STA410 Assignment 2

Zikun Chen
1001117882

September 6, 2018

Question 1

(a) Define the following transformation:

$$Y = U$$

$$X = \frac{V}{U}$$

Then the inverse transformation is:

$$U = Y$$

$$V = XY$$

The Jacobian of the transformation is:

$$|J| = \begin{vmatrix} \frac{\partial U}{\partial Y} & \frac{\partial U}{\partial X} \\ \frac{\partial V}{\partial Y} & \frac{\partial V}{\partial X} \end{vmatrix} = \begin{vmatrix} 1 & 0 \\ X & Y \end{vmatrix} = |Y| = Y$$

We know that $f_{UV}(u,v) = \frac{1}{|\mathcal{C}_h|}$ for $(u,v) \in \mathcal{C}_h = \{(u,v) : 0 \leq u \leq \sqrt{h(\frac{v}{u})}\}$
Therefore:

$$\begin{aligned} g(u,x) &= g(y,x) \\ &= f_{UV}(y,x)|J| \\ &= \frac{1}{|\mathcal{C}_h|} \cdot y \\ &= \frac{u}{|\mathcal{C}_h|} \text{ for } (u,x) \in \{(u,x) : 0 \leq u \leq \sqrt{h(x)}\} \end{aligned}$$

The marginal density of X is:

$$\begin{aligned} f_X(x) &= \int_0^{\sqrt{h(x)}} \frac{u}{|\mathcal{C}_h|} du \\ &= \frac{1}{|\mathcal{C}_h|} \int_0^{\sqrt{h(x)}} u du \\ &= \frac{1}{|\mathcal{C}_h|} \frac{(\sqrt{h(x)})^2 - 0}{2} \\ &= \frac{1}{2|\mathcal{C}_h|} h(x) \qquad\qquad \text{where } \gamma = \frac{1}{2|\mathcal{C}_h|} > 0 \end{aligned}$$

(b) Let $(u, v) \in \mathcal{C}_h$

Then $0 \leq u \leq \sqrt{h(x)} \leq \max_x \sqrt{h(x)}$

$0 \leq u = \frac{v}{x} \leq \sqrt{h(x)}$

If $x > 0$, then $v \geq 0$, and $v \leq x\sqrt{h(x)} \leq \max_x x\sqrt{h(x)}$

Otherwise, $x < 0$, then $v < 0$, and $v \geq x\sqrt{h(x)} \geq \min_x x\sqrt{h(x)}$

Therefore, $(u, v) \in \mathcal{D}_h$

(c) **Explanation:**

Since by part a) the marginal distribution for $X \sim N(0, 1)$ is

$$f_X(x) = \frac{1}{2|\mathcal{C}_h|} h(x) = \frac{1}{2|\mathcal{C}_h|} e^{\frac{-x^2}{2}} = \frac{1}{\sqrt{2\pi}} e^{\frac{-x^2}{2}}$$

Therefore, $|\mathcal{C}_h| = \sqrt{\frac{\pi}{2}}$ if in the end X comes from N(0,1).

The theoretical probability of U^* and V^* coming from $|\mathcal{C}_h| \subseteq |\mathcal{D}_h|$ is $\frac{|\mathcal{C}_h|}{|\mathcal{D}_h|} = \frac{\sqrt{\frac{\pi}{2}}}{u_+(v_+ - v_-)} \approx 0.73$.

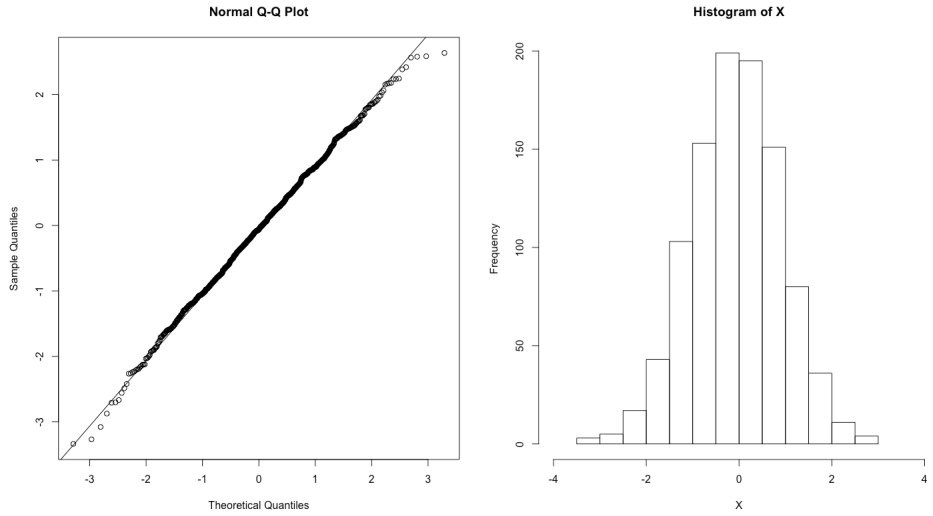
When we sample U^* and V^* in R from $|\mathcal{D}_h|$, if it turns out that they also lie in $|\mathcal{C}_h|$, then we accept them, otherwise we reject. So we do not need to compute M explicitly.

The counted probability of accepted proposals from the R program turns out to be also around 0.73, which is in line with the theoretical value.

Implementation:

```
#Question 1 Rejection Sampling
stdnormal <- function(n) {
  u <- NULL
```

```
  v <- NULL
  rejections <- 0
  uplus <- 1
  vminus <- -sqrt(2/exp(1))
  vplus <- sqrt(2/exp(1))
  for (i in 1:n) {
    reject <- T
    while(reject) {
      ustar <- runif(1, 0, uplus)
      vstar <- runif(1, vminus, vplus)
      #check if ustar and vstar are in Ch
      if (ustar <= exp((-vstar^2)/(4*ustar^2))) {
        u <- c(u, ustar)
        v <- c(v, vstar)
        reject <- F
      }
      else rejections <- rejections + 1
    }
  }
  x <- v/u

  # calculate proposal acceptance rate
  accept.rate <- n/(n+rejections)
  r <- list(x=x,accept.rate=accept.rate)
  r
}

y <- stdnormal(1000)

#Acceptence Rate
y$accept.rate

#Normal Q-Q Plot
qqnorm(y$x)
qqline(y$x)

#Histogram
hist(y$x, main="Histogram of X", xlab = "X", xlim = c(-4,4))
```



Question 2

(a) We want to minimize $f(\theta_1, \cdots, \theta_n) = \sum_{i=1}^{n}(y_i - \theta i)^2 + \lambda \sum_{i=2}^{n}(\theta_i - \theta_{i-1})^2$
Take partial derivatives and set them to zero, we get: First:

$$\frac{\partial f}{\partial \theta_1} = 0$$

$$2(y_1 - \hat{\theta}_1)(-1) + 2\lambda(\hat{\theta}_2 - \hat{\theta}_1)(-1) = 0$$
$$\hat{\theta}_1 - y_1 + \lambda\hat{\theta}_1 - \lambda\hat{\theta}_2 = 0$$
$$y_1 = (1+\lambda)\hat{\theta}_1 - \lambda\hat{\theta}_2$$

For $j = 2, \cdots, n-1$:

$$\frac{\partial f}{\partial \theta_j} = 0$$

$$2(y_j - \hat{\theta}_1)(-1) + \lambda(2(\hat{\theta}_j - \hat{\theta}_{j-1}) + 2(\hat{\theta}_{j+1} - \hat{\theta}_j)(-1)) = 0$$
$$(\hat{\theta}_1 - y_j) + \lambda(\hat{\theta}_j - \hat{\theta}_{j-1} + \hat{\theta}_j - \hat{\theta}_{j+1}) = 0$$
$$y_j = -\lambda\hat{\theta}_{j-1} + (1+2\lambda)\theta j - \lambda\hat{\theta}_{j+1}$$

Finally:

$$\frac{\partial f}{\partial \theta_n} = 0$$

$$2(y_n - \hat{\theta}_n)(-1) + 2\lambda(\hat{\theta}_n - \hat{\theta}_{n-1}) = 0$$
$$\hat{\theta}_n - y_n + \lambda\hat{\theta}_n - \lambda\hat{\theta}_{n-1} = 0$$
$$y_n = -\lambda\hat{\theta}_{n-1} + (1+\lambda)\hat{\theta}_n$$

(b) We write the equations in a) into matrix form $\mathbf{y} = A_\lambda \hat{\boldsymbol{\theta}}$

where $\mathbf{y} = (y_1, \cdots, y_n)^T$, $\hat{\boldsymbol{\theta}} = (\hat{\theta}_1, \cdots, \hat{\theta}_n)^T$ and

$$A_\lambda = \begin{bmatrix} 1+\lambda & -\lambda & 0 & 0 & \cdots & 0 \\ -\lambda & 1+2\lambda & -\lambda & 0 & \cdots & 0 \\ 0 & -\lambda & 1+2\lambda & -\lambda & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & 0 & -\lambda & 1+2\lambda & -\lambda \\ 0 & 0 & \cdots & 0 & -\lambda & 1+\lambda \end{bmatrix}$$

The Jacobi method is guaranteed to converge because A_λ is strictly diagonally dominant:

$$|a_{ii}| > \sum_{j \neq i} |a_{ij}|$$

for each row i.

When $i = 1$, $|a_{11}| = 1 + \lambda > \lambda = \sum_{j \neq 1} |a_{1j}|$

When $i = 2, \cdots, n-1$, $|a_{ii}| = 1 + 2\lambda > 2\lambda = \sum_{j \neq 1} |a_{1j}|$

When $i = n$, $|a_{nn}| = 1 + \lambda > \lambda = \sum_{j \neq n} |a_{nj}|$

A_λ is both symmetric and strictly diagonally dominant, therefore, by the diagonal dominance theorem, A_λ is positive definite.

Therefore, the G-S method is also guaranteed to converge for $\mathbf{y} = A_\lambda \hat{\boldsymbol{\theta}}$

(c) **Implementation:**

```
# Question 2 G-S Method
seidel <- function(y,lambda,theta,max.iter=50,eps=1.e-6) {
  n <- length(y)
  # Define initial estimates if unspecified
  if (missing(theta)) theta <- rep(mean(y),n)
  # Compute objective function for initial estimates
  obj <- sum((y-theta)^2)+lambda*sum(diff(theta)^2)
  # The function diff(theta) computes first differences of the vector theta
  no.conv <- T
  # Do Gauss-Seidel iteration until convergence or until max.iter iterations
  iter <- 0
  theta.old <- theta
  while(no.conv) {
    theta[1] <-  (y[1] + lambda*theta[2])/(1+lambda)
    # Update theta[2],..., theta[n-1]
    for (j in 2:(n-1)) {
      theta[j] <-  (y[j]+lambda*(theta[j+1]+theta[j-1]))/(1+2*lambda)
    }
    theta[n] <- (y[n]+lambda*theta[n-1])/(1+lambda)
```

```
    # Compute new objective function for current estimates
    obj.new <- sum((y-theta)^2)+lambda*sum(diff(theta)^2)
    iter <- iter + 1
    # Now set no.conv to F if either convergence or iter=max.iter
    # and update the value of the objective function variable obj
    if (obj==obj.new) no.conv <- F
    if (iter==max.iter) no.conv <- F
    obj <- obj.new
    theta.old <- theta
  }
  r <- list(y=y,theta=theta,obj=obj,niters=iter)
  r
}

y <- c(rep(0,250),rep(1,250),rep(0,50),rep(1,450)) + rnorm(1000,0,0.1)

#plot y and its estimate theta with different values of lambda
lambda <- 1
plot(c(1:1000), y, col="blue")
points(c(1:1000), seidel(y,lambda)$theta, col="red")

lambda <- 10
plot(c(1:1000), y, col="blue")
points(c(1:1000), seidel(y,lambda)$theta, col="red")

lambda <- 100
plot(c(1:1000), y, col="blue")
points(c(1:1000), seidel(y,lambda)$theta, col="red")
```

As we can see from following graphs (blue for y, red for theta estimates), when λ gets bigger, the change from the previous theta and the next theta gets smaller, especially for the transition thetas between 0 and 1. As λ grows, more and more transition points emerge and the smoothness of the thetas increases. The effect of minimizing the lag difference dominates and the accuracy of the estimates diminishes (values of the objective function in the end increase when λ grows). The algorithm takes more iterations to converge as well.

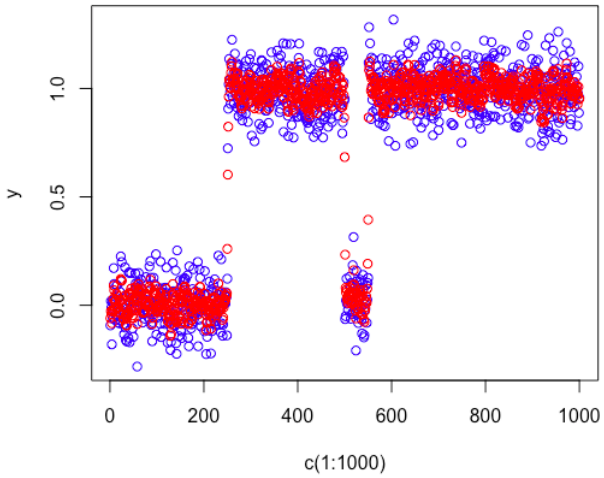


Figure 1: $\boldsymbol{\lambda = 1}$

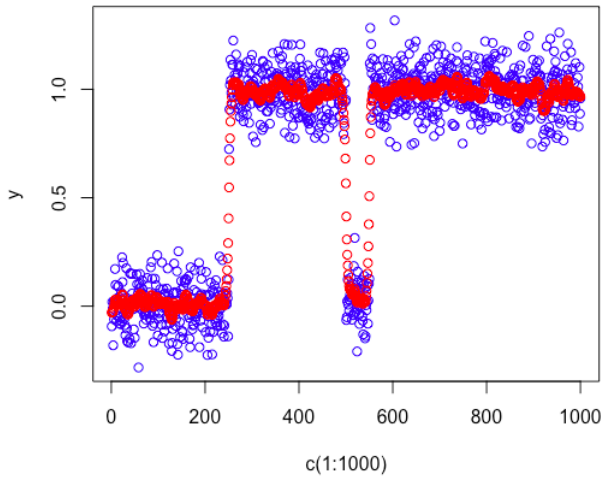


Figure 2: $\boldsymbol{\lambda = 10}$

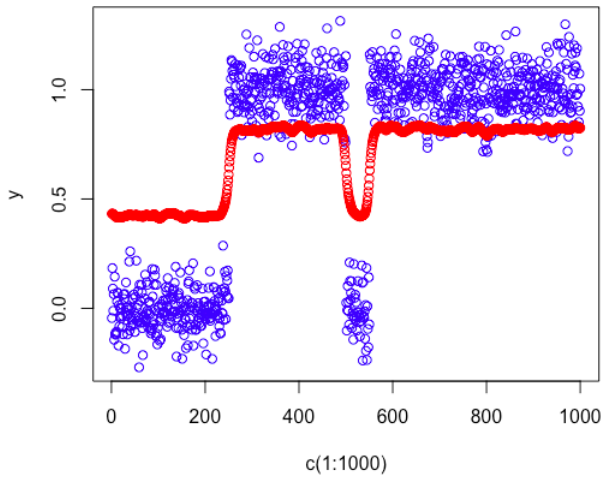


Figure 3: $\boldsymbol{\lambda = 100}$