Goal Programming

Mirdha Suci Ananda, Yuda Hendriawan, Andi Aqil, Sumihar Christian[*]

Department of Mathematical Computation and Data Science
INSTITUT TEKNOLOGI SEPULUH NOPEMBER

May 10, 2019

Abstract

Simulating Goal Programming on a product store sales problem with 3 types Goal Programming initiation method: Lexicographical, Weighted and Chebyshev using Pyomo for mathematical modelling in Python and GLPK for solving the optimization problem using simplex method

I. Introduction

Ordinary linear program models are not able to solve management cases that require certain goals to reach all objectives optimally at the same time. In the linear program model there is a Slack variable in the constraint function in the form of a delimiter and a surplus in the constraint function in the form of conditions. In solving the case of a linear program the two variables function to accommodate the advantages and disadvantages of the left hand side value of a constraint function so that it equals the value of the right hand side. However, both of these variables are completely uncontrollable in solving the case of linear programs, so the linear program model was developed by A. Charles and W. M. Cooper as a goal programming model. If there are variables in a linear program that have characteristics similar to the Slack and Surplus variables, and are in a constraint equation, then the variable is controlled so that the left segment value of a constraint is equal to the value of the right segment. The goal programming model is able to solve cases of linear programs that have more than one goal to be achieved. The goal or target is a constant value on the right hand side of the constraint function Basically the structure of goal programming and linear programming is the same. The concept of goal programming is to introduce additional auxiliary variables called deviations that act not as variable decision, but only as facilitators to formulate models. This deviation is the difference between the desired target value and the results obtained. The goal programming model is an extension of the linear program model so that all as- sumptions, formation notations, mathematical models, formulation procedures, models and solutions are no different. The difference lies only in

II. General Form

III. Problem

i. Variables

x_1 = number of shirts,
x_2 = number of jackets

ii. Constraints

$2x_1 + 4x_2 \leq 600$ Cotton Stock
$5x_1 + 3x_2 \leq 700$ Linen Stock
$100x_1 + 90x_2 \geq 18000$ Weekly Profit minimum
$2x_1 + 2x_2 \leq 380$ Man-hours available
$x_1 + x_2 \leq 200$ Machine Limitation
$x_1, x_2 \geq 60$ Minimum Production's Contract

[*] A thank you or further information

IV. Methods

i. Lexicographical

Lexicographical Goal Programming use priority levels for each goals. The linear program will run separately for each priority level, running in sequence of order from the highest to lowest priority.

This will be the priority level for each of the goals :

- **Priority Level 1** Man-hours (Goal 4)
- **Priority Level 2** Weekly Profit (Goal 3)
- **Priority Level 3** Cotton and Linen stocks to buy (Goal 1 and Goal 2)

The first linear program is to solve Priority Level 1, which is to minimize man hour excess.

Priority Level 1 LP
Minimize d_4^+
Subject to

$$\begin{aligned} 2x_1 + 4x_2 + d_1^- - d_1^+ &= 600 \\ 5x_1 + 3x_2 + d_2^- - d_2^+ &= 700 \\ 100x_1 + 90x_2 + d_3^- - d_3^+ &= 18000 \\ 2x_1 + 2x_2 + d_4^- - d_4^+ &= 600 \\ x_1 + x_2 &\leq 200 \\ x_1, x_2 &\geq 60 \\ d_1^-, d_1^+, d_2^-, d_2^+, d_3^-, d_3^+, d_4^-, d_4^+ &\geq 0 \end{aligned}$$

This linear program can be solved using any method such as Simplex Method, Graphical Method or using any Solver. The results we get is $d_4^+ = 0$ which fully satisfy the goal.

The idea is to use the previous results for the next priority goal we want to accomplish. We will use the results from first Linear Program as a constraints for the next one. The second Linear Program is to achieve weekly profit.

Priority Level 2 LP
Minimize d_3^-
Subject to

$$\begin{aligned} 2x_1 + 4x_2 + d_1^- - d_1^+ &= 600 \\ 5x_1 + 3x_2 + d_2^- - d_2^+ &= 700 \\ 100x_1 + 90x_2 + d_3^- - d_3^+ &= 18000 \\ 2x_1 + 2x_2 + d_4^- - d_4^+ &= 600 \\ d_4^+ &= 0 \\ x_1 + x_2 &\leq 200 \\ x_1, x_2 &\geq 60 \\ d_1^-, d_1^+, d_2^-, d_2^+, d_3^-, d_3^+, d_4^-, d_4^+ &\geq 0 \end{aligned}$$

The results we get from the second Linear Program is $d_3^- = 0$ which fully satisfy the second level priority goal.

The third Linear Program will use the second Linear Program's constraints and its results.

Priority Level 3 LP
Minimize $d_1^+ + d_2^+$
Subject to

$$\begin{aligned} 2x_1 + 4x_2 + d_1^- - d_1^+ &= 600 \\ 5x_1 + 3x_2 + d_2^- - d_2^+ &= 700 \\ 100x_1 + 90x_2 + d_3^- - d_3^+ &= 18000 \\ 2x_1 + 2x_2 + d_4^- - d_4^+ &= 600 \\ d_4^+ &= 0 \\ d_3^- &= 0 \\ x_1 + x_2 &\leq 200 \\ x_1, x_2 &\geq 60 \\ d_1^-, d_1^+, d_2^-, d_2^+, d_3^-, d_3^+, d_4^-, d_4^+ &\geq 0 \end{aligned}$$

The result for this linear program is $d_1^+ = 0$, $d_1^- = 20$, $d_2^+ = 50$ and $d_2^- = 0$

This will be the final results :

$$\begin{aligned} x_1 &= 90 \\ x_2 &= 100 \\ d_1^+ &= 0,\ d_1^- = 20 \\ d_2^+ &= 50, d_2^- = 0 \\ d_3^+ &= 0,\ d_3^- = 0 \\ d_4^+ &= 0,\ d_4^- = 0 \end{aligned}$$

```
// Lexicograhical.py
# first priority level
model.obj = Objective(expr = model.p4)

# Define the constraints
model.con1 = Constraint(expr = 2 *
    model.x1 +
    4 * model.x2 + model.n1 - model.p1 ==
        600)
model.con2 = Constraint(expr = 5 *
    model.x1 +
    3 * model.x2 + model.n2 - model.p2 ==
        700)
model.con3 = Constraint(expr = 100 *
    model.x1 +
    90 * model.x2 + model.n3 - model.p3 ==
        18000)
model.con4 = Constraint(expr = 2 *
    model.x1 +
    2 * model.x2 + model.n4 - model.p4 ==
        380)
model.con5 = Constraint(expr = model.x1 +
    model.x2 <= 200)
model.con6 = Constraint(expr = model.x1 >=
    60)
model.con7 = Constraint(expr = model.x2 >=
    60)

# Solve the Goal Programming problem of the
# first priority level
opt.solve(model)

# Retrieve the value of the first priority
    level
p4 = model.p4.value

# Define the objective function of the
# second priority level
model.obj = Objective(expr = model.n3)

# Add a constraint for the value of the
    first
# priority level
model.con8 = Constraint(expr = model.p4 ==
    p4)

# Solve the Goal Programming problem of the
# second priority level
opt.solve(model)

# Retrieve the value of the second
    priority level
n3 = model.n3.value

# Define the objective function of the
# third priority level
model.obj = Objective(expr = model.p1 +
    model.p2)

# Add a constraint for the value of the
    second
# priority level
model.con9 = Constraint(expr = model.n3 ==
    n3)

# Solve the Goal Programming problem of the
# third priority level
opt.solve(model)

# Print the values of the decision
    variables
print("x1 = ", model.x1.value)
print("x2 = ", model.x2.value)

# Print the achieved values for each goal
if model.n1.value > 0:
    print("The first goal is underachieved
        by ",
         model.n1.value)
elif model.p1.value > 0:
    print("The first goal is overachieved
        by ",
         model.p1.value)
else:
    print("The first goal is fully
        satisfied")

if model.n2.value > 0:
    print("The second goal is
        underachieved by ",
         model.n2.value)
elif model.p2.value > 0:
    print("The second goal is overachieved
        by ",
         model.p2.value)
else:
    print("The second goal is fully
        satisfied")

if model.n3.value > 0:
    print("The third goal is underachieved
        by ",
         model.n3.value)
elif model.p3.value > 0:
    print("The third goal is overachieved
        by ",
         model.p3.value)
else:
    print("The third goal is fully
        satisfied")

if model.n4.value > 0:
    print("The fourth goal is
        underachieved by ",
         model.n4.value)
elif model.p4.value > 0:
    print("The fourth goal is overachieved
        by ",
         model.p4.value)
else:
    print("The fourth goal is fully
        satisfied")
```

ii. Weighted

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

```
model.obj = Objective(expr = (1 / 600) *
    model.p1 +
    (1 / 700) * model.p2 + (2 / 18000) *
        model.n3 +
    (3 / 380) * model.p4)

# Define the constraints
model.con1 = Constraint(expr = 2 *
    model.x1 +
    4 * model.x2 + model.n1 - model.p1 ==
        600)
model.con2 = Constraint(expr = 5 *
    model.x1 +
    3 * model.x2 + model.n2 - model.p2 ==
        700)
model.con3 = Constraint(expr = 100 *
    model.x1 +
    90 * model.x2 + model.n3 - model.p3 ==
        18000)
model.con4 = Constraint(expr = 2 *
    model.x1 +
    2 * model.x2 + model.n4 - model.p4 ==
        380)
model.con5 = Constraint(expr = model.x1 +
    model.x2 <= 200)
model.con6 = Constraint(expr = model.x1 >=
    60)
model.con7 = Constraint(expr = model.x2 >=
    60)

# Solve the Goal Programming problem
opt.solve(model)

# Print the values of the decision
    variables
print("x1 = ", model.x1.value)
print("x2 = ", model.x2.value)

# Print the achieved values for each goal
if model.n1.value > 0:
    print("The first goal is underachieved
        by ",
         model.n1.value)
elif model.p1.value > 0:
    print("The first goal is overachieved
        by ",
         model.p1.value)
else:
    print("The first goal is fully
        satisfied")

if model.n2.value > 0:
    print("The second goal is
        underachieved by ",
         model.n2.value)
elif model.p2.value > 0:
    print("The second goal is overachieved
        by ",
         model.p2.value)
else:
    print("The second goal is fully
        satisfied")
```

```
if model.n3.value > 0:
    print("The third goal is underachieved
        by ",
         model.n3.value)
elif model.p3.value > 0:
    print("The third goal is overachieved
        by ",
         model.p3.value)
else:
    print("The third goal is fully
        satisfied")

if model.n4.value > 0:
    print("The fourth goal is
        underachieved by ",
         model.n4.value)
elif model.p4.value > 0:
    print("The fourth goal is overachieved
        by ",
         model.p4.value)
else:
    print("The fourth goal is fully
        satisfied")
```

iii. Chebyshev

1. Determine whether a constraint is soft or hard.
2. Add a negative and a positive deviational variable on each constraint. Determine the type of the constraint and add a constraint of the deviational variable(s) to be penalized. For each deviational variable, select a weight.
3. Use a normalization method to scale the deviations.
4. Each hard constraint is written as a typical linear programming constraint.
5. Add bound constraints to the problem (if applicable).

```
# Define the decision variables
model.x1 = Var(within =
    NonNegativeIntegers)
model.x2 = Var(within =
    NonNegativeIntegers)

# Define the deviational variables
model.n1 = Var(within =
    NonNegativeIntegers)
model.p1 = Var(within =
    NonNegativeIntegers)
model.n2 = Var(within =
    NonNegativeIntegers)
model.p2 = Var(within =
    NonNegativeIntegers)
model.n3 = Var(within =
    NonNegativeIntegers)
model.p3 = Var(within =
    NonNegativeIntegers)
model.n4 = Var(within =
    NonNegativeIntegers)
model.p4 = Var(within =
    NonNegativeIntegers)

# Define the variable of maximal deviation
# from amongst the set of goals
model.l = Var(within=NonNegativeReals)

# Define the objective function
model.obj = Objective(expr = model.l)

# Define the constraints
model.con1 = Constraint(expr = (1 / 600) *
    model.p1 <= model.l)
model.con2 = Constraint(expr = (1 / 700) *
    model.p2 <= model.l)
model.con3 = Constraint(expr = (2 / 18000)
    *
    model.n3 <= model.l)
model.con4 = Constraint(expr = (3 / 380) *
    model.p4 <= model.l)
model.con5 = Constraint(expr = 2 *
    model.x1 +
    4 * model.x2 + model.n1 - model.p1 ==
        600)
model.con6 = Constraint(expr = 5 *
    model.x1 +
    3 * model.x2 + model.n2 - model.p2 ==
        700)
model.con7 = Constraint(expr = 100 *
    model.x1 +
    90 * model.x2 + model.n3 - model.p3 ==
        18000)
model.con8 = Constraint(expr = 2 *
    model.x1 +
    2 * model.x2 + model.n4 - model.p4 ==
        380)
```

```
model.con9 = Constraint(expr = model.x1 +
    model.x2 <= 200)
model.con10 = Constraint(expr = model.x1
     >= 60)
model.con11 = Constraint(expr = model.x2
     >= 60)

# Solve the Goal Programming problem
opt.solve(model)

# Print the values of the decision
     variables
print("x1 = ", model.x1.value)
print("x2 = ", model.x2.value)

# Print the achieved values for each goal
if model.n1.value > 0:
    print("The first goal is underachieved
         by ",
          model.n1.value)
elif model.p1.value > 0:
    print("The first goal is overachieved
         by ",
          model.p1.value)
else:
    print("The first goal is fully
         satisfied")

if model.n2.value > 0:
    print("The second goal is
         underachieved by ",
          model.n2.value)
elif model.p2.value > 0:
    print("The second goal is overachieved
         by ",
          model.p2.value)
else:
    print("The second goal is fully
         satisfied")

if model.n3.value > 0:
    print("The third goal is underachieved
         by ",
          model.n3.value)
elif model.p3.value > 0:
    print("The third goal is overachieved
         by ",
          model.p3.value)
else:
    print("The third goal is fully
         satisfied")

if model.n4.value > 0:
    print("The fourth goal is
         underachieved by ",
          model.n4.value)
elif model.p4.value > 0:
    print("The fourth goal is overachieved
         by ",
          model.p4.value)
else:
    print("The fourth goal is fully
         satisfied")
```

V. Results

Table 1: *Solution for Lexicographical*

Goal	Target	Achieved Value
1	600	580
2	700	750
3	18000	18000
4	380	380

Table 2: *Solution for Weighted*

Goal	Target	Achieved Value
1	600	614
2	700	716
3	18000	17830
4	380	380

Table 3: *Solution for Chebyshev*

Goal	Target	Achieved Value
1	600	614
2	700	716
3	18000	17830
4	380	380

From the result we have Chebyshev that has most balanced solution

References

[Figueredo and Wolf, 2009] Figueredo, A. J. and Wolf, P. S. A. (2009). Assortative pairing and life history strategy - a cross-cultural study. *Human Nature*, 20:317–330.