金融计量学 HW3

23210980044 加兴华

Problem 1 (Example 1.32)

Show that the Ornstein-Uhlenbeck process is a Gaussian process with zero mean function and covariance function $\frac{\sigma^2}{2c}(e^{c(t+s)} - e^{c(t-s)})$ for $s < t$.

$$
\begin{aligned}
Ornstei-Uhlenbeck\ process: \quad & X_t = e^{ct}X_0 + \sigma e^{ct}\int_0^t e^{-cs}dB_s, \\
let \quad & Z_t = \int_0^t e^{-cs}dB_s, \\
& Z_t = \lim \sum e^{-ct_{i-1}}\Delta_i B \sim \lim N(0, \sum e^{-ct_{i-1}}\Delta_i) \quad (B_s \sim N(0,s)\&\text{独立增量}) \\
\therefore \quad & Z_t \overset{L_2}{\sim} N(0, \int_0^t e^{-cs}ds) = N(0, \frac{1-e^{-2ct}}{2c}) \\
\therefore \quad & X_t \sim N(e^{ct}X_0, \sigma^2 e^{2ct}\frac{1-e^{-2ct}}{2c}) = N(0, \sigma^2\frac{e^{2ct}-1}{2c}) \quad (\text{取}X_0 = 0\text{时}) \\
\text{显然}\{X_t\}\text{的任意}FIDIS\text{服从高斯分布，}\therefore \quad & \{X_t\}\text{为高斯过程且均值为0;} \\
& c(t,s) = Cov(X_t, X_s) = E(X_tX_s) - EX_tEX_s = E(X_tX_s) \\
\text{此外，} \quad & Z_t - Z_s = \lim \sum_{t_0=s}^{t_n=t} e^{-ct_{i-1}}\Delta_i B\text{由}B_t\text{独立增量可推仍然独立增量} \\
\therefore \quad & c(t,s) = \sigma^2 e^{ct+cs}E(Z_tZ_s) = \sigma^2 e^{ct+cs}(EZ_s^2 + EZ_sEZ_{t-s}) = \sigma^2 e^{c(t+s)}\frac{1-e^{-2cs}}{2c} \\
\therefore \quad & c(t,s) = \frac{\sigma^2}{2c}(e^{c(t+s)} - e^{c(t-s)})
\end{aligned}
$$

Problem 2 (Exercise 3.1)

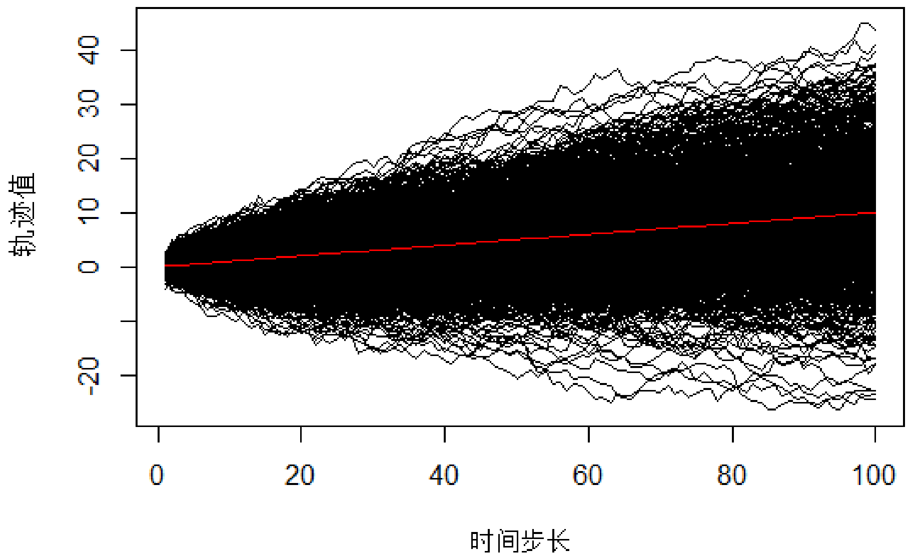
Write an algorithm for Brownian motion with drift $q > 0$ such that

$$\tilde{B}_t = B_t + qt, \quad t \in [0, T],$$

where B is the standard Brownian motion. Moreover, simulate 10000 sample paths with $q = 0.1$ and sketch the paths in one plot.

```r
driftBt<-function( n, drift, var=1) {
  # n: 轨迹点的数量
  # drift: 漂移参数
  # var: Bt的标准差

  # 生成离散正态并累加获得布朗运动轨迹
  Bt <- cumsum(rnorm(n, mean = 0, sd = var))
  # 生成偏移量
  offset <- seq(0, n-1) * drift
  # 添加偏移量到布朗运动轨迹
  drift_Bt <- Bt + offset
  return(drift_Bt)
}
```

```r
n <- 100
q <- 0.1
repeat_time <- 1000
# 生成1000次带偏移的布朗运动轨迹
trajectories <- replicate(repeat_time, driftBt(n,q))
# 绘制轨迹
plot(1:n, trajectories[,1], type = "l",
     xlab = "时间步长", ylab = "轨迹值",ylim = range(trajectories))
for (i in 2:repeat_time) {
  lines(1:n, trajectories[,i])
}
lines(1:n,0.1*1:n,col='red')
```



Problem 3 (Example 3.8)

We want to estimate $Cauchy(0,1)$ tail probability p with the following four equivalent forms

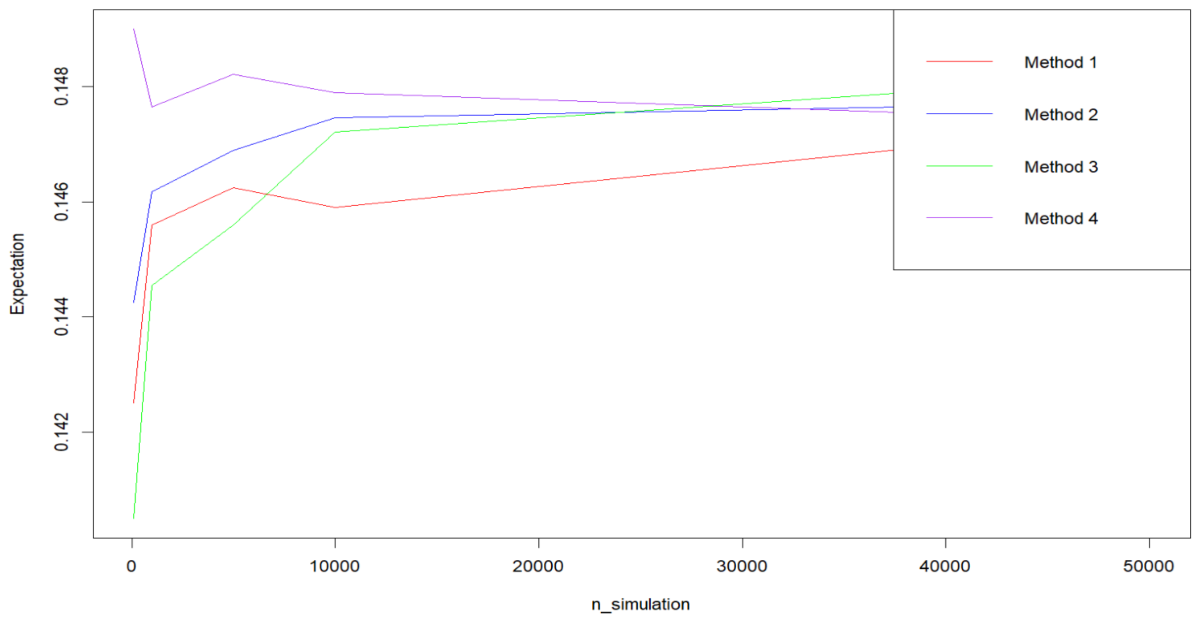
$$p = \int_2^\infty \frac{1}{\pi(1+x^2)}dx = \frac{1}{2}\int_{|x|>2} \frac{1}{\pi(1+x^2)}dx = \frac{1}{2} - \int_0^2 \frac{1}{\pi(1+x^2)}dx = \int_0^{1/2} \frac{x^{-2}}{\pi(1+x^{-2})}dx.$$

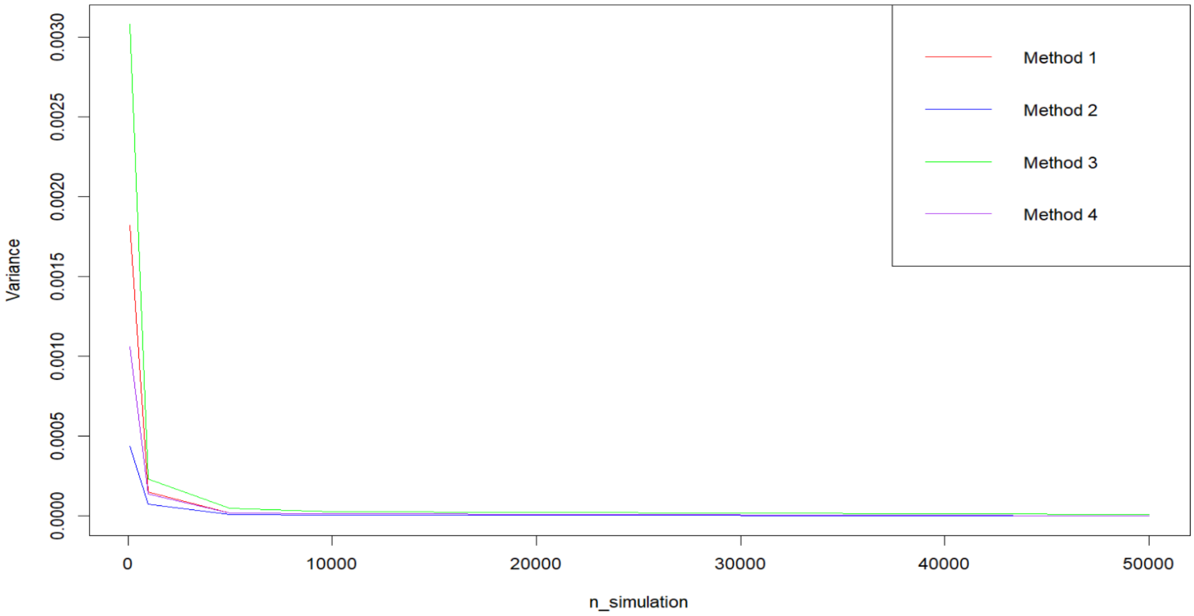
Construct four approximations based on the four representations by Monte Carlo method and monitor their variances. What can you conclude?

```r
simulate_and_ptail<- function(n_simulations,n_repeats) {
  probs<-matrix(nrow = 4, ncol = n_repeats)
  for ( i in 1:n_repeats) {
    # 生成服从柯西分布的随机样本
    samples <- rcauchy(n_simulations, location = 0, scale = 1)
    # 积分1计算尾巴大于2的样本比例
    probs[1,i] <- sum(samples > 2) / n_simulations
    # 积分2计算尾巴大于2的样本比例
    probs[2,i] <- 0.5* sum(abs(samples) > 2) / n_simulations
    # 积分3计算尾巴大于2的样本比例
    probs[3,i] <- 0.5- sum(0<samples & samples<2) / n_simulations
    # 积分4计算尾巴大于2的样本比例
    probs[4,i] <-  sum(0<samples & samples<0.5) / n_simulations
  }
  # 返回四种方法的尾端概率的均值方差，拼成一行
  return(c(rowMeans(probs),apply(probs, MARGIN = 1, FUN = var)))
}
```

```
# 设置模拟次数和重复次数
n_simulations <- c(100, 1000, 5000, 10000, 50000)
n_repeats <- 20

#
probs= matrix(nrow = length(n_simulations) ,ncol = 8 )
for ( i in 1:length(n_simulations)) {
  probs[i,] <- simulate_and_ptail(n_simulations[i],20)
}

# 提取期望和方差数据，每列表示一种方法随模拟次数的变化
expectation <- probs[, 1:4]
variance <- probs[, 5:8]


# 绘制期望随n_simulation的变化
plot(n_simulations,expectation[, 1], type = "l", xlab = "n_simulation", ylab = "Expectation", col =
"red", ylim = range(expectation))
lines(n_simulations,expectation[, 2], col = "blue")
lines(n_simulations,expectation[, 3], col = "green")
lines(n_simulations,expectation[, 4], col = "purple")
legend("topright", legend = c("Method 1", "Method 2", "Method 3", "Method 4"), col = c("red",
"blue", "green", "purple"), lty = 1)

# 绘制方差随n_simulation的变化
plot(n_simulations,variance[, 1], type = "l", xlab = "n_simulation", ylab = "Variance", col =
"red", ylim = range(variance))
lines(n_simulations,variance[, 2], col = "blue")
lines(n_simulations,variance[, 3], col = "green")
lines(n_simulations,variance[, 4], col = "purple")
legend("topright", legend = c("Method 1", "Method 2", "Method 3", "Method 4"), col = c("red",
"blue", "green", "purple"), lty = 1)
```

output:





以上图表为各同样本容量重复实验20次的结果，从方差来看方法二的模拟方差下降最快，从成本角度考虑是最优的。

Problem 4 (Example 3.10)

Suppose $X \sim f$, the density of $t(\nu, \theta, \sigma^2)$ with location θ and scale σ. Consider $\mathbb{E}h_i(X)$ with h_i being

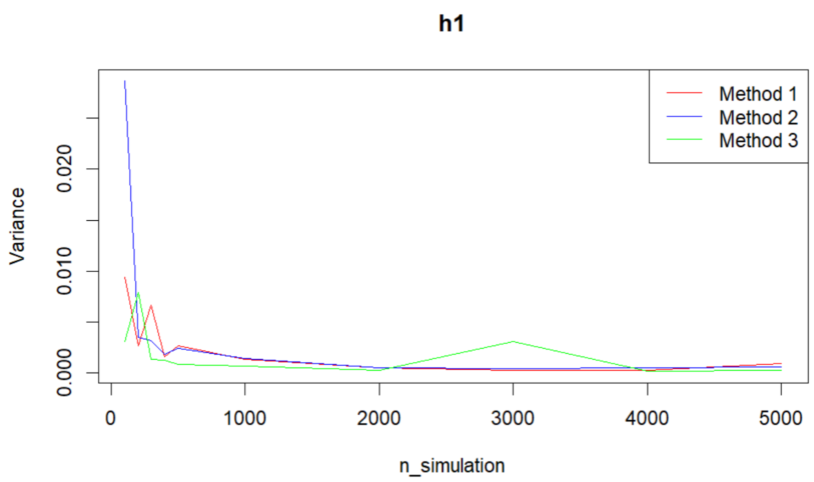
$$h_1(x) = \sqrt{\left|\frac{x}{1-x}\right|}, \quad h_2(x) = x^5 I(x \geq 2.1), \quad h_3(x) = \frac{x^5}{1+(x-3)^2} I(x \geq 0).$$

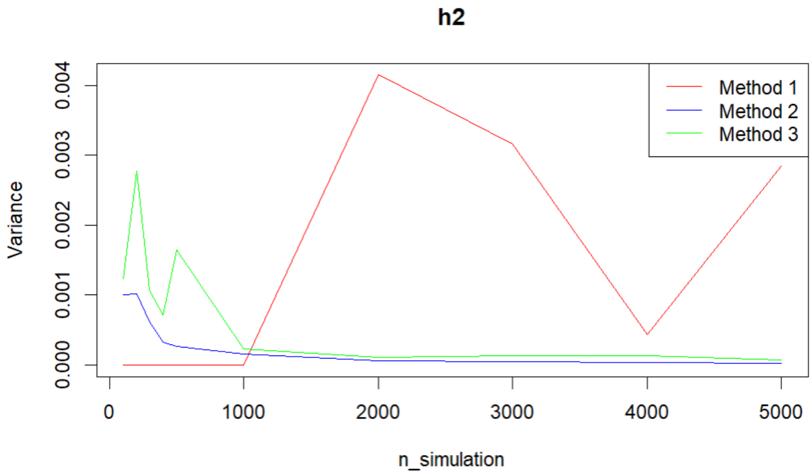
Compare the approximation results of Monte Carlo method, importance sample with instrumental distribution $Cauchy(0,1)$ and importance sample with instrumental distribution $N\left(0, \frac{\nu}{\nu-2}\right)$.

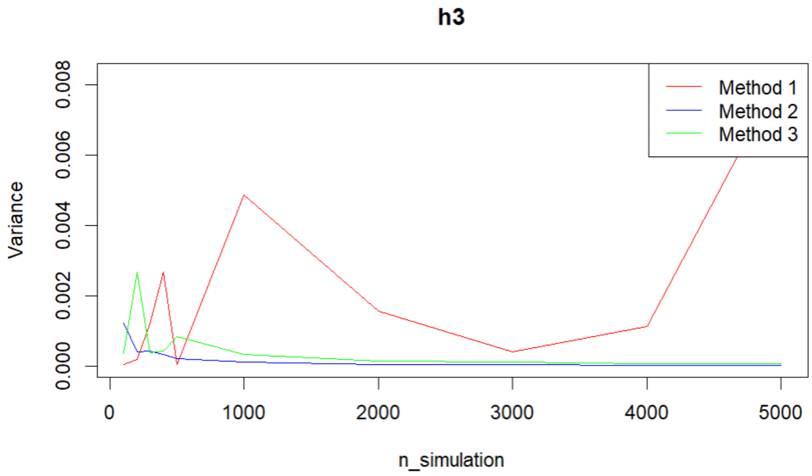
```
## 蒙特卡洛方法模拟
MC_simulate_mean<- function(n_simulations,h,f) {
  samples <- f(n_simulations)
  ehm <- mean(h(samples))
  evar_ehm <- var(h(samples))/n_simulations
  return(c(ehm,evar_ehm))
}
## 重要性采样模拟
IS_simulate_mean <- function(n_simulations,h,f,g) {
  samples <- g(n_simulations)
  ehm <- (h(samples) %*% (f(samples)/g(samples))) / n_simulations
  evar_ehm <- (h(samples)^2 %*% (f(samples)^2/g(samples)^2)) / n_simulations^2 -ehm^2/
n_simulations
  return(c(ehm,evar_ehm))
}
f <- function(item) {
  if (length(item)==1) {
    return(rt(item,df=df)*sigma+theta)
  }
  return(dt((item-theta)/sigma,df=df)/sigma)
}
g1 <- function(item) {
  if (length(item)==1) {
    return(rcauchy(item, location = 0, scale = 1))
  }
  return(dcauchy(item, location = 0, scale = 1))
}
g2 <- function(item) {
  if (length(item)==1) {
```

```r
    return(rnorm(item,mean=0,sd= (df/(df-2))^0.5))
  }
  return(dnorm(item,mean=0,sd= (df/(df-2))^0.5))
}
h1 <- function(x) {return(abs(x/(1-x))^0.5)}
h2 <- function(x) {return(x^5*(x>2.1))}
h3 <- function(x) {return(x^5/(1+(x-3)^2)*(x>=0))}
## 画每个hi的三种模拟方法方差图
myplot<- function(n_simulations,h){
  method1 <- matrix(nrow=length(n_simulations),ncol=2)
  method2 <- matrix(nrow=length(n_simulations),ncol=2)
  method3 <- matrix(nrow=length(n_simulations),ncol=2)
  for (i in 1:length(n_simulations)) {
    method1[i,]<-MC_simulate_mean(n_simulations[i],h,f)
    method2[i,]<-IS_simulate_mean(n_simulations[i],h,f,g1)
    method3[i,]<-IS_simulate_mean(n_simulations[i],h,f,g2)
  }
  plot(n_simulations,method1[,2], type = "l", xlab = "n_simulation", ylab = "Variance", col =
"red", ylim = range(rbind(method1[,2],method2[,2],method3[,2])),main=paste(deparse(substitute(h))))
  lines(n_simulations,method2[,2], col = "blue" )
  lines(n_simulations,method3[,2], col = "green")
  legend("topright", legend = c("Method 1", "Method 2", "Method 3"), col = c("red", "blue",
"green"), lty = 1)
}

df <- 10 # 自由度
theta <- 0 # t位置参数
sigma <- 0.5 # t标准差
myplot(c(100,200,300,400,500,1000,2000,3000,4000,5000),h1)
myplot(c(100,200,300,400,500,1000,2000,3000,4000,5000),h2)
myplot(c(100,200,300,400,500,1000,2000,3000,4000,5000),h3)
```

在df=10，θ=0，σ=0.5设定下，均值估计的方差如下：







可见对于估计$Eh_1(x)$，MC方法最好；对于$Eh_2(x)$和$Eh_3(x)$，使用柯西分布的IS方法最好。但本题在不同的参数组下方差图表现相当不同，因此前面只是一个局部结论。