Bayesian Econometrics for Business Economics: Assignment 1

By Group 15

Kacper Kaznowski	XXXXXXX
Lanlan Hou	2801069
Wa Chak Sou	2796840

School of Business and Economics

12th November 2025

Model with normal distribution and Gibbs sampling

a. With a non-informative prior (flat prior) for μ

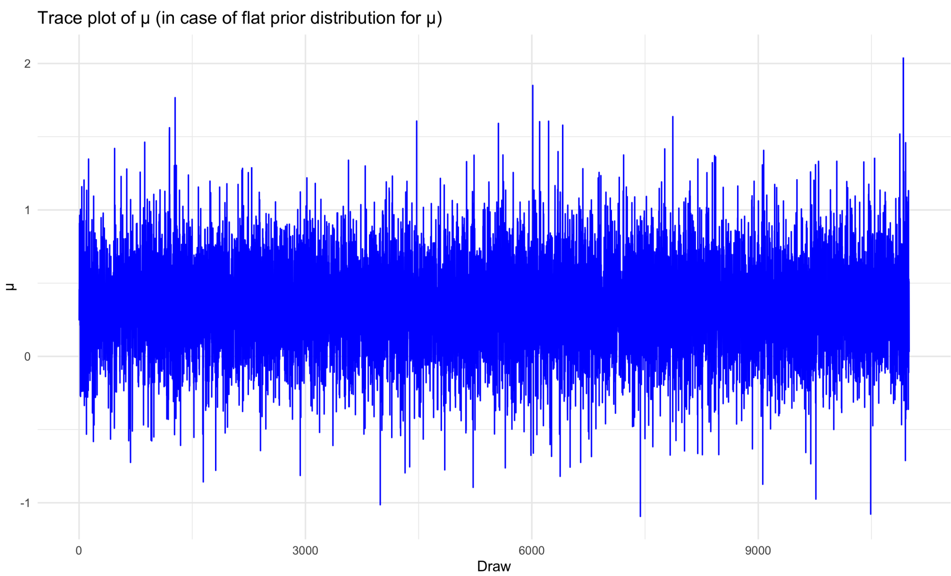

Figure 1.1 Trace plot of μ (in case of flat prior distribution for μ)

The posterior probabilities are: $\Pr(\mu > 0|y) = 0.8766$ and $\Pr(\mu < 0|y) = 0.1235$.

b. With a normal prior density for μ: $m_{prior} = 0$, $v_{prior} = 10000$

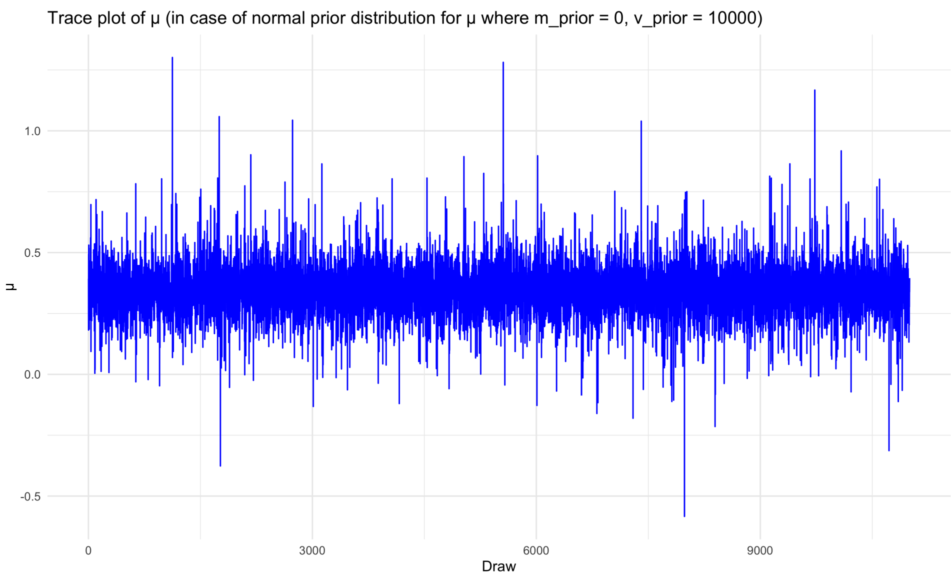

Figure 1.2 Trace plot of μ (in case of normal prior distribution for μ where $m_{prior} = 0$, $v_{prior} = 10000$)

The posterior probabilities are: $\Pr(\mu > 0|y) = 0.9962$ and $\Pr(\mu < 0|y) = 0.0039$.

c. With a normal prior density for μ: $m_{prior} = 0.5$, $v_{prior} = 0.0625$

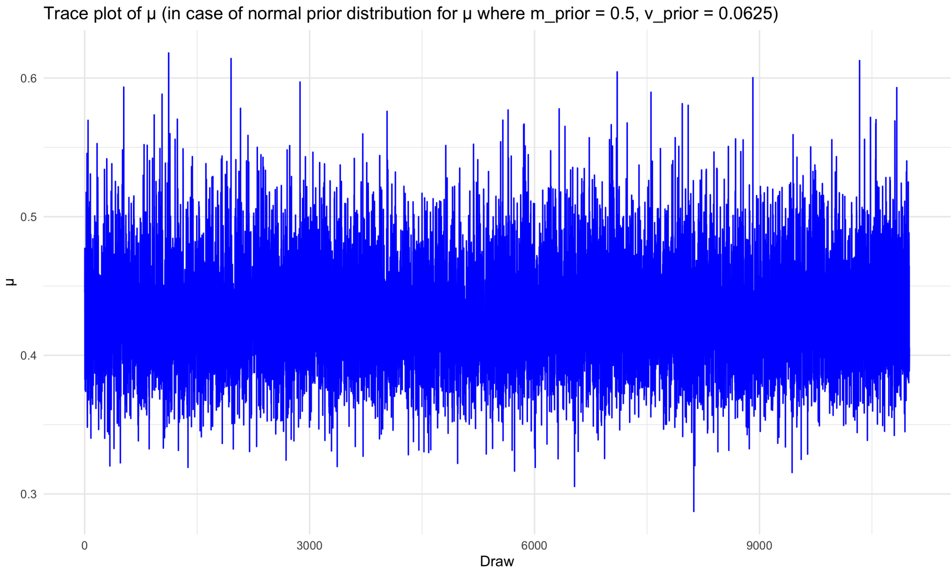

Figure 1.3 Trace plot of μ (in case of normal prior distribution for μ where $m_{prior} = 0.5$, $v_{prior} = 0.0625$)

The posterior probabilities are: $\Pr(\mu > 0|y) = 1.00$ and $\Pr(\mu < 0|y) = 0.00$.

d. A classical/frequentist two-sided test

Prior Distribution for μ	Test Statistic	Accept Region	Decision (5% level)
(Improper) Non-informative prior	116.0842	[-2.2622, 2.2622]	Reject H_0
$\mu \sim N(0, 10000)$	371.2741	[-2.2622, 2.2622]	Reject H_0
$\mu \sim N(0.5, 0.0625)$	1144.5673	[-2.2622, 2.2622]	Reject H_0

Table 1.4: Classical two-sided t-test results for cases (a), (b), and (c)

Appendix

```r
library(openxlsx)
raw <- read.xlsx('student_groups_stocks.xlsx', sheet = 1)

groupNumber <- 15
nameOfStock <- raw$Stock.Name[groupNumber]
startDate <- raw$Start.Date[groupNumber]
endDate <- raw$`End.Date.(+10y)`[groupNumber]

# Compute the start date of the final five years
library(lubridate)

originalDate <- as.Date(startDate)
finalFiveYearsStartDate <- originalDate %m+% years(5)

# Extract the daily stock price from source
library(quantmod)

getSymbols(nameOfStock, src = 'yahoo', from = finalFiveYearsStartDate, to = endDate)
date <- index(HSBC)

# Adjusted Close
adjustedPrice <- as.numeric(HSBC[,'HSBC.Adjusted'])

# Log Returns
compoundedReturns <- numeric(length(adjustedPrice) - 1)
for(i in (2:length(adjustedPrice))){
  compoundedReturns[i-1] <- log(adjustedPrice[i] / adjustedPrice[i-1])
}
# Squared Log Returns
squaredCompoundedReturns <- compoundedReturns ^ 2

# Function to compute the ACF and plot the graph
plotACF <- function(input_list, input_maxLags, objectName, savePath) {
  ACFResult <- acf(input_list, lag.max = input_maxLags, plot = FALSE)

  ACFValues <- ACFResult$acf[-1]
  lags <- ACFResult$lag[-1]
  numberOfObservations <- length(input_list)
  confidenceBands <- 1.96 * 1 / sqrt(length(input_list))

  library(ggplot2)
  plot_df <- data.frame(lag = lags, acf = ACFValues)

  ggplot(plot_df, aes(x = lag, y = acf)) +
    geom_bar(aes(color = 'ACF'), stat = 'identity',
             fill ='blue', width = 0.05) +
    geom_hline(yintercept = 0, color = 'black') +
    geom_hline(aes(yintercept = -confidenceBands,
               color = '95% Confidence Interval'), linetype = 'dashed') +
    geom_hline(aes(yintercept = confidenceBands,
               color = '95% Confidence Interval'), linetype = 'dashed') +
    scale_color_manual(name = 'Components',
                       values = c('ACF' = 'blue',
```

```
                                    '95% Confidence Interval' = 'red')) +
    labs(title = paste('ACF for HSBC', objectName, 'for', input_maxLags
        , 'Lags (2015M11-2020M11)'),
         x = 'Lag', y = 'ACF') +
    theme_minimal() +
    theme(legend.position = c(0.95, 0.95),
          legend.justification = c("right", "top"),
          legend.title = element_text(size = 12, face = "bold"),
          legend.text = element_text(size = 10))
  ggsave(savePath, width = 10, height = 6, dpi = 300)
}

returns_result <- plotACF(compoundedReturns,
                          input_maxLags = 50,
                          objectName = 'Log Returns',
                          savePath = 'figures/returns_acf_plot.png')

squaredReturns_result <- plotACF(squaredCompoundedReturns,
                                 input_maxLags = 50,
                                 objectName = 'Squared Log Returns',
                                 savePath = 'figures/squaredReturns_acf
                                    _plot.png')
```

Figure 1: Question 1(a)

```
# Function to perform the Ljung-Box test and report the result
LBTest <- function(input_series, input_maxLags, steps){
  steppedLags <- seq(steps, input_maxLags, by = steps)

  result_df <- data.frame(Lag = integer(),
                          Statistic = numeric(),
                          Crit_Value = numeric(),
                          P_Value = numeric())

  for (lag in steppedLags) {
    lb_test <- Box.test(input_series, lag = lag, type = 'Ljung-Box')
    testStatistic <- as.numeric(lb_test$statistic)
    pValue <- as.numeric(lb_test$p.value)
    criticalValue <- qchisq(0.95, df = lag)

    newRow <- data.frame(Lag = lag,
                         Statistic = round(testStatistic, 4),
                         Critical_Value = round(criticalValue, 4),
                         P_Value = signif(pValue, 4))

    result_df <- rbind(result_df,newRow)
  }
  return(result_df)
}

returns_LBTest <- LBTest(compoundedReturns, 50, 10)
returns_LBTest
returns_LBTest <- LBTest(squaredCompoundedReturns, 50, 10)
returns_LBTest
```

Figure 2: Question 1(b)

```
# compute the MA volatility
window <- 300
```

```

hat_y <- numeric(length(compoundedReturns)-window)
MA_volatility <- numeric(length(compoundedReturns)-window)

for (t in window:length(compoundedReturns)){
  hat_y[t-window+1] <- mean(compoundedReturns[(t-window+1):(t)])
  MA_volatility[t-window+1] <- sqrt(1/(window-1) * sum(sapply(0:(window-1),
                                function(j) (compoundedReturns[t-j])^2)))
}

# compute the EWMA volatility
lambda <- 0.94

EWMA_volatility <- numeric(length(compoundedReturns)+1)

for (t in 1:length(compoundedReturns)){
  EWMA_volatility[t+1] <- sqrt((1 - lambda) * (compoundedReturns[t]^2)
                          + lambda * EWMA_volatility[t]^2)
}

MA_df <- data.frame(MA_vol = MA_volatility,
                    EWMA_vol = EWMA_volatility[window:length(compoundedReturns)],
                    date = date[window:length(compoundedReturns)])

ggplot(MA_df, aes(x = date)) +
  geom_line(aes(y = MA_vol, color = 'MA Volatility')) +
  geom_line(aes(y = EWMA_vol, color = 'EWMA Volatility')) +
  labs(title = paste('MA Volatility (W=', window,
                     ') and EWMA Volatility (  =', lambda,
                     ') for HSBC Log Returns (2015M11-2020M11)'),
       x = 'Date', y = 'Volatility', color = 'Model') +
  theme_minimal()
ggsave('figures/ma_plot.png', dpi = 300)
```

Figure 3: Question 2(a)

```
# calculate standrized residuals
residual <- (compoundedReturns[(window+1):length(compoundedReturns)])
residual_MA <- residual / MA_volatility[1:(length(hat_y)-1)]
residual_EWMA <- residual / EWMA_volatility[window:(length(compoundedReturns)-1)]

# residual autocorrelation checking
MA_result <- plotACF(residual_MA,
                     input_maxLags = window,
                     objectName = 'Log Returns',
                     savePath = 'figures/residuals_MA_acf_plot.png')
EWMA_result <- plotACF(residual_EWMA,
                       input_maxLags = window,
                       objectName = 'Squared Log Returns',
                       savePath = 'figures/residuals_EWMA_acf_plot.png')

# Distribution of Standardized Residuals
residual_df <- data.frame(MA_vol =  residual_MA,
                          EWMA_vol = residual_EWMA)
```

```

ggplot(residual_df, aes(x = MA_vol)) +
  geom_histogram(aes(y = ..density..),
                 bins = 50, fill = 'skyblue', color = 'blue') +
  stat_function(fun = dnorm,
                args = list(mean = mean(residual_MA, na.rm = TRUE),
                            sd = sd(residual_MA, na.rm = TRUE)),
                color = 'black', linewidth = 1, linetype = 'dashed') +
  labs(title = 'Under MA Volatility Models',
       x = 'Standardized Residuals', y = 'Density')
ggsave('figures/MA_residual_distribution.png', width = 6, height = 6,
    dpi = 300)

ggplot(residual_df, aes(x = EWMA_vol)) +
  geom_histogram(aes(y = ..density..),
                 bins = 50, fill = 'orange', color = 'darkorange') +
  stat_function(fun = dnorm,
                args = list(mean = mean(residual_EWMA, na.rm = TRUE),
                            sd = sd(residual_EWMA, na.rm = TRUE)),
                color = 'black', linewidth = 1, linetype = 'dashed') +
  labs(title = 'Under EWMA Volatility Models',
       x = 'Standardized Residuals', y = 'Density')
ggsave('figures/EWMA_residual_distribution.png', width = 6, height = 6,
     dpi = 300)

# Ljung-Box Test
MA_LBTest <- LBTest((residual_MA^2), 50, 10)
MA_LBTest
EWMA_LBTest <- LBTest((residual_EWMA^2), 50, 10)
EWMA_LBTest

library('tseries')

# Jarque-Bera Test
jarque.bera.test(residual_MA)
jarque.bera.test(residual_EWMA)
```

Figure 4: Question 2(b)