```
R version 3.5.0 (2018-04-23) -- "Joy in Playing"
Copyright (C) 2018 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Previously saved workspace restored]

> #porfolio a11
> install.packages("rootSolve")
Installing package into ‘C:/Users/s1155058334/Documents/R/win-library/3.5’
(as ‘lib’ is unspecified)
--- Please select a CRAN mirror for use in this session ---
trying URL 'https://mirror-hk.koddos.net/CRAN/bin/windows/contrib/3.5/rootSolve_1.7.zip'
Content type 'application/zip' length 787735 bytes (769 KB)
downloaded 769 KB

package ‘rootSolve’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\s1155058334\AppData\Local\Temp\Rtmp6v8b2V\downloaded_packages
> library(rootSolve)
Warning message:
package ‘rootSolve’ was built under R version 3.5.2
> install.packages("gtools")
Installing package into ‘C:/Users/s1155058334/Documents/R/win-library/3.5’
(as ‘lib’ is unspecified)
trying URL 'https://mirror-hk.koddos.net/CRAN/bin/windows/contrib/3.5/gtools_3.8.1.zip'
Content type 'application/zip' length 325812 bytes (318 KB)
downloaded 318 KB

package ‘gtools’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\s1155058334\AppData\Local\Temp\Rtmp6v8b2V\downloaded_packages
> library(gtools)
Warning message:
package ‘gtools’ was built under R version 3.5.2
>
> #####Consider porfolios on derivatives based on 10 underlying correlated assets
> #####investigate the loss probability, which is critical to estimating VAR
> rm(list=ls())
> set.seed(1111)
> r<-rep(0.05,10)
> S0<-c(100,50,30,100,80,20,50,200,150,10)
> K<-S0
> ######sigma<-to be determined
> dt<-0.04
> T<-rep(0.5,10)
>
> ##############################################################################################
##############################################################################
> #####In order to suimulate the loss, we have to be able to samples the change in asset price
s(assumed to follow multivariate normal),
> #####Hence we need to approximate the SIGMA, given the asset price are correlated, SIGMA is
Not diagonal
> #####Indetify sigma and SIGMA with the given covariance matrix of the annual log return of t
he assets
> covm<-matrix(c(
+ c(0.289,0.069,0.008,0.069,0.084,0.085,0.081,0.052,0.075,0.114),
+ c(0.069,0.116,0.020,0.061,0.036,0.088,0.102,0.070,0.005,0.102),
```

```
+ c(0.008,0.020,0.022,0.013,0.009,0.016,0.019,0.016,0.010,0.017),
+ c(0.069,0.061,0.013,0.079,0.035,0.090,0.090,0.051,0.031,0.075),
+ c(0.084,0.036,0.009,0.035,0.067,0.055,0.049,0.029,0.022,0.062),
+ c(0.085,0.088,0.016,0.090,0.055,0.147,0.125,0.073,0.016,0.112),
+ c(0.081,0.102,0.019,0.090,0.049,0.125,0.158,0.087,0.016,0.127),
+ c(0.052,0.070,0.016,0.051,0.029,0.073,0.087,0.077,0.014,0.084),
+ c(0.075,0.005,0.010,0.031,0.022,0.016,0.016,0.014,0.143,0.033),
+ c(0.114,0.102,0.017,0.075,0.062,0.112,0.127,0.084,0.033,0.176)),
+ 10,10,byrow=TRUE)
>
> sigma<-sqrt(diag(covm))
> corrm<-matrix(0,10,10)
> for(i in 1:10){
+ for(j in 1:10){
+ corrm[i,j]<-covm[i,j]/sqrt(covm[i,i])/sqrt(covm[j,j])
+ }
+ }
> SIGMA<-matrix(0,10,10)
> for(i in 1:10){
+ for(j in 1:10){
+ SIGMA[i,j]<-S0[i]*S0[j]*exp(2*0.05*0.04)*(exp(corrm[i,j]*sigma[i]*sigma[j]*0.04)-1)
+ }
+ }
> ############################################################################################
####################################################################################
>
>
> #####define functions for the caluculating the value of a unit of the option(long position),
 under risk neutral framework
> Call<-function(S,T,t,sigma,r,K){
+ d1<-(log(S/K)+(r+0.5*sigma^2)*(T-t))/sigma/sqrt(T-t)
+ d2<-d1-sigma*sqrt(T-t)
+ c<-S*pnorm(d1)-K*exp(-r*(T-t))*pnorm(d2)
+ return(c)
+ }
> Put<-function(S,T,t,sigma,r,K){
+ d1<-(log(S/K)+(r+0.5*sigma^2)*(T-t))/sigma/sqrt(T-t)
+ d2<-d1-sigma*sqrt(T-t)
+ p<-(-S)*pnorm(-d1)+K*exp(-r*(T-t))*pnorm(-d2)
+ return(p)
+ }
> #####All the partial derivatives, thus the greeks, are evaluated at t=0
> #####Calculating the European call options' and European put options' greeks and thus, the p
ortfolio's greeks
> d1<-(log(S0/K)+(r+0.5*sigma^2)*T)/sigma/sqrt(T)
> d2<-d1-sigma*sqrt(T)
>   #####caluculating theta of the European Call option on the 10 assets
>   thetac<-(-S0)*dnorm(d1)*sigma/2/sqrt(T)-r*K*exp(-r*T)*pnorm(d2)
>   #####caluculating theta of the European Put options on the 10 assets
>   thetap<-(-S0)*dnorm(d1)*sigma/2/sqrt(T)+r*K*exp(-r*T)*pnorm(-d2)
>   #####caluculating delta of the European Call option on the 10 assets
>   deltac<-pnorm(d1)
>   #####caluculating delta of the European Put option on the 10 assets
>   deltap<-pnorm(d1)-1
>   #####caluculating delta of the European Call option on the 10 assets
>   gammac<-dnorm(d1)/S0/sigma/sqrt(T)
>   #####caluculating delta of the European Put option on the 10 assets
>   gammap<-dnorm(d1)/S0/sigma/sqrt(T)
>
>
> #####characterize the portfolio (e.g. short 50 ATM calls and short 50 ATM puts on each asset
s)
> weightc<-rep(-50,10)
> weightp<-rep(-50,10)
> #####calculating initial value of the portfolio (value at time 0)
> V0<-sum(weightc*Call(S0,T,0,sigma,r,K)+weightp*Put(S0,T,0,sigma,r,K))
> V0
[1] -7488.298
>
>
> #####consider Delta-GAMMA approximation of the portfolio loss
> #####caluculating theta of the portfolio consisting of mix of the options on the 10 assets
> THETA<-sum(weightc*thetac+weightp*thetap)
> #####caluculating delta of the portfolio(by assets) consisting of mix of the options on the
```

```
10 assets
> delta<-matrix(weightc*deltac+weightp*deltap,10,1)
> #####caluculating gamma of the portfolio(by pairs of assets) consisting of mix of the option
s on the 10 assets
> GAMMA<-diag(weightc*gammac+weightp*gammap,10)
> #####Caluculating parameters for the Delta-Gamma approximatino on the portfolio loss
> a0=-THETA*dt
> a=-delta
> A=-1/2*GAMMA
> #-----------------------------------------------------------------------------------------------
---------------------------------------------------------------------------#
>
>
> #####
> #####step1: Express Q in diagonal form
> Ct<-t(chol(SIGMA))
> ED<-eigen(t(Ct)%*%A%*%Ct)
> U<-ED$vectors
> LAMBDA<-diag(ED$values,10)
> C<-Ct%*%U
> b<-t(C)%*%a
> #define a function to calculate Q
> Q<-function(Z){t(b)%*%Z+t(Z)%*%LAMBDA%*%Z}
>
>
> #####
> #####step2: Identify the IS distribution Z~N(thetax*B(thetax)%*%b,B(thetax)), B(thetax)=solv
e(I-2thetax*LAMBDA)
> ###Given x, find thetax that makes E[Q]=(x-a0) under the IS chagne of measure (assume D-G ap
proximation is exact)
> ###The x is adjusted so that the loss probability is close to 1.1%, xstd=3.2 under the origi
nal distribution of Z
> vecb<-as.vector(b)
> veclambda<-diag(LAMBDA)
> xstd<-3.2
> x<-(a0+sum(veclambda))+xstd*sqrt(sum(vecb^2)+2*sum(veclambda^2))
> ###To identify thetax, we numerically solve psipithetax=(x-a0), notice that E[Q]=psipitheta
for general theta
> psipithetax<-function(thetax){
+ (thetax*vecb[1]^2*(1-thetax*veclambda[1])/(1-2*thetax*veclambda[1])^2 + veclambda[1]/(1-2*th
etax*veclambda[1])
+ +thetax*vecb[2]^2*(1-thetax*veclambda[2])/(1-2*thetax*veclambda[2])^2 + veclambda[2]/(1-2*th
etax*veclambda[2])
+ +thetax*vecb[3]^2*(1-thetax*veclambda[3])/(1-2*thetax*veclambda[3])^2 + veclambda[3]/(1-2*th
etax*veclambda[3])
+ +thetax*vecb[4]^2*(1-thetax*veclambda[4])/(1-2*thetax*veclambda[4])^2 + veclambda[4]/(1-2*th
etax*veclambda[4])
+ +thetax*vecb[5]^2*(1-thetax*veclambda[5])/(1-2*thetax*veclambda[5])^2 + veclambda[5]/(1-2*th
etax*veclambda[5])
+ +thetax*vecb[6]^2*(1-thetax*veclambda[6])/(1-2*thetax*veclambda[6])^2 + veclambda[6]/(1-2*th
etax*veclambda[6])
+ +thetax*vecb[7]^2*(1-thetax*veclambda[7])/(1-2*thetax*veclambda[7])^2 + veclambda[7]/(1-2*th
etax*veclambda[7])
+ +thetax*vecb[8]^2*(1-thetax*veclambda[8])/(1-2*thetax*veclambda[8])^2 + veclambda[8]/(1-2*th
etax*veclambda[8])
+ +thetax*vecb[9]^2*(1-thetax*veclambda[9])/(1-2*thetax*veclambda[9])^2 + veclambda[9]/(1-2*th
etax*veclambda[9])
+ +thetax*vecb[10]^2*(1-thetax*veclambda[10])/(1-2*thetax*veclambda[10])^2 + veclambda[10]/(1-
2*thetax*veclambda[10]))-(x-a0)
+ }
> curve(psipithetax)
> abline(h=0,v=0)
> uni<-uniroot.all(psipithetax,c(0,0.05))
> uni
 [1] 0.002000000 0.004358305 0.006874815 0.008000000 0.013293517 0.013653790
 [7] 0.033000000 0.033500000 0.043500000 0.044561547
>
> ###choose the thetax that makes a valid change of measure
> k<-0
> for(i in 1:length(uni)){
+ if(sum(sign(1-2*uni[i]*veclambda))==length(veclambda)){
+ k<-i
+ break}
+ }
```

```
> (thetax<-uni[k])
[1] 0.002
> psipithetax(thetax)
[1] -25.962
> ax<-thetax-0.0001
> bx<-thetax+0.0001
> while(abs(psipithetax(thetax))>0.0000001){
+ thetax<-(ax+bx)/2
+ ifelse(sign(psipithetax(thetax))==sign(bx),bx<-(ax+bx)/2,ax<-(ax+bx)/2)
+ }
> thetax
[1] 0.002011482
> psipithetax(thetax)
[1] -7.838776e-08
>
> ###identify the IS distribution
> Bthetax<-solve(diag(10)-2*thetax*LAMBDA)
> muthetax<-thetax*Bthetax%*%b
>
> ###generate 5000000 samples of Q under IS change of measure,
> Qsamples<-rep(0,5000000)
> for(j in 1:5000000){
+ Z<-muthetax+chol(Bthetax)%*%matrix(rnorm(10),10,1)
+ Qsamples[j]<-Q(Z)
+ }
> ###check whether E[Q] approximately equals (x-a0) under the importance sampling change of me
asure
> mean(Qsamples)
[1] 1650.825
> x-a0
[1] 1651.413
> a0
[1] -293.8096
> x
[1] 1357.603
> thetax
[1] 0.002011482
>
> ###display the parameters
> SIGMA
             [,1]      [,2]      [,3]      [,4]       [,5]       [,6]      [,7]       [,8]
    [,9]      [,10]
 [1,] 116.7367645 13.874448 0.9640019 27.748897 27.0331255  6.8388740 16.291307  41.810201  45
.248199 4.5887309
 [2,]  13.8744485 11.673555 1.2052917 12.263854  5.7872520  3.5403355 10.261802  28.151618   1
.506163 2.0523603
 [3,]   0.9640019  1.205292 0.7955243  1.566660  0.8676191  0.3856625  1.145004   3.856625   1
.807576 0.2048873
 [4,]  27.7488970 12.263854 1.5666598 31.776834 11.2527648  7.2418852 18.104713  41.005338  18
.686132 3.0165466
 [5,]  27.0331255  5.787252 0.8676191 11.252765 17.2438418  3.5379986  7.879142  18.645201  10
.606991 1.9944240
 [6,]   6.8388740  3.540336 0.3856625  7.241885  3.5379986  2.3683831  5.032611  11.743951   1
.928312 0.9016093
 [7,]  16.2913074 10.261802 1.1450042 18.104713  7.8791418  5.0326111 15.913560  35.000344   4
.820781 2.5566688
 [8,]  41.8102008 28.151618 3.8566247 41.005338 18.6452008 11.7439514 35.000344 123.884471  16
.872058 6.7582814
 [9,]  45.2481988  1.506163 1.8075759 18.686132 10.6069910  1.9283124  4.820781  16.872058 129
.586094 1.9892485
[10,]   4.5887309  2.052360 0.2048873  3.016547  1.9944240  0.9016093  2.556669   6.758281   1
.989248 0.7093155
> THETA
[1] 7345.241
> a0
[1] -293.8096
> delta
            [,1]
 [1,] -10.095990
 [2,]  -8.870803
 [3,] -11.440019
 [4,]  -8.907325
 [5,]  -9.021754
 [6,]  -9.008682
```

```
 [7,]  -9.075234
 [8,]  -8.921218
 [9,]  -8.986282
[10,]  -9.196358
> a
           [,1]
 [1,] 10.095990
 [2,]  8.870803
 [3,] 11.440019
 [4,]  8.907325
 [5,]  9.021754
 [6,]  9.008682
 [7,]  9.075234
 [8,]  8.921218
 [9,]  8.986282
[10,]  9.196358
> GAMMA
           [,1]      [,2]      [,3]      [,4]    [,5]      [,6]      [,7]       [,8]       [,9
]     [,10]
 [1,] -1.015696  0.000000   0.00000  0.000000  0.0000  0.000000  0.000000  0.0000000  0.000000
0   0.00000
 [2,]  0.000000 -3.230791   0.00000  0.000000  0.0000  0.000000  0.000000  0.0000000  0.000000
0   0.00000
 [3,]  0.000000  0.000000 -12.15427  0.000000  0.0000  0.000000  0.000000  0.0000000  0.000000
0   0.00000
 [4,]  0.000000  0.000000   0.00000 -1.957053  0.0000  0.000000  0.000000  0.0000000  0.000000
0   0.00000
 [5,]  0.000000  0.000000   0.00000  0.000000 -2.6546  0.000000  0.000000  0.0000000  0.000000
0   0.00000
 [6,]  0.000000  0.000000   0.00000  0.000000  0.0000 -7.169207  0.000000  0.0000000  0.000000
0   0.00000
 [7,]  0.000000  0.000000   0.00000  0.000000  0.0000  0.000000 -2.764975  0.0000000  0.000000
0   0.00000
 [8,]  0.000000  0.000000   0.00000  0.000000  0.0000  0.000000  0.000000 -0.9910735  0.000000
0   0.00000
 [9,]  0.000000  0.000000   0.00000  0.000000  0.0000  0.000000  0.000000  0.0000000 -0.969298
3   0.00000
[10,]  0.000000  0.000000   0.00000  0.000000  0.0000  0.000000  0.000000  0.0000000  0.000000
0 -13.08943
> A
           [,1]     [,2]     [,3]      [,4]   [,5]     [,6]     [,7]      [,8]      [,9]    [,
10]
 [1,] 0.5078481 0.000000 0.000000 0.0000000 0.0000 0.000000 0.000000 0.0000000 0.0000000 0.000
000
 [2,] 0.0000000 1.615395 0.000000 0.0000000 0.0000 0.000000 0.000000 0.0000000 0.0000000 0.000
000
 [3,] 0.0000000 0.000000 6.077133 0.0000000 0.0000 0.000000 0.000000 0.0000000 0.0000000 0.000
000
 [4,] 0.0000000 0.000000 0.000000 0.9785266 0.0000 0.000000 0.000000 0.0000000 0.0000000 0.000
000
 [5,] 0.0000000 0.000000 0.000000 0.0000000 1.3273 0.000000 0.000000 0.0000000 0.0000000 0.000
000
 [6,] 0.0000000 0.000000 0.000000 0.0000000 0.0000 3.584604 0.000000 0.0000000 0.0000000 0.000
000
 [7,] 0.0000000 0.000000 0.000000 0.0000000 0.0000 0.000000 1.382488 0.0000000 0.0000000 0.000
000
 [8,] 0.0000000 0.000000 0.000000 0.0000000 0.0000 0.000000 0.000000 0.4955368 0.0000000 0.000
000
 [9,] 0.0000000 0.000000 0.000000 0.0000000 0.0000 0.000000 0.000000 0.0000000 0.4846492 0.000
000
[10,] 0.0000000 0.000000 0.000000 0.0000000 0.0000 0.000000 0.000000 0.0000000 0.0000000 6.544
713
>
> b
             [,1]
 [1,] 345.1896215
 [2,] -48.3465643
 [3,]  -4.3262964
 [4,]   2.3124371
 [5,]   3.4478696
 [6,]   4.3636705
 [7,]   3.0918849
 [8,]  -3.2100455
 [9,]   0.6804609
```

```
[10,]   0.2279155
> LAMBDA
         [,1]     [,2]     [,3]     [,4]     [,5]    [,6]    [,7]     [,8]    [,9]    [,10]
 [1,] 150.9541  0.00000  0.00000  0.00000  0.00000 0.00000 0.00000 0.000000 0.00000 0.000000
 [2,]   0.0000 65.71888  0.00000  0.00000  0.00000 0.00000 0.00000 0.000000 0.00000 0.000000
 [3,]   0.0000  0.00000 36.74488  0.00000  0.00000 0.00000 0.00000 0.000000 0.00000 0.000000
 [4,]   0.0000  0.00000  0.00000 14.98134  0.00000 0.00000 0.00000 0.000000 0.00000 0.000000
 [5,]   0.0000  0.00000  0.00000  0.00000 11.31876 0.00000 0.00000 0.000000 0.00000 0.000000
 [6,]   0.0000  0.00000  0.00000  0.00000  0.00000 6.58597 0.00000 0.000000 0.00000 0.000000
 [7,]   0.0000  0.00000  0.00000  0.00000  0.00000 0.00000 4.06192 0.000000 0.00000 0.000000
 [8,]   0.0000  0.00000  0.00000  0.00000  0.00000 0.00000 0.00000 3.108247 0.00000 0.000000
 [9,]   0.0000  0.00000  0.00000  0.00000  0.00000 0.00000 0.00000 0.000000 1.59312 0.000000
[10,]   0.0000  0.00000  0.00000  0.00000  0.00000 0.00000 0.00000 0.000000 0.00000 1.216817
> muthetax
               [,1]
 [1,]  1.7680495948
 [2,] -0.1321998926
 [3,] -0.0102118134
 [4,]  0.0049497446
 [5,]  0.0072661947
 [6,]  0.0090163351
 [7,]  0.0063225891
 [8,] -0.0065387124
 [9,]  0.0013775639
[10,]  0.0004607033
> Bthetax
         [,1]     [,2]     [,3]     [,4]     [,5]     [,6]     [,7]     [,8]    [,9]    [,10]
 [1,] 2.546364 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.00000 0.000000
 [2,] 0.000000 1.359406 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.00000 0.000000
 [3,] 0.000000 0.000000 1.173466 0.000000 0.000000 0.000000 0.000000 0.000000 0.00000 0.000000
 [4,] 0.000000 0.000000 0.000000 1.064135 0.000000 0.000000 0.000000 0.000000 0.00000 0.000000
 [5,] 0.000000 0.000000 0.000000 0.000000 1.047707 0.000000 0.000000 0.000000 0.00000 0.000000
 [6,] 0.000000 0.000000 0.000000 0.000000 0.000000 1.027216 0.000000 0.000000 0.00000 0.000000
 [7,] 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 1.016612 0.000000 0.00000 0.000000
 [8,] 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 1.012663 0.00000 0.000000
 [9,] 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 1.00645 0.000000
[10,] 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.00000 1.004919
>
>
> ##
> #-----------------------------------------------------------------------------------------------
----#
> ##
>
> ###define a function that calculate the Loss
> L<-function(dS){
+ Sdt<-S0+as.vector(dS)
+ Vdt<-sum(weightc*Call(Sdt,T,dt,sigma,r,K)+weightp*Put(Sdt,T,dt,sigma,r,K))
+ return(V0-Vdt)
+ }
> ###define a function that calculate the likelihood ratio
> likelihood<-function(Z){
+ p1<-sum((1/2)*((thetax*vecb)^2/(1-2*thetax*veclambda)-log(1-2*thetax*veclambda)))
+ p2<-thetax*Q(Z)
+ return(exp(p1-p2))
+ }
>
> ##
> #-----------------------------------------------------------------------------------------------
----#
> ##
>
>
> #####
> #####step3: Define k strata
> ###plot the empirical CDF
> Qsamples<-sort(Qsamples,decreasing=FALSE)
> ECDFa11<-ecdf(Qsamples)
> #plot.ecdf(Qsamples)
> ###mimics the quantiles of Q using the 5000000 samples of Q generated in the previous step
> stratabyQ<-rep(0,40-1)
> for(i in 1:39){stratabyQ[i]<-quantile(Qsamples,0.025*i)}
> stratabyQ
 [1]  -76.56625  -11.55883   50.28230  111.64152  173.81427  237.34323  301.69094  367.20228
```

```
434.14867  502.21868
[11]  571.73910  642.58454  714.75469  788.27529  863.79594  941.05995 1020.01836 1101.25785 1
185.33138 1271.64779
[21] 1361.30476 1454.34990 1551.15875 1652.27427 1758.08316 1869.61144 1988.21583 2114.49881 2
249.24635 2394.90176
[31] 2554.03300 2728.29257 2922.90657 3143.17842 3398.48632 3705.56495 4093.17303 4624.60020 5
502.39092
> ECDFall(stratabyQ)
 [1] 0.025 0.050 0.075 0.100 0.125 0.150 0.175 0.200 0.225 0.250 0.275 0.300 0.325 0.350 0.375
 0.400 0.425 0.450 0.475 0.500
[21] 0.525 0.550 0.575 0.600 0.625 0.650 0.675 0.700 0.725 0.750 0.775 0.800 0.825 0.850 0.875
 0.900 0.925 0.950 0.975
>
> ###calculate the optimal alocation of samples size for each strata
> options(warn=-1)
> bins<-c(stratabyQ,.Machine$double.xmax)
> vars<-matrix(0,40,10000)
> counts<-rep(0,40)
> while(sum(counts)!=400000){
+ Z<-muthetax+chol(Bthetax)%*%matrix(rnorm(10),10,1)
+ k<-tail(binsearch(function(y) bins[y]-(Q(Z)), range=c(1, length(bins)))$where,1)
+ if(counts[k]<10000){
+ counts[k]<-counts[k]+1
+ vars[k,counts[k]]<-ifelse(L(C%*%Z)>x,1,0)*likelihood(Z)
+ }
+ else{}
+ }
> (dumvar<-apply(vars,1,var))
 [1] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+0
0 0.000000e+00 0.000000e+00
[10] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+0
0 0.000000e+00 0.000000e+00
[19] 0.000000e+00 0.000000e+00 5.008468e-05 6.444592e-05 3.273394e-05 1.884583e-05 4.712198e-0
4 1.060794e-03 3.755794e-05
[28] 2.335285e-05 1.599740e-05 1.047093e-05 6.842962e-06 4.230171e-06 2.516573e-06 1.412242e-0
6 7.194037e-07 3.474490e-07
[37] 1.384737e-07 4.202009e-08 7.923998e-09 3.495510e-10
> #Since we assume equiprobable strata, pj=1/k, where k is the number of strata, which is 40 i
n the case
> (qj<-sqrt(dumvar)/sum(sqrt(dumvar)))
 [1] 0.0000000000 0.0000000000 0.0000000000 0.0000000000 0.0000000000 0.0000000000 0.000000000
0 0.0000000000 0.0000000000
[10] 0.0000000000 0.0000000000 0.0000000000 0.0000000000 0.0000000000 0.0000000000 0.000000000
0 0.0000000000 0.0000000000
[19] 0.0000000000 0.0000000000 0.0660104110 0.0748785922 0.0533653180 0.0404918454 0.202475157
2 0.3037914121 0.0571624496
[28] 0.0450743947 0.0373065165 0.0301823217 0.0243995625 0.0191839888 0.0147966945 0.011084457
9 0.0079112718 0.0054980095
[37] 0.0034709123 0.0019120025 0.0008302945 0.0001743875
>
>
> #####
> #####step4: Perform the simulation
> ###define a function to generate estimates of P{L>xp} using three methods: SMC, IS, ISSQ, IS
SQO
> options(warn=-1)
> run<-function(n,strata){
+
+ results<-rep(0,4)
+ SMC<-0
+ IS<-0
+ ISSQ<-0
+ ISSQO<-0
+
+ bins<-c(stratabyQ,.Machine$double.xmax)
+ binscount<-rep(0,strata)
+ binscountpi<-rep(0,strata)
+
+ nj<-round(n*qj)
+ nj[match(max(nj),nj)]<-nj[match(max(nj),nj)]+(n-sum(nj))
+
+
+ for(i in 1:n){
+ Z1<-matrix(rnorm(10),10,1)
```

```
+ #Standard Monte Carlo
+ dS1<-C%*%Z1
+ L1<-L(dS1)
+ SMC<-SMC+(ifelse(L1>x,1,0)*(1/n))
+
+ Z2<-muthetax+chol(Bthetax)%*%Z1
+ #Monte Carlo (IS)
+ dS2<-C%*%Z2
+ L2<-L(dS2)
+ IS<-IS+(ifelse(L2>x,1,0)*likelihood(Z2)*(1/n))
+ kthbins<-tail(binsearch(function(y) bins[y]-(Q(Z2)), range=c(1, length(bins)))$where,1)
+ #Monte Carlo (IS and Stratification)
+ if(binscount[kthbins]<(n/strata)){
+ binscount[kthbins]<-binscount[kthbins]+1
+ ISSQ<-ISSQ+(ifelse(L2>x,1,0)*likelihood(Z2)*(1/n))
+ }
+ else{
+ }
+ #Monte Carlo (IS and Stratification with optimized smaple size for each strata)
+ if(binscountpi[kthbins]<nj[kthbins]){
+ binscountpi[kthbins]<-binscountpi[kthbins]+1
+ ISSQO<-ISSQO+(ifelse(L2>x,1,0)*likelihood(Z2)*(1/nj[kthbins])*(1/strata))
+ }
+ else{
+ }
+ }
+ results[1]<-SMC
+ results[2]<-IS
+
+
+ while(sum(binscount)<n){
+ Z2<-muthetax+chol(Bthetax)%*%matrix(rnorm(10),10,1)
+ kthbins<-tail(binsearch(function(y) bins[y]-(Q(Z2)), range=c(1, length(bins)))$where,1)
+ #Monte Carlo (IS and Stratification) continue...
+ if(binscount[kthbins]<(n/strata)){
+ binscount[kthbins]<-binscount[kthbins]+1
+ ISSQ<-ISSQ+(ifelse(L(C%*%Z2)>x,1,0)*likelihood(Z2)*(1/n))
+ }
+ else{
+ }
+ #Monte Carlo (IS and Stratification with optimized sample size for each strata) continue...
+ if(binscountpi[kthbins]<nj[kthbins]){
+ binscountpi[kthbins]<-binscountpi[kthbins]+1
+ ISSQO<-ISSQO+(ifelse(L(C%*%Z2)>x,1,0)*likelihood(Z2)*(1/nj[kthbins])*(1/strata))
+ }
+ else{
+ }
+ }
+ results[3]<-ISSQ
+
+ while(sum(binscountpi)<n){
+ Z2<-muthetax+chol(Bthetax)%*%matrix(rnorm(10),10,1)
+ kthbins<-tail(binsearch(function(y) bins[y]-(Q(Z2)), range=c(1, length(bins)))$where,1)
+ #Monte Carlo (IS and Stratification with optimized sample size for each strata) continue...
+ if(binscountpi[kthbins]<nj[kthbins]){
+ binscountpi[kthbins]<-binscountpi[kthbins]+1
+ ISSQO<-ISSQO+(ifelse(L(C%*%Z2)>x,1,0)*likelihood(Z2)*(1/nj[kthbins])*(1/strata))
+ }
+ else{
+ }
+ }
+ results[4]<-ISSQO
+
+ return(results)
+ }
> run(1000,40)
[1] 0.00900000 0.01184504 0.01080188 0.01063524
> run(10000,40)
[1] 0.01040000 0.01090044 0.01050546 0.01061639
>
> ###define a function to generate the replications
> replication<-function(N,n,strata){
+ dum<-c(0,0,0,0)
+ for(i in 1:N){
```

```
+ dum<-rbind(dum,run(n,strata))
+ }
+ return(tail(dum,-1))
+ }
>
>
> #####
> #####Step5:evaluate the performance of the algorithm
> SAMPLES<-replication(10000,10000,40)
> (ISratio<-var(SAMPLES[,1])/var(SAMPLES[,2]))
[1] 18.06257
> (ISSQratio<-var(SAMPLES[,1])/var(SAMPLES[,3]))
[1] 228.1924
> (ISSQOratio<-var(SAMPLES[,1])/var(SAMPLES[,4]))
[1] 1411.825
>
> n<-10000
> strata<-40
> var(SAMPLES[,1])
[1] 1.04743e-06
> (sum(sqrt(dumvar)*(1/strata)))^2/n
[1] 7.183897e-10
> (theoreticalISSQOratio<-var(SAMPLES[,1])/((sum(sqrt(dumvar)*(1/strata)))^2/n))
[1] 1458.025
>
> q()
> save.image("C:\\Users\\s1155058334\\Desktop\\portfolio a11  (5.3) os5\\a11os5workspace")
>
```