Estimation of Term Structure in CN

201657015

2019.4.14

```
knitr::opts_chunk$set(echo = TRUE)
```

```
library(readxl)    # 程序包：文件读取
library(lubridate) # 程序包：时间数据
library(stringr)   # 程序包：文本数据
library(limSolve)  # 程序包：解方程组
```

在这篇报告中我们使用了在2019年4月15日还未到期的国债数据，市场上总共存在215类这样的债券。我们先从Wind获取了债券的各类指标以及在2019年4月14日之前最新的一次收盘价。总共用到的变量包括“证券代码”、“证券简称”、“债券面值”、“起息日期”、“到期日期”、“债券期限”、“票面利率” “计息方式” “利率类型”、“每年付息次数”、“年付息日1”、“年付息日2” 、“前收盘价”。

```
# 读取数据
bond <- read_xlsx('D://R/TermStructure/CN_bond_19-04-15.xlsx')
bond <- bond[!is.na(bond$前收盘价),]
row.names(bond) <- 1:nrow(bond)
```

```
## Warning: Setting row names on a tibble is deprecated.
```

```
# 展示数据
head(bond)
```

起息日期 <S3: POSIXct>	到期日期 <S3: POSIXct>	债券期限 <dbl>	票面利率 <dbl>	计息方式 <chr>	利率类型 <chr>	每年付息次数 <dbl>	年付息日1 <chr>	年付息日2 <chr>	前收盘价 <dbl>
2018-11-19	2068-11-19	50	3.82	单利	固定利率	2	5月19日	11月19日	103.7622
2018-05-21	2068-05-21	50	4.13	单利	固定利率	2	5月21日	11月21日	101.5001
2017-11-20	2067-11-20	50	4.37	单利	固定利率	2	5月20日	11月20日	109.4186
2017-05-22	2067-05-22	50	4.08	单利	固定利率	2	5月22日	11月22日	107.0644
2016-11-21	2066-11-21	50	3.48	单利	固定利率	2	5月21日	11月21日	95.5840
2016-05-23	2066-05-23	50	3.70	单利	固定利率	2	5月23日	11月23日	100.5326

6 rows | 4-13 of 13 columns

在使用数据之前，需要对数据进行清洗，因为提取时的数据混合了字符和数值类型，需要先将数据类型统一。另外，我们只使用付息日（以及到期日）在4，7，10，1四个月份的债券，这样的债券总共有76只，如此，我们的现金流（利息及本金）只专注在每月的这四个月上，每次现金流的间隔是一个季度。

```
# 第一次付息日期处理
pay_1st <- unlist(bond[,11])
pay_1st_month <- unlist(strsplit(str_extract(pay_1st, '(.|..)月'),'月'))
na_loc <- which(is.na(pay_1st_month))
pay_1st_month[na_loc] <- substr(pay_1st,5,6)[na_loc]
pay_1st_month <- as.numeric(pay_1st_month)

bond_in_1st <- pay_1st_month %in% c(4,7,10,1)
pay_1st_month <- pay_1st_month[bond_in_1st]
# 第二次付息日期处理
pay_2cd <- unlist(bond[,12])
pay_2cd_month <- unlist(strsplit(str_extract(pay_2cd, '(.|..)月'),'月'))
na_loc <- which(is.na(pay_2cd_month))
pay_2cd_month[na_loc] <- substr(pay_2cd,5,6)[na_loc]
pay_2cd_month <- as.numeric(pay_2cd_month)

bond_in_2cd <- pay_2cd_month %in% c(4,7,10,1)
pay_2cd_month <- pay_2cd_month[bond_in_2cd]


# 债券筛选
bond_in <- bond_in_1st + bond_in_2cd
bond_fit <- bond[as.logical(bond_in),]
# 展示数据
head(bond_fit)
```

证券代码 <chr>	证券简称 <chr>	债券面值 <dbl>	起息日期 <S3: POSIXct>	到期日期 <S3: POSIXct>	债券期限 <dbl>	票面利率 <dbl>	计息方式 <chr>	利率类型 <chr>	每年付息次数 <dbl>
180024.IB	18附息国债24	100	2018-10-22	2048-10-22	30	4.08	单利	固定利率	2
180017.IB	18附息国债17	100	2018-07-23	2048-07-23	30	3.97	单利	固定利率	2
170022.IB	17附息国债22	100	2017-10-23	2047-10-23	30	4.28	单利	固定利率	2
170015.IB	17附息国债15	100	2017-07-24	2047-07-24	30	4.05	单利	固定利率	2
160008.IB	16附息国债08	100	2016-04-25	2046-04-25	30	3.52	单利	固定利率	2
150025.IB	15附息国债25	100	2015-10-20	2045-10-20	30	3.74	单利	固定利率	2

6 rows | 1-10 of 13 columns

我们每只债券现金流发生的时点，组成一个矩阵。该矩阵有76行（76只债券），有20列（5年，20个季度）。最初的矩阵只包含0-1元素，1表示该债券在此时点发生现金流（付息），否则没有发生。

```
# 提取现金流时点
# 提取债券指标
# 到期年月
n <- 5
nmax <- 4*n
end_year <- year(bond_fit$到期日期)
end_year[end_year > (2019+n)] <- 2019+n
end_month <- month(bond_fit$到期日期)
end_time <- end_year + 1/12 * end_month
# 债券期限
duration <- bond_fit$债券期限
# 票面利率
coupon <- bond_fit$票面利率

# 自变量
fill_col <- function(lst){
  if(length(lst) < nmax){
    n_fill <- nmax - length(lst)
    lst <- c(lst, rep(0, n_fill))
  }
  return(lst)
}

# Start here
df_fit <- c()
for(i in 1:nrow(bond_fit)){
  if(duration[i] <= 1){
    if(pay_1st_month[i] == 7) df_fit <- c(df_fit, fill_col(c(1,0,0,0)))
    else if(pay_1st_month[i] == 10) df_fit <- c(df_fit, fill_col(c(0,1,0,0)))
    else if(pay_1st_month[i] == 1) df_fit <- c(df_fit, fill_col(c(0,0,1,0)))
    else if(pay_1st_month[i] == 4) df_fit <- c(df_fit, fill_col(c(0,0,0,1)))
    else print('Warning: ...[0]')
  }
  else{
    if(pay_1st_month[i] == pay_2cd_month[i]){
      if(pay_1st_month[i] == 7) df_fit <- c(df_fit, fill_col(rep(c(1,0,0,0), end_year[i]-2019)))
      else if(pay_1st_month[i] == 10) df_fit <- c(df_fit, fill_col(rep(c(0,1,0,0), end_year[i]-2019)))
      else if(pay_1st_month[i] == 1) df_fit <- c(df_fit, fill_col(rep(c(0,0,1,0), end_year[i]-2019)))
      else if(pay_1st_month[i] == 4) df_fit <- c(df_fit, fill_col(rep(c(0,0,0,1), end_year[i]-2019)))
      else print('Warning: ...[1]')
    }
    else{
      if(pay_1st_month[i] == 7 && pay_2cd_month[i] == 1) df_fit <- c(df_fit, fill_col(rep(c(1,0,1,0),
 end_year[i]-2019)))
      else if(pay_1st_month[i] == 10 && pay_2cd_month[i] == 4) df_fit <- c(df_fit, fill_col(rep(c(0,1,
0,1), end_year[i]-2019)))
      else if(pay_1st_month[i] == 1 && pay_2cd_month[i] == 7) df_fit <- c(df_fit, fill_col(rep(c(1,0,1
,0), end_year[i]-2019)))
      else if(pay_1st_month[i] == 4 && pay_2cd_month[i] == 10) df_fit <- c(df_fit, fill_col(rep(c(0,1,
0,1), end_year[i]-2019)))
      else print('Warning: ...[2]')
    }
  }
}


df_fit <- matrix(df_fit, nrow = nmax)
```

```
df_fit <- t(df_fit)
head(df_fit)
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12] [,13]
## [1,]    0    1    0    1    0    1    0    1    0     1     0     1     0
## [2,]    1    0    1    0    1    0    1    0    1     0     1     0     1
## [3,]    0    1    0    1    0    1    0    1    0     1     0     1     0
## [4,]    1    0    1    0    1    0    1    0    1     0     1     0     1
## [5,]    0    1    0    1    0    1    0    1    0     1     0     1     0
## [6,]    0    1    0    1    0    1    0    1    0     1     0     1     0
##      [,14] [,15] [,16] [,17] [,18] [,19] [,20]
## [1,]     1     0     1     0     1     0     1
## [2,]     0     1     0     1     0     1     0
## [3,]     1     0     1     0     1     0     1
## [4,]     0     1     0     1     0     1     0
## [5,]     1     0     1     0     1     0     1
## [6,]     1     0     1     0     1     0     1
```

接着，我们把实际发生的现金流数值加入矩阵，使用的是债券面值乘以票面利率取得。

```
# 现金流矩阵
cashflow <- df_fit
# 利息与到期现金加入现金流
for(i in 1:nrow(cashflow)){
  cashflow[i,] <- cashflow[i,] * coupon[i]
  cashflow[i,max(which((cashflow[i,] == coupon[i])))] <- 100 + coupon[i]
}
head(cashflow)
```

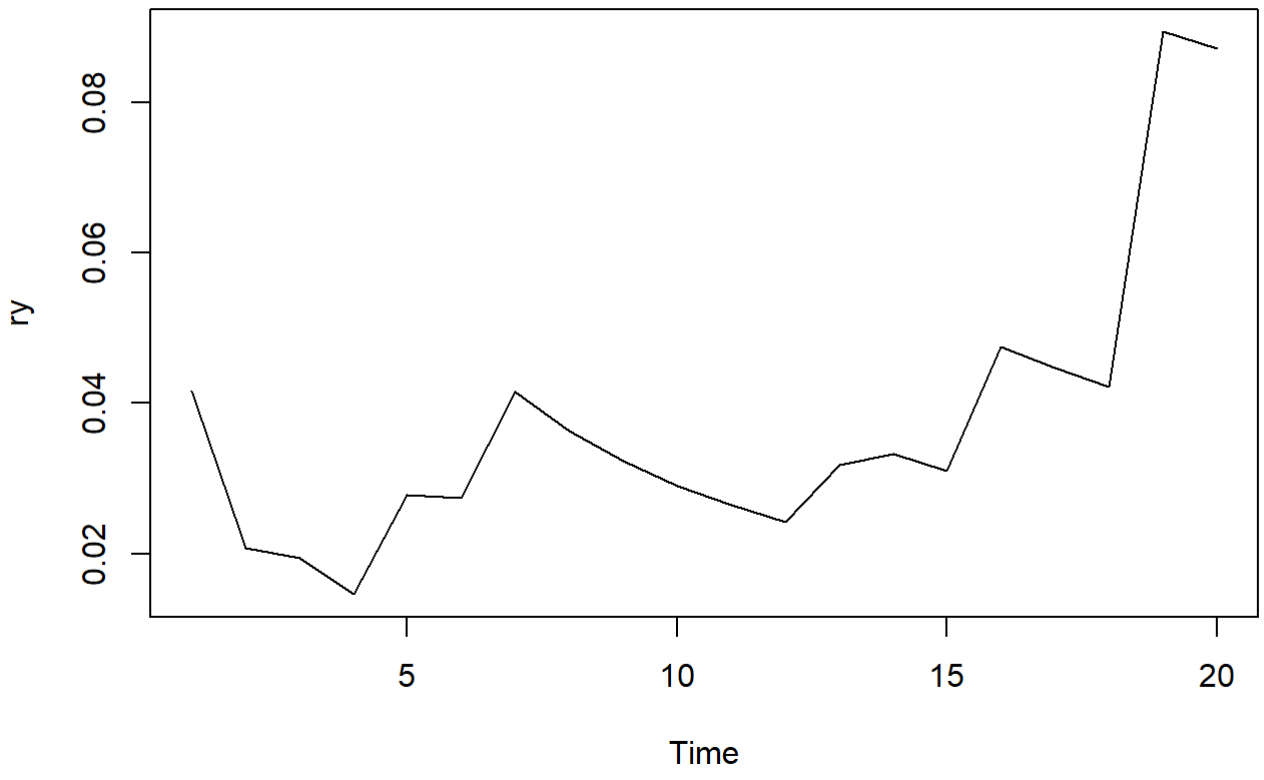
```
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12] [,13]
## [1,] 0.00 4.08 0.00 4.08 0.00 4.08 0.00 4.08 0.00  4.08  0.00  4.08  0.00
## [2,] 3.97 0.00 3.97 0.00 3.97 0.00 3.97 0.00 3.97  0.00  3.97  0.00  3.97
## [3,] 0.00 4.28 0.00 4.28 0.00 4.28 0.00 4.28 0.00  4.28  0.00  4.28  0.00
## [4,] 4.05 0.00 4.05 0.00 4.05 0.00 4.05 0.00 4.05  0.00  4.05  0.00  4.05
## [5,] 0.00 3.52 0.00 3.52 0.00 3.52 0.00 3.52 0.00  3.52  0.00  3.52  0.00
## [6,] 0.00 3.74 0.00 3.74 0.00 3.74 0.00 3.74 0.00  3.74  0.00  3.74  0.00
##      [,14] [,15] [,16] [,17] [,18]  [,19]  [,20]
## [1,]  4.08  0.00  4.08  0.00  4.08   0.00 104.08
## [2,]  0.00  3.97  0.00  3.97  0.00 103.97   0.00
## [3,]  4.28  0.00  4.28  0.00  4.28   0.00 104.28
## [4,]  0.00  4.05  0.00  4.05  0.00 104.05   0.00
## [5,]  3.52  0.00  3.52  0.00  3.52   0.00 103.52
## [6,]  3.74  0.00  3.74  0.00  3.74   0.00 103.74
```

最后，我们使用受约束的回归，令回归的系数x1>x2>..x20，这符合对债券期限结构的假设。随后，把估计的回归系数转换为利率。（x=1/(1+r)）得到利率的期限结构。

```
# 受约束回归
y <- bond_fit$前收盘价
X <- cashflow
A <- X
B <- y
N <- nmax
G_0 <- rbind(cbind(rep(0,N-1),-1*diag(N-1)),rep(0,N))

G <- G_0 + diag(N)
H <- rep(0, N)

reg <- lsei(A = A, B = B, G = G, H = H, type=2)
# 系数
coe <- reg$X
r <- 1/coe-1
# ts.plot(r)
# r
# 年化利率
m <- seq(3,60,3)
ry <- r * 12/m
ts.plot(ry)
```




```
ry
```

```
##  [1] 0.04159932 0.02079966 0.01945108 0.01458831 0.02780732 0.02738632
##  [7] 0.04149802 0.03631077 0.03227624 0.02904862 0.02640783 0.02420718
## [13] 0.03185131 0.03321369 0.03099944 0.04746020 0.04466843 0.04218685
## [19] 0.08937727 0.08718535
```

如上为利率的期限结构估计，使用的是年化利率。可以看出，短期内，三个月的年化利率最高，这可能是因为市场对短期资金的需求比较高。在6个月到1年以内，利率不超过2%。在1年到3年内的时间区间，利率约在3%上下波动，最高可以达到4.1%。利率在第三年末达到4.7%。第四年，利率期限结构的陡峭程度突然剧烈，上半年还在4.5%以内，下半年就达到了接近9%，这比较不符合常理。可能的原因是自由度的限制，我们只有76只债券作为观测，需要估计20个变量，并且越到后期可以使用的有效观测就越少，所以估计结果对样本较为敏感。如果想得到更接近总体的值，那么需要更多的债券以提升准确度。