1. Consider our usual regression model $Y = X\beta + \epsilon$, with Y $n \times 1$, X $n \times p$ and prior $\beta \sim N_p(\beta_0, \sigma^2 V_0)$ but take σ^2 *known*.

   (a) Calculate the Bayes factor for $H_0 : \beta_j = 0$ versus $H_0 : \beta_j \neq 0$ using the Savage-Dickey ratio. Simplify as best possible.

$\underset{n\times 1}{Y} = \underset{n\times p}{X}\,\underset{p\times 1}{\beta} + \underset{n\times 1}{\varepsilon}$, $\beta \sim N_p(\beta_0, \sigma^2 V_0)$, σ^2 known $\rightarrow$ $\beta_j \sim N(\beta_0, \sigma^2 V_{0jj})$

a) Bayes factor for $H_0 : \beta_j = 0$ vs. $H_A : \beta_j \neq 0$

Savage Dickey Ratio

Rearrange Bayes Thm on reduced parameter space (β_j, k)

$$f(\beta_j, k \mid Y) = \frac{f(Y \mid \beta_j, k) \cdot f(\beta_j, k)}{f(Y)}$$

$$\hookrightarrow \frac{f(Y \mid \beta_j, k)}{f(Y)} = \frac{f(\beta_j, k \mid Y)}{f(\beta_j, k)}$$

$\bar{\beta}_j = (X_j'X_j + (V_{0jj})^{-1})^{-1}(X_j'X_j\hat{\beta}_j + V_{0jj}^{-1}\beta_0)$

evaluate both sides at $\beta_j = 0$

posterior of $\beta_j \mid Y$ evaluated @ $\beta_j = 0$: $\beta_j \mid Y \sim N\left(\bar{\beta}_j, \sigma^2 (X_j'X_j + V_{0jj}^{-1})^{-1}\right)$

let $\Sigma = X_j'X_j + V_{0jj}^{-1}$

$$B_{01} = \boxed{\frac{f(Y \mid \beta_j, k)\big|_{\beta_j = 0}}{f(Y)}} = \left.\frac{\boxed{f(\beta_j, k \mid Y)}}{\boxed{f(\beta_j, k)}}\right|_{\beta_j = 0}$$

Bayes factor B_{01}

prior of β_j evaluated @ $\beta_j = 0$

pdf of normal: $\frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$ for $N(\mu, \sigma)$

$$= \frac{\left(2\pi\left(\frac{\sigma^2}{\Sigma}\right)\right)^{-1/2} e^{-\frac{1}{2}\left(\frac{\Sigma}{\sigma^2}\right)^2 (0 - \bar{\beta}_j)^2}}{\left(2\pi(\sigma^2 V_{0jj})\right)^{-1/2} e^{-\frac{1}{2}\left(\frac{1}{\sigma^2 V_{0jj}}\right)^2 (0 - \beta_0)^2}}$$

$$= (\Sigma \cdot V_{0jj})^{1/2} e^{\left(-\frac{1}{2\sigma^4}\Sigma^2 \bar{\beta}_j^2\right) - \left(-\frac{1}{2\sigma^4}\cdot\frac{1}{V_{0jj}^2}\cdot\beta_0^2\right)}$$

$$= (\Sigma \cdot V_{0jj})^{1/2} e^{-\frac{1}{2\sigma^4}\left(\Sigma^2 \bar{\beta}_j^2 - \frac{\beta_0^2}{V_{0jj}^2}\right)}$$

$$= \left[(X_j'X_j + V_{0jj}^{-1}) \cdot V_{0jj}\right]^{1/2} e^{-\frac{1}{2\sigma^4}\left[(X_j'X_j + V_{0jj}^{-1})^2 \cdot \left(\frac{X_j'X_j\hat{\beta}_j + \frac{\beta_0}{V_{0jj}}}{X_j'X_j + V_{0jj}^{-1}}\right)^2 - \frac{\beta_0^2}{V_{0jj}^2}\right]}$$

$$= (X_j'X_j V_{0jj} + 1)^{1/2} e^{-\frac{1}{2\sigma^4}\left[\left(X_j'X_j\hat{\beta}_j + \frac{\beta_0}{V_{0jj}}\right)^2 - \frac{\beta_0^2}{V_{0jj}^2}\right]}$$

$$= (X_j'X_j V_{0jj} + 1)^{1/2} e^{-\frac{1}{2\sigma^4}\left[(X_j'X_j)(X_jX_j')\hat{\beta}_j^2 + 2X_j'X_j\hat{\beta}_j\beta_0 V_{0jj}^{-1} + \frac{\beta_0^2}{V_{0jj}^2} - \frac{\beta_0^2}{V_{0jj}^2}\right]}$$

$$\boxed{B_{01} = (X_j'X_j V_{0jj} + 1)^{1/2} e^{-\frac{1}{2\sigma^4}\left[(X_j'X_j)\left(X_jX_j'\hat{\beta}_j^2 + 2\hat{\beta}_j\beta_0 V_{0jj}^{-1}\right)\right]}}$$

202C Homework 5

Lillian Chen

11/14/2021

Question 2

a) What is the value of c?

Solution: The value of c, which we are trying to estimate, is $1/\sqrt{2\pi}$ which would make p(x) proportional to a density q(x).

```
(c <- 1/sqrt(2*pi))
```

```
## [1] 0.3989423
```

b) Take density $g_1(x)$ to be the density of a $N(0,1)$ random variable. Sample 1000 samples from g_1, and calculate the normalizing constant $\hat{c}$.

Solution: Utilizing Monte Carlo procedures we can estimate the normalizing constant $\hat{c}$ as follows:

$$\hat{c}^{-1} = \frac{1}{1000}\sum_{n=1}^{1000}\frac{p(x^n)}{g_1(x^n)}$$

Carrying out the simulation, we estimate $\hat{c} = 0.3989423$, same as the value of c stated in 2a.

```
set.seed(1234)
chat_g1 <- function(x){
  g1 <- dnorm(x)
  p <- exp(-0.5*x^2)
  chat <- 1/(sum(p/g1)/1000)
  return(chat)
}

# sample 1000 samples from g_1, calculate normalizing constant
chat_g1(rnorm(1000))
```

```
## [1] 0.3989423
```

c) Repeat the previous step 200 times, saving $\hat{c}_j$ for $j = 1, ..., 200$.

Solution: All the $\hat{c}_j$ estimates appear to be the same, with the value of 0.3989423.

```r
set.seed(1234)
sampleg1 <- replicate(n = 200, expr = rnorm(1000))
cjhat_g1 <- vector()

for (j in 1:ncol(sampleg1)){
  cjhat_g1[j] <- chat_g1(sampleg1[,j])
}
```

d) Calculate the bias of the estimator, SD of the estimator and root mean squared error (RMSE). [Note: Answers are very simple.]

Solution: The bias of the estimator, SD of the estimator, and RMSE are all equal to 0 for $g_1(x)$.

```r
#g = g of x function representing density of a RV distributed ~ distr
#cjs = vector with all of the cjhats
# distr = the distribution of a random variable

chatstats <- function(cjs, g, distr){
  c <- 1/sqrt(2*pi)
  bias <- c - mean(cjs)
  sd <- sd(cjs)
  rmse <- sqrt(bias^2 + sd^2)
  dfq2 <- data.frame(Density = g,
                     Distribution = distr,
                     Bias = bias,
                     SD = sd,
                     RMSE = rmse)
  return(dfq2)
}
```

```r
(tbl.g1 <- chatstats(cjs = cjhat_g1, g = "g_1(x)", distr = "N(0,1)"))
```

```
##   Density Distribution Bias SD RMSE
## 1  g_1(x)       N(0,1)    0  0    0
```

e) Repeat these four steps for (a) $g_2(x)$ is the dnesity of a $N(0,4)$ random variable and (b) similarly g_3 for $t(0,1,6)$. The random variable $t(a,b,c)$ is a t with c degrees of freedom, center a and scale parameter b. By generating a standard $z \sim t(0,1,c)$ using the rt command in R, then $\sqrt{(b)} * z + a$ produces a random t with center a, scale parameter b and df c.

Solution:

```r
set.seed(1234)
chat_g2 <- function(x){
  g2 <- dnorm(x, sd = 2)
  p <- exp(-0.5*x^2)
  chat <- 1/(sum(p/g2)/1000)
  return(chat)
}

set.seed(1234)
```

```r
sampleg2 <- replicate(n = 200, expr = rnorm(1000, sd = 2))
cjhat_g2 <- vector()

for (j in 1:ncol(sampleg2)){
  cjhat_g2[j] <- chat_g2(sampleg2[,j])
}

(tbl.g2 <- chatstats(cjs = cjhat_g2, g = "g_2(x)", distr = "N(0,4)"))
```

```
##   Density Distribution         Bias          SD        RMSE
## 1  g_2(x)       N(0,4) 0.0006277268 0.008956905 0.008978875
```

```r
set.seed(1234)
chat_g3 <- function(x){
  g3 <- dt(x, df = 6)
  p <- exp(-0.5*x^2)
  chat <- 1/(sum(p/g3)/1000)
  return(chat)
}

set.seed(1234)
sampleg3 <- replicate(n = 200, expr = rt(1000, df = 6))
cjhat_g3 <- vector()

for (j in 1:ncol(sampleg3)){
  cjhat_g3[j] <- chat_g3(sampleg3[,j])
}

(tbl.g3 <- chatstats(cjs = cjhat_g3, g = "g_3(x)", distr = "t(0,1,6)"))
```

```
##   Density Distribution          Bias          SD        RMSE
## 1  g_3(x)     t(0,1,6) -0.0004377492 0.002426966 0.002466128
```

f) Report your results in a modest table (should be 3 rows, one for each g, and 3 columns for the bias, SD, and RMSE).

```r
(table1 <- rbind(tbl.g1, tbl.g2, tbl.g3))
```

```
##   Density Distribution          Bias          SD        RMSE
## 1  g_1(x)       N(0,1)  0.0000000000 0.000000000 0.000000000
## 2  g_2(x)       N(0,4)  0.0006277268 0.008956905 0.008978875
## 3  g_3(x)     t(0,1,6) -0.0004377492 0.002426966 0.002466128
```

g) Briefly discuss your conclusions.

Solution: Sampling from $g_1(x)$, the density of a $N(0,1)$ RV, yields us an unbiased estimator $\hat{c}$ since the probability density function of $g_1(x)$ is proportional to the normal density. Sampling from $g_2(x)$, the density of a $N(0,4)$ RV, yields us higher values of bias, SD, and RMSE than both sampling from $g_1(x)$ and $g_3(x)$. This makes sense since it is the normal density with a different (larger) variance. Sampling from $g_3(x)$ yields a less biased estimator than sampling from $g_2(x)$, which makes sense since the variance of the t distribution for $g_3(x)$ is equal to 1; however, t distributions have larger tails and tend to not approximate the normal quite as well with lower degrees of freedom (df $= 6$ in $g_3(x)$).

Question 3

a) Estimate the mean of the unknown density that is proportional to $p(x)$. You will have to estimate both the normalizing constant and the mean for each iteration (of 200 iterations).

b) Estimate the variance of the unknown density that is proportional to $p(x)$. You will have the estimate the normalizing constant, mean, and variance.

c) Note: You can (and should!) do all three simulations (normalizing constant, mean, variance) at once, but it was easier to explain the tasks as three separate problems.

Solution:

```r
meanvar_g1 <- function(x){
  g1 <- dnorm(x)
  p <- exp(-0.5*x^2)
  chat <- 1/(sum(p/g1)/1000)
  e_x <- (sum(x*(p/g1))/ 1000)*chat
  e_x2 <- (sum(x^2*(p/g1))/ 1000)*chat
  var <- e_x2 - (e_x)^2

  mean_g1 <- vector()
  var_g1 <- vector()
  for (j in 1:ncol(x)){
    mean_g1[j] <- e_x
    var_g1[j] <- var
  }

  df <- data.frame(mean = mean_g1,
                   var = var_g1)
  return(df)
}

meanvar1 <- meanvar_g1(sampleg1)
```

```r
meanvar_g2 <- function(x){
  g2 <- dnorm(x, sd = 2)
  p <- exp(-0.5*x^2)
  chat <- 1/(sum(p/g2)/1000)
  e_x <- (sum(x*(p/g2))/ 1000)*chat
  e_x2 <- (sum(x^2*(p/g2))/ 1000)*chat
  var <- e_x2 - (e_x)^2

  mean_g2 <- vector()
  var_g2 <- vector()
  for (j in 1:ncol(x)){
    mean_g2[j] <- e_x
    var_g2[j] <- var
  }

  df <- data.frame(mean = mean_g2,
                   var = var_g2)
  return(df)
}
```

```
meanvar2 <- meanvar_g2(sampleg2)
```

```
meanvar_g3 <- function(x){
  g3 <- dt(x, df = 6)
  p <- exp(-0.5*x^2)
  chat <- 1/(sum(p/g3)/1000)
  e_x <- (sum(x*(p/g3))/ 1000)*chat
  e_x2 <- (sum(x^2*(p/g3))/ 1000)*chat
  var <- e_x2 - (e_x)^2

  mean_g3 <- vector()
  var_g3 <- vector()
  for (j in 1:ncol(x)){
    mean_g3[j] <- e_x
    var_g3[j] <- var
  }

  df <- data.frame(mean = mean_g3,
                   var = var_g3)
  return(df)
}

meanvar3 <- meanvar_g3(sampleg3)
```

```
tbl.meanvar <- data.frame(
  Mean.Est = c(mean(meanvar1$mean), mean(meanvar2$mean), mean(meanvar3$mean)),
  Var.Est = c(mean(meanvar1$var), mean(meanvar2$var), mean(meanvar3$var)))
```

```
(finaltable <- cbind(table1, tbl.meanvar))
```

```
##   Density Distribution          Bias          SD        RMSE     Mean.Est
## 1  g_1(x)       N(0,1)  0.0000000000 0.000000000 0.000000000 0.0030688656
## 2  g_2(x)       N(0,4)  0.0006277268 0.008956905 0.008978875 0.0020923270
## 3  g_3(x)     t(0,1,6) -0.0004377492 0.002426966 0.002466128 0.0004344096
##    Var.Est
## 1 0.994614
## 2 0.999850
## 3 1.002331
```

Question 4

a) The actual true density corresponding to p(x) is not the clear best choice for g in this simulation, especially according to the results obtained in this simulation. We see that while the bias, standard deviation, and RMSE is perfectly at 0 for $g_1(x)$, the estimate for the mean and variance is closer to 0 and 1 for $g_2(x)$ and $g_3(x)$.

b) If I could only pick 1 of the 3 g densities, I would pick $g_1(x)$ because it is an unbiased estimator and the estimates for mean and variance are still fairly accurate and close to the original mean and variance.

c) None of the bias estimates are significantly different from zero. To test this, we can run a one-sample t-test to test each of the bias estimates against the null hypothesis $H_0 : bias = 0$ by identifying the critical value corresponding to $1 - \alpha/2$ and $df = 999$. The test statistics for each of the bias estimates are $TS = -0.070$ for $g_2(x)$ and $TS = 0.180$ for $g_3(x)$. Since the absolute values of each of these test statistics do not exceed the critical value of 1.96, we conclude that both bias estimates are not significantly different from zero (p = 0.472 and p = 0.572, respectively).

```r
# critical value
qt(0.975, df = 999)
```

```
## [1] 1.962341
```

```r
# t test for cjhat_g1 yields TS = 0
biasg1 <- cjhat_g1 - c
(tsg1 <- mean(biasg1) / sd(biasg1))
```

```
## [1] NaN
```

```r
pt(tsg1, df = 999)
```

```
## [1] NaN
```

```r
# t test for cjhat_g2
biasg2 <- cjhat_g2 - c
(tsg2 <- mean(biasg2) / sd(biasg2))
```

```
## [1] -0.070083
```

```r
pt(tsg2, df = 999)
```

```
## [1] 0.4720708
```

```r
# t test for cjhat_g3
biasg3 <- cjhat_g3 - c
(tsg3 <- mean(biasg3) / sd(biasg3))
```

```
## [1] 0.1803689
```

```r
pt(tsg3, df = 999)
```

```
## [1] 0.5715502
```

d) I found it surprising that the mean and variance estimates for $g_1(x)$ were not as accurate as the estimates from the other g function. I am also surprised that the estimates were quite accurate for all three g functions and that the bias estimates were not significantly different from zero.