MIS 381N - Project 1

Group 3 - Scroggings, Thelakkat, Tanwisuth, Wooten

February 13, 2018

## Inputs

#Liabilities   
l = c(12000,18000,20000,20000,16000,15000,12000,10000)  
  
#Bond prices  
p = c(102,99,101,98,98,104,100,101,102,94)  
  
#Coupons  
coupons = c(5,3.5,5,3.5,4,9,6,8,9,7)  
  
#Maturities  
m = c(1,2,2,3,4,5,5,6,7,8)

## Q1. Formulate portfolio construction problem as a linear program. Clearly list and describe the decision variables, the objective, and all the constraints.

Decision Variables: Amount of each of the 10 bonds i.e. x1,...,x10

Objective: Minimize the total cost of the bonds 102x1 + 99x2 + 101x3 + 98x4 + 98x5 + 104x6 + 100x7 + 101x8 + 102x9 + 94x10

Constraints: cash flow inward from coupon and maturity payments must equal cash flow outward to liabilities for each year (1-8)

#Objective  
c = p  
  
#Constraints  
A = matrix(0,length(l),length(c))  
for (i in seq\_along(coupons)){  
 A[1:m[i]-1,i] = coupons[i]  
 #Matrix with maturity and coupon payments  
 A[m[i],i] = 100 + coupons[i]  
}  
  
#Direction  
dir = rep("=",length(l))  
  
#Constraints Vector   
b = l

## Q2. Solving the test case

#Solve LP  
s = lp("min",c,A,dir,b)  
sol = s$solution  
  
#Optimal Solution  
for (i in seq\_along(sol)){  
 cat("Bond",i,"\t:\t",sol[i],"\n")  
}

## Bond 1 : 62.13613   
## Bond 2 : 0   
## Bond 3 : 125.2429   
## Bond 4 : 151.5051   
## Bond 5 : 156.8078   
## Bond 6 : 123.0801   
## Bond 7 : 0   
## Bond 8 : 124.1573   
## Bond 9 : 104.0899   
## Bond 10 : 93.45794

## Q3. Writing the function

The dedicate\_g3 function will construct a portfolio for any set of liabilities and bonds. The function takes four inputs: P, C, M, and L. P is the vector containing the prices of the bonds. C is the vector containing the coupon payments for the bonds. M is the vector containing the maturity (in years) for the bonds. Finally, L is the vector of non-negative liabilities for each year. The function outputs the optimal number of each bond to purchase to minimize price of bonds

dedicate\_g3 <- function(P,C,M,L){  
 #Objective  
 c = P  
   
 #Constraints  
 #Initialize constraints matrix with zeros  
 A = matrix(0,length(L), length(C))  
   
 for (i in seq\_along(C)){  
 A[1:M[i]-1,i] = C[i]  
   
 #Matrix with maturity and final coupon payments  
 A[M[i],i] = 100 + C[i]  
 }  
   
 #Direction  
 dir = rep("=",length(L))  
   
 #Constraints Vector  
 b = L  
   
 #Solution  
 s = lp("min",c,A,dir,b,compute.sens=1)  
 return(s)  
}

Let's test the function using our test case from problem 2.

#Inputs  
#Liabilities  
l = c(12000,18000,20000,20000,16000,15000,12000,10000)  
  
#Bond prices  
p = c(102,99,101,98,98,104,100,101,102,94)  
  
#Coupons  
coup = c(5,3.5,5,3.5,4,9,6,8,9,7)  
  
#Maturities  
m = c(1,2,2,3,4,5,5,6,7,8)  
  
#Calling the function  
s = dedicate\_g3(p,coup,m,l)  
sol = s$solution  
  
#Optimal Solution  
for (i in seq\_along(sol)){  
 cat("Bond",i,"\t:\t",sol[i],"\n")  
}

## Bond 1 : 62.13613   
## Bond 2 : 0   
## Bond 3 : 125.2429   
## Bond 4 : 151.5051   
## Bond 5 : 156.8078   
## Bond 6 : 123.0801   
## Bond 7 : 0   
## Bond 8 : 124.1573   
## Bond 9 : 104.0899   
## Bond 10 : 93.45794

The results match!

## Q4. Construct a dedicated portfolio

Bond information was collected from the Wall Street Journal (<http://online.wsj.com/mdc/public/page/2_3020-treasury.html#treasuryB>) and stored as a csv file named "TreasuryQuotes\_wsj.csv".

#Reading bond information  
Bonds = read.csv("TreasuryQuotes\_wsj.csv")

Create input vectors

#Limit bonds to only those that mature/pay coupons at the end of June/December  
dates = c("6/30/2017","12/31/2017","6/30/2018","12/31/2018","6/30/2019","12/31/2019","6/30/2020","12/31/2020","6/30/2021","12/31/2021","6/30/2022","12/31/2022")  
  
Bonds = Bonds[Bonds$Maturity %in% dates,]  
  
#Price Vector  
P = Bonds$Asked  
  
#Coupon Vector  
C = Bonds$Coupon/2  
  
#Maturities Vector (in periods/half-years)  
#Assign each date to period number  
periods = seq\_along(dates)  
names(periods) = dates  
  
#Initialize maturities vector with zeros  
M = rep(0,length(Bonds$Maturity))  
  
#Add period number corresponding to each maturity date to maturities vector  
for (i in seq\_along(M)){  
 date = toString(Bonds$Maturity[i])  
 period = periods[date]  
   
 M[i] = period  
}  
  
#Non-negative liabilities vector  
L = 1000000\*c(9,9,10,10,6,6,9,9,10,10,5,3)

Finally, we can solve for the optimal amount of each bond to purchase using the function we defined in Q3.

options("scipen"=100, "digits"=4)  
#Calling the function  
s = dedicate\_g3(P,C,M,L)  
  
optimal\_solution = data.frame(Bonds$Maturity,Bonds$Coupon,Bonds$Asked,s$solution)  
names(optimal\_solution) = c("Maturity","Coupon","Price","Amount to Purchase")  
optimal\_solution

## Maturity Coupon Price Amount to Purchase  
## 1 6/30/2017 0.625 100.05 0  
## 2 6/30/2017 0.750 100.09 0  
## 3 6/30/2017 2.500 100.77 80607  
## 4 12/31/2017 0.750 99.93 0  
## 5 12/31/2017 1.000 100.16 0  
## 6 12/31/2017 2.750 101.72 81615  
## 7 6/30/2018 0.625 99.50 0  
## 8 6/30/2018 1.375 100.55 0  
## 9 6/30/2018 2.375 101.90 92737  
## 10 12/31/2018 1.250 100.16 0  
## 11 12/31/2018 1.375 100.35 93838  
## 12 12/31/2018 1.500 100.62 0  
## 13 6/30/2019 1.000 99.18 0  
## 14 6/30/2019 1.625 100.62 54484  
## 15 12/31/2019 1.125 99.03 0  
## 16 12/31/2019 1.625 100.44 54926  
## 17 6/30/2020 1.625 100.03 85372  
## 18 6/30/2020 1.875 100.90 0  
## 19 12/31/2020 1.750 100.02 0  
## 20 12/31/2020 2.375 102.39 86066  
## 21 6/30/2021 1.125 96.91 0  
## 22 6/30/2021 2.125 101.12 97088  
## 23 12/31/2021 2.000 100.34 0  
## 24 12/31/2021 2.125 100.89 98120  
## 25 6/30/2022 2.125 100.42 49162  
## 26 12/31/2022 2.125 99.95 29685

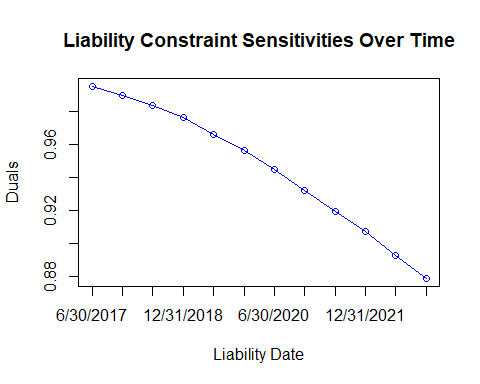
### Sensitivity Analysis

Next we can evaluate the sensitivity of this model to the liability constraints.

optimal\_sol\_sens = data.frame(dates,L,s$duals[0:length(L)],s$duals.from[0:length(L)],s$duals.to[0:length(L)])  
names(optimal\_sol\_sens) = c("Date","Liability","Duals","Duals From","Duals To")  
  
optimal\_sol\_sens

## Date Liability Duals Duals From Duals To  
## 1 6/30/2017 9000000 0.9953 838507 1000000000000000019924668064446  
## 2 12/31/2017 9000000 0.9899 726286 610724618  
## 3 6/30/2018 10000000 0.9837 616161 715007482  
## 4 12/31/2018 10000000 0.9764 551647 1236112160  
## 5 6/30/2019 6000000 0.9663 507379 1051909041  
## 6 12/31/2019 6000000 0.9567 462752 687507465  
## 7 6/30/2020 9000000 0.9450 393387 696044713  
## 8 12/31/2020 9000000 0.9319 291183 484665463  
## 9 6/30/2021 10000000 0.9192 188027 547274633  
## 10 12/31/2021 10000000 0.9072 83775 552983176  
## 11 6/30/2022 5000000 0.8930 31540 553752372  
## 12 12/31/2022 3000000 0.8789 0 475588235

plot(periods, optimal\_sol\_sens$Duals, main="Liability Constraint Sensitivities Over Time", xlab="Liability Date", ylab="Duals", type='o', col='blue', xaxt='n')  
axis(1, at=1:12, labels=dates[1:12])



The duals (i.e. sensitivity of the constraint for each half-year) can be interperted as the time value of money. In other words, 1 dollar on 12/31/2022 is worth 88 cents today. The Duals From & To are the range of liabilities for that date for which the duals/sensitivity applies.