

PROGRAMMING PROJECT ONE: PORTFOLIO ALLOCATION
(DUE WEDNESDAY APRIL 30, 2008)

The object of this project is to setup and solve a nontrivial linear programming problem. The particular problem we will examine comes from Chapter 13, Section 1 in your book and you are encouraged to read the discussion found there.

We will use Matlab (which is available on the math department computers) to solve this problem. Part of the goal of this project is to become familiar with solving linear programs on a computer. I will provide a short tutorial on the use of the Matlab command “linprog” on the course webpage and possibly discuss it in class.

INTRODUCTION

We will solve the problem of selecting an appropriate mix of assets to include in an investment portfolio. Suppose we are given a selection of n potential investments. We will let R_j denote the return on investment j in the next investment period. Of course, we do not know *a priori* the values of the R_j . They represent the future return on an investment made now. We can, however, try to estimate the behavior of the R_j using historical data.

RANDOM VARIABLES

In particular, we treat the R_j as *random variables* — unknown quantities associated with a probability distribution. We might not be able to say that R_j will have a particular value in the next time period, but we are able to estimate the probability that it will have a certain value. For our purposes, we will only be interested in a single quantity associated with a random variable, the expected value.

Given a random variable R and a set of observed values x_1, \dots, x_n of R , we can estimate the expected value of R as

$$(1) \quad \mathbb{E}R \approx \frac{1}{n} \sum_j x_j.$$

Let us now consider the particular case of an investment R_j . For each investment j , we will be given a set

$$(2) \quad R_j(1), R_j(2), \dots, R_j(T)$$

for T previous observations of the return on investment j . This is the historical data available for each investment. Each observation is the return on that investment over one time interval in the past. We will then estimate the expected value of the random variable R_j as

$$(3) \quad \mathbb{E}R_j \approx \frac{1}{T} \sum_t R_j(t).$$

We interpret this quantity as the amount of money we expect to make in the next investment interval if we invest one dollar in the j^{th} investment.

PORTFOLIOS

A portfolio is determined by specifying what fraction of one’s assets are put into each asset. That is, a portfolio is a sequence of nonnegative real numbers x_1, \dots, x_n such that

$$(4) \quad x_1 + \dots + x_n = 1.$$

The return on a given portfolio in the next time period would then be

$$(5) \quad R = \sum_j x_j R_j.$$

We will call the expected value of R ,

$$(6) \quad \mathbb{E}R = \sum_j x_j \mathbb{E}R_j,$$

the reward associated with the given portfolio. This is the amount of money we expect — based on our estimates — to earn in the next investment period for each dollar invested.

There is always a tradeoff between the risk associated with an investment and its rewards. We will measure the risk associated with single investment R_j as

$$(7) \quad \mathbb{E} |R_j - \mathbb{E}R_j|.$$

And we will measure the risk associated with a portfolio as

$$(8) \quad \mathbb{E} |R - \mathbb{E}R| = \mathbb{E} \left| \sum_j x_j (R_j - \mathbb{E}R_j) \right|.$$

This is, once again, a random variable whose value we do not actually know. Again, we will estimate it using the historical data available to us. That is, we can estimate the risk associated with the given portfolio as:

$$(9) \quad \frac{1}{T} \sum_{t=1}^T \left| \sum_j x_j (R_j(t) - r_j) \right|$$

where r_j is the estimated expected value of R_j .

THE PROBLEM

Our problem is to find an investment portfolio with the right mix of risk and reward. We introduce a constant μ which measures the amount of risk an investor is willing to take and consider the problem:

$$(10) \quad \begin{aligned} &\text{maximize: } \mu \sum_j x_j \mathbb{E}R_j - \mathbb{E} \left| \sum_j x_j (R_j - \mathbb{E}R_j) \right| \\ &\text{subject to: } \sum_j x_j = 1 \\ & \quad \quad \quad x \geq 0. \end{aligned}$$

Replacing the expected values in (10) with our estimated expected values, we arrive at the problem:

$$(11) \quad \begin{aligned} &\text{maximize: } \mu \sum_j x_j r_j - \frac{1}{T} \sum_{t=1}^T \left| \sum_j x_j (R_j(t) - r_j) \right| \\ &\text{subject to: } \sum_j x_j = 1 \\ & \quad \quad \quad x \geq 0, \end{aligned}$$

where r_j is the estimated expected value

$$(12) \quad r_j = \frac{1}{T} \sum_{t=1}^T R_j(t).$$

THE PROJECT

1. Reformulate the problem (11) as a linear program in standard form. This involves eliminating the absolute values in the objective function by introducing new variables and replacing the inequalities by equalities by introducing slack variables. Write down your formulation and explain how the absolute values were eliminated.

2. Download the matlab script “returns.m” from the course website. When run, this script initializes a matrix T which contains the data in Table 13.1 on page 213 of your book. **Please use this file to load the data instead of trying to enter the values from the table yourself. The values as they appear in returns.m are the official values we will use for this project.**

The matrix T is 24×9 and the entry T_{ij} gives the value of $R_j(t)$ for $t = i$.

3. Write a matlab script “expreturns.m” which estimates the r_j using formula (12). Your script should store the values in a column vector called r and print the value of r . Make sure the estimated expected value of the j^{th} investment is stored in the j^{th} entry of r . Your script should execute the script “returns.m” rather than relying on the user to load it before calling “expreturns.m.”

3. Write a matlab script “constraints.m” which initializes the matrix Aeq and vector beq in your constraint equations

$$(13) \quad Aeq \cdot x = beq$$

from Step One. Once again, your script should print Aeq as well as initializing it. Also, don’t forget the constraint $\sum_j x_j = 1$. Your script should execute “expreturns.m” rather than relying on the user to load it.

5. Write a matlab script “solvelp.m” which uses the command “linprog” to solve the linear program you just wrote down for a particular value of μ . Your script should use the variable mu to store μ . It should assume that mu has been set before it is called and it should report the value of mu , the solution vector, and the value of the maximum of the objective function. **Keep in mind that matlab’s linprog command expects to minimize the objective function!** Your script should execute constraint.m rather than relying on the user to do it for you.

6. Write a matlab script “solveall.m” which uses your script “solvelp.m” to solve the LP for the cases: $\mu = 0$, $\mu = 1$, and $\mu = 10$.

Challenge: I implemented solvelp.m, constraint.m, and expreturns.m in 12 lines of matlab code. See if you can beat or tie that.