)
[1] 0
>
> # Retrieve the value of the objective function
> get.objective(toyCompanyModel)
[1] 160
>
> # Retrieve the values of the decision variables
> get.variables(toyCompanyModel)
[1] 40 20
>
> # Retrieve the values of the constraints
> get.constraints(toyCompanyModel)
[1] 100  60
```

Return Value

NOMEMORY (-2)	Out of memory
OPTIMAL (0)	An optimal solution was obtained
SUBOPTIMAL (1)	The model is sub-optimal. Only happens if there are integer variables and there is already an integer solution found. The solution is not guaranteed the most optimal one. • A timeout occured (set via set_timeout or with the -timeout option in lp_solve) • set_break_at_first was called so that the first found integer solution is found (-f option in lp_solve) • set_break_at_value was called so that when integer solution is found that is better than the specified value that it stops (-o option in lp_solve) • set_mip_gap was called (-g/-ga/-gr options in lp_solve) to specify a MIP gap • An abort function is installed (put_abortfunc) and this function returned TRUE • At some point not enough memory could not be allocated
INFEASIBLE (2)	The model is infeasible
UNBOUNDED (3)	The model is unbounded
DEGENERATE (4)	The model is degenerative
NUMFAILURE (5)	Numerical failure encountered
USERABORT (6)	The abort routine returned TRUE. See put_abortfunc
TIMEOUT (7)	A timeout occurred. A timeout was set via set_timeout
PRESOLVED (9)	The model could be solved by presolve. This can only happen if presolve is active via set_presolve
ACCURACYERROR (25)	Accuracy error encountered

7.6 - Assembly Line Problem

```
> library(lpSolveAPI)
> assemblyModel <- make.lp(0, 2) # two variables
> lp.control(assemblyModel, sense= "maximize")
> set.objfn(assemblyModel, c(12,19))
> add.constraint(assemblyModel, c(3,6), "<=", 540)
> add.constraint(assemblyModel, c(5,5), "<=", 450)
> add.constraint(assemblyModel, c(4,8), "<=", 480)
> set.bounds(assemblyModel, lower = c(0,0), columns = c(1, 2))
> set.bounds(assemblyModel, upper = c(Inf,Inf), columns = c(1, 2))
> RowNames <- c("Constraint 1", "Constraint 2", "Constraint 3")
> ColNames <- c("Soldiers", "Trains")
> dimnames(assemblyModel) <- list(RowNames, ColNames)
> solve(assemblyModel)
[1] 0
> assemblyModel
Model name:
                Soldiers    Trains
Maximize              12        19
Constraint 1           3         6  <=  540
Constraint 2           5         5  <=  450
Constraint 3           4         8  <=  480
Kind                 Std       Std
Type                Real      Real
Upper                Inf       Inf
Lower                  0         0
>
> # We need to convert the objective function to the original version
> 1470-get.objective(assemblyModel)
[1] 180
>
> get.variables(assemblyModel)
[1] 60 30
>
> get.constraints(assemblyModel)
[1] 360 450 480
```

$$\min z = 1470 - 12s - 19t$$

$$s.t. 3s + 6t \leq 540$$

$$5s + 5t \leq 450$$

$$4s + 8t \leq 480$$

$$s, t \geq 0$$