MULTIOBJECTIVE OPTIMIZATION: PORTFOLIO OPTIMIZATION BASED ON GOAL PROGRAMMING METHODS

MARY CATHERINE ROBERTS, AVERY ST. DIZIER, AND JOSHUA VAUGHAN

Abstract. While solution methods are well-known for optimization problems with a single objective function, there are many common real world scenarios in which a single function does not suffice. Multiobjective or multicriteria optimization is the branch of optimization that deals with the case of two or more objective functions. An important example is the biobjective problem that arises in portfolio optimization. In this paper we will examine the use of the weighted sum of deviations and Chebyshev goal programming methods to find optimal allocations of capital in various assets that attains a desired level of return with a minimal amount of risk.

1. Introduction

Optimization problems play a decisive role in helping investors make informed decisions about their investments and strategies. However, most simple modeling techniques only optimize exactly one objective. The single-objective optimization problem seeks to minimize (or maximize) a single function $f : \mathbb{R}^N \to \mathbb{R}$ over its domain. That is, it attempts to compute

$$\min_{x \in \mathbb{R}^N} f(x) \tag{1.1}$$

The function f is called the *objective function*.

Often it is desirable to restrict the domain by introducing additional constraints on the vectors x over which we minimize f. When we do this, equation **??** becomes what is called a *constrained optimization problem*. The formulation for the single-objective constrained optimization problem is

$$\min_{x \in C} f(x) \tag{1.2}$$

where

$$C = \{x : h(x) = 0, g(x) \leq 0\}$$

is the set of equality and inequality constraints. If a vector $x \in \mathbb{R}^N$ satisfies the constraints of the problem, that is, if $x \in C$, then x is said to be a *feasible solution* of **??**. The set of all feasible solutions is called the *feasible region*, and the image of the feasible region under f is called the *objective space*.

Single-objective constrained optimization problems are enticing because solution methods are well-known and often only involve concepts from calculus. However in many real-world scenarios, the single-objective approach proves inadequate. The portfolio optimization problem is one such instance. When creating an investment portfolio, the primary goal for investors is to maximize profit while minimizing risk. Since the return and risk of any investment portfolio are closely interrelated, investors need ways to balance the inherent risk-return trade-off.

To accurately address the problem of portfolio optimization, we require multiple objective functions. The multiple-objective constrained optimization problem can be stated as

$$\min_{x \in C} F(x) = \begin{bmatrix} f_1(x) \\ f_2(x) \\ \vdots \\ f_N(x) \end{bmatrix} \quad \text{for} \quad n \geq 2, \tag{1.3}$$

where

$$C = \{x : h(x) = 0, g(x) \leq 0\}.$$

In addition, we require that $F : \mathbb{R}^n \to \mathbb{R}^N$, $h : \mathbb{R}^n \to \mathbb{R}^{Ne}$, and $g : \mathbb{R}^n \to \mathbb{R}^{Ni}$ are twice continuously differentiable, where n denotes the number of variables, N the number of objectives, and Ne and Ni the number of equality and inequality constraints, respectively. Feasible solutions, the feasible region, and the objective space are defined analogously to the single-objective problem. A solution x to the multiobjective problem **??** is feasible if $x \in C$, the set of all such $x \in \mathbb{R}^n$ forms the feasible region, and the image of the feasible region under F forms the objective space, a subset of $\mathbb{R}^N$.

Since a single vector will not likely optimize each objective function simultaneously, we must introduce a way to measure the optimality of a solution across multiple objective functions. This leads us to the following definition.

Definition. The vector $F(\hat{x})$ is said to *dominate* the vector $F(\bar{x})$ if and only if $f_i(\hat{x}) \leq f_i(\bar{x})$ for all $i \in \{1, 2, \ldots, N\}$ and $f_j(\hat{x}) < f_j(\bar{x})$ for some $j \in \{1, 2, \ldots, N\}$. A point $x^* \in C$ is said to be *locally Pareto optimal* if and only if there exists an open neighborhood $\mathcal{B}(x^*)$ of x^* such that $F(x^*)$ dominates $F(x)$ for all $x \in \mathcal{B}(x^*) \cap C$. If $F(x^*)$ dominates $F(x)$ for all $x \in C$, then x^* is said to be *globally Pareto optimal*.

Although finding a globally Pareto optimal point is favorable, most methods can guarantee that a point is at most locally optimal. In this paper, we treat the problem of portfolio optimization as a biobjective optimization problem, taking risk and return as our objective functions.

2. Statement of the Problem

In the general biobjective portfolio optimization problem, we consider an investor who is building a portfolio consisting of n assets, and we let $\{1, 2, ..., n\}$ denote the set of these assets. We assume that the investment period for each asset is one year, and measure the return on a portfolio as the percentage change in capital over that year. For this reason, we may assume that the total amount of capital to be invested is equal to 1, and consider the allocation of capital among the different assets as a percentage of capital. We further assume that the investor will distribute all of his capital among the n assets. We define the *allocation vector* to be the vector $x = (x_1, x_2, ..., x_n) \in \mathbb{R}^n$ where x_i denotes the percent of capital invested in asset i for $i = 1, 2, \ldots, n$. Since the investor has to allocate all of his available capital. we obtain the constraint on that $\sum_{i=1}^{n} x_i = 1$.

Let $r = (r_1, r_2, ..., r_n) \in \mathbb{R}^n$ where r_i is the mean yearly return for asset i over a given period of time. To represent return as a function, we define

$$f_1(x) = x \cdot r^T = \sum_{i=1}^{n} x_i r_i.$$

This function tells us the expected return on a portfolio given an allocation vector. To measure the risk of a portfolio, we consider the covariance matrix

$$V = \begin{pmatrix} \sigma_{11} & \dots & \sigma_{1n} \\ \vdots & \ddots & \vdots \\ \sigma_{n1} & \dots & \sigma_{nn} \end{pmatrix}$$

where σ_{ij} is the covariance of asset i and asset j, and σ_{ii} is the variance of asset i. Using this matrix, we can describe risk using the matrix product

$$f_2(x) = x \cdot V \cdot x^T = \sum_{i,j=1}^{n} x_i x_j \sigma_{ij}.$$

This equation denotes the percent of the total assets that are vulnerable to external forces if i and j move together. When we calculate the covariance of an asset with itself, we calculate the variance, which is a measurement of how far, on average, the data of a set is away from the mean of the data. The greater the variance, the riskier the asset. One important characteristic about the covariance matrix is always positive semi-definite, meaning that $x \cdot V \cdot x^T \geq 0$ for all $x \in \mathbb{R}^n$.

Then the biobjective portfolio optimization problem can be stated as

$$\min_{x \in C} \begin{bmatrix} x \cdot r^T \\ x \cdot V \cdot x^T \end{bmatrix} \tag{2.1}$$

where $C = \{x \in \mathbb{R}^n : \sum_{i=1}^{n} x_i = 1, x_i \geq 0 \text{ for } i = 1, 2, \dots, n\}$. Potential investors must decide how to allocate capital among a variety of assets with varying levels of risk and return. When the correlation between the values of two assets is highly positive, the returns on these investments will move together. So we want create a diversified portfolio consisting of several unrelated assets that yield high levels of return. Ideally, this helps balance out the rise and fall of returns across different markets by avoiding strong positive correlations between assets. Since different investors are willing to accept varying amounts of risk, there is no single optimal strategy. An individual investor's optimal portfolio depends upon their risk preferences.

Ultimately, we seek an optimal allocation of capital among potential investments, including US government securities, various stocks, and gold. In this paper we consider six such assets: US Treasury Bills (TBills), US Treasury Bonds (TBonds), stocks in the NASDAQ Composite Index (NASDAQ), stocks in the Dow Jones Index (DowJones), and stocks in the Standard & Poor 500 Index (S&P 500), and gold. We observe annual return data for each asset from 1980-2011.

3. Goal Programming Methods

Since solution techniques to the single objective optimization problem are well-known, one plausible way to solve the biobjective problem is to combine the two objective functions into a single aggregate objective function that incorporates each objective function in a meaningful way. However, there is no canonical way to build such a function. We will examine two methods for generating the new objective function in the following section, but we must first introduce the concept of goal programming.

Goal programming methods work by fixing a goal value for each objective function and measuring the deviations of the values of the objective functions from

their goal value over the feasible region. That is, instead of optimizing the objective functions, we choose a goal target value for each and then minimize the difference between each function and its goal. Hence we are not optimizing the functions themselves, but rather forcing them to be as close as possible to their goal values. More formally, the biobjective optimization problem can be reformulated into a goal programming problem by assigning to each f_i a goal value g_i and minimizing the deviation $(f_i - g_i)^+$ for $i = 1, 2$ over the feasible region, where $+$ refers to the positive part of the function.

For our problem, we define $g_1 = r^*$, where r^* denotes the desired level of return on the portfolio, and let $g_2 = 0$. We call the vector $g = (r^*, 0) \in \mathbb{R}^2$ the *goal vector*. This is strictly ideal; we do not require that the goal vector lie in the objective space. To achieve the necessary flexibility for investor preferences to be taken into account, we also introduce a weight vector $w = (w_1, w_2) \in \mathbb{R}^2$ with $w_1, w_2 \in (0, 1)$, such that $w_1 + w_2 = 1$. However as stated, this is still a multiobjective problem; but the weighted sums of deviations and Chebyshev goal programming methods provide ways to reduce our reformulated problem into a single objective problem. Both methods can be generalized to the N-dimensional case, with goal vector $g = (g_1, g_2, \ldots, g_N) \in \mathbb{R}^N$ and weight vector $w = (w_1, w_2, \ldots, w_N)$ such that $w_1, w_2, \ldots, w_N \in (0, 1)$ and $\sum_{i=1}^N w_i = 1$.

3.1. **Method 1: Weighted Sum of Deviations.** The first method we consider is the called the weighted sum of deviations (WSD) method. In general, the WSD method is formulated as

$$\min_{x \in C} \sum_{j=1}^{N} w_j (f_j(x) - g_j)^+$$

where C denotes the set of constraints defined in Equation **??**. Here, the function

$$F = \sum_{j=1}^{N} w_j (f_j - g_j)^+$$

is the aggregate objective function mentioned previously.

Using the WDS method for the biobjective portfolio optimization problem, we obtain the aggregate objective function F given by

$$F(x) = w_1 \left(\left[\sum_{i=1}^{n} x_i r_i \right] - r^* \right)^+ + w_2 \left(\sum_{i,j=1}^{n} x_i x_j \sigma_{ij} \right)^+$$

$$= w_1 \left(\left[\sum_{i=1}^{n} x_i r_i \right] - r^* \right)^+ + w_2 \sum_{i,j=1}^{n} x_i x_j \sigma_{ij}$$

since V is positive semi-definite.

Then our original biobjective problem can be reformulated as

$$\min_{x \in C} F(x), \tag{3.1}$$

where $C = \{x \in \mathbb{R}^n : \sum_{i=1}^n x_i = 1 \text{ and } x_i \geq 0 \text{ for } i = 1, 2, \ldots, n\}$.

For the biobjective problem, the constraints $w_1 + w_2 = 1$ and $w_1, w_2 \in (0, 1)$ on the weight vector $w = (w_1, w_2)$ imply that $w_2 = 1 - w_1$, and hence we can rewrite

$w = (w_1, 1 - w_1)$ subject to the constrain that $w_1 \in (0, 1)$. Then it is easy to see that the aggregate objective function becomes

$$F(x) = w_1 \left(\left[\sum_{i=1}^{n} x_i r_i \right] - r^* \right)^+ + (1 - w_1) \sum_{i,j=1}^{n} x_i x_j \sigma_{ij}.$$

That is, F is a convex combination of the functions $|x \cdot r^T - r^*|$ and $|x \cdot V \cdot x^T|$. So we see that combining the two objective functions in this way yields a new curve that is a weighted average of the deviations of each objective from its goal.

3.2. **Method 2: Chebyshev Goal Programming.** The Chebyshev goal programming method uses essentially the same concepts as the WSD method, except instead of minimizing the sum of deviations, we minimize only the maximum weighted deviation of any function from its goal. Mathematically, we the Chebyshev method is formulated as

$$\min_{x \in C} \left[\max_j w_j (f_j(x) - g_j)^+ \right] \text{ for } j = 1, 2, \ldots N,$$

taking C to be the same set of constraints as defined in Equation **??**. In this method, the aggregate objective function is

$$F = \max_j (w_j |f_j - g_j|) \text{ for } j = 1, 2, \ldots, N.$$

In the Chebyshev method, the aggregate objective function for the portfolio optimization problem is

$$F(x) = \max\{w_1 (f_1 - r^*)^+, w_2 (f_2)^+\}$$

$$= \max \left\{ w_1 \left(\left[\sum_{i=1}^{n} x_i r_i \right] - r^* \right)^+, w_2 \sum_{i,j=1}^{n} x_i x_j \sigma_{ij} \right\}$$

So the biobjective problem becomes

$$\min_{x \in C} F(x), \tag{3.2}$$

where $C = \{x \in \mathbb{R}^n : \sum_{i=1}^{n} x_i = 1 \text{ and } x_i \geq 0 \text{ for } i = 1, 2, \ldots, n\}$.

Here, instead of minimizing the weighted average of the deviations, as in the WSD method, we minimize the largest deviation, bearing in mind that when the largest deviation is minimized, the other deviation will be small as well.

4. A Fundamental Theorem

Until now, we have merely asserted that we can reformulate our biobjective problem into a goal programming problem to generate an optimal solution. But we must prove that an optimal solution to the goal programming problem is also an optimal solution to the original problem. This leads us to the following theorem. The general case is also true, but we restrict ourselves to the biobjective case.

Theorem. *If x^* is the unique minimizer for the goal programming problem, then x^* is a Pareto optimal point for the biobjective optimization problem* **??**.

Proof. We will prove this theorem for the goal programming problem from the WSD method. Assume that x^* is the unique global minimizer of

$$\min_{x \in C}[w_1(x \cdot r^T - r^*)^+ + w_2(x \cdot V \cdot x^T)] \tag{4.1}$$

and suppose that x^* is not a global Pareto optimal point for the biobjective problem (reference the biobjective problem). Then there exists a point $\bar{x} \in C$ such that either $\bar{x} \cdot r^T < x^* \cdot r^T$ or $\bar{x} \cdot V \cdot \bar{x} < x^* \cdot V \cdot x^{*T}$. Hence we have that $w_1(\bar{x} \cdot r^T - r^*)^+ + w_2(\bar{x} \cdot V \cdot \bar{x}) < w_1(x^* \cdot r^T - r^*)^+ + w_2(x^* \cdot V \cdot x^{*T})$. But this implies that $\bar{x}$ is a global minimizer of **??**, which is a contradiction. The case for the Chebyshev method follows similarly. □

Although this theorem deals with global Pareto optimal points, the result holds for local Pareto optimal points as well, and the proof follows similarly.

5. The Pareto Front

By varying our choices of w, we can generate a set called the *Pareto optimal curve* or *Pareto front*. The Pareto front maps the relation between our two objective functions. That is, we map variance versus expected return and by varying weights we generate the Pareto curve. As we vary our choice of w we obtain a sequence of optimal portfolios on the Pareto front ranging from the portfolio with the smallest overall variance to the portfolio with the highest expected return. By generating points on the Pareto front, we can approximate the actual Pareto optimal curve. Once we have this approximation, the investor no longer has to consider the entire objective space when choosing an optimal allocation vector, but only the portion that is on the Pareto front.

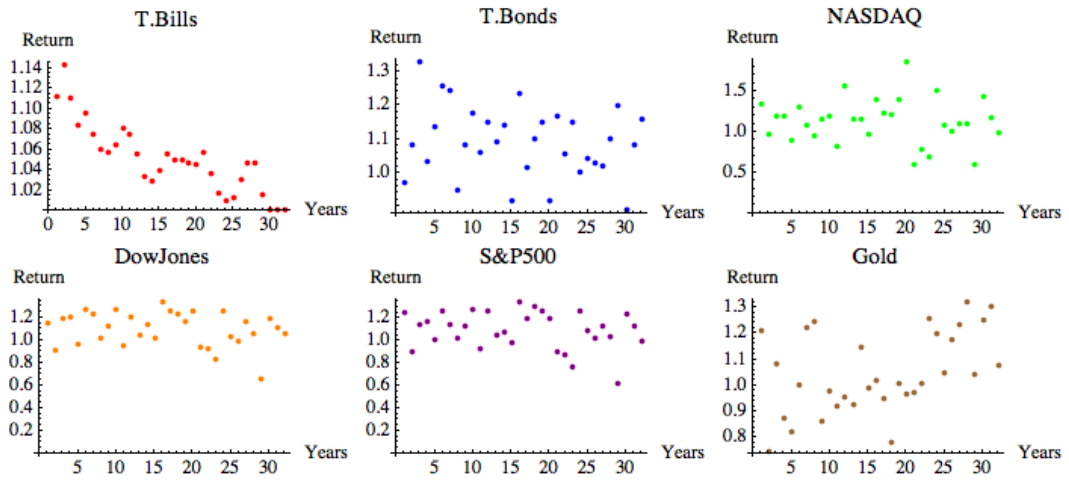
The Pareto front occurs on the boundary of the objective space. When the objective space is convex, we can often generate the entire Pareto front using the methods we discussed here. However, if the objective space is not convex, then these methods are not sufficient for generating the entire curve, and some Pareto points will be missing. Other techniques have been developed for this case, and other cases when traditional methods are not successful in generating the entire curve. In (NBI) so and so talks about the Normal Boundary Intersection method, which generate at least as much of the Pareto curve as the methods discussed in this paper.

6. Results

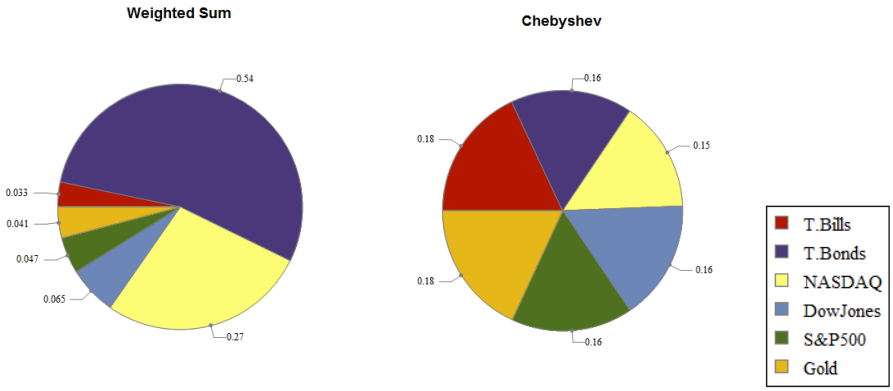
Using the historic return data compiled in Appendix A, we created and worked an example biobjective portfolio optimization problem to demonstrate the uses of the two methods. For our particular problem, we elected to let $n = 6$, that is we chose to have six assets between which to allocate our capital. We chose T.Bills, T.Bonds, NASDAQ, Dow Jones, S&P500, Gold for our six assets. For simplicity, we decided to use an even weighting, $w = (\frac{1}{2}, \frac{1}{2})$, and we chose $r^* = 1.1$, a 10% return. In Mathematica, we entered the formulas for both methods and evaluated the problem using the FindMinimum command on each function. For a general standard of comparison, we also worked the problem twice using only single objective functions to demonstrate the efficiency of the biobjective methods. First we focused only on maximizing profit without regard to risk, that is finding

$\min f_1(x) - r^*$, and then we focused only on minimizing risk without regard to profit, $\min f_2(x)$.

Using Mathematica we obtained the following scatter plots for the data. (The code is included in Appendix B).



The results of our calculations using the biobjective methods are summarized in the following tables:

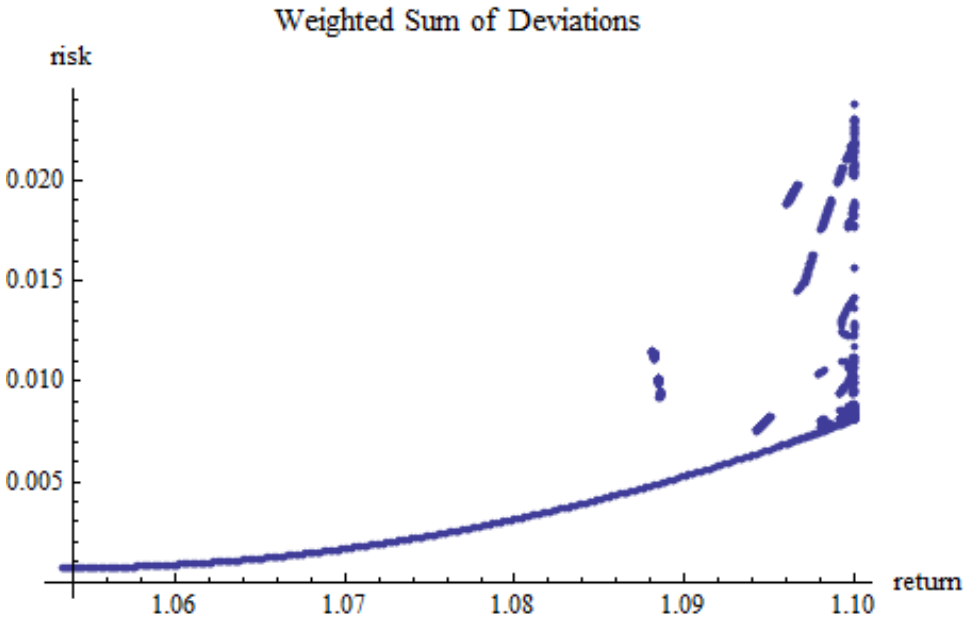


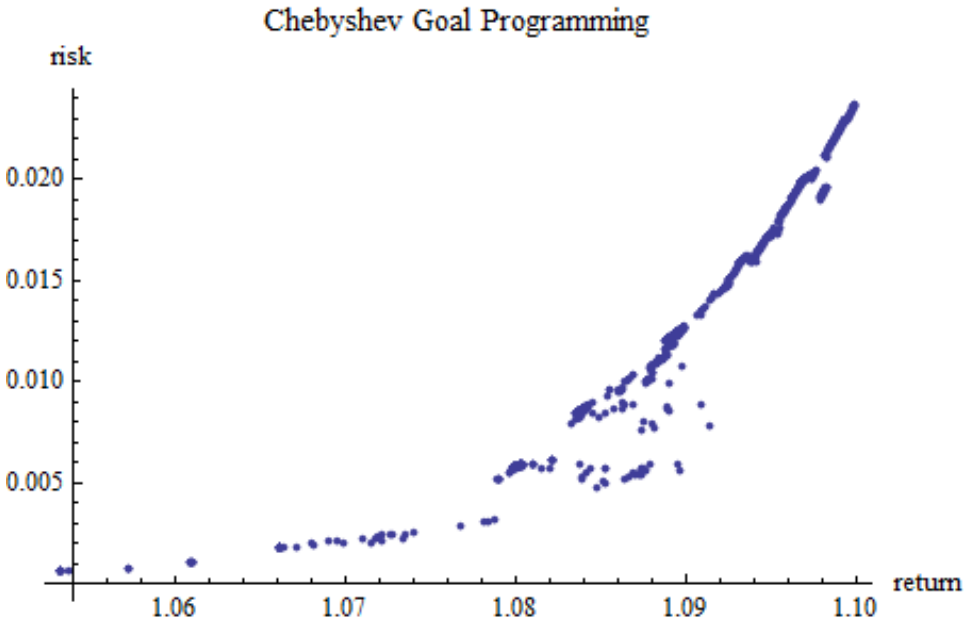
For comparison, we juxtapose the two biobjective method results with those obtained by taking either function to be the single objective and optimizing it exclusively.

	min $f_1(x) - r^*$	min $f_2(x)$
T.Bills	12.00%	84.00%
T.Bonds	12.00%	3.20%
NASDAQ	39.00%	0.75%
DowJones	13.00%	0.28%
S&P500	12.00%	0.22%
Gold	12.00%	12.00%
...	...	...
Return	1.09669	1.05327
Variance	0.0198787	0.00069933

	Weighted Sum	Chebyshev
T.Bills	3.30%	18.00%
T.Bonds	54.00%	16.00%
NASDAQ	27.00%	15.00%
DowJones	6.50%	16.00%
S&P500	4.70%	16.00%
Gold	4.10%	18.00%
...	...	...
Return	1.09925	1.08369
Variance	0.00955972	0.00853878

By varying our choice of weight vector w, we obtained the following approximations for the Pareto front using both methods.





7. Conclusion

The biobjective portfolio optimization problem highlights the difficulty that arises in optimization problems with multiple objectives. We chose to treat the problem as a function of only the allocation vector x, however we could reformulate **??** and **??** as a quadratic programming problem, taking x and w as variables. For the WSD method, we get the aggregate objective function $F : \mathbb{R}^n \times \mathbb{R}^2 \to \mathbb{R}$ given by

$$F(x,w) = w_1 \left(\left[\sum_{i=1}^{n} x_i r_i \right] - r^* \right)^+ + w_2 \sum_{i,j=1}^{n} x_i x_j \sigma_{ij}.$$

Doing this we get the equivalent quadratic programming problem

$$\min_{(x,w)\in C} F(x), \tag{7.1}$$

where $C = \{(x,w) \in \mathbb{R}^n \times \mathbb{R}^2 : \sum_{i=1}^{n} x_i = 1, w_1 + w_2 = 1, x_i \geq 0 \text{ for } i = 1,2,\ldots,n, \text{ and } w_j \geq 0 \text{ for } j = 1,2\}$. In the Chebyshev method, we obtain the quadratic programming problem

$$\min_{(x,w)\in C} F(x), \tag{7.2}$$

where $C = \{(x,w) \in \mathbb{R}^n \times \mathbb{R}^2 : \sum_{i=1}^{n} x_i = 1, w_1 + w_2 = 1, x_i \geq 0 \text{ for } i = 1,2,\ldots,n, \text{ and } w_j \geq 0 \text{ for } j = 1,2\}$. By viewing this as a quadratic problem, we could simultaneously optimize in both w and x, giving the investor an even more precise solution.

For our portfolio optimization problem, we used only yearly data for each asset. However, the values of assets can vary a great deal over the course of a year. We could modify our problem to incorporate data that gives a more accurate description of how the value of each asset changes, and this would give us a more accurate model for portfolio optimization. Another possible addition could include research into

other types of assets, including options which are defined as a contract conveying a right to buy or sell designated securities, commodities, or property interest at a specified price during a stipulated period, for multi-year investments to maximize returns.

8. Acknowledgments

We would like to thank our graduate mentor Emily McHenry, and our professor Humberto Munoz for all their help and guidance on this project.

References

[1] Marker, R.T. & Arora, J.S. The weighted sum method for multi-objective optimization: new insights. *Struct Multidisc Optim*, 41:853-862, 2010.

[2] Das, I. & Dennis, J.E. Normal boundary intersection: a new method for generating the Pareto surface in nonlinear multicriteria optimization problems. *SIAM Journal on Optimization*, 8:631-657, 1998.

[3] Collete, Y. & Siarry, P. *Multiobjective optimization: principles and case studies*. Springer, Berlin, 2003.

[4] Steuer, R.E. *Multiple criteria optimization: theory, computations, and application*. John Wiley & Sons, Inc., New York, 1986.

9. Appendix A

Table 1. This is a table of the historical return data we used in constructing our example problem

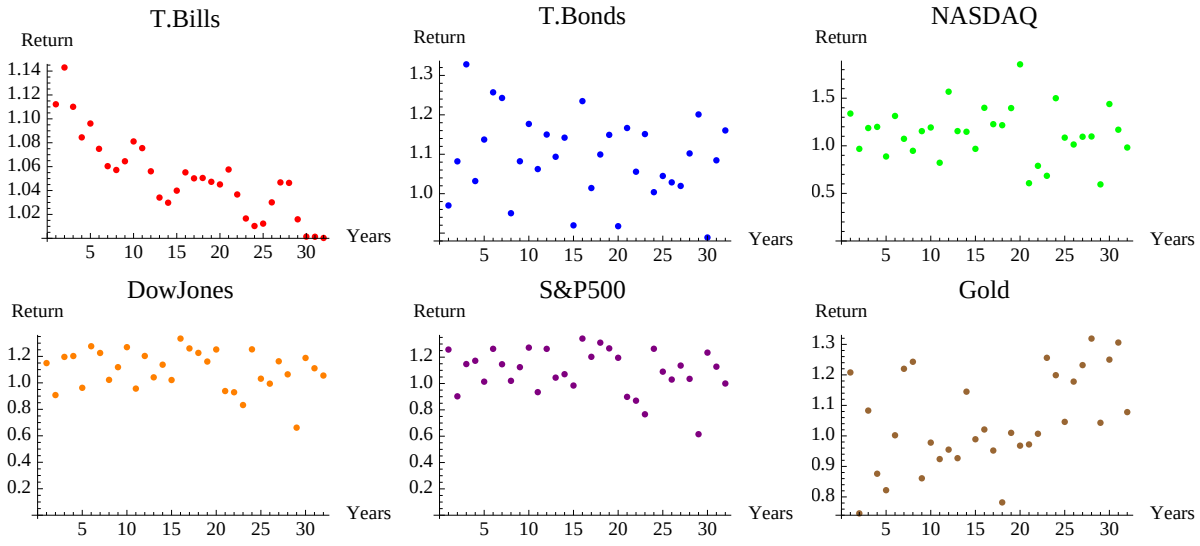
Year	T Bills	T Bonds	NASDAQ	DowJones	S&P 500	Gold
1980	1.1122	0.9701	1.3388	1.1493	1.2577	1.208
1981	1.143	1.082	0.9679	0.9077	0.9027	0.746
1982	1.1101	1.3281	1.1867	1.1961	1.1476	1.083
1983	1.0845	1.032	1.1987	1.2027	1.1727	0.876
1984	1.0961	1.1373	0.8878	0.9626	1.014	0.822
1985	1.0749	1.2571	1.3136	1.2766	1.2633	1.002
1986	1.0604	1.2428	1.0736	1.2258	1.1462	1.22
1987	1.0572	0.9504	0.9474	1.0226	1.0203	1.243
1988	1.0645	1.0822	1.1541	1.1185	1.124	0.861
1989	1.0811	1.1769	1.1926	1.2696	1.2725	0.978
1990	1.0755	1.0624	0.822	0.9566	0.9344	0.924
1991	1.0561	1.15	1.5684	1.2032	1.2631	0.955
1992	1.0341	1.0936	1.1545	1.0417	1.0446	0.927
1993	1.0298	1.1421	1.1475	1.1372	1.0706	1.145
1994	1.0399	0.9196	0.968	1.0214	0.9846	0.989
1995	1.0552	1.2348	1.3992	1.3345	1.3411	1.021
1996	1.0502	1.0143	1.2271	1.2601	1.2026	0.952
1997	1.0505	1.0994	1.2164	1.2264	1.3101	0.782
1998	1.0473	1.1492	1.3963	1.161	1.2667	1.01
1999	1.0451	0.9175	1.8559	1.2522	1.1953	0.968
2000	1.0576	1.1666	0.6071	0.9382	0.8986	0.972
2001	1.0367	1.0557	0.7895	0.929	0.8696	1.007
2002	1.0166	1.1512	0.6847	0.8324	0.7663	1.256
2003	1.0103	1.0038	1.5001	1.2532	1.2638	1.199
2004	1.0123	1.0449	1.0859	1.0315	1.0899	1.046
2005	1.0301	1.0287	1.0137	0.9939	1.03	1.178
2006	1.0468	1.0196	1.0952	1.1629	1.1362	1.232
2007	1.0464	1.1021	1.0981	1.0643	1.0353	1.319
2008	1.0159	1.201	0.5946	0.6616	0.6151	1.043
2009	1.0014	0.8888	1.4389	1.1882	1.2345	1.25
2010	1.0013	1.0846	1.1691	1.1102	1.1278	1.306
2011	1.0003	1.1604	0.982	1.0553	1.0	1.078

Portfolio Optimization

By Mary Catherine Roberts, Avery St. Dizier, and Joshua Vaughan

```
(*Annual return data for each investment option in percents*)

tbills = {11.22, 14.30, 11.01, 8.45, 9.61, 7.49, 6.04, 5.72, 6.45,
   8.11, 7.55, 5.61, 3.41, 2.98, 3.99, 5.52, 5.02, 5.05, 4.73, 4.51,
   5.76, 3.67, 1.66, 1.03, 1.23, 3.01, 4.68, 4.64, 1.59, .14, .13, .03};

tbonds = {-2.99, 8.20, 32.81, 3.20, 13.73, 25.71, 24.28, -4.96, 8.22, 17.69,
   6.24, 15.00, 9.36, 14.21, -8.04, 23.48, 1.43, 9.94, 14.92, -8.25, 16.66,
   5.57, 15.12, .38, 4.49, 2.87, 1.96, 10.21, 20.10, -11.12, 8.46, 16.04};

nasdaq = {33.88, -3.21, 18.67, 19.87, -11.22, 31.36, 7.36, -5.26, 15.41, 19.26,
   -17.80, 56.84, 15.45, 14.75, -3.20, 39.92, 22.71, 21.64, 39.63, 85.59, -39.29,
   -21.05, -31.53, 50.01, 8.59, 1.37, 9.52, 9.81, -40.54, 43.89, 16.91, -1.80};

dowjones = {14.93, -9.23, 19.61, 20.27, -3.74, 27.66, 22.58, 2.26, 11.85, 26.96,
   -4.34, 20.32, 4.17, 13.72, 2.14, 33.45, 26.01, 22.64, 16.10, 25.22, -6.18,
   -7.10, -16.76, 25.32, 3.15, -0.61, 16.29, 6.43, -33.84, 18.82, 11.02, 5.53};

sp500 = {25.77, -9.73, 14.76, 17.27, 1.40, 26.33, 14.62, 2.03, 12.40, 27.25,
   -6.56, 26.31, 4.46, 7.06, -1.54, 34.11, 20.26, 31.01, 26.67, 19.53, -10.14,
   -13.04, -23.37, 26.38, 8.99, 3.00, 13.62, 3.53, -38.49, 23.45, 12.78, 0.00};

gold = {20.8, -25.4, 8.3, -12.4, -17.8, 0.2, 22.0, 24.3, -13.9,
   -2.2, -7.6, -4.5, -7.3, 14.5, -1.1, 2.1, -4.8, -21.8, 1.0, -3.2,
   -2.8, 0.7, 25.6, 19.9, 4.6, 17.8, 23.2, 31.9, 4.3, 25.0, 30.6, 7.8};

(*Collective list of all data sets*)
data = {tbills, tbonds, nasdaq, dowjones, sp500, gold};

(*Converting the percents to positive decimals*)
For[i = 1, i ≤ Length[data], i++,
  For[j = 1, j ≤ Length[data[[i]]], j++,
   If[data[[i]][[j]] ≥ 0, data[[i]][[j]] = data[[i]][[j]] / 100 + 1,
    data[[i]][[j]] = 1 + data[[i]][[j]] / 100]]];
```

```
(*Graphs of the returns of each
 investment option relative to years after 1980*)
options = {"T.Bills", "T.Bonds", "NASDAQ", "DowJones", "S&P500", "Gold"};
For[i = 1; plots = {};
 colors = {Red, Blue, Green, Orange, Purple, Brown}, i ≤ Length[data], i++,
 plots = Append[plots,
   ListPlot[data[[i]], PlotStyle → colors[[i]], AxesLabel → {"Years"
, "Return"}, PlotLabel → options[[i]]]]]
Grid[Partition[plots, 3]]

(*Constructing a chart of the data by year*)
For[dates = {}; i = 1, i ≤ 32, i++, dates = Append[dates, (1979 + i)]];
Grid[Join[{Join[{""}, options]}, Join[{dates}, data] // Transpose],
 Frame → All, Background → {{Gray, None}, {LightGray, None}}]

(*Constructing a second chart with the mean, variance and standard deviation*)
Grid[Transpose[Join[{{"", "Mean", "Variance", "Standard Deviation"}},
   Transpose[Join[{options}, {Mean[data // Transpose],
      Variance[data // Transpose], StandardDeviation[data // Transpose]}]]]],
 Frame → All, Background → {{Gray, None}, {LightGray, None}}]
```



	T.Bills	T.Bonds	NASDAQ	DowJones	S&P500	Gold
1980	1.1122	0.9701	1.3388	1.1493	1.2577	1.208
1981	1.143	1.082	0.9679	0.9077	0.9027	0.746
1982	1.1101	1.3281	1.1867	1.1961	1.1476	1.083
1983	1.0845	1.032	1.1987	1.2027	1.1727	0.876
1984	1.0961	1.1373	0.8878	0.9626	1.014	0.822
1985	1.0749	1.2571	1.3136	1.2766	1.2633	1.002
1986	1.0604	1.2428	1.0736	1.2258	1.1462	1.22
1987	1.0572	0.9504	0.9474	1.0226	1.0203	1.243
1988	1.0645	1.0822	1.1541	1.1185	1.124	0.861
1989	1.0811	1.1769	1.1926	1.2696	1.2725	0.978
1990	1.0755	1.0624	0.822	0.9566	0.9344	0.924
1991	1.0561	1.15	1.5684	1.2032	1.2631	0.955
1992	1.0341	1.0936	1.1545	1.0417	1.0446	0.927
1993	1.0298	1.1421	1.1475	1.1372	1.0706	1.145
1994	1.0399	0.9196	0.968	1.0214	0.9846	0.989
1995	1.0552	1.2348	1.3992	1.3345	1.3411	1.021
1996	1.0502	1.0143	1.2271	1.2601	1.2026	0.952
1997	1.0505	1.0994	1.2164	1.2264	1.3101	0.782
1998	1.0473	1.1492	1.3963	1.161	1.2667	1.01
1999	1.0451	0.9175	1.8559	1.2522	1.1953	0.968
2000	1.0576	1.1666	0.6071	0.9382	0.8986	0.972
2001	1.0367	1.0557	0.7895	0.929	0.8696	1.007
2002	1.0166	1.1512	0.6847	0.8324	0.7663	1.256
2003	1.0103	1.0038	1.5001	1.2532	1.2638	1.199
2004	1.0123	1.0449	1.0859	1.0315	1.0899	1.046
2005	1.0301	1.0287	1.0137	0.9939	1.03	1.178
2006	1.0468	1.0196	1.0952	1.1629	1.1362	1.232
2007	1.0464	1.1021	1.0981	1.0643	1.0353	1.319
2008	1.0159	1.201	0.5946	0.6616	0.6151	1.043
2009	1.0014	0.8888	1.4389	1.1882	1.2345	1.25
2010	1.0013	1.0846	1.1691	1.1102	1.1278	1.306
2011	1.0003	1.1604	0.982	1.0553	1.	1.078

	T.Bills	T.Bonds	NASDAQ	DowJones	S&P500	Gold
Mean	1.05136	1.09216	1.12736	1.09833	1.09379	1.04994
Variance	0.00116085	0.0109041	0.0761365	0.0225648	0.0280729	0.0243316
Standard Deviation	0.0340712	0.104423	0.275928	0.150216	0.16755	0.155986

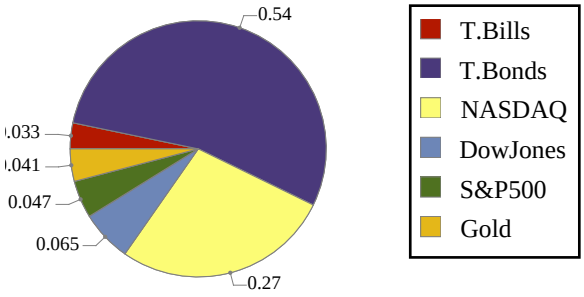
```
(*Definitions of our objective functions, goals,
weights, and other variables we will need*)

(*The historical mean return value of each investment option*)
meanReturn = Mean[Transpose[data]];

(*The covariant matrix of our data with entry aij =
 Covariance[option i, option j] and aii = Variance[option i]*)
covariants = Covariance[Transpose[data]];

(*A list of the variables representing the
 percent each variable will have in our allocation*)
variables = Map[Subscript[x, #] &, options];

(*A list of our objective functions*)
f = {variables.meanReturn, variables.covariants.variables} // Simplify;

(*A list of our goals for each objective*)
g = {1.1, 0};

(*A list of our weights for each objective*)
w = {.5, .5};

(*Our standard constraints: xi≥0 for every x and that ∑xi=1*)
constraints = Total[variables] == 1 && Apply[And, Thread[Greater[variables, 0]]];

(*The Weighted Sum of Deviations function is derived by taking a weighted
 sum of the deviations of each objective function from its goal*)
weightedSum = Sum[Abs[(f[[i]] - g[[i]])] w[[i]], {i, 1, Length[f]}] // Simplify;

(*The Chebyshev function is derived by finding the
 maximum deviation of any objective function from its goal*)
chebyshev = Max[Table[Abs[(f[[l]] - g[[l]])] w[[l]], {l, 1, Length[f]}]] // Simplify;
```

```
(*This section is our calculations using the two different methods*)

(*
                                    Weighted Sum of Deviations:
  We combine our two objective functions into a weighted average of
   their deviations from the goals using weights derived from our risk
   preferences. We minimize this function using the original constraints.
*)
results2 = Quiet[FindMinimum[{weightedSum, constraints}, variables]];

(*
                                    Chebyshev:
 We create a new function to minimize by finding
  the maximum deviation of any objective funtion from its
  respective goal and attempting to minimize just this deviation.
*)
results3 = Quiet[FindMinimum[{chebyshev, constraints}, variables]];


(*Constructing a chart summarizing the
 results of the calculations using the two methods*)
For[results = {results2, results3}; resultsD = Table[{}, {Length[results]}];
 j = 1, j ≤ Length[resultsD], j++,
 For[i = 1, i ≤ Length[variables], i++,
  resultsD[[j]] = Append[resultsD[[j]],
     PaddedForm[(variables[[i]] /. results[[j]][[2]][[i]]) * 100, {4, 2}]];]]
For[j = 1, j ≤ Length[resultsD], j++,
 For[i = 1, i ≤ Length[resultsD[[j]]], i++,
   resultsD[[j]][[i]] = StringJoin[ToString[resultsD[[j]][[i]]], "%"];]
Grid[Join[{Join[{""}, options, {"...", "Return", "Variance"}]},
   {Join[{"Weighted Sum"}, resultsD[[1]], {"..."}, f /. results[[1]][[2]]]},
   {Join[{"Chebyshev"}, resultsD[[2]], {"..."}, f /. results[[2]][[2]]]}] //
  Transpose, Frame → All, Background → {{Gray, None}, {LightGray, None}}]
```

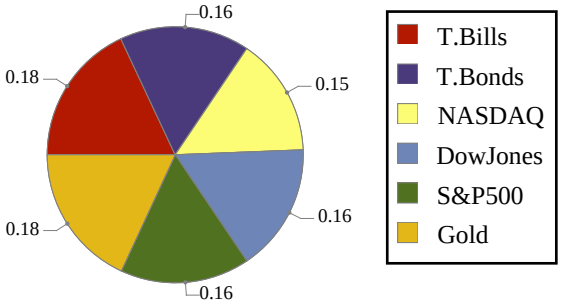
	Weighted Sum	Chebyshev
T.Bills	3.27%	18.02%
T.Bonds	53.96%	16.42%
NASDAQ	27.48%	14.94%
DowJones	6.47%	16.17%
S&P500	4.73%	16.36%
Gold	4.09%	18.09%
...	...	...
Return	1.09925	1.08369
Variance	0.00955972	0.00853878

```
(*Pie Charts of each allocation*)
For[name = {"Weighted Sum", "Chebyshev"};
 picharts = Table[{}, {Length[results]}]; i = 1, i ≤ Length[results], i++,
 picharts[[i]] = PieChart[variables /. results[[i]][[2]],
   ChartStyle → 61, PerformanceGoal → "Accuracy", ChartLegends → options,
   ChartLayout → "Stacked", PlotLabel → Style[name[[i]], "Title", FontSize → 14],
   LabelingFunction → (Placed[NumberForm[#, 2], "RadialCallout"] &)]]
Grid[Partition[picharts, 1]]
```

Weighted Sum



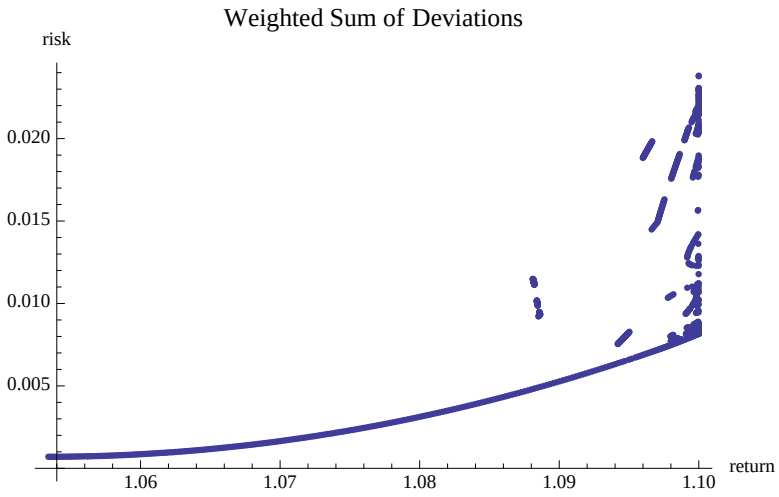
Chebyshev

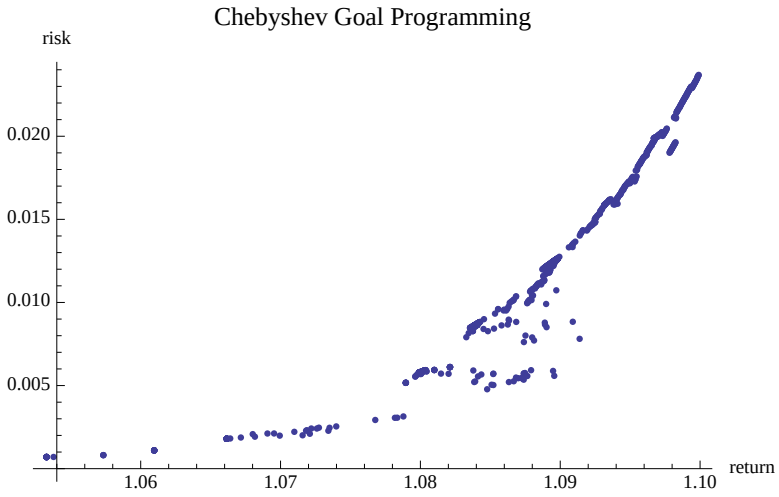


```
(*Single function versions*)
f1results = Quiet[FindMinimum[{Abs[f[[1]] - g[[1]]], constraints}, variables]];
f2results = FindMinimum[{f[[2]], constraints}, variables];

For[fresults = {f1results, f2results};
 fresultsD = Table[{}, {Length[fresults]}]; j = 1, j ≤ Length[fresultsD], j++,
 For[i = 1, i ≤ Length[variables], i++,
  fresultsD[[j]] = Append[fresultsD[[j]],
     PaddedForm[(variables[[i]] /. fresults[[j]][[2]][[i]]) * 100, {2, 2}]];]]
For[j = 1, j ≤ Length[fresultsD], j++,
 For[i = 1, i ≤ Length[fresultsD[[j]]], i++,
   fresultsD[[j]][[i]] = StringJoin[ToString[fresultsD[[j]][[i]]], "%"];]
Grid[Join[{Join[{""}, options, {"...", "Return", "Variance"}]},
   {Join[{"min f₁(x) - r*"}, fresultsD[[1]], {"..."}, f /. fresults[[1]][[2]]]},
   {Join[{"min f₂(x)"}, fresultsD[[2]], {"..."}, f /. fresults[[2]][[2]]]}] //
  Transpose, Frame → All, Background → {{Gray, None}, {LightGray, None}}]
```

	min $f_1(x) - r^*$	min $f_2(x)$
T.Bills	12.00%	84.00%
T.Bonds	12.00%	3.20%
NASDAQ	39.00%	0.75%
DowJones	13.00%	0.28%
S&P500	12.00%	0.22%
Gold	12.00%	12.00%
...	...	...
Return	1.09669	1.05327
Variance	0.0198787	0.00069933

```
(*This cell plots the Pareto fronts for
 Weighted Average and Chebyshev by varying weights*)
(*IT IS NOT EVALUATABLE.*)
experiment2[w1_] :=
  Module[{},
   w = {w1, 1 - w1};
   weightedSum = Sum[Abs[(f[[i]] - g[[i]])] w[[i]], {i, 1, Length[f]}] // Simplify;
   results2 = Quiet[FindMinimum[{weightedSum, constraints}, variables]];
   f /. results2[[2]]];

experiment3[w1_] :=
  Module[{},
   w = {w1, 1 - w1};
   chebyshev =
    Max[Table[Abs[(f[[l]] - g[[l]])] w[[l]], {l, 1, Length[f]}]] // Simplify;
   Quiet[results3 = FindMinimum[{chebyshev, constraints}, variables]];
   f /. results3[[2]]];
ListPlot[Table[experiment2[i], {i, .001, .999, .001}], PlotRange → All,
 PlotLabel → "Weighted Sum of Deviations", AxesLabel → {"return", "risk"}]
ListPlot[Table[experiment3[i], {i, .001, .999, .001}], PlotRange → All,
 PlotLabel → "Chebyshev Goal Programming", AxesLabel → {"return", "risk"}]
```


Weighted Sum of Deviations
risk
0.020
0.015
0.010
0.005
1.06
1.07
1.08
1.09
1.10
return


Chebyshev Goal Programming
risk
0.020
0.015
0.010
0.005
1.06
1.07
1.08
1.09
1.10
return