Fixed Income Assignment 8

Nitish Ramkumar, Justin Ge, Carlos Quicazan, Yuying Wang

```
library(knitr)
setwd("C:/_UCLA/Quarter3/237F_FixedIncome/Assignment/Assignment8")
corr_mat <- read.csv("corrin.csv",header=FALSE)
corchol <- read.csv("corchol.csv",header=FALSE)
dt_raw <- read.csv("pfilea.csv",header=FALSE)
vol_raw <- read.csv("sigma.csv",header=FALSE)
dt_val <- dt_raw[1:20,]
vol <- vol_raw[1:19,]

set.seed(1234)
noOfSims <- 1000


####Carlos Code
corrin = as.matrix(read.csv(file ="corrin.csv",header=F, sep=","))
corchol = as.matrix(read.csv(file ="corchol.csv",header=F, sep=","))
pfilea = as.matrix(read.csv(file ="pfilea.csv",header=F, sep=","))
sigma = as.matrix(read.csv(file ="sigma.csv",header=F, sep=","))
pfilea = pfilea[1:20]
sigma = sigma[1:20]
sigma_i = sigma[-1]
rho = corrin
dt = 1/2
string_model= function(s,r,D){
  n = length(D)
  D_matrix = cbind(c(D[1],D[1:(n-1)]),matrix(0,ncol = n,nrow = n))
  for (i in 1:(n-1)){
    aux = max(n-i,1)
    sigma_i = sigma_i[1:aux];
    r = r[1:aux,1:aux]
    if (i <= (n-1))
    {
      D_matrix[i,i+1] = 1
      rt = (1/D_matrix[i,i]-1)*2
      Z = rnorm(aux)
      L = t(chol(r))
      dZ = (L %*% Z)*sqrt(dt)
      D_matrix[(i+1):n,i+1] = D_matrix[(i+1):n,i] + D_matrix[(i+1):n,i]*rt*dt+sigma_i*dt*dZ
    }
    if (i == (n-1)) { D_matrix[i+1,i+2] = 1 }
  }
  return(D_matrix)
}
s = sigma[-1]
r = rho[1:19,1:19]
r = rho[-1,-1]
result = array( dim = c(length(pfilea), (length(pfilea)+1), noOfSims))
for (i in 1:noOfSims){
  result[,,i] = string_model(s,r,pfilea)
```

```
}

all_sims <- as.list(rep(0,noOfSims))
for(i in c(1:noOfSims)){
  all_sims[[i]] <- result[,,i]
}
####
```

Question 1

1) Get CMS rates using par rate formula
2) Calculate spot rates at various points for all iterations
3) Get payoff using spot rates and CMS rates (as strike)
4) Discount the payoff back to time 0

```
#use dt provided in excel sheet to calculate cms
cms <- sapply(c(1:length(dt_val)), function(m){2 * 100 * (1 - dt_val[m])/(sum(dt_val[1:m]))})/100

calculateDiscountedPayoffs <- function(all_sims,cms){
  #Seperate All dts for moving one step forward
  all_dis_fac <- sapply(all_sims,function(x){diag(x)})

  #L(t-0.5) is the forward rate at t-0.5, 0.5 steps forward (i.e. until t)
  #a.k.a spot at that moment
  forward_rates <- 2*((1/all_dis_fac)-1)

  #Calculate payoff of caplets or swaptions
  payoff <- 0.5*sapply(c(1:length(all_sims)),function(count){pmax(forward_rates[,count]-cms
                                                                   ,0.0)})

  #Calculate present value of caplet
  dis_cf <- apply(all_dis_fac,2,cumprod)
  discount_payoffs <- payoff*dis_cf
  return(discount_payoffs)
}

discounted_payoffs <- calculateDiscountedPayoffs(all_sims,cms)
answer_count <- c(2,3,4,5,7,10)
answers <- sapply(2*answer_count,function(x){mean(apply(discounted_payoffs[2:x,],2,sum))})
kable(cbind(answer_count,answers),col.names = c("cap count","price"))
```

cap count	price
2	0.0395718
3	0.0739968
4	0.1133892
5	0.1504881
7	0.2250123
10	0.3351856

Question 2

1) For every path, check if at exercise step, rate is greater than strike.
2) If it is, generate paths from this point (using the string pattern logic, but different initial dt values)

3) Calculate payoff at each possible step in these newly generated path. This is the fixed leg - floating leg price. Floating leg price = 1. fixed leg is current value of all coupon payments until expiry
4) Discount each payoff using a discount factor
5) Add up cashflows according to what swaption is requested.

```r
calculateSwaptionPrice <- function(all_sims,coupon,exercisestep,expirystep){
  all_dis_fac <- sapply(all_sims,function(x){diag(x)})
  forward_rates <- 2*((1/all_dis_fac)-1)

  exercise_val <- forward_rates[exercisestep,]
  itm <- exercise_val > coupon
  payoffs <- rep(0,length(exercise_val))
  for(count in which(itm)){
      dts <-  all_sims[[count]][-c(1:(exercisestep-1)),exercisestep]
      noOfSim2 <- 10
      all_sims2 <- as.list(rep(0,noOfSim2))
      for(incount in c(1:noOfSim2)){
        all_sims2[[incount]] <- string_model(s,r,dts)
      }

      #Get present value of fixed rate
      all_dis_fac_2 <- sapply(all_sims2,function(x){diag(x)})
      dis_cf_2 <- apply(all_dis_fac_2,2,cumprod)
      fixedcf_pv <- coupon * dis_cf_2[-1,]
      total_coupon <- apply(fixedcf_pv[1:expirystep,],2,sum)
      coupon_principal <- total_coupon + dis_cf_2[expirystep,]
      discount_payoffs <- coupon_principal - 1

      payoffs[count] <- mean(discount_payoffs)
  }
  mean(payoffs)
}

n <- 2
expiry <- 2*c(1:4)
forwards_1 <- sapply(expiry, function(m){2 * 100 *
    (dt_val[n] - dt_val[n+m])/(sum(dt_val[(n+1):(n+m)]))})/100
swaption_price1 <- sapply(c(1:length(forwards_1)),function(x)
                    {calculateSwaptionPrice(all_sims,forwards_1[x],n,expiry[x])})
kable(cbind(c(1:4),swaption_price1),col.names = c("SwpCnt","Price"))
```

SwpCnt	Price
1	0.0148109
2	0.0246161
3	0.0366040
4	0.0472055

```
n <- 4
expiry <- 2*c(1:3)
forwards_2 <- sapply(expiry, function(m){2 * 100 *
    (dt_val[n] - dt_val[n+m])/(sum(dt_val[(n+1):(n+m)]))})/100
swaption_price2 <- sapply(c(1:length(forwards_2)),function(x)
                  {calculateSwaptionPrice(all_sims,forwards_2[x],n,expiry[x])})
kable(cbind(c(1:3),swaption_price2),col.names = c("SwpCnt","Price"))
```

SwpCnt	Price
1	0.0254584
2	0.0476065
3	0.0706884

```
n <- 10
expiry <- 2*c(1,2,5)
forwards_3 <- sapply(expiry, function(m){2 * 100 *
    (dt_val[n] - dt_val[n+m])/(sum(dt_val[(n+1):(n+m)]))})/100
swaption_price3 <- sapply(c(1:length(forwards_3)),function(x)
                  {calculateSwaptionPrice(all_sims,forwards_3[x],n,expiry[x])})
kable(cbind(c(1:3),swaption_price3),col.names = c("SwpCnt","Price"))
```

SwpCnt	Price
1	0.0210879
2	0.0469440
3	0.1219572

Question 3

1) For every swaption, for every spot maturity (t = 0.5,1,. . . 20), bump the spot rate by 0.0001 on both sides and use that to find DV01.

2) Calculation of Dv01 is performed by using the price swaption function from the previous question.

```
calculateDV01 <- function(pfilea, coupon, exercisestep, expirystep){
  answers <- c()
  for(count in 1:length(pfilea)){
    pfilea_up <- pfilea
    pfilea_down <- pfilea
    rt_up = (1/(pfilea_up[count]^count)-1)*2 + 0.0001
    rt_down = (1/pfilea_down[count]^count-1)*2 - 0.0001
    pfilea_up[count] <- (1 + rt_up/2)^(-count)
    pfilea_down[count] <- (1 + rt_down/2)^(-count)

    noOfSims <- 10
    all_sims_up <- as.list(rep(0,noOfSims))
    for (i in 1:noOfSims){
      all_sims_up[[i]] <- string_model(s,r,pfilea_up)
    }
```

```r
    price_up <- calculateSwaptionPrice(all_sims_up,coupon,exercisestep,expirystep)

    all_sims_down <- as.list(rep(0,noOfSims))
    for (i in 1:noOfSims){
      all_sims_down[[i]] <- string_model(s,r,pfilea_down)
    }
    price_down <- calculateSwaptionPrice(all_sims_down,coupon,exercisestep,expirystep)

    #answers[count,1] <- count
    answers[count] <- (price_down - price_up)/2
  }
  return(answers)
}

n <- 2
expiry <- 2*c(1:4)
dv01_1 <- sapply(c(1:length(forwards_1)),function(x){calculateDV01(pfilea, forwards_1[x],
                                                                    n, expiry[x])})

kable(cbind(c(1:20),dv01_1),col.names=c("bump","Swptn1","Swptn2","Swptn3","Swptn4"))
```

bump	Swptn1	Swptn2	Swptn3	Swptn4
1	-0.0026494	0.0021921	0.0021205	0.0177390
2	0.0031288	-0.0128035	0.0071000	-0.0052949
3	-0.0038785	-0.0218694	-0.0162859	-0.0230666
4	0.0041228	-0.0119427	0.0119155	0.0201801
5	-0.0024906	0.0139868	-0.1509512	-0.0192269
6	0.0061102	-0.0011539	-0.0233206	0.0019059
7	0.0011122	0.0063933	0.0105080	0.0190031
8	0.0055463	0.0072253	0.0048500	-0.0080472
9	-0.0110673	-0.0094304	0.0278922	-0.0140043
10	0.0010914	-0.0035193	0.0025065	-0.0104877
11	-0.0060006	0.0035446	-0.0146674	-0.0197036
12	-0.0043700	-0.0008499	-0.0092633	0.0302660
13	0.0027295	0.0250987	-0.0132722	-0.0561150
14	-0.0134225	0.0052774	-0.0146287	-0.0246422
15	-0.0099686	0.0004912	-0.0056790	0.0004508
16	-0.0002611	-0.0051275	-0.0261775	-0.0083539
17	0.0119755	-0.0032833	0.0061416	-0.0029684
18	-0.0064392	-0.0026533	0.0073645	0.0168263
19	-0.0021017	0.0253583	0.0205540	0.0123573
20	-0.0070801	-0.0044530	-0.0181276	0.0066473

```r
n <- 4
expiry <- 2*c(1:3)
dv01_2 <- sapply(c(1:length(forwards_2)),function(x){calculateDV01(pfilea, forwards_2[x],
                                                                    n, expiry[x])})

kable(cbind(c(1:20),dv01_2),col.names=c("bump","Swptn5","Swptn6","Swptn7"))
```

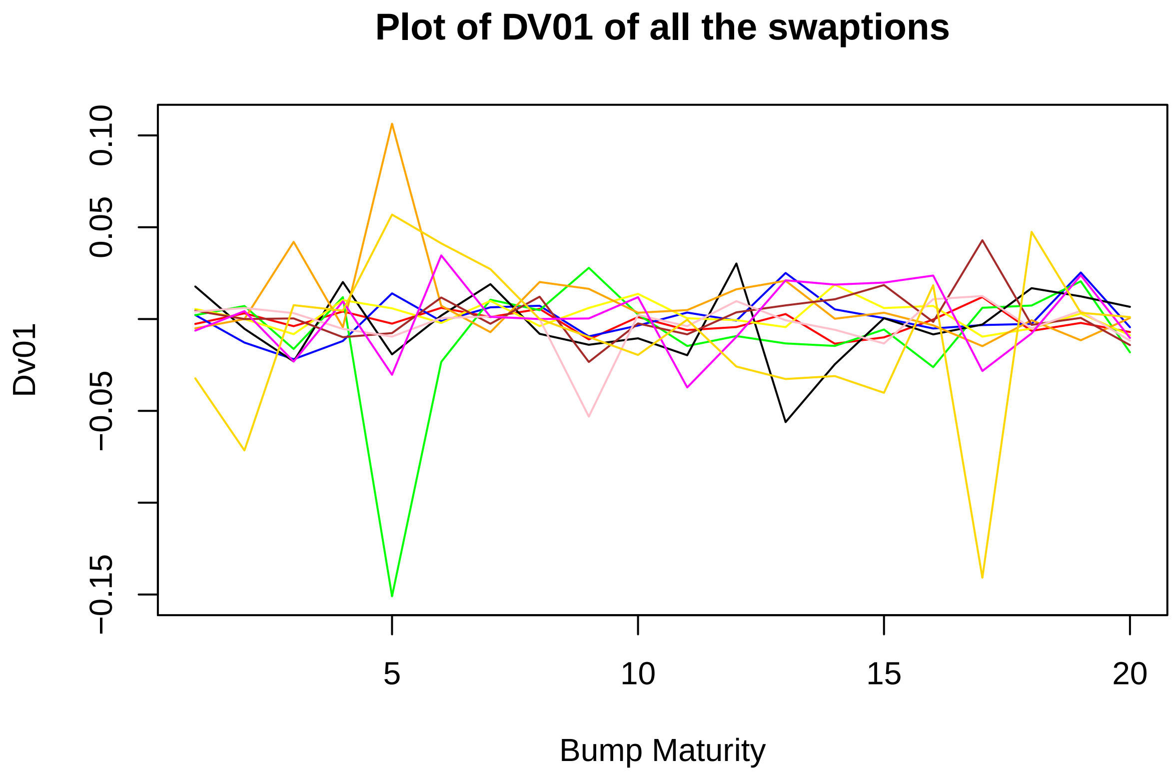
bump	Swptn5	Swptn6	Swptn7
1	0.0054564	-0.0050534	0.0045565
2	0.0000000	0.0000000	0.0000000
3	-0.0082280	0.0420333	0.0003306
4	0.0102885	-0.0044700	-0.0098795
5	0.0058259	0.1063799	-0.0076406
6	-0.0021349	0.0072115	0.0117310
7	0.0098984	-0.0071290	-0.0025610
8	-0.0038248	0.0201803	0.0121736
9	0.0060478	0.0164512	-0.0233802
10	0.0137234	0.0034462	-0.0023883
11	0.0004288	0.0048773	-0.0083187
12	-0.0005710	0.0162189	0.0036626
13	-0.0044003	0.0208362	0.0074465
14	0.0187331	0.0001792	0.0107667
15	0.0059768	0.0034380	0.0185175
16	0.0071590	-0.0034381	-0.0015336
17	-0.0095677	-0.0147326	0.0429190
18	-0.0051339	-0.0006536	-0.0031607
19	0.0030226	-0.0115527	0.0006256
20	-0.0088281	0.0005820	-0.0141687

```r
n <- 10
expiry <- 2*c(1,2,5)
dv01_3 <- sapply(c(1:length(forwards_3)),function(x){calculateDV01(pfilea, forwards_3[x],
                                                      n, expiry[x])})

kable(cbind(c(1:20),dv01_3),col.names=c("bump","Swptn8","Swptn9","Swptn10"))
```

bump	Swptn8	Swptn9	Swptn10
1	0.0040802	-0.0062988	-0.0322694
2	0.0060234	0.0043120	-0.0715480
3	0.0032198	-0.0231340	0.0075669
4	-0.0053619	0.0096267	0.0045734
5	-0.0095416	-0.0303440	0.0568812
6	0.0005364	0.0346598	0.0412111
7	0.0014598	0.0011864	0.0271723
8	0.0000000	0.0000000	0.0000000
9	-0.0531402	0.0003248	-0.0097559
10	0.0022021	0.0119130	-0.0195457
11	-0.0039013	-0.0372228	-0.0003453
12	0.0097603	-0.0097041	-0.0258637
13	-0.0006632	0.0210829	-0.0326403
14	-0.0058338	0.0187684	-0.0310438
15	-0.0133058	0.0198491	-0.0401439
16	0.0108123	0.0236903	0.0184382
17	0.0125344	-0.0282795	-0.1408282
18	-0.0050668	-0.0078116	0.0475005
19	0.0044727	0.0238307	0.0034290
20	-0.0117836	-0.0102565	0.0010709

```
matplot(x=c(1:20),y=cbind(dv01_1,dv01_2,dv01_3),type="l",
        col=c("Red","Blue","Green","Black","Yellow","Orange",
              "Brown","Pink","Magenta","Gold"),lty=1,
        main="Plot of DV01 of all the swaptions",xlab="Bump Maturity",ylab="Dv01")
```



Question 4

1) Use the simulated interest rate paths, to find the payoff, which is current rate on every path - previous rate on that path. Once we do that, we get cashflow at every step in every path.

2) Discount these cashflows

3) Add all the cashflows necessary for the 5 year cap (which is cashflow 2 to 10).

```
calculateResetPayoffs <- function(all_sims){
  #Seperate All dts for moving one step forward
  all_dis_fac <- sapply(all_sims,function(x){diag(x)})

  #L(t-0.5) is the forward rate at t-0.5, 0.5 steps forward (i.e. until t)
  #a.k.a spot at that moment
  forward_rates <- 2*((1/all_dis_fac)-1)

  #Calculate payoff of caplets
```

```
  payoff <- 0.5*sapply(c(1:length(all_sims)),function(count){
    pmax(forward_rates[2:nrow(forward_rates),count]-
           forward_rates[(1:nrow(forward_rates)-1),count],0.0)})

  #Calculate present value of caplet
  dis_cf <- apply(all_dis_fac[2:nrow(all_dis_fac),],2,cumprod)
  discount_payoffs <- payoff*dis_cf
  return(discount_payoffs)
}

reset_discount_payoffs <- calculateResetPayoffs(all_sims)
#only 1-9, ignoring of first already performed
price <- mean(apply(discounted_payoffs[1:9,],2,sum))
price
```

```
## [1] 0.1314874
```

Question 5

1) Calculate the cms rate for every maturity for every simulation.

2) Use this to subtract the strike and to get payoff.
3) Discount the values to t=0 and add up the payoffs

```
calculateCmsPayoffs <- function(all_sims,strike){
  #Seperate All dts for moving one step forward
  all_dis_fac <- sapply(all_sims,function(x){diag(x)})
  dis_cf <- apply(all_dis_fac,2,cumprod)

  #L(t-0.5) is the forward rate at t-0.5, 0.5 steps forward (i.e. until t)
  #a.k.a spot at that moment
  #forward_rates <- 2*((1/all_dis_fac)-1)
  cms_rates <- t(sapply(c(2:nrow(dis_cf)), function(m){2 * 100 * (1 -dis_cf[m,])/
                                                  (apply(dis_cf[c(1:m),],2,sum))})/100)
  #Calculate payoff of caplets
  payoff <- 0.5*sapply(c(1:length(all_sims)),function(count){
    pmax(cms_rates[,count]-strike,0.0)})

  #Calculate present value of caplet
  dis_cf <- apply(all_dis_fac[2:nrow(all_dis_fac),],2,cumprod)
  discount_payoffs <- payoff*dis_cf
  return(discount_payoffs)
}

cms_payoffs <- calculateCmsPayoffs(all_sims,0.05)
#only 1-9, ignoring of first already performed
price <- mean(apply(cms_payoffs[1:9,],2,sum))
price
```

```
## [1] 0.05318682
```