Linear Programming using R

Data Driven Decision Making

Muhammad Mattin Sharif

40418446

Contents

Problem-1: Customized Automotive Tech

Decision Variables

Let x_{ij} be a binary variable that equals 1 if product *j* is produced in plant *i*, and 0 otherwise.

$$x_{ij} = \begin{cases} 1, & \text{if product } j \text{ is produced in plant } i \\ 0, & \text{otherwise} \end{cases}$$

Objective Function

The goal is to maximize the total number of batches manufactured, which is given by the sum of the products of each plant's capacity and the decision variables.

$$\text{maximize } Z = \sum_{i=1}^{5} \sum_{j=1}^{4} C_{ij} * x_{ij}$$

Where $C_{ij}$is the capacity of the plant *i* for the product *j*, and double summation is over all plants *i* and all products *j.*

Product	1	2	3	4
Plant-1	1200	-	600	1000
Plant-2	1400	1200	800	1000
Plant-3	600	-	200	600
Plant-4	800	-	-	1200
Plant-5	800	1400	1000	1600

Plugging in the provided capacity values into the objective function, we get:

$$\begin{aligned} \text{Maximize } Z \ = \ & 1200x_{1,1} + 1400x_{2,1} + 600x_{3,1} + 800x_{4,1} + 800x_{5,1} + 1200x_{2,2} \\ & + 1400x_{5,2} + 600x_{1,3} + 800x_{2,3} + 200x_{3,3} + 1000x_{5,3} + 1000x_{1,4} \\ & + 1000x_{2,4} + 600x_{3,4} + 1200x_{4,4} + 1600x_{5,4} \end{aligned}$$

Constraints

Each product must be produced in only one plant.

(For product 1) : $x_{1,1} + x_{2,1} + x_{3,1} + x_{4,1} + x_{5,1} = 1$

(For product 2) : $x_{2,2} \qquad + x_{5,2} = 1$

(For product 3) : $x_{1,3} + x_{2,3} + x_{3,3} \qquad + x_{5,3} = 1$

(For product 4) : $x_{1,4} + x_{2,4} + x_{3,4} + x_{4,4} + x_{5,4} = 1$

Each plant can be scheduled for the production of at most one of the products.

(For plant 1) : $x_{1,1} + x_{1,2} + x_{1,3} + x_{1,4} \leq 1$

(For plant 2) : $x_{2,1} + x_{2,2} + x_{2,3} + x_{2,4} \leq 1$

(For plant 3) : $x_{3,1} + x_{3,2} + x_{3,3} + x_{3,4} \leq 1$

(For plant 4) : $x_{4,1} + x_{4,2} + x_{4,3} + x_{4,4} \leq 1$

Plants 1, 3, and 4 cannot manufacture product 2, and plant 4 cannot manufacture product 3.

$x_{1,2} = x_{3,2} = x_{4,2} = x_{4,3} = 0$

The decision variables are binary.

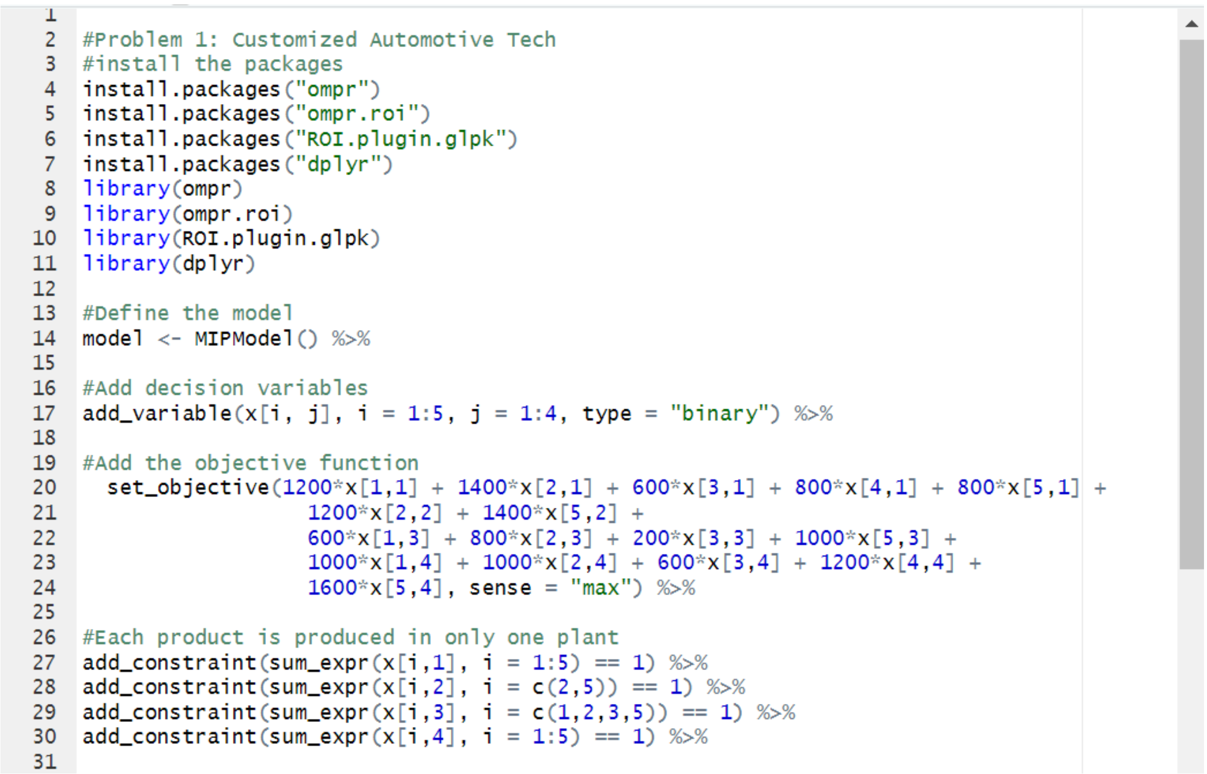
$x_{ij} \in \{0,1\}$ for all *i* and *j*.

Explanation

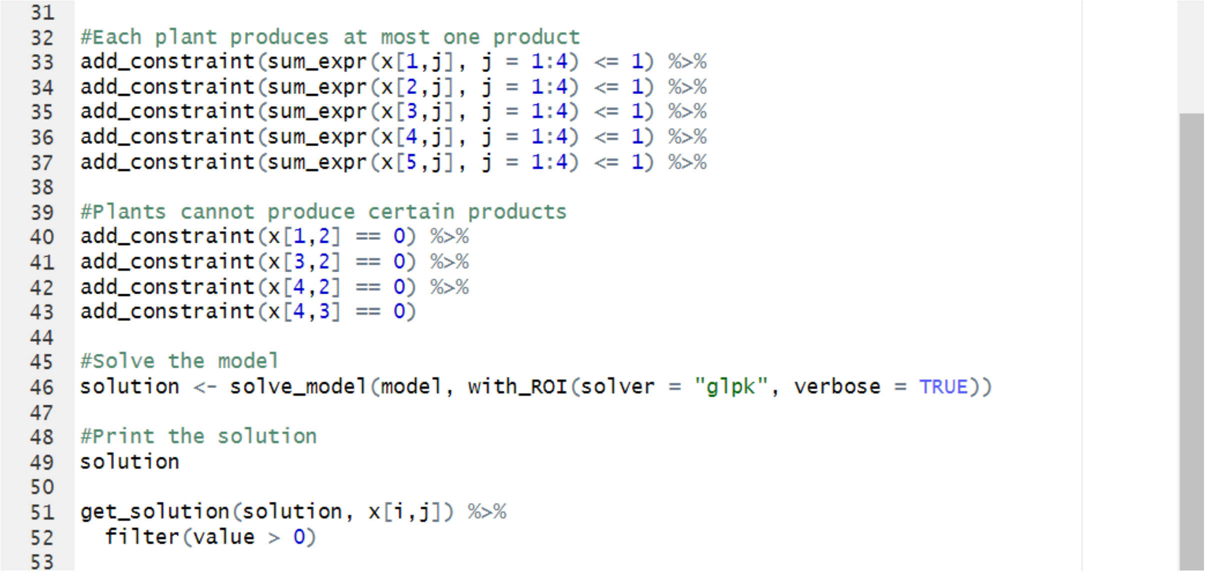
- The objective function corresponds to the total number of batches produced across all plants and products, as per the given capacity table.
- The constraints for each product ensure that each product is only made in one plant.
- The constraints for each plant ensure that each plant only makes at most one product.
- The specific constraints for plant-product combinations that are not allowed are set to 0, which represents these restrictions.
- The binary nature of the decision variables corresponds to the yes/no decision of whether a product is made in a certain plant.

This model is designed to maximize production while adhering to the limitations of plant capacities and production possibilities. The result of this model will tell us the optimal allocation of products to plants for maximum batch production.

R code:

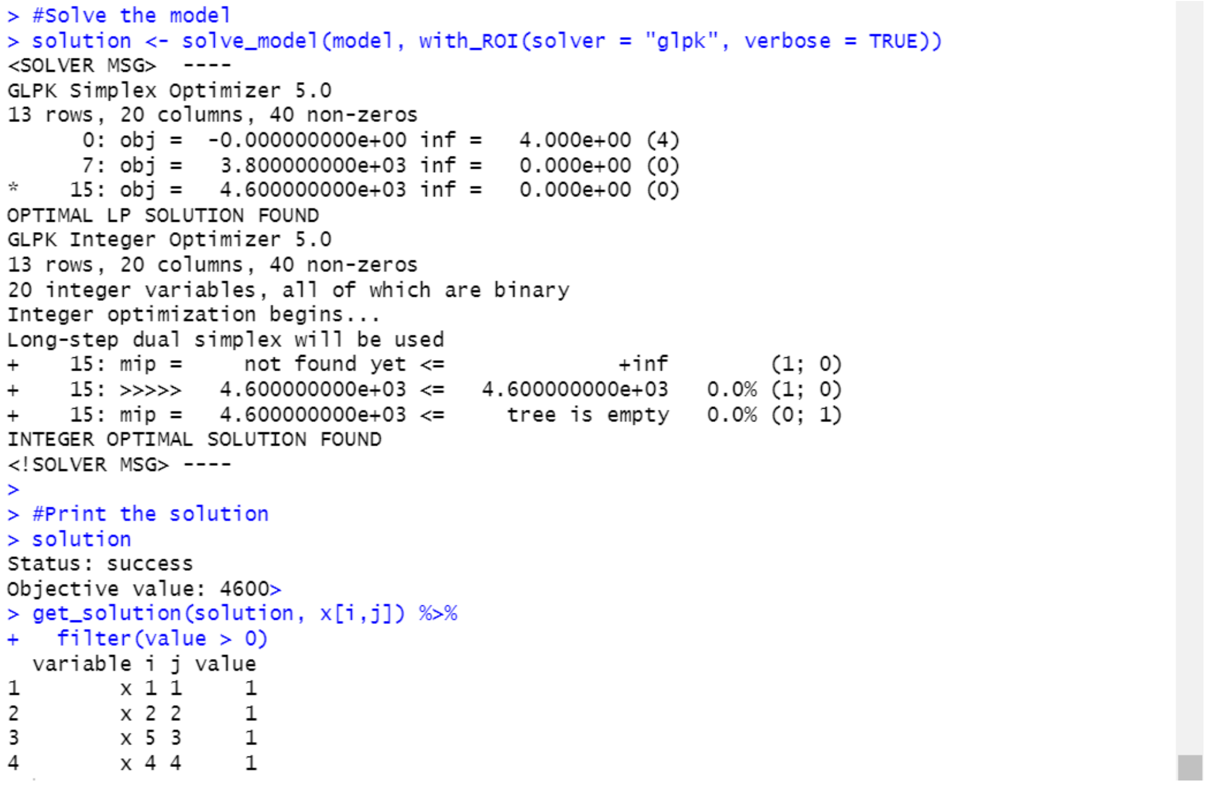


```

#Problem 1: Customized Automotive Tech
#install the packages
install.packages("ompr")
install.packages("ompr.roi")
install.packages("ROI.plugin.glpk")
install.packages("dplyr")
library(ompr)
library(ompr.roi)
library(ROI.plugin.glpk)
library(dplyr)

#Define the model
model <- MIPModel() %>%

#Add decision variables
add_variable(x[i, j], i = 1:5, j = 1:4, type = "binary") %>%

#Add the objective function
  set_objective(1200*x[1,1] + 1400*x[2,1] + 600*x[3,1] + 800*x[4,1] + 800*x[5,1] +
                1200*x[2,2] + 1400*x[5,2] +
                600*x[1,3] + 800*x[2,3] + 200*x[3,3] + 1000*x[5,3] +
                1000*x[1,4] + 1000*x[2,4] + 600*x[3,4] + 1200*x[4,4] +
                1600*x[5,4], sense = "max") %>%

#Each product is produced in only one plant
add_constraint(sum_expr(x[i,1], i = 1:5) == 1) %>%
add_constraint(sum_expr(x[i,2], i = c(2,5)) == 1) %>%
add_constraint(sum_expr(x[i,3], i = c(1,2,3,5)) == 1) %>%
add_constraint(sum_expr(x[i,4], i = 1:5) == 1) %>%

```



```

#Each plant produces at most one product
add_constraint(sum_expr(x[1,j], j = 1:4) <= 1) %>%
add_constraint(sum_expr(x[2,j], j = 1:4) <= 1) %>%
add_constraint(sum_expr(x[3,j], j = 1:4) <= 1) %>%
add_constraint(sum_expr(x[4,j], j = 1:4) <= 1) %>%
add_constraint(sum_expr(x[5,j], j = 1:4) <= 1) %>%

#Plants cannot produce certain products
add_constraint(x[1,2] == 0) %>%
add_constraint(x[3,2] == 0) %>%
add_constraint(x[4,2] == 0) %>%
add_constraint(x[4,3] == 0)

#Solve the model
solution <- solve_model(model, with_ROI(solver = "glpk", verbose = TRUE))

#Print the solution
solution

get_solution(solution, x[i,j]) %>%
  filter(value > 0)

```

R Results:



```
> #Solve the model
> solution <- solve_model(model, with_ROI(solver = "glpk", verbose = TRUE))
<SOLVER MSG>  ----
GLPK Simplex Optimizer 5.0
13 rows, 20 columns, 40 non-zeros
      0: obj =  -0.000000000e+00 inf =   4.000e+00 (4)
      7: obj =   3.800000000e+03 inf =   0.000e+00 (0)
*    15: obj =   4.600000000e+03 inf =   0.000e+00 (0)
OPTIMAL LP SOLUTION FOUND
GLPK Integer Optimizer 5.0
13 rows, 20 columns, 40 non-zeros
20 integer variables, all of which are binary
Integer optimization begins...
Long-step dual simplex will be used
+    15: mip =     not found yet <=              +inf        (1; 0)
+    15: >>>>>   4.600000000e+03 <=   4.600000000e+03   0.0% (1; 0)
+    15: mip =   4.600000000e+03 <=     tree is empty   0.0% (0; 1)
INTEGER OPTIMAL SOLUTION FOUND
<!SOLVER MSG> ----
>
> #Print the solution
> solution
Status: success
Objective value: 4600>
> get_solution(solution, x[i,j]) %>%
+   filter(value > 0)
  variable i j value
1        x 1 1     1
2        x 2 2     1
3        x 5 3     1
4        x 4 4     1
```

Managerial Results

The production manager can schedule the plants in such a way that the total production across all plants and products will be 4600 batches in one production period.

- Product 1 should be produced in Plant 1
- Product 2 should be produced in Plant 2
- Product 3 should be produced in Plant 5
- Product 4 should be produced in Plant 4

Problem-2: AI Chip

Decision Variables:

Let x_{ij} represent the number of million chips sold to customer *i* from factory *j*. Here, *i* ranges over the set {1, 2, 3, 4} corresponding to the four customers, and *j* is in {A, B} for the two factories.

- x_{1A}: Sold to customer 1 from factory A
- x_{2A}: Sold to customer 2 from factory A
- x_{3A}: Sold to customer 3 from factory A
- x_{4A}: Sold to customer 4 from factory A
- x_{1B}: Sold to customer 1 from factory B
- x_{2B}: Sold to customer 2 from factory B
- x_{3B}: Sold to customer 3 from factory B
- x_{4B}: Sold to customer 4 from factory B

Objective Function:

Our objective is to maximize the total profit (*Z*), which is calculated as the sum of individual profits from selling chips to each customer from each factory. The profit from each sale is the difference between the offered price and the sum of the production and delivery costs, all multiplied by the number of chips sold.

$$\text{maximize } z = \sum_{i=1}^{4} \sum_{j=A}^{B} \left(P_i - C_j - D_{ij}\right) * \ x_{ij}$$

Where P_i is the price offered by customer *i*, C_j is the production cost at factory *j*, and D_{ij} is the delivery cost from factory *j* to customer *i*.

Plugging in the provided values into the objective function, we get:

$$\begin{aligned} Maximize\ Z = (1950 - 1150 - 300)x_{1A} + (1950 - 1250 - 600)x_{1B} \\ + (1850 - 1150 - 400)x_{2A} + \cdots + (1800 - 1250 - 250)\ x_{4B} \end{aligned}$$

Customer	Profit from Fab-A (₰/chip)	Profit from Fab-B (₰/chip)
1	500	100
2	300	300
3	300	350
4	200	300

Constraints:

Demand Constraints:

Each customer's demand cannot be exceeded.

(For customer 1) : $x_{1A} + x_{1B} \leq 36$

(For customer 2) : $x_{2A} + x_{2B} \leq 46$

(For customer 3) : $x_{3A} + x_{3B} \leq 11$

(For customer 4) : $x_{4A} + x_{4B} \leq 35$

Capacity Constraints: Each factory's capacity cannot be exceeded.

(For factory A) : $x_{1A} + x_{2A} + x_{3A} + x_{4A} \leq 50$

(For factory A) : $x_{1B} + x_{2B} + x_{3B} + x_{4B} \leq 42$

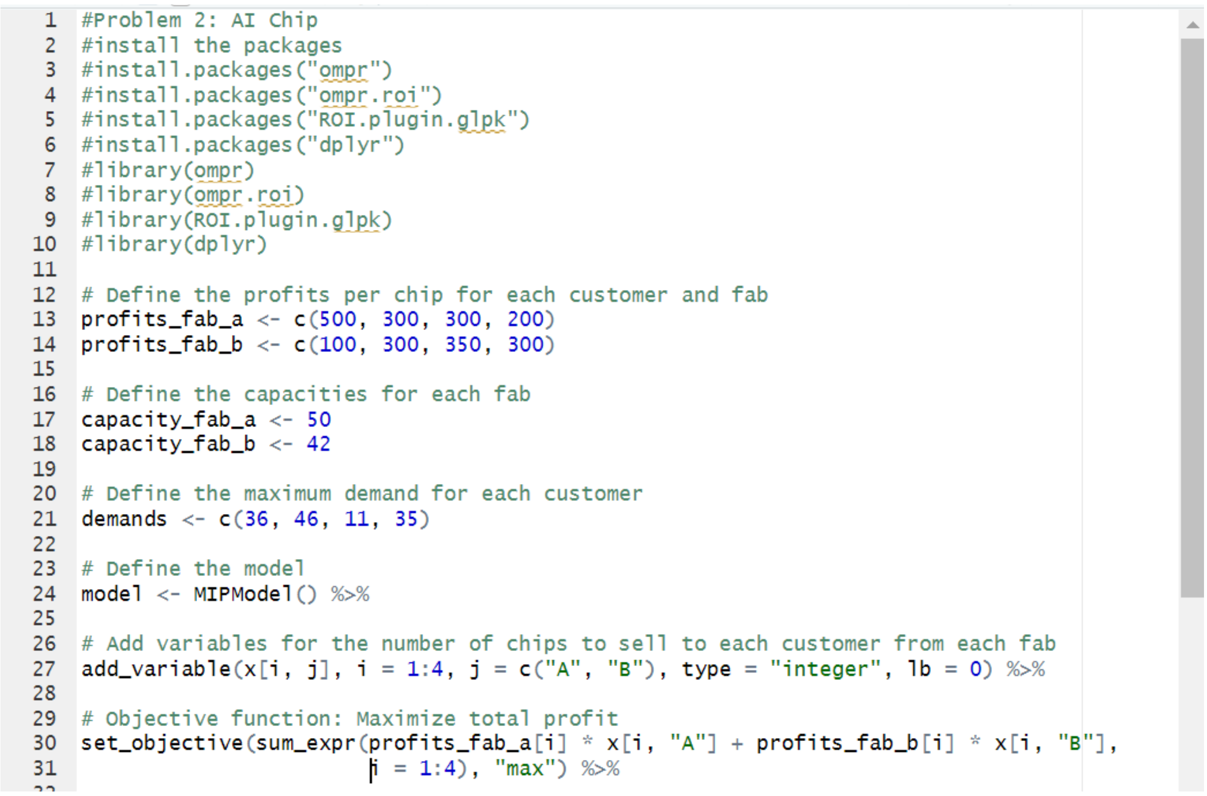
Non-Negativity Constraints: The number of chips sold must be non-negative.

$x_{ij} > 0$ for all *i* and *j*

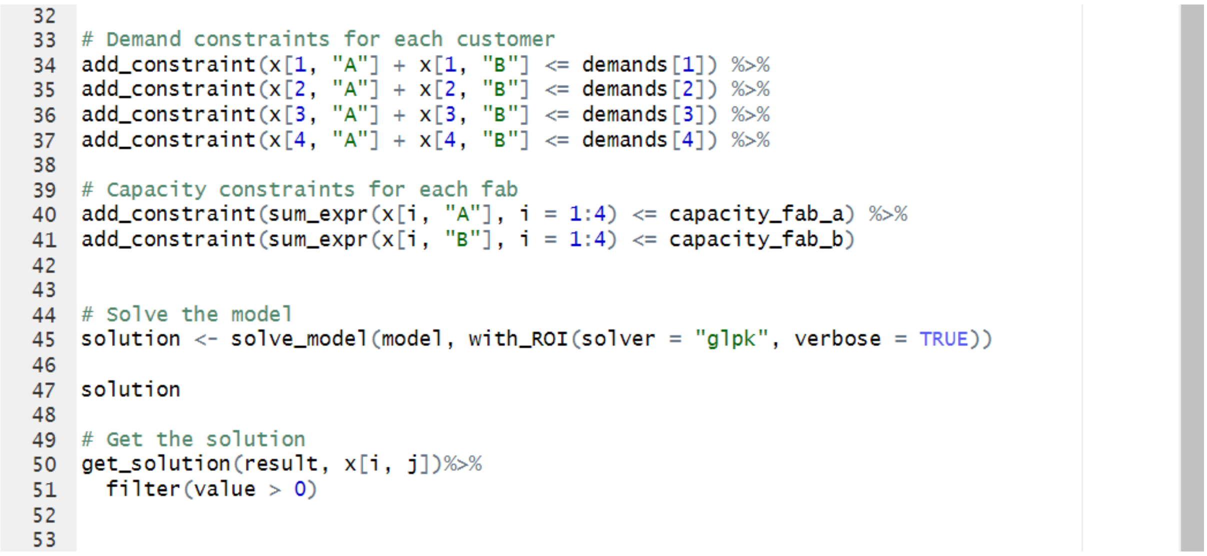
By solving this linear programming problem, we will be able to find out how many chips Teranikx should agree to sell to each customer to maximize total profit, as well as calculate what that maximum profit is.

The model includes the specific unit profit for each customer-factory combination after considering the respective production and delivery costs. The demand constraints ensure that sales do not exceed the customer's maximum demand, and the capacity constraints ensure that the factories do not produce more than their capacity. Solving this linear programming model will provide the optimal number of chips TeraNix should sell to each customer and the maximum total profit that can be achieved.

R code:

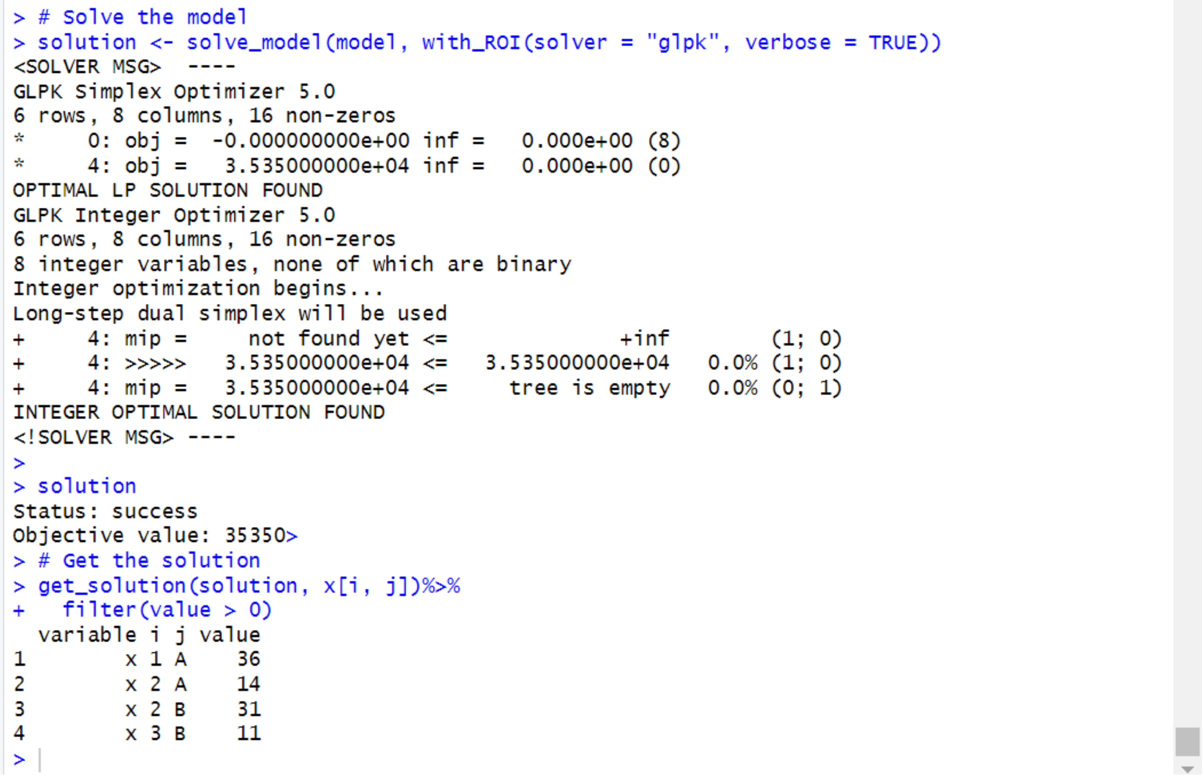


```r
#Problem 2: AI Chip
#install the packages
#install.packages("ompr")
#install.packages("ompr.roi")
#install.packages("ROI.plugin.glpk")
#install.packages("dplyr")
#library(ompr)
#library(ompr.roi)
#library(ROI.plugin.glpk)
#library(dplyr)

# Define the profits per chip for each customer and fab
profits_fab_a <- c(500, 300, 300, 200)
profits_fab_b <- c(100, 300, 350, 300)

# Define the capacities for each fab
capacity_fab_a <- 50
capacity_fab_b <- 42

# Define the maximum demand for each customer
demands <- c(36, 46, 11, 35)

# Define the model
model <- MIPModel() %>%

# Add variables for the number of chips to sell to each customer from each fab
add_variable(x[i, j], i = 1:4, j = c("A", "B"), type = "integer", lb = 0) %>%

# Objective function: Maximize total profit
set_objective(sum_expr(profits_fab_a[i] * x[i, "A"] + profits_fab_b[i] * x[i, "B"],
                       i = 1:4), "max") %>%
```



```r

# Demand constraints for each customer
add_constraint(x[1, "A"] + x[1, "B"] <= demands[1]) %>%
add_constraint(x[2, "A"] + x[2, "B"] <= demands[2]) %>%
add_constraint(x[3, "A"] + x[3, "B"] <= demands[3]) %>%
add_constraint(x[4, "A"] + x[4, "B"] <= demands[4]) %>%

# Capacity constraints for each fab
add_constraint(sum_expr(x[i, "A"], i = 1:4) <= capacity_fab_a) %>%
add_constraint(sum_expr(x[i, "B"], i = 1:4) <= capacity_fab_b)


# Solve the model
solution <- solve_model(model, with_ROI(solver = "glpk", verbose = TRUE))

solution

# Get the solution
get_solution(result, x[i, j])%>%
  filter(value > 0)


```

R results:



```
> # Solve the model
> solution <- solve_model(model, with_ROI(solver = "glpk", verbose = TRUE))
<SOLVER MSG>  ----
GLPK Simplex Optimizer 5.0
6 rows, 8 columns, 16 non-zeros
*     0: obj =  -0.000000000e+00 inf =   0.000e+00 (8)
*     4: obj =   3.535000000e+04 inf =   0.000e+00 (0)
OPTIMAL LP SOLUTION FOUND
GLPK Integer Optimizer 5.0
6 rows, 8 columns, 16 non-zeros
8 integer variables, none of which are binary
Integer optimization begins...
Long-step dual simplex will be used
+     4: mip =     not found yet <=              +inf        (1; 0)
+     4: >>>>>   3.535000000e+04 <=   3.535000000e+04   0.0% (1; 0)
+     4: mip =   3.535000000e+04 <=     tree is empty   0.0% (0; 1)
INTEGER OPTIMAL SOLUTION FOUND
<!SOLVER MSG> ----
>
> solution
Status: success
Objective value: 35350>
> # Get the solution
> get_solution(solution, x[i, j])%>%
+   filter(value > 0)
  variable i j value
1        x 1 A    36
2        x 2 A    14
3        x 2 B    31
4        x 3 B    11
> 
```

Managerial Results:

The maximum total profit that Teranikx can achieve is 35350 Million ₷.

The optimal distribution of chip sales to each customer from each factory is as follows:

- Sell 36 million chips to Customer 1 from Factory A
- Sell 14 million chips to Customer 2 from Factory A
- Sell 31 million chips to Customer 2 from Factory B
- Sell 11 million chips to Customer 3 from Factory B

Problem-3: Make-to-Stock Chemotherapy Drugs

Decision Variables:

Let's define the decision variables for the number of vials of EU and US constituents used in Chemo1 and Chemo2:

- x_{1EU}: Number of vials of EU constituent used in Chemo1
- x_{1US}: Number of vials of US constituent used in Chemo1
- x_{2EU}: Number of vials of EU constituent used in Chemo2
- x_{2US}: Number of vials of US constituent used in Chemo2

These variables are non-negative integers since we can't use a negative amount of constituents and we deal with whole vials.

Objective Function:

We want to maximize the monthly profit, which is the revenue from selling the chemotherapy compounds minus the cost of the constituents used.

$$Maximize\ Z = \sum_{i=1}^{2} \sum_{j=EU}^{US} \left(SP_i - C_j\right) * x_{ij}$$

Where SP_i is the selling price of chemo in *i*, while C_j is the cost to manufacture in region *j.*

Simplifying the profit function:

$$Maximize\ Z = (1200x_{1EU} + 1200x_{1US} + 1400x_{2EU} + 1400x_{2US}) - (800x_{1EU} + 1500x_{1US} + 800x_{2EU} + 1500x_{2US})$$

Constraints:

D-metric and P-metric for Chemo1 and Chemo2:

$87x_{1EU} + 98x_{1US} \leq 88\ (x_{1EU} + x_{1US})$ P-metric Constraints for Chemo1 with a minimum P-metric of 88

$87x_{1EU} + 98x_{1US} \leq 93\ (x_{2EU} + x_{2US})$ P-metric Constraints for Chemo2 with a minimum P-metric of 93

$25x_{1EU} + 15x_{1US} \leq 23\ (x_{1EU} + x_{1US})$ D-metric Constraints for Chemo1 considering they must not exceed 23

$25x_{2EU} + 15x_{2US} \leq 23\ (x_{2EU} + x_{2US})$ D-metric Constraints for Chemo2 considering they must not exceed 23

Demand and Delivery Constraints:

$x_{1EU} + x_{1US} \leq 200{,}000$ For Chemo-1

$x_{1EU} + x_{1US} \geq 100{,}000$ For Chemo-1

$x_{2EU} + x_{2US} \leq 40{,}000$ For Chemo-2

$x_{2EU} + x_{2US} \geq 10{,}000$ For Chemo-2

Inventory Constraints:

$x_{1EU} + x_{2EU} \leq 80{,}000$ For EU constituents

$x_{1US} + x_{2US} \leq 120{,}000$ For US constituents

Non-Negativity Constraints:

$x_{1EU}, x_{2EU}, x_{1US}, x_{2US} \geq 0$

Explanation:

- The objective function is derived from the profit per vial sold minus the cost per vial of the constituents.
- The D-metric and P-metric constraints ensure that the final product meets the quality standards.
- The demand constraints ensure that production does not exceed the maximum demand and that at least the minimum delivery amount is met.
- The inventory constraints ensure that the usage of constituents does not exceed the available inventory.
- The non-negativity constraints ensure that we do not have negative quantities, which is not physically possible.

Calculation of D-metrics and P-metrics:

After solving the problem, calculate the D-metrics and P-metrics for Chemo1 and Chemo2 using the optimal values of $x_{1EU}, x_{2EU}, x_{1US}, x_{2US}$.

For Chemo1:

$$D_{Chemo1} = \frac{x_{1EU} * 25 + x_{1US} * 15}{x_{1EU} + x_{1US}}$$

$$P_{Chemo1} = \frac{x_{1EU} * 87 + x_{1US} * 98}{x_{1EU} + x_{1US}}$$

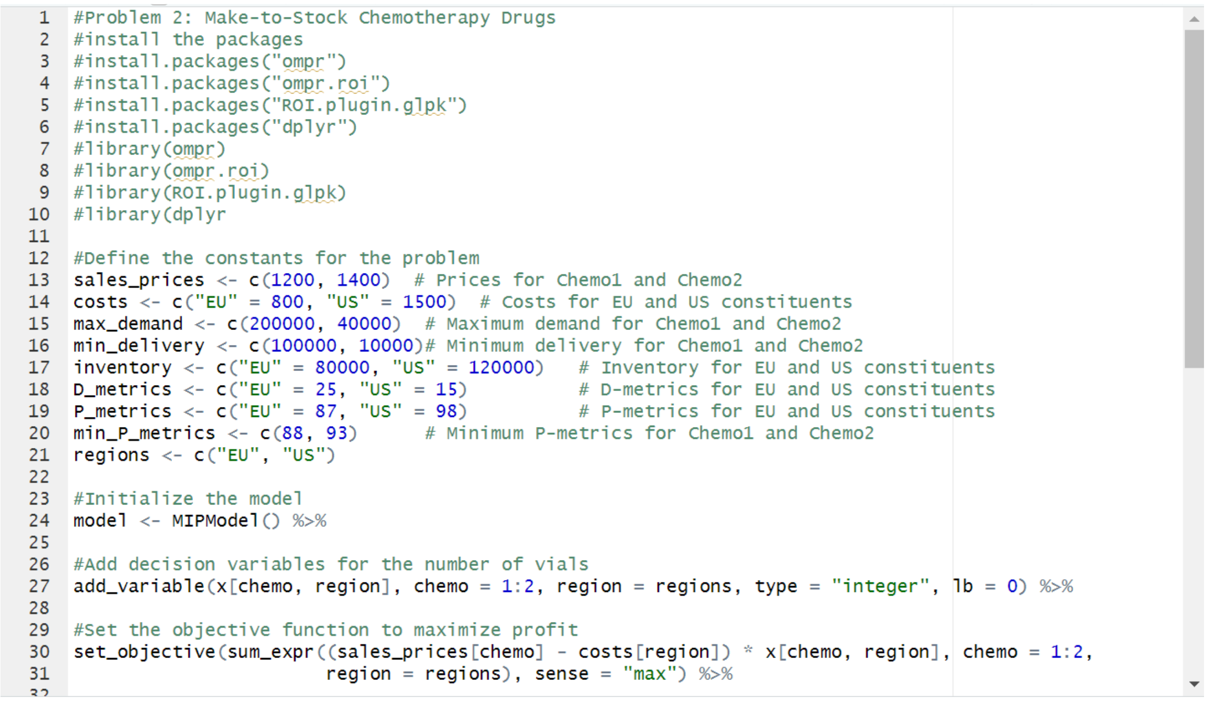
For Chemo2:

$$D_{Chemo2} = \frac{x_{2EU} * 25 + x_{2US} * 15}{x_{2EU} + x_{2US}}$$

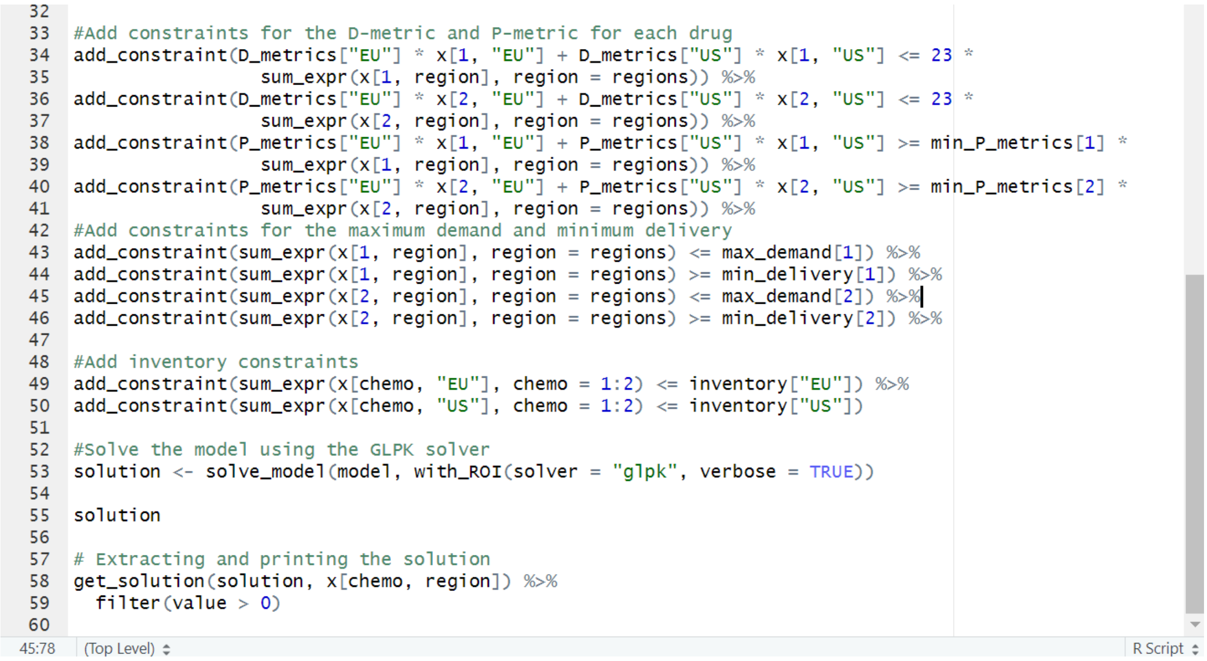
$$P_{Chemo2} = \frac{x_{2EU} * 87 + x_{2US} * 98}{x_{2EU} + x_{2US}}$$

This model aims to maximize monthly profit by blending the right amount of EU and US constituents to produce Chemo1 and Chemo2 while meeting all the specified metrics and constraints. After solving, the model will provide the optimal number of vials to produce, the maximum profit, and the D-metrics and P-metrics for each type of chemotherapy drug.

R code:

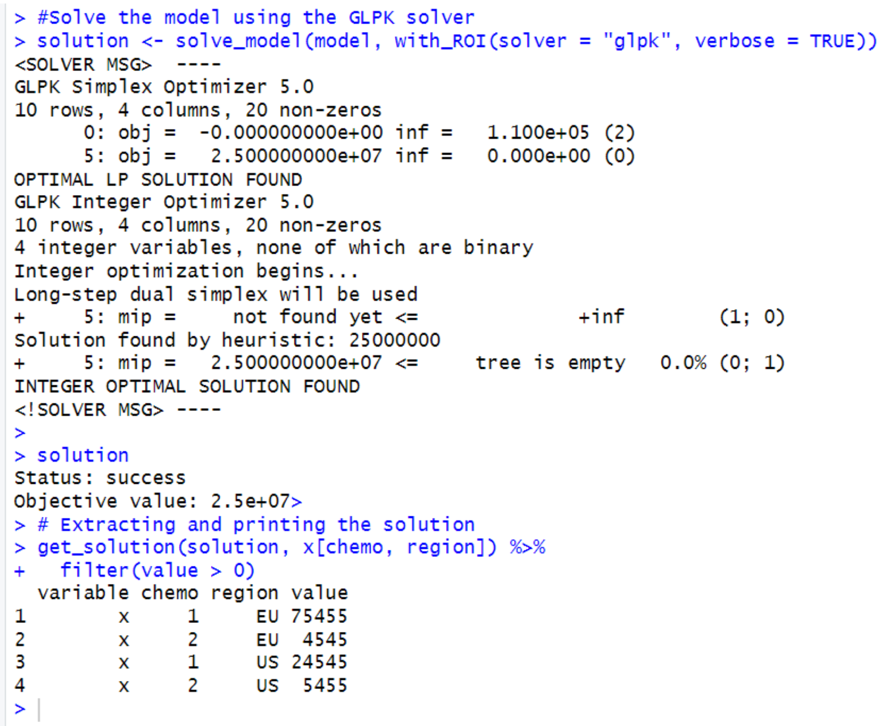


```r
#Problem 2: Make-to-Stock Chemotherapy Drugs
#install the packages
#install.packages("ompr")
#install.packages("ompr.roi")
#install.packages("ROI.plugin.glpk")
#install.packages("dplyr")
#library(ompr)
#library(ompr.roi)
#library(ROI.plugin.glpk)
#library(dplyr

#Define the constants for the problem
sales_prices <- c(1200, 1400)  # Prices for Chemo1 and Chemo2
costs <- c("EU" = 800, "US" = 1500)  # Costs for EU and US constituents
max_demand <- c(200000, 40000)  # Maximum demand for Chemo1 and Chemo2
min_delivery <- c(100000, 10000)# Minimum delivery for Chemo1 and Chemo2
inventory <- c("EU" = 80000, "US" = 120000)   # Inventory for EU and US constituents
D_metrics <- c("EU" = 25, "US" = 15)          # D-metrics for EU and US constituents
P_metrics <- c("EU" = 87, "US" = 98)          # P-metrics for EU and US constituents
min_P_metrics <- c(88, 93)      # Minimum P-metrics for Chemo1 and Chemo2
regions <- c("EU", "US")

#Initialize the model
model <- MIPModel() %>%

#Add decision variables for the number of vials
add_variable(x[chemo, region], chemo = 1:2, region = regions, type = "integer", lb = 0) %>%

#Set the objective function to maximize profit
set_objective(sum_expr((sales_prices[chemo] - costs[region]) * x[chemo, region], chemo = 1:2,
                       region = regions), sense = "max") %>%
```



```r

#Add constraints for the D-metric and P-metric for each drug
add_constraint(D_metrics["EU"] * x[1, "EU"] + D_metrics["US"] * x[1, "US"] <= 23 *
                 sum_expr(x[1, region], region = regions)) %>%
add_constraint(D_metrics["EU"] * x[2, "EU"] + D_metrics["US"] * x[2, "US"] <= 23 *
                 sum_expr(x[2, region], region = regions)) %>%
add_constraint(P_metrics["EU"] * x[1, "EU"] + P_metrics["US"] * x[1, "US"] >= min_P_metrics[1] *
                 sum_expr(x[1, region], region = regions)) %>%
add_constraint(P_metrics["EU"] * x[2, "EU"] + P_metrics["US"] * x[2, "US"] >= min_P_metrics[2] *
                 sum_expr(x[2, region], region = regions)) %>%
#Add constraints for the maximum demand and minimum delivery
add_constraint(sum_expr(x[1, region], region = regions) <= max_demand[1]) %>%
add_constraint(sum_expr(x[1, region], region = regions) >= min_delivery[1]) %>%
add_constraint(sum_expr(x[2, region], region = regions) <= max_demand[2]) %>%
add_constraint(sum_expr(x[2, region], region = regions) >= min_delivery[2]) %>%

#Add inventory constraints
add_constraint(sum_expr(x[chemo, "EU"], chemo = 1:2) <= inventory["EU"]) %>%
add_constraint(sum_expr(x[chemo, "US"], chemo = 1:2) <= inventory["US"])

#Solve the model using the GLPK solver
solution <- solve_model(model, with_ROI(solver = "glpk", verbose = TRUE))

solution

# Extracting and printing the solution
get_solution(solution, x[chemo, region]) %>%
  filter(value > 0)
```

Results:



```
> #Solve the model using the GLPK solver
> solution <- solve_model(model, with_ROI(solver = "glpk", verbose = TRUE))
<SOLVER MSG>  ----
GLPK Simplex Optimizer 5.0
10 rows, 4 columns, 20 non-zeros
      0: obj =  -0.000000000e+00 inf =   1.100e+05 (2)
      5: obj =   2.500000000e+07 inf =   0.000e+00 (0)
OPTIMAL LP SOLUTION FOUND
GLPK Integer Optimizer 5.0
10 rows, 4 columns, 20 non-zeros
4 integer variables, none of which are binary
Integer optimization begins...
Long-step dual simplex will be used
+      5: mip =     not found yet <=              +inf        (1; 0)
Solution found by heuristic: 25000000
+      5: mip =   2.500000000e+07 <=     tree is empty   0.0% (0; 1)
INTEGER OPTIMAL SOLUTION FOUND
<!SOLVER MSG> ----
>
> solution
Status: success
Objective value: 2.5e+07>
> # Extracting and printing the solution
> get_solution(solution, x[chemo, region]) %>%
+   filter(value > 0)
  variable chemo region value
1        x     1     EU 75455
2        x     2     EU  4545
3        x     1     US 24545
4        x     2     US  5455
> 
```

Computing the D-metrics and P-metrics:

For Chemo1:

$$D_{Chemo1} = \frac{x_{1EU} * 25 + x_{1US} * 15}{x_{1EU} + x_{1US}} = \frac{(75455 * 25) \ + \ (24545 * 15)}{75455 + 24545} = \ 22.5455$$

$$P_{Chemo1} = \frac{x_{1EU} * 87 + x_{1US} * 98}{x_{1EU} + x_{1US}} = \frac{(75455 * 87) + (24545 * 98)}{75455 + 24545} = 89.6999$$

For Chemo2:

$$D_{Chemo2} = \frac{x_{2EU} * 25 + x_{2US} * 15}{x_{2EU} + x_{2US}} = \frac{(4545 * 25) + (5455 * 15)}{4545 + 5455} = 19.545$$

$$P_{Chemo2} = \frac{x_{2EU} * 87 + x_{2US} * 98}{x_{2EU} + x_{2US}} = \frac{(4545 * 87) + (5455 * 98)}{4545 + 5455} = 93.0005$$

Managerial Results:

Quantities of constituents blended into Chemo1 and Chemo2:

The quantities of the EU and US constituents to blend into Chemo1 and Chemo2 for maximum monthly profit are:

- Chemo1: 75,455 vials of EU constituents and 24,545 vials of US constituents.
- Chemo2: 4,545 vials of EU constituents and 5,455 vials of US constituents.

Maximum Monthly Profit:

The maximum monthly profit given by the objective value in the solution output is €25,000,000.

D-metrics of Chemo1 and Chemo2:

The D-metrics for Chemo1 and Chemo2, which represent the weighted average based on the proportion of EU and US constituents in each drug, are:

- Chemo1: D-metric = 22.545
- Chemo2: D-metric = 19.545

P-metrics of Chemo1 and Chemo2:

The P-metrics for Chemo1 and Chemo2, which also represent the weighted average based on the proportion of EU and US constituents, are:

- Chemo1: P-metric = 89.699
- Chemo2: P-metric = 93.000