Statistical Computing and Simulation: Assignment 5

Deparment of Statistics, NCCU
葉佐晨　高崇哲
{112354016,112354020}@nccu.edu.tw

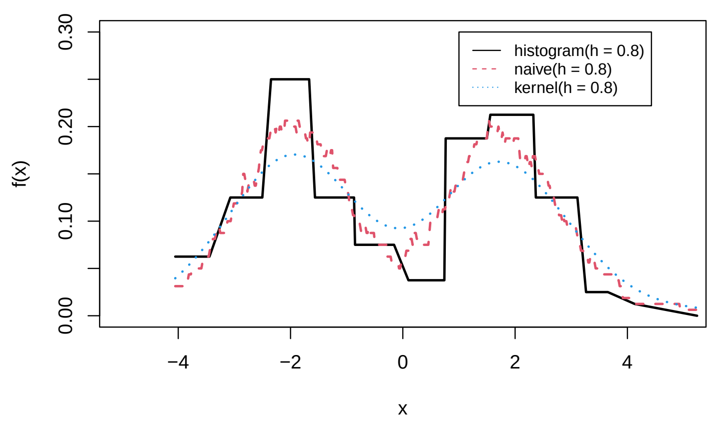
2024-06-10

Statistical Computing and Simulation

Assignment 5, Due June 11/2024

Question 1

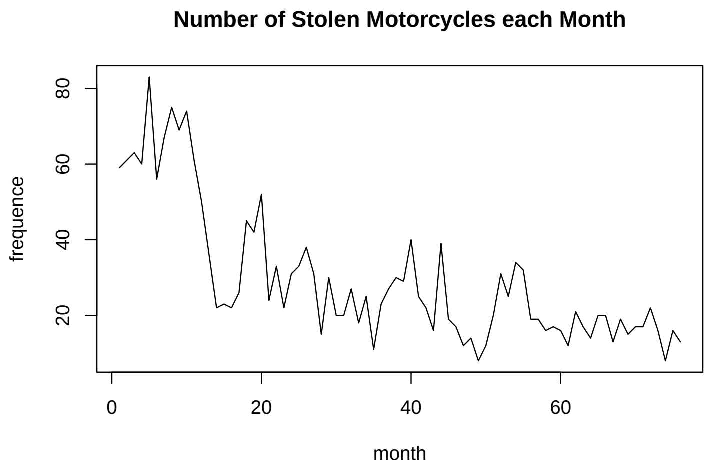
First, simulate 100 observations from a mixed distribution of N(2,1) and N(2,1), each with probability 0.5. Then, use at least 3 density estimating methods to smooth the observations. You need to specify the parameters in the smoothing methods, and compare the results.



我們使用了histogram、 naïve與normal kernel三種估計密度函數的方法， h均設置為0.8， 可以發現histogram的震盪幅度最大， naïve則可看出資料的局部特性，而normal kernel最平滑。

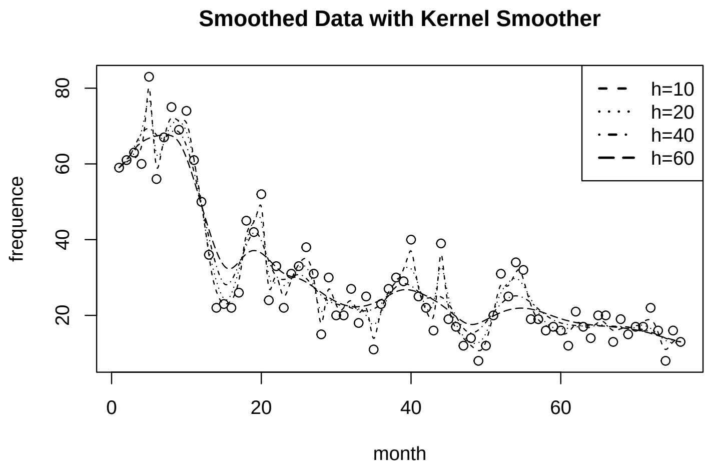
Question 2

Crime data are available in many countries and we can use them the explore whether there are hot spots and/or peak seasons. Explore the reported cases of stolen motorcycles provided by Taipei City and evaluate which month(s) has the highest reported cases of stolen motorcycles (via density estimation methods). (Bonus: Explore if there are hot spots in stolen motorcycles.)



Kernel Smooth

下圖呈現了使用不同的bandwidth的Kernel smoother所估計的密度函數。可以看到在`bandwidth=10`時，估計密度函數保留了多數的特徵，不過由於不夠平滑，容易受到個別樣本的影響，較難看出整體的分布情況，因此需要進一步加大bandwidth。隨著bandwidth的提高，估計函數也更加平滑，可以看出整體的趨勢。

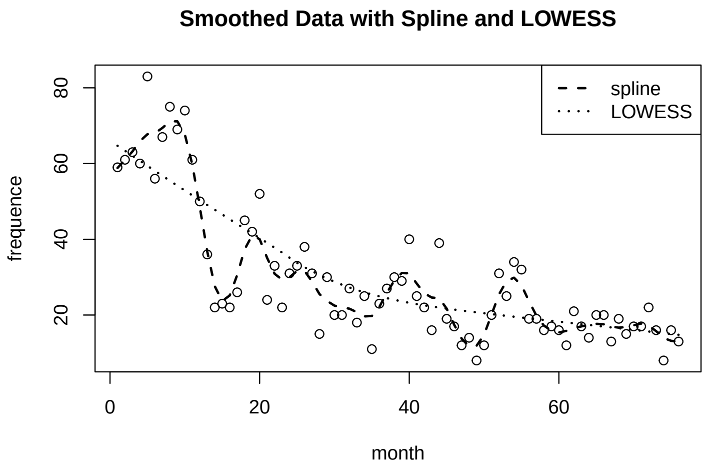


我們使用`bandwidth=60`的Kernel Smoother的估計結果尋找其函數值最高的月份，結果顯示根據smooth的結果，2018年8月為最高通報數量的月份。

```
## [1] "Highest month is 2018-08-01"
```

Spline and LOWESS

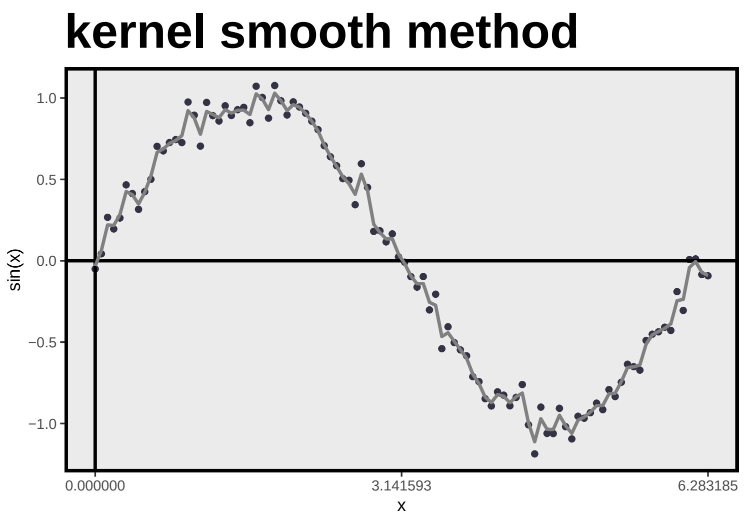
另外我們也分別使用Smoothing Spline以及LOWESS的方法估計密度函數。



Question 3

Let x be 100 equally spaced points on $[0,2\pi]$ and generate random sample $y_i = \sin x_i + \epsilon_i$ with $\epsilon_i \sim$ N(0,0.09). Apply at least 3 linear smoothers and compare the differences, with respect to mean squares error (i.e., bias^2 and variance) from 1,000 simulation runs.

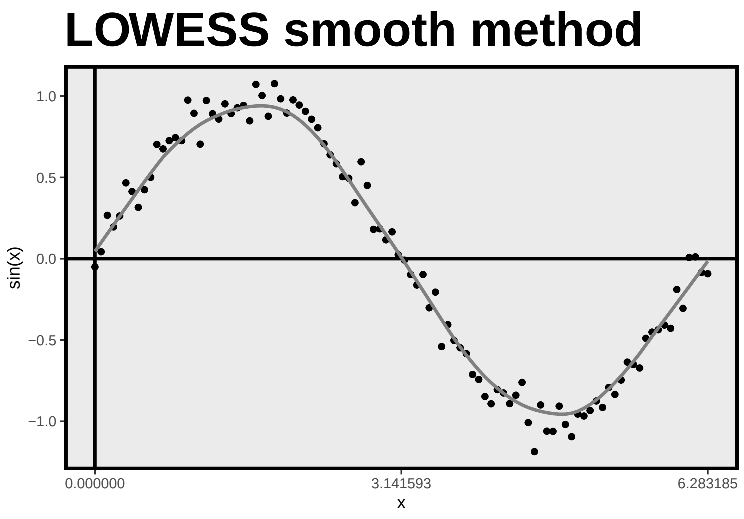
kernel smooth



```
## MSE of kernel smooth: 0.004182453
```

使用kernel smooth重複模擬1000次, MSE為0.004182。

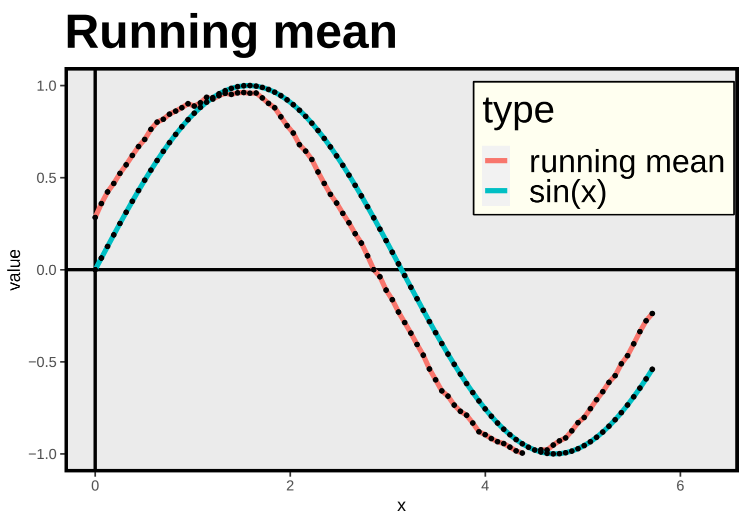
LOWESS



```
## MSE of LOWESS: 0.001636628
```

使用LOWESS重複模擬1000次, MSE為0.001636。

Running means



```
## MSE of Running mean: 0.004040543
```

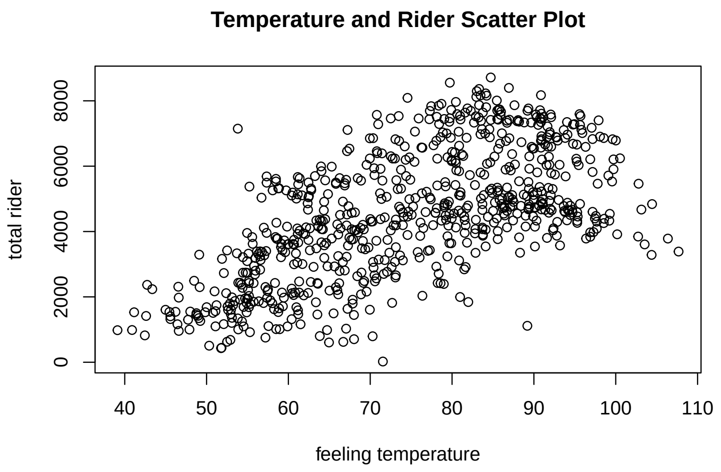
使用Running means重複模擬1000次, MSE為0.004041。

透過EDA與MSE比較三種方法，可以得到LOWESS Smooth所模擬出來的誤差最小，在實際EDA 呈現上也與理論函數最為接近。

Question 4

Use `MCMCregress` in the module `MCMCpack` to obtain MCMC estimation of regression analysis. Duplicate the analysis in the lecture notes and apply the MCMC on the `bikes.csv` data. Compare your results with the regular simple linear regression.

由體感溫度以及總騎乘人數的散佈圖可以發現似乎有一個正向的關係，但其趨勢有一點曲線的樣式，因此我們在模型中加入體感溫度的二次項，更好的擬合兩者間的相關性。



lm

下列為使用`lm`估計模型的結果，可以看出二次項為顯著，顯示確實有正向影響遞減的現象。

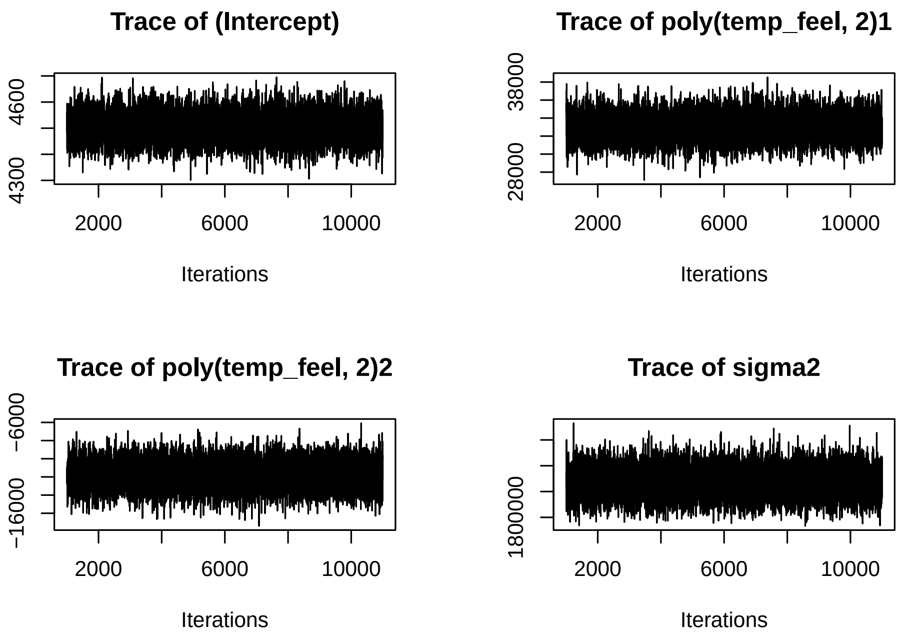
```
##
## Call:
## lm(formula = riders_total ~ poly(temp_feel, 2), data = bikes)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4648.8 -1042.4  -130.1  1148.7  4751.5
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)           4504.35      53.26   84.57  < 2e-16 ***
## poly(temp_feel, 2)1  33030.34    1440.08   22.94  < 2e-16 ***
## poly(temp_feel, 2)2 -11780.56    1440.08   -8.18 1.26e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1440 on 728 degrees of freedom
## Multiple R-squared:  0.4489, Adjusted R-squared:  0.4474
## F-statistic: 296.5 on 2 and 728 DF,  p-value: < 2.2e-16
```

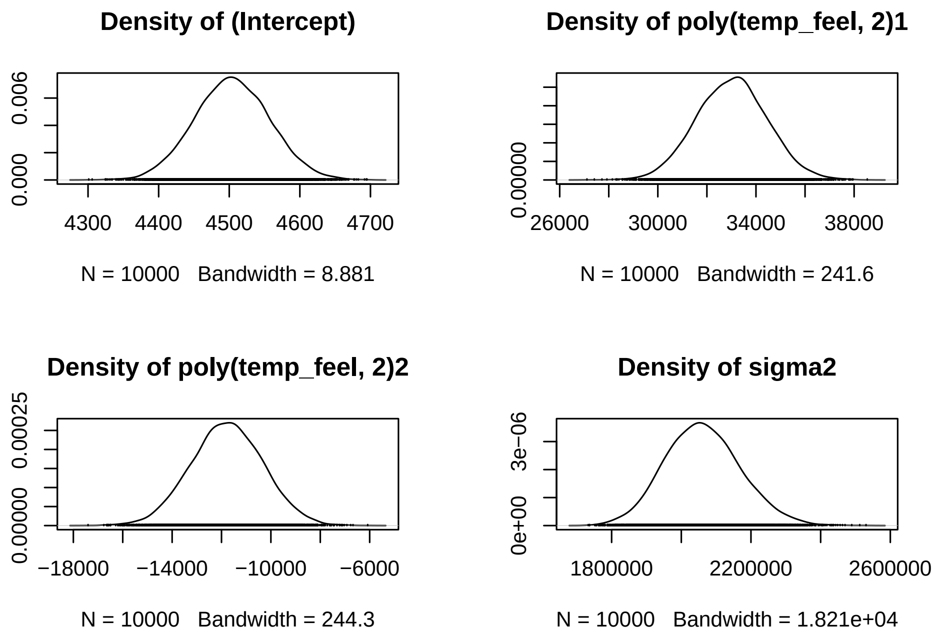
MCMC

下方結果為使用Monte-Carlo Markov Chain估計線性迴歸係數的結果，估計值和`lm`的結果相當接近，表示MCMC方法可以很有效的估計，另外的優點則是可以看到估計值的分布。

```
##
## Iterations = 1001:11000
## Thinning interval = 1
## Number of chains = 1
## Sample size per chain = 10000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##                         Mean        SD  Naive SE Time-series SE
## (Intercept)             4504     52.86    0.5286          0.528
## poly(temp_feel, 2)1   33018   1438.16   14.3816         14.382
## poly(temp_feel, 2)2  -11781   1454.42   14.5442         14.466
## sigma2              2061874 108369.34 1083.6934       1083.693
##
## 2. Quantiles for each variable:
##
##                        2.5%     25%     50%     75%   97.5%
## (Intercept)            4402    4468    4504    4540    4609
## poly(temp_feel, 2)1   30236   32034   33034   33971   35859
## poly(temp_feel, 2)2  -14639  -12743  -11773  -10790   -8923
## sigma2              1860254 1986590 2057815 2132101 2286788
```

下圖分別是模擬的結果以及係數的估計分布。





Question 5

We will apply Bayesian computing (Normal + Normal → Normal) to construct Taiwan's life tables, use Taiwan's mortality data in 2020. Try different prior distributions and compare your analysis results to the official abridged life tables. For example, you may treat the official life table as the prior. Also, you need to specify the parameters used.

```
## Normal + Normal → Normal:

## Suppose prior_variance = 0.01 and data_variance = 0.01, error: 0.02702967

## Suppose prior_variance = 0.02 and data_variance = 0.01, error: 0.03603956

## Suppose prior_variance = 0.01 and data_variance = 0.02, error: 0.01801978

## Beta + Binomial → Beta:

## Suppose alpha_prior = 2 and beta_prior = 20, error: 0.01209802

## Suppose alpha_prior = 2 and beta_prior = 10, error: 0.01219968

## Suppose alpha_prior = 2 and beta_prior = 5, error: 0.01225052
```

我們使用兩種組合來進行貝氏計算，分別是Normal + Normal以及Beta + Binomial，在Normal + Normal的部分，我們可以調整不同的先驗分配變異數與實際資料變異數來分配權重，當先驗分配的假設變異數較小，則代表對先驗分配的相信程度越高，後驗分配會與先驗分配的誤差越小。

在Beta + Binomial，我們可以調整先驗分配的α與β值，在α固定的情況下，β越大,，則代表對先驗分配的相信程度越高，後驗分配會與先驗分配的誤差越小。

Question 6

You can use `MCMClogit` in the module `MCMCpack` to obtain MCMC estimation of logistic regression analysis. Conduct the logistic regression via the `glm` and MCMC, using the data `birthwt`, and comment on the results you found. (Note: `data("birthwt", package = "MASS")`)

glm

下方結果為使用懷孕時是否抽菸`smoke`、過去早產次數`ptl`、高血壓`ht`三個變數預測新生兒是否體重過輕`low`，三者的係數皆為顯著，表示皆能有效地預測新生兒體重過輕的情況。我們將`glm`的估計結果作為基準，比較和其他估計結果的差異。

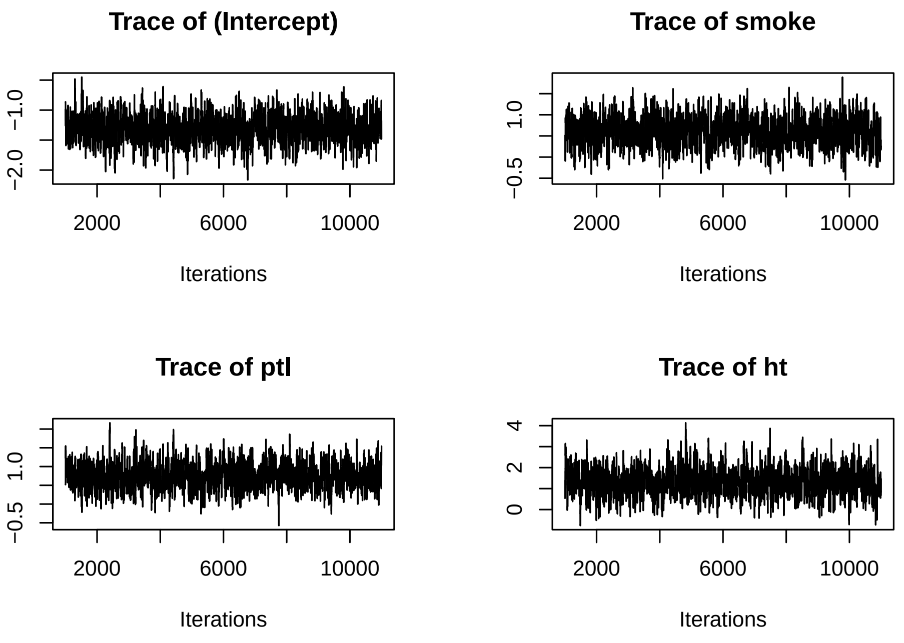
```
##
## Call:
## glm(formula = low ~ smoke + ptl + ht, family = binomial(link = "logit"),
##     data = birthwt)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -1.2873     0.2320  -5.549 2.88e-08 ***
## smoke         0.5840     0.3326   1.756   0.0792 .
## ptl           0.7257     0.3232   2.245   0.0248 *
## ht            1.2670     0.6171   2.053   0.0401 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 234.67  on 188  degrees of freedom
## Residual deviance: 220.51  on 185  degrees of freedom
## AIC: 228.51
##
## Number of Fisher Scoring iterations: 4
```

MCMC

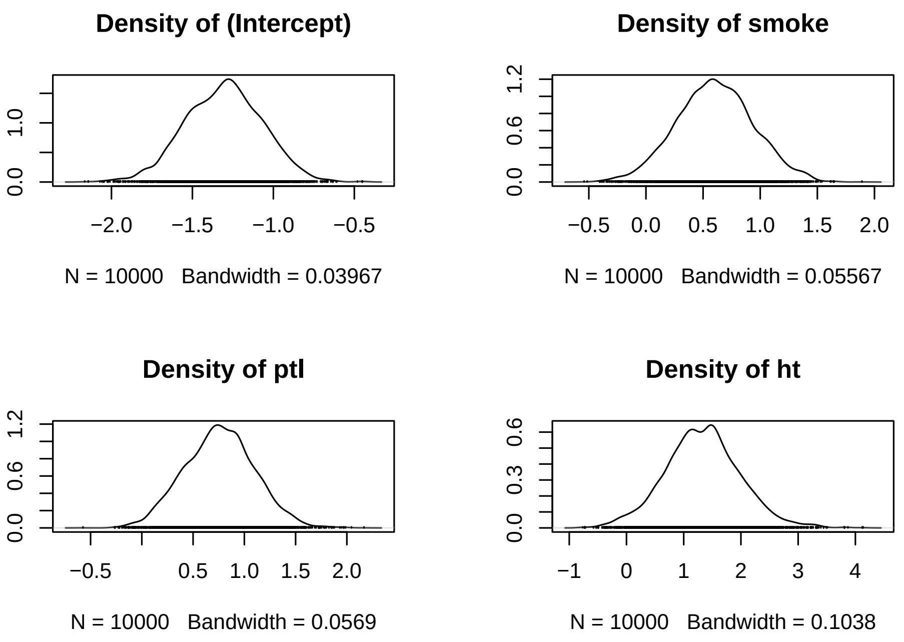
我們使用Monte-Carlo Markov Chain方法估計Logistic Regression模型，估計的結果和`glm`差異不大。

```
##
## Iterations = 1001:11000
## Thinning interval = 1
## Number of chains = 1
## Sample size per chain = 10000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##                Mean     SD Naive SE Time-series SE
## (Intercept) -1.3055 0.2361 0.002361       0.008703
## smoke        0.6042 0.3345 0.003345       0.012148
## ptl          0.7568 0.3404 0.003404       0.012659
## ht           1.3315 0.6561 0.006561       0.025116
##
## 2. Quantiles for each variable:
##
##                 2.5%     25%     50%     75%   97.5%
## (Intercept) -1.77763 -1.4704 -1.2989 -1.1454 -0.8506
## smoke       -0.04900  0.3837  0.6012  0.8277  1.2773
## ptl          0.10434  0.5238  0.7572  0.9776  1.4386
## ht           0.03766  0.9073  1.3266  1.7351  2.6688
```

除了估計的結果以外，我們還可以看到模擬的過程以及分布，下圖為MCMC的估計路徑以及估計分布。



Trace of (Intercept)
Trace of smoke
Trace of ptl
Trace of ht
Iterations
2000
6000
10000
−2.0
−1.0
−0.5
1.0
0
2
4



Density of (Intercept)
N = 10000 Bandwidth = 0.03967
Density of smoke
N = 10000 Bandwidth = 0.05567
Density of ptl
N = 10000 Bandwidth = 0.0569
Density of ht
N = 10000 Bandwidth = 0.1038

Appendix

R code

question 1

```r
set.seed(SEED)
random_unif = runif(100)
random_norm1 = rnorm(100, -2, 1)
random_norm2 = rnorm(100, 2 ,1)

mixed_data = (random_unif < 0.5) * random_norm1 + (random_unif > 0.5) * random_norm2

###. histogram density estimator
histest = function(x, h) {
  w = function(x, a, b) {
    if (x <= b & x >= a) {return(1)}
    else {return(0)}}
  n = length(x)
  sx = seq(min(x), max(x), by = h)
  a = sx[-length(sx)]
  b = sx[-1]
  ni = NULL
  for (j in 1:length(a)) {
    ni[j] = sum(x <= b[j] & x >= a[j])}
  t1 = NULL
  for (i in sort(x)){
    t0 = NULL
    for (j in 1:length(a)){
      wei = w(i, a[j], b[j])
      t0 = c(t0,wei)}
    y = 1/n * sum(ni/h * t0)
    t1 = c(t1,y)}
  return(t1)}

###. naive density estimator
naiveest = function(x, h) {
  w = function(y) {
    if (abs(y) < 1) {return(1/2)}
    else {return(0)}
  }

  n = length(x)
  sx = seq(min(x), max(x), length = 500)
  t1 = NULL

  for (i in sx) {
    t0 = NULL
    for (j in x) {
      wei = w((i - j)/h)
      t0 = c(t0, wei)
    }
    y = 1/n * sum(1/h * t0)
    t1 = c(t1, y)
  }
  return(t1)
}

###. kernel density estimator
kernelest = function(x, h) {
  w <- function(y) {dnorm(y)}
  n <- length(x)
  sx <- seq(min(x), max(x), length = 500)

  t1 <- NULL
  for (i in sx) {
    t0 <- NULL
    for (j in x) {
      wei <- w((i - j)/h)
      t0 <- c(t0, wei)
    }
    y <- 1/n * sum(1/h * t0)
    t1 <- c(t1, y)
  }
  return(t1)
}

y1 <- histest(mixed_data, 0.8)
xa <- seq(min(mixed_data), max(mixed_data), length = 500)
```

```r
y2 <- naiveest(mixed_data, 0.8)
y3 <- kernelest(mixed_data,0.8)

par(mfrow = c(1,1))
plot(sort(mixed_data), y1, typ = "l", xlab = "x", ylab = "f(x)", xlim = c(-5, 5),
     ylim = c(0, 0.3), lwd = 2)
matplot(cbind(xa,xa),cbind(y2,y3),typ=c("l", "l"),
        col = c(2, 4),lty = c(2:3), lwd = 2, main = " Three Density Estimates ", add = TRUE)
legend(1,0.3,c("histogram(h = 0.8)","naive(h = 0.8)","kernel(h = 0.8)")
       ,col=c(1,2,4),lty = c(1:3),cex = 0.8)
```

question 2

```r
# motor series plot
motor <- read_excel("motor_bikes.xlsx",
                    sheet = "motorcycles")

colnames(motor) <- c("pno", "case", "date", "time", "location")
motor[motor$date == "110420", "date"] <- 1110420

motor <- mutate(motor,
                year = as.numeric(substr(date, 1, 3)) + 1911,
                month = as.numeric(substr(date, 4, 5)),
                day = as.numeric(substr(date, 6, 7))) %>%
  mutate(date = ymd(sprintf("%04d%02d%02d", year, month, day))) %>%
  mutate(mon_num = interval(min(date), date) %/% months(1) + 1)

st_num <- table(motor$mon_num) %>%
  data.frame() %>%
  mutate(Var1 = as.numeric(Var1))
colnames(st_num) <- c("mon", "num")

num_ts <- ts(table(motor$mon_num))
num_m <- as.matrix(st_num)
ts.plot(num_ts, xlab = "month", ylab = "frequence",
        main = "Number of Stolen Motorcycles each Month")
# kernel smooth
```

```r
num_smooth_h10 <- smooth_ksmooth(num_m, bandwidth = 10)
num_smooth_h20 <- smooth_ksmooth(num_m, bandwidth = 20)
num_smooth_h60 <- smooth_ksmooth(num_m, bandwidth = 60)
num_smooth_h40 <- smooth_ksmooth(num_m, bandwidth = 40)

plot(st_num$mon, st_num$num,
     main = "Smoothed Data with Kernel Smoother",
     xlab = "month", ylab = "frequence")
lines(num_smooth_h10, lty = 2)
lines(num_smooth_h20, lty = 3)
lines(num_smooth_h40, lty = 4)
lines(num_smooth_h60, lty = 5)

legend("topright", legend = c("h=10", "h=20", "h=40", "h=60"),
       lty = c(2, 3, 4, 5), lwd = 2)
# max month
max_mon <- num_smooth_h60[which.max(num_smooth_h60[,2]), 1]
print(paste("Highest month is", (min(motor$date) + months(as.integer(max_mon)))))
# spline and LOWESS
num_spline <- smooth.spline(num_m)
num_lowess <- lowess(num_m)

plot(num ~ mon, data = num_m,
     main = "Smoothed Data with Spline and LOWESS",
     xlab = "month", ylab = "frequence")
lines(num_spline, lty = 2, lwd = 2)
lines(num_lowess, lty = 3, lwd = 2)
legend("topright", legend = c("spline", "LOWESS"),
       lty = c(2, 3), lwd = 2)
```

question 3

```r
set.seed(SEED)
a <- seq(0, 2*pi, length = 100)
b <- sin(a) + rnorm(100, 0, 0.09)
c <- sin(a)
data <- ksmooth(a, b, kernel = "normal", bandwidth = 0.1) %>% as.data.frame()
data <- cbind(b, c, data) %>% as.data.frame()
```

```r
ggplot(data,aes(x = x)) + labs(title = "kernel smooth method",x = "x",y = "sin(x)")+
  geom_point(aes(y = b),col = "#333344")+
  geom_vline(xintercept = 0, size=1)+
  geom_hline(yintercept = 0, size=1)+
  geom_line(aes(y = y),col = "#808080", lwd=1)+
  scale_x_continuous(breaks = c(0:2*pi))+
  theme(panel.grid.major = element_line(NA), panel.grid.minor
        = element_line(NA))+
  theme(panel.background = element_rect(color = "black",size = 2))+
  theme(plot.title = element_text(size = 30, face = "bold"))+
  theme(legend.title=element_text(size = 24))+
  theme(legend.text=element_text(size = 20))

MSE <- c()
for (i in 1:1000) {
  b <- NULL
  b <- sin(seq(0, 2*pi, length = 100)) + rnorm(100, 0, 0.09)
  b1 <- ksmooth(a, b, kernel = "normal", bandwidth = 0.1)$y
  MSE[i] <- mean((b1 - c)^2)
}
MSE <- mean(MSE)
cat("MSE of kernel smooth:", MSE)
set.seed(SEED)
a <- seq(0, 2*pi, length = 100)
b <- sin(a) + rnorm(100, 0, 0.09)
c <- sin(a)

data <- lowess(x = a, y = b, f = 0.23)[c("x", "y")] %>% as.data.frame()
data <- cbind(b,c,data) %>% as.data.frame()
ggplot(data,aes(x = x))+ labs(title = "LOWESS smooth method", x = "x", y = "sin(x)") +
  geom_point(aes(y = b)) +
  geom_vline(xintercept = 0,size = 1) +
  geom_hline(yintercept = 0,size = 1) +
  geom_line(aes(y = y),col = "#808080",lwd = 1) +
  scale_x_continuous(breaks = c(0:2*pi)) +
  theme(panel.background = element_rect(colour = "black",size = 2)) +
  theme(panel.grid.major = element_line(NA),panel.grid.minor
        =element_line(NA))+
  theme(plot.title = element_text(size = 30, face = "bold")) +
  theme(legend.title = element_text(size = 24))+
  theme(legend.text = element_text(size = 20))

MSE <- c()
for (i in 1:1000) {
  b <- NULL
  b <- sin(seq(0, 2*pi, length = 100)) + rnorm(100, 0, 0.09)
  b1 <- lowess(x = a, y = b, f = 0.23)$y
  MSE[i] <- mean((b1 - c)^2)
}
MSE <- mean(MSE)
cat("MSE of LOWESS:", MSE)
set.seed(SEED)
a <- seq(0, 2*pi, length = 100)
b <- sin(seq(0, 2*pi, length = 100)) + rnorm(100, 0, 0.09)
c <- sin(seq(0, 2*pi, length = 100))
mse = c()
for (k in 1:20){
  r <- running_mean(b, binwidth = k)
  x = NULL
  for(i in 1:(100 - k + 1)){
    x[i] <- mean(a[i:(i + k - 1)])
  }
  mse[k] <- mean((sin(x) - r)^2)
  num = which(mse == min(mse))
}

b <- running_mean(b, binwidth = which(mse == min(mse)))
model <- lm(b ~ poly(seq(0,2*pi, length = 100 - num + 1), 3))
y <- fitted.values(model)
data <- cbind(a[1:length(b)],b,c[1:length(b)]) %>% as.data.frame()
colnames(data) <- c("x","running mean","sin(x)")
data2 <- gather(data,key = "type",value = "value",2:3)

data2$type %<>% as.factor()
ggplot(data2) + labs(title = "Running mean")+
  theme(panel.grid.major = element_blank(),panel.grid.minor = element_blank())+
  xlim(0,2*pi) + ylim(-1,1) +
  geom_vline(xintercept = 0,size = 1)+
  geom_hline(yintercept = 0,size = 1)+
  geom_line(mapping = aes(x = x,y = value, color = type),lwd = 1.5)+
  geom_point(mapping = aes(x = x, y = value), color = "black",size = 1)+
  theme(legend.text = element_text(size = 16))+
  theme(legend.position = c(0.8,0.8))+
  theme(legend.background = element_rect(size = 0.5, linetype = "solid",fill
                                        = "#FFFFF0",colour = "black"))+
  theme(panel.background = element_rect(color = '#000000',size = 2))+
  theme(plot.title = element_text(size = 30, face = "bold"))+
  theme(legend.title = element_text(size = 24))+
  theme(legend.text = element_text(size = 20))

a <- seq(0,2*pi, length = 100)
b <- sin(seq(0,2*pi, length = 100)) + rnorm(100, 0, 0.09)
c <- sin(seq(0,2*pi, length = 100))

# MSE
c <- sin((a[-1] + a[-100])/2)
for (i in 1:1000) {
  b <- NULL
  b <- sin(seq(0,2*pi, length=100)) + rnorm(100,0,0.09)
  b1 <- running_mean(b, binwidth=2)
  MSE[i] <- mean((b1-c)^2)
}
MSE <- mean(MSE)
cat("MSE of Running mean:", MSE)
```

question 4

```
# bikes data
bikes <- read_excel("motor_bikes.xlsx", sheet = "Bikes")
plot(bikes$temp_feel, bikes$riders_total,
     xlab = "feeling temperature", ylab = "total rider",
     main = "Temperature and Rider Scatter Plot")
# lm
md1 <- lm(riders_total ~ poly(temp_feel, 2), data = bikes)
summary(md1)
# MCMC
```

```r
posterior <- MCMCregress(riders_total ~ poly(temp_feel, 2),
            b0 = 0, b1 = 0.1, sigma.mu = 5, sigma.var = 25,
            verbose = 0, data = bikes)
summary(posterior)
par(mfrow=c(2,2))
plot(posterior, density = FALSE, auto.layout = FALSE)
par(mfrow=c(2,2))
plot(posterior, trace = FALSE, auto.layout = FALSE)
```

question 5

```r
life_table_df <- read_excel('2020_birth.xlsx', col_names = FALSE)
mortality_df <- read_excel('2020_death.xlsx')




prior_means <- life_table_df[[2]]
death_counts <- life_table_df[[3]]
total_population <- life_table_df[[4]]
data_means <- as.numeric(mortality_df[1, ]) / 1000

NormalNormal <- function(prior_variance, data_variance) {
  posterior_variance = 1 / (1/prior_variance + 1/data_variance)
  posterior_means = posterior_variance * (prior_means/prior_variance + data_means/data_variance)

  posterior_df <- data.frame(
    Age = 0:(length(posterior_means) - 1),
    Prior_Mean = prior_means,
    Data_Mean = data_means,
    Posterior_Mean = posterior_means,
    Posterior_Variance = posterior_variance
  )

  error <- posterior_df$Posterior_Mean - posterior_df$Prior_Mean
  return(sum(abs(error)))
}

BetaBinomial <- function(alpha_prior, beta_prior) {
  alpha_posterior = alpha_prior + death_counts
  beta_posterior = beta_prior + total_population - death_counts
  posterior_means = alpha_posterior / (alpha_posterior + beta_posterior)

  result_df = data.frame(
    Age = 0:(length(posterior_means) - 1),
    Official_Mortality_Rate = prior_means,
    Death_counts = round(death_counts, 0),
    Total_population = round(total_population, 0),
    Posterior_Mean = posterior_means)

  error <- result_df$Posterior_Mean - result_df$Official_Mortality_Rate
  return(sum(abs(error)))
}

cat("Normal + Normal → Normal:\n")
cat("Suppose prior_variance = 0.01 and data_variance = 0.01, error:",
    NormalNormal(0.01, 0.01), "\n")
cat("Suppose prior_variance = 0.02 and data_variance = 0.01, error:",
    NormalNormal(0.02, 0.01), "\n")
cat("Suppose prior_variance = 0.01 and data_variance = 0.02, error:",
    NormalNormal(0.01, 0.02), "\n")

cat("\n")
cat("Beta + Binomial → Beta:\n")
cat("Suppose alpha_prior = 2 and beta_prior = 20, error:",
    BetaBinomial(2, 20), "\n")
cat("Suppose alpha_prior = 2 and beta_prior = 10, error:",
    BetaBinomial(2, 10), "\n")
cat("Suppose alpha_prior = 2 and beta_prior = 5, error:",
    BetaBinomial(2, 5), "\n")
```

question 6

```r
data("birthwt", package = "MASS")

# glm
md1 <- glm(low ~  smoke + ptl + ht, family = binomial(link = "logit"), data = birthwt)
```

```r
summary(md1)
# MCMC
posterior <- MCMClogit(low ~ smoke + ptl + ht, b0=0, B0=.001,
                       data=birthwt)
summary(posterior)
par(mfrow=c(2,2))
plot(posterior, density = FALSE, auto.layout = FALSE)
par(mfrow=c(2,2))
plot(posterior, trace = FALSE, auto.layout = FALSE)
```