Lexicographic goal programming approach for portfolio optimization

Mehdi Jafarian
Azizolah Jafari
Abstract

The mean variance model of portfolio optimization that was introduced by Markowitz includes two conflicted objective functions. These two criteria, risk and return does not encompass all of the information about investment. Information like liquidity, annual dividends and performance in later years. Thus portfolio selection is a usual multi-objective decision making (MODM) problem. This paper will investigate the optimum portfolio for a private investor, taking into account 6 criteria. It is well known that GP, based on preemptive priorities and target values, has been successful in solving MODM problems. In this paper we classify the MODM model with consideration of conflict between objective functions. And this is because of the fact that Decision maker (DM) does not have enough information about the trade-off between objective functions. Then we solve this model which was constructed by prioritizing objectives, with Lexicographic Goal Programming (LGP). Finally we determine the goals, and illustrate our proposed LGP model by an example.

Key words: Lexicographic Goal Programming (LGP), portfolio optimization, multi-objective decision making (MODM), priority, decision maker (DM)

1. Introduction

1.1 Portfolio optimization:

The traditional mean–variance model developed by Markowitz [6] has been the basis of portfolio theory. It is the first systematic treatment of investor’s conflicting objectives of high return versus low risk. On one hand, the risk of a portfolio, represented by its variance, is to be minimized, while on the other hand the expected return of the portfolio is to be maximized. This traditional portfolio theory has been applied successfully in a variety of situations in which investments are comprised of stocks, bonds, real estate, private equity, and similar instruments.

* Corresponding author. Tel.: +9821-66068715; fax: +9821-66012428.
E-mail address: jafarian.mehdi@gmail.com
* Iranian Academic Center for Education, Culture, and Research
1.2 Pareto optimal solutions

In the context of multi-criteria programming, solving Markowitz’s [6] model is understood as generating its Pareto optimal (efficient) solutions. The analytic derivation of the mean–variance efficient portfolio frontier is given by Merton [12].

Note that a solution \( x_e \) in the feasible space is called an efficient solution if there are no other solutions fulfilling followings.

\[
\begin{align*}
  f_2(x) & \leq f_2(x_e) \quad \& \quad f_1(x) < f_1(x_e) \\
  \text{or} \\
  f_2(x) & < f_2(x_e) \quad \& \quad f_1(x) \leq f_1(x_e)
\end{align*}
\]

In fact efficient solutions are those for which none of the criteria can be improved without deterioration of the other criteria. The proposed tool in Markowitz model for finding the efficient portfolios is linear criteria algorithm. In fact the model solves a quadratic parametric problem [6].

Konno [9] formulates a linear piecewise function for risk for making the objective function linear. He claims that this model equals Markowitz’s model, when the vector of returns is multivariate normally distributed.

All of these models do not incorporate other information that is preferred by DM. in fact investors do not buy actually efficient portfolios but rather those behind the efficient frontier. Ghandforoush [5] proposed the measures of objective and subjective for investment that his idea led to a simple linear programming model.

Matthias Ehrgott [10] proposed a model for portfolio optimization extending the Markowitz mean–variance model. Based on cooperation with standard and poor’s he used five specific objectives related to risk and return and allowed consideration of
individual preferences through the construction of decision-maker specific utility functions and an additive global utility function.

Goal Programming (GP) introduced by Charnes and Cooper [1] is an appropriate for solving portfolio models. The majority of GP applications in the literature have been implemented using various methods such as lexicographic GP (preemptive GP), weighted GP (Archimedean GP), and MINMAX GP (Chebyshev GP) [3]. In this case DM can determine the goals and priorities of objectives. Useful details of critical issues and applications in GP are found in [4, 11, 15].

Waieel F. Abd El-Wahed [14] introduces a solution method based on the theory of fuzzy sets and GP for multiple objective decision making (MODM) problems. The solution method, called hybrid fuzzy-goal programming (HFGP), combines and extends the attractive features of both fuzzy set theory and goal programming for MODM problems. HFGP approach has been introduced to determine weights and weights to the objectives under the same priorities using the concept of fuzzy membership functions along with the notion of degree of conflict among objectives. Also, HFGP converts a MODM problem into a lexicographic goal programming problem by fixing the priorities and aspiration levels appropriately.

A common weakness of all approaches described above is that arbitrary selection of goals and priorities can lead to undesirable results; because they do not consider the trade-off and degree of conflict among objectives. Therefore DM should pay attention to conflict between objectives when determining goals and priorities. On the other hand DM doesn’t have enough Information about these trade-off and conflict among objectives.
Thus we propose a model that prioritizes objectives according to the degree of non-conflict among objectives. In section 2 first we describe Markowitz’s model and its extensions according to the above discussion. Then we develop our model based on Lexicographic Goal Programming (LGP). In order to prioritizing objectives we must first determine the degree of non-conflict among objectives. We do this by means of the gradient of each objective. Finally we determine the goals by using membership function expressing degrees of individual optimalities [15]. In section 3 we illustrate our approach by a numerical example.

2. Model description

2.1. Markowitz’s model

The so called model of Markowitz is in the following form:

\[
\begin{align*}
\text{Max} & \quad \sum_{i=1}^{n} \mu_i x_i \\
\text{Min} & \quad \sum_{i=1}^{n} \sum_{j=1}^{n} \sigma_{ij} x_i x_j \\
\text{S.t.} & \quad \sum_{i=1}^{n} x_i \leq M \\
& \quad x_i \geq 0 \quad ; \quad i = 1, K, n
\end{align*}
\]

Here, \( n \) denotes the number of available assets; \( x_i \) represents the investment portion in asset \( i \in \{1, \ldots, n\} \) where \( x \) is the \( n \) dimensional solution vector. \( \mu_i \), denotes the expected return of asset \( i \). \( \sigma_{ij} \), denotes covariance between returns of asset \( i, j \). The constraint \( \sum_{i=1}^{n} x_i \leq M \) is capital constraint where \( M \) is total amount of capital.
The proposed tool in Markowitz model for finding the efficient portfolios is linear criteria algorithm. In fact the model solves a nonlinear parametric problem

$$\text{Min} \quad x' \sigma x - \lambda \mu x \quad [6].$$

Note that the first objective, \( f_1(x) = \sum \mu_i x_i \) is a linear function while the second objective, \( f_2(x) = -\sum \sum \sigma_{ij} x_i x_j \) is a quadratic function of \( x \). In order to have a vector maximization problem (VMP) we have replaced the minimization of the variance in (1) by the equivalent maximization of the negative variance. Also without loss of generality we assume that the capital constraint is \( \sum x_i \leq 1.\)

2.2 Objective hierarchy

Most of investors prefer portfolios that lie behind the non-dominated frontier. In Markowitz’s model this shows that all of decision information is not reflected in risk and return. A portfolio that is optimized based on risk and return may result in poor objectives like liquidity, 1 and 3 years performance, annual revenue. Thus they can’t be aggregated in one objective called return. Figure 2.2 shows an example of an objective hierarchy extending the classical Markowitz model in the sense that the two classical criteria risk and return are replaced by six more specific objective functions. Matthias Ehrgott [10] split the global utility somewhat similar to the one follows.
It was indicated that the expected return as used in the Markowitz model should be broken down into the criteria 1 year return, 3-year return annual dividend and liquidity in order to improve the possibilities of the individual investor to articulate subjective preferences. This gain of flexibility seems to outweigh the additional time required for the consideration of further objectives. The objective, Standard and Poor’s star ranking, describes to what extent an investment fund follows a specific market index and is applied particularly in the case that a portfolio consists exclusively of investment funds. It evaluates the out- or under-performance divided by the tracking error over three years and rewards funds that closely follow the market index. First we split the expected return of Markowitz model into 5 other objectives illustrated in above figure. This will show the interests of investor better. Then we show the total risk of investment risk functions as showed above. In what follows we try to determine the explicit form of objective functions.

2.2.1 One year return
If $P_{it}$ denotes the price of asset $i$ in period $t$ then the return rate in one year will be:

$$\mu_i = \frac{P_{it} - P_{t-1,i}}{P_{t-1,i}}$$

Where $T$ is present. In fact $\mu_i$ is the relative change in price of asset $i$ over the last year.

This is a good approximate of expected return over future year. We do not need to know the statistical distribution of $\mu_i$.

2.2.2 Three year return

We denote the 3 years return by $\mu_i'$ and define it in this way:

$$\mu_i' = \frac{P_{it} - P_{t-3,i}}{P_{t-3,i}}$$

Thus the objective functions for 2 years and 3 years return will be:

$$f_1(x) = \sum_{i=1}^{n} \mu_i x_i$$

and

$$f_2(x) = \sum_{i=1}^{n} \mu_i' x_i.$$  

2.2.3 Annual dividend

The annual dividend of an asset is the dividend paid, relative to the highest price of the asset and will be denoted by $d_i$. $d_i = \frac{d_i^a}{P_i^h}$

We denote the annual dividend objective function by $f_3(x) = \sum_{i=1}^{n} d_i x_i$.

2.2.4 Liquidity

The degree of liquidity of an asset $i$ will be denoted by $l_i$ and equals the number of days which company index is open in one year divided by total days that stock market
works in a year. The total liquidity of portfolio would be the linear combination of $X_i$ and is denoted by $f_5(x) = \sum_{i=1}^{n} l_i x_i$.

2.2.5 Risk

And finally as called before the risk function would be $f_i(x) = -\sum_{i=1}^{n} \sum_{j=1}^{n} \sigma_{ij} x_i x_j$. See [6].

2.2.6 S&P star ranking

The Standard and Poor’s Fund Services evaluates the performance of most investment funds contained in their database on an annual basis which results in a performance ranking (star ranking). The ranking is based on the performance of an investment fund in comparison to the sector index and assigns between one star (for a relatively poor performance) and up to five stars (for a very good performance). We will assume in the following that the ranking is additive in the sense that the ranking of a portfolio of investment funds can be obtained as the weighted sum of the rankings of the individual investment funds in the portfolio. Consequently, the fourth objective function can be written as

$$f_6(x) = \sum_{i=1}^{n} s_{ri} x_i$$

Where $s_{ri}$ denotes the number of stars assigned to investment fund $i$.

Therefore our MODM model would be of this form:

$$\begin{align*}
\text{Max} & \quad [f_1(x), f_2(x), f_3(x), f_4(x), f_5(x), f_6(x)] \\
\text{S.t.} & \quad \sum_{i=1}^{n} x_i \leq 1 \\
& \quad x_i \geq 0; \quad i = 1, \ldots, n
\end{align*}$$
2.3. Deriving LGP model

Now we have an MODM model that some its objective are in conflict with each other. Solving these types of model involves trade-off decisions in order to achieving an optimized solution.

Most of solving MODM methods do not prioritize objective functions, but one of the goal programming methods based on prioritizing objectives and goal determining have been successful in solving MODM problems. We convert the problem solving to a more general form of GP called Lexicographic Goal Programming (LGP) with consideration of conflict between objectives. It means that we prioritize objective functions according to the conflict among them as they can form L prioritized classes \((1 \leq L \leq 6)\). If we assume

\[ d_k \text{ be deviation variable of } k\text{th objective and } \lambda_{i,k} \text{ be the weight of deviation variable } d_k \]

which lies in \(l\)th class, our LGP model would be of this form:

\[
\begin{align*}
\text{Min } & \sum_{j=1}^{L} \sum_{k \in K_j} P_j (\lambda_{k,1}^+ d_k^+ + \lambda_{k}^- d_k^-) \\
\text{S.t. } & f_k (x) - d_k^+ + d_k^- = g_k ; k = 1, ..., 6 \\
& \sum_{j=1}^{n} x_j \leq 1 \\
& d_k^+ d_k^- = 0 ; k = 1, ..., 6 \\
& d_k^+, d_k^- \geq 0 ; k = 1, ..., 6 ; i = 1, ..., n
\end{align*}
\]

Here the GP formulation has be modified into a general form by introducing the preemptive priorities, \(P_l\), in place of, or together with, the weights\(\lambda_{k}^+, \lambda_{k}^- \geq 0\). When the objective functions \(f_k (.)\) are divided into L ordinal ranking classes, \((1 \leq L \leq 6)\), having the preemptive priorities \(P_1, ..., P_l\) in decreasing order, it may be convenient to write:
This means that there is no real number \( t > 0 \) such that \( tp_{i+1} \geq P_i \). \( K_i \neq \emptyset \) is index set of \( l \)th prioritized class. Further details concerning the algorithm, extensions, and applications can be found in the text of Lee [13], Ignizio [8], and Romero [4].

2.4. Determining non-conflict degree among objectives

For determining objectives priorities first we must specify the degree of conflict between objectives. We use Cohon’s [7] procedure of gradients of the objectives to determine the regions of conflict. We assume \( \theta_{ij} \) is the angle between gradients of objectives \( f_i \) and \( f_j \). It’s obvious that \( \theta_{ij} \) can be computed from the following relation:

\[
\cos \theta_{ij} = \frac{\text{grad}(f_i) \cdot \text{grad}(f_j)}{|\text{grad}(f_i)| \cdot |\text{grad}(f_j)|}
\]

In this way we can specify a \( 6 \times 6 \) matrix of angles between gradients.

If \( \theta_{ij} = 0 \) then objectives \( f_i \) and \( f_j \) can be maximized simultaneously. In this case there is no conflict between these two functions and their gradients are in the same direction. If \( 0 \leq \theta_{ij} \leq \frac{\pi}{2} \) then objectives \( f_i \) and \( f_j \) have a conflict with each other. if \( \frac{\pi}{2} \leq \theta_{ij} \leq \pi \) then the degree of conflict between objectives is more than previous case. And if \( \theta_{ij} = \pi \) then we have a maximum conflict between objectives because the gradients of function are in opposite direction.

We can define the non-conflict function as follows. This definition uses the theory of fuzzy sets [2]:

\[ P_i \gg P_{i+1}, \quad l = 1, \ldots, L - 1 \]
In the case of \( \delta_{ij} = 0 \) we have complete conflict among objectives and when \( \delta_{ij} = 1 \) objectives are in complete non-conflict state. These \( \delta_{ij} \)s construct a symmetric matrix \( \Delta \) which its entries indicate a numerical measure of the degrees of non-conflict among the objectives [2].

\[
\Delta_{6\times6} = \begin{pmatrix}
1 & \delta_{12} & \emptyset & \delta_{16} \\
\delta_{21} & 1 & \emptyset & \delta_{26} \\
\emptyset & \emptyset & \emptyset & \emptyset \\
\delta_{61} & \emptyset & \delta_{65} & 1
\end{pmatrix}
\]

2.5. Specifying priorities:

2.5.1 Gradient vector of risk function

Note that the gradient vector of risk function will be

\[
\text{grad}(f_2) = (-2\sigma_i x_i - \sum_{j=2}^n \sigma_{ij} x_j, \ldots, -2\sigma_j x_j - \sum_{i=2}^n \sigma_{ij} x_i, \ldots, -2\sigma_n x_n - \sum_{j=2}^n \sigma_{nj} x_j)
\]

that its direction is not constant and each of its components is linear functions. In order to obtaining a constant gradient of risk function we can obtain the mean value of each one of the components.
\[
\text{mean}(f(x_1, \ldots, x_n)) = \frac{\int_{a_1}^{b_1} \cdots \int_{a_n}^{b_n} f(x_1, \ldots, x_n) \, dx_1 \cdots dx_n}{(b_n - a_n)(b_{n-1} - a_{n-1}) \cdots (b_1 - a_1)}
\]

Now we can write:

\[
\text{mean}(-2\sigma_i^2 x_i - \sum_{j=1}^{n} \sigma_{ij} x_j) = \int_{0 \leq x_i \leq 1} (-2\sigma_i^2 x_i - \sum_{j=1, j \neq i}^{n} \sigma_{ij} x_j) \, dx_i \cdots dx_n
\]

\[
= \frac{1}{2} (-2\sigma_i^2 - \sum_{j=2}^{n} \sigma_{1j}, \ldots, -2\sigma_i^2 - \sum_{j=2, j \neq i}^{n} \sigma_{ij}, \ldots, -2\sigma_i^2 - \sum_{j=2}^{n-1} \sigma_{ij})
\]

We can drop the \(\frac{1}{2}\) coefficient because it does not have any effect on the direction of gradient vector. Finally the gradient vector takes the form:

\[
\text{grad}(f_i) = (-2\sigma_i^2 - \sum_{j=2}^{n} \sigma_{1j}, \ldots, -2\sigma_i^2 - \sum_{j=2, j \neq i}^{n} \sigma_{ij}, \ldots, -2\sigma_i^2 - \sum_{j=2}^{n-1} \sigma_{ij})
\]

So the matrix \(\Delta\) would be a matrix of constants.

2.5.2 Determining priorities

Matrix \(\Delta\) prioritizes objectives and finally facilitates achievement to specified goals. Because of the fact that the elements of \(i\)th row in \(\Delta\) shows the degree of non-conflict among \(f_i\) and other objectives then we define the total amount of support of the objective \(f_i\) gets from all the other objectives as:

\[
w_k = \frac{\sum_{j=1}^{6} \delta_{ij}}{5}
\]

Waiel F. Abd El-Wahed [14] in his article defined \(w_k\) as follows:

\[
w_k = \frac{\sum_{j=1}^{6} \delta_{ij}}{6}
\]
The difference between these two definitions is that the second gets total support from all objectives including $f_k$ but the first one ignores the non-conflict degree with itself.

Now we can sort obtained $w_k$'s. $w_k$ corresponds to function $f_k$ and consequently each objective $f_k$ has been sorted according to its degree of non-conflict with other objectives.

2.5.3 Obtaining goals

Now we determine the goals from obtained $w_k$'s. First we obtain the best and worst solution for each objective:

$$f_k^+ = \max_{x \in X} f_k(x) \quad \text{and} \quad f_k^- = \min_{x \in X} f_k(x) \quad \text{for} \ k = 1, \ldots, 6$$

where $X$ is feasible space of our LGP model.

Linear membership functions expressing degrees of individual optimalities can be defined as [15]:

$$\mu_k(f_k) = \begin{cases} 
1 & ; f_k(x) > f_k^+ \\
\frac{f_k(x) - f_k^-}{f_k^+ - f_k^-} & ; f_k^- \leq f_k(x) \leq f_k^+ \\
0 & ; f_k(x) < f_k^-
\end{cases} \quad (6)$$

Each objective should have an aspiration level according to degree of non-conflict with other objectives. This means that $w_k$'s have a great effect on the goal of $f_k$ in LGP model. Note that the larger the $w_k$ the more satisfactory of objective $f_k$. Now $g_k$ can be found in this way:

$$g_k = \mu_k^{-1}(w_k) \quad k = 1, \ldots, 6 \quad (7)$$
These goals are more compatible with ranking of objectives than DM’s goals. However DM’s goal may differ with those we calculated by above formula. As called before this is because of the fact that DM does not have any technical information about trade-off and conflict between objectives, and his/her goals may not be feasible even when we do not prioritize objectives. So we can’t use DM’s goal with our obtained ranking. Each time we want to use him/her goals we can calculate \( w_i \)s by above formula and if it does not violate obtained ranking we can change goals.

The only remaining thing for completing our model is computing weights of deviation variables. If there is only one objective in each class then there is no need for computing these weights.

3. Numerical illustration

We test our approach, by solving a portfolio optimization problem. Let us use the set of historical annual data of 350 assets. We also assume that an investor wants to allocate one unit of wealth among some of these assets on which historical three year returns and other data have been calculated using the so called formulas.

First we obtain max and min of each objective in feasible space. For doing so we must solve an optimization problem which does not take significant time except for risk function. We computed the min of risk function by lingo and after 173 seconds its local optimum with 12 iterations was found. The min of objectives \( f_2, f_3, f_4, f_6 \) and the max of \( f_5 \) are nearly zero. Results are shown in table 3.1
Table 3.1 maximum and minimums of objectives

<table>
<thead>
<tr>
<th></th>
<th>$f_1$</th>
<th>$f_2$</th>
<th>$f_3$</th>
<th>$f_4$</th>
<th>$f_5$</th>
<th>$f_6$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max</td>
<td>89.23</td>
<td>108.6</td>
<td>10</td>
<td>0</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>Min</td>
<td>-30</td>
<td>0</td>
<td>0</td>
<td>-1170</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

So we can compute $\mu_i$s from (6)

\[
\mu_1(f_i) = \frac{f_i + 30}{119.23} \quad \mu_2(f_2) = \frac{f_2}{108.6} \quad \mu_3(f_3) = \frac{f_3}{10} \quad \mu_4(f_4) = \frac{f_4 + 1170}{1170} \\
\mu_5(f_5) = \frac{f_5}{1} \quad \mu_6(f_6) = \frac{f_6}{5}
\]

Here we obtain $\Delta$ from the angle between gradients of objectives and using (4) by a C++ program. Computing this matrix and the degree of non-conflict by our C++ program took less than one second:

\[
\Delta = \begin{bmatrix}
1 & 0.7160 & 0.6324 & 0.3313 & 0.7097 & 0.7064 \\
0.7116 & 1 & 0.6839 & 0.3439 & 0.7868 & 0.7317 \\
0.6324 & 0.6839 & 1 & 0.4025 & 0.6996 & 0.6710 \\
0.3312 & 0.3439 & 0.4025 & 1 & 0.3377 & 0.3631 \\
0.7097 & 0.7868 & 0.6996 & 0.3377 & 1 & 0.7630 \\
0.7064 & 0.7317 & 0.6710 & 0.3631 & 0.7630 & 1
\end{bmatrix}
\]

After calculating matrix $\Delta$ we computed $w_i$s from (5) with following results:

\[
w_1 = 0.5229 \quad w_2 = 0.63665 \quad w_3 = 0.6143 \quad w_4 = 0.3618 \quad w_5 = 0.6468 \quad w_6 = 0.6323
\]

As we expect the non-conflict degree for risk function is less than those of other’s.

Thus the obtained ranking is: $f_3 > f_2 > f_6 > f_5 > f_1 > f_4$

And priority classes will be: $f_3 \in P_1 \quad f_2 \in P_2 \quad f_6 \in P_3 \quad f_5 \in P_4 \quad f_1 \in P_5 \quad f_4 \in P_6$

Now we compute goals from (7) and obtained $\mu_i$s:
\[ g_1 = 31.9 \quad g_2 = 68.04 \quad g_3 = 6.1 \quad g_4 = -748.8 \quad g_5 = 0.68 \quad g_6 = 3.15 \]

Thus we have completed our LGP model as described in (3):

\[
\min \sum_{i=1}^{n} P(d_i^+ + d_i^-)
\]

\[ S.t \]

\[ f_i(x) - d_i^+ + d_i^- = 31.9 \]
\[ f_i(x) - d_i^+ + d_i^- = 68.04 \]
\[ f_i(x) - d_i^+ + d_i^- = 6.1 \]
\[ f_i(x) - d_i^+ + d_i^- = -748.8 \]
\[ f_i(x) - d_i^+ + d_i^- = 0.68 \]
\[ f_i(x) - d_i^+ + d_i^- = 3.15 \]

\[ \sum_{i=1}^{n} x_i \leq 1 \]
\[ d_i^+ d_i^- = 0 \quad d_i \geq 0 \quad k = 1, \ldots, 6 \]
\[ x_i \geq 0 \quad i = 1, \ldots, 350 \]

We solved this model by ordinary GP and got results as shown in table 3.2 within 9'52" and 102 iterations:

**Table 3.2 results for (8) without prioritizing**

<table>
<thead>
<tr>
<th>Objective</th>
<th>( f_1 )</th>
<th>( f_2 )</th>
<th>( f_3 )</th>
<th>( f_4 )</th>
<th>( f_5 )</th>
<th>( f_6 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( P_i )</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>( d_i^+ )</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( d_i^- )</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.2254</td>
<td>0</td>
</tr>
<tr>
<td>Achieved</td>
<td>31.9</td>
<td>68.04</td>
<td>6.1</td>
<td>-748.8</td>
<td>0.45</td>
<td>3.15</td>
</tr>
<tr>
<td>Goal</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
This is not interesting result because liquidity is in first ranking class. Thus we must solve our model by LGP. Second attempt is illustrated in table 3.3. We gave weights to priority classes as shown in following table. The obtained results were within 9′:12″ and 368 iterations:

Table 3.3 results for (8) using weak priority weights

<table>
<thead>
<tr>
<th>Objective</th>
<th>$f_1$</th>
<th>$f_2$</th>
<th>$f_3$</th>
<th>$f_4$</th>
<th>$f_5$</th>
<th>$f_6$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P_i$</td>
<td>10</td>
<td>10^4</td>
<td>10^2</td>
<td>1</td>
<td>10^5</td>
<td>10^3</td>
</tr>
<tr>
<td>$d_k^+$</td>
<td>4.02</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$d_k^-$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Achieved</td>
<td>35.92</td>
<td>68.04</td>
<td>6.1</td>
<td>-748.8</td>
<td>0.68</td>
<td>3.15</td>
</tr>
<tr>
<td>Goal</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

$f_i$ is in priority class $P_i$, we can also change priority coefficients and increase their differences and make priorities more prominent to obtain better results.

Finally we solved this model by Lingo as table 3.4 shows the results were obtained within 9′:51″ and 108 iterations:

Table 3.4 results for (8) using more significant priority weights

<table>
<thead>
<tr>
<th>Objective</th>
<th>$f_1$</th>
<th>$f_2$</th>
<th>$f_3$</th>
<th>$f_4$</th>
<th>$f_5$</th>
<th>$f_6$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P_i$</td>
<td>10^2</td>
<td>10^8</td>
<td>10^4</td>
<td>1</td>
<td>10^{10}</td>
<td>10^6</td>
</tr>
<tr>
<td>$d_k^+$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>53.26</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$d_k^-$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Achieved</td>
<td>31.9</td>
<td>68.04</td>
<td>6.1</td>
<td>-695.54</td>
<td>0.68</td>
<td>3.15</td>
</tr>
<tr>
<td>Goal</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

It is obvious from the above table that all of the objectives except risk have been exactly satisfied. The achieved value for risk is less than its goal and this is a good result and there is no need for changing coefficients of priorities.
4. Conclusions

In this paper, we investigated the Lexicographic goal programming, for portfolio optimization problem. Here we specified goals and rankings of objectives without DM’s opinion. For future research we can make our approach more interactive by considering DM's preferences. For example it is possible to obtain weights of objectives by pair wise comparative approach or any other approach and rank them according to these weights; In this case the goals could be obtained from weights by the same way we used here.

References


