Solutions to BDA Assignment 1, 2020/2021 Semester 2

Name: Chen Tao **Student ID: s2022911**

IMPORTANT: EVERY ANSWER NEEDS TO HAVE AT LEAST ONE SENTENCE OF EXPLANATION **CLEARLY DEMONSTRATING THAT YOU UNDERSTAND WHAT YOU ARE DOING**.
SOLUTIONS WITHOUT EXPLANATION WILL BE MARKED AS 0, IRRESPECTIVELY WHETHER THE COMPUTATIONS ARE CORRECT OR NOT.

FOR THE RESULTS, YOU CAN COPY PASTE THE PLOTS AND NUMBERS FROM KAGGLE.
THE SOLUTION HAS TO BE CLEARLY EXPLAINED AND READABLE WITHOUT LOOKING AT YOUR CODE.

BESIDES SUBMITTING THIS REPORT, YOU ARE ALSO REQUIRED TO SUBMIT YOUR IPYNB NOTEBOOK FROM KAGGLE. YOUR CODE HAS TO BE ABLE TO RUN THROUGH ALL QUESTIONS AND REPRODUCE EVERY RESULT IN THIS REPORT.

Please note that the results of the code for the questions of this assignment may have some (slight) variability over time due to randomness.

Problem 1 - Gray Whale Data

Question (a)

The parameter b is the coefficient of the autoregressive process of order 1 (AR(1)), $\{x_t\}$, so it determines the degree of dependence of the log population size x_t at any time t on the previous values. The greater the parameter b, the greater the dependence of x_t on previous values would be, and vice versa.

The parameter u is the intercept of the AR(1) process $\{x_t\}$. This parameter, together with b, determines the mean of the process $\{x_t\}$. We can see that

$$E(x_t) = b * E(x_{t-1}) + u = \cdots = b^{(t-p)}E(x_p) + \sum_{k=0}^{t-p-1} b^k u = b^{(t-p)}E(x_p) + \frac{1-b^{(t-p)}}{1-b}u$$

Hence, if $|b| < 1$ and $p \to -\infty$, the mean of x_t is $\mu = u/(1-b)$.

The parameter σ^2 is the variance of the white noise ω_t of the AR(1) process $\{x_t\}$. It controls the fluctuation amplitude, or in other words, variance, of the process $\{x_t\}$. At any time t, if σ^2 is large, x_t will tend to stay away bx_{t-1}, and if σ^2 is small, x_t is more likely to approach bx_{t-1}.

The parameter η^2 is the variance of the white noise v_t. The effect of it is similar to σ^2. It controls the fluctuation amplitude, or in other words, variance, of the process $\{y_t\}$. If η^2 is large, the observed log population size y_t will tend to deviate from the log population size x_t, and if η^2 is small, y_t will be likely to be near to x_t.

Question (b)

There are some years in the data set `graywhales` are missing, so, to begin with, I will add the missing years and denote `NA` for the estimates of those years.

```
# Initialize some variables
n = 47 # The total period of years
# Rebuild the variable `Count`
Year = 1951:1997; Count = rep(NA, n)
for (k in Year){
  if (k %in% graywhales[, 'Year']){
    ind = which(graywhales[, 'Year'] == k)
    Count[(k - Year[1] + 1)] = graywhales[, 'Count'][ind]
  }
}
```

The model string that contains the BUGS code for specifying the model is shown as follows. Because we are required to fit the model between years 1951-1997 and the index of the BUGS language also starts from 1, I use `x[1]` and `y[1]` to represent the states in the year 1951, and use `x[2]` and `y[2]` to represent the states in 1952, … , and finally use `x[47]` and `y[47]` to represent the states in 1997.

```
# The JAGS model
model_string = 'model{
  # The likelihood
  x[1] ~ dnorm(mu0, prec0)
  y[1] ~ dnorm(x[1], tau_y)
  for (t in 2:n){
    y[t] ~ dnorm(x[t], tau_y)
    x[t] ~ dnorm(mu[t-1], tau_x)
    mu[t-1] = b*x[t-1] + u
  }

  # The priors for the parameters b and u
  b ~ dunif(low, up)
  u ~ dexp(lambda)

  # The priors for the precisions tau_x and tau_y
  tau_x ~ dgamma(a1, b1)
  tau_y ~ dgamma(a2, b2)

  # Calculate the variances
  sigma_sq = 1/tau_x
  eta_sq = 1/tau_y
}'
```

In the BUGS code, we no longer need to make a specific treatment on the fact that the observations are not available (NA) for every year in the period. JAGS will treat response variables that are NA as unobserved stochastic nodes. If we include the names of these variables in the argument `variable.names` of the function `coda.samples`, samples from the posterior predictive distribution of these variables are included in the MCMC output of `coda.samples`. Therefore, I will first extract the variable names of the missing observations of the response variable `Count` and then include them among the variable names in the argument `variable.names`.

```
# Extract the variable names of the missing observations
```

```
NA.indicator = c()
for(i in 1:n){
  if(is.na(Count[i])){
    NA.indicator = c(NA.indicator, paste('y[', i, ']'))
  }
}
```

The function `jags.model` is used to compile the JAGS model and the `n.adapt` argument of it specifies the burn-in period of the model (in our question 2000). The function `coda.samples` is used to take samples from the posterior distributions of the parameters, and for this question, we need to obtain 10000 posterior samples. I will run the model for 500000 iterations after the burn-in period and perform a thinning of 50 due to the high correlation of the chains for the regression coefficient b and the intercept u. Also, I will run two chains in total herein, because we need to compute the Gelman-Rubin diagnostics and produce the Gelman-Rubin plots in question (c).

```
data = list(y = log(Count), n = n, mu0 = log(2500), prec0 = 1, low = 0, up = 1, lambda = 1,
            a1 = 0.1, b1 = 0.1, a2 = 0.1, b2 = 0.1)
model = jags.model(textConnection(model_string), data = data, n.chain = 2, n.adapt = 2000)
sample = coda.samples(model, variable.names = c('b', 'u', 'sigma_sq', 'eta_sq', 'x', NA.indicator),
                      n.iter = 500000, thin = 50)
```

```
Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 24
   Unobserved stochastic nodes: 74
   Total graph size: 203

Initializing model
```

In the output of the code, the 24 observed stochastic nodes are the 24 observed log population sizes y_t that are not missing. The 74 unobserved stochastic nodes consist of the 23 missing y_t, and the 47 log population sizes x_t (include the states of the year 1951, namely `y[1]` and `x[1]`), and the four parameters b, u, σ^2, η^2.

Question (c)

For this question, we only need to study the parameters b, u, σ^2 and η^2, so firstly I extract the samples of these parameters (i.e. exclude the samples of the variables x and y).

```
sample2 = sample[, c('b', 'u', 'sigma_sq', 'eta_sq')]
```

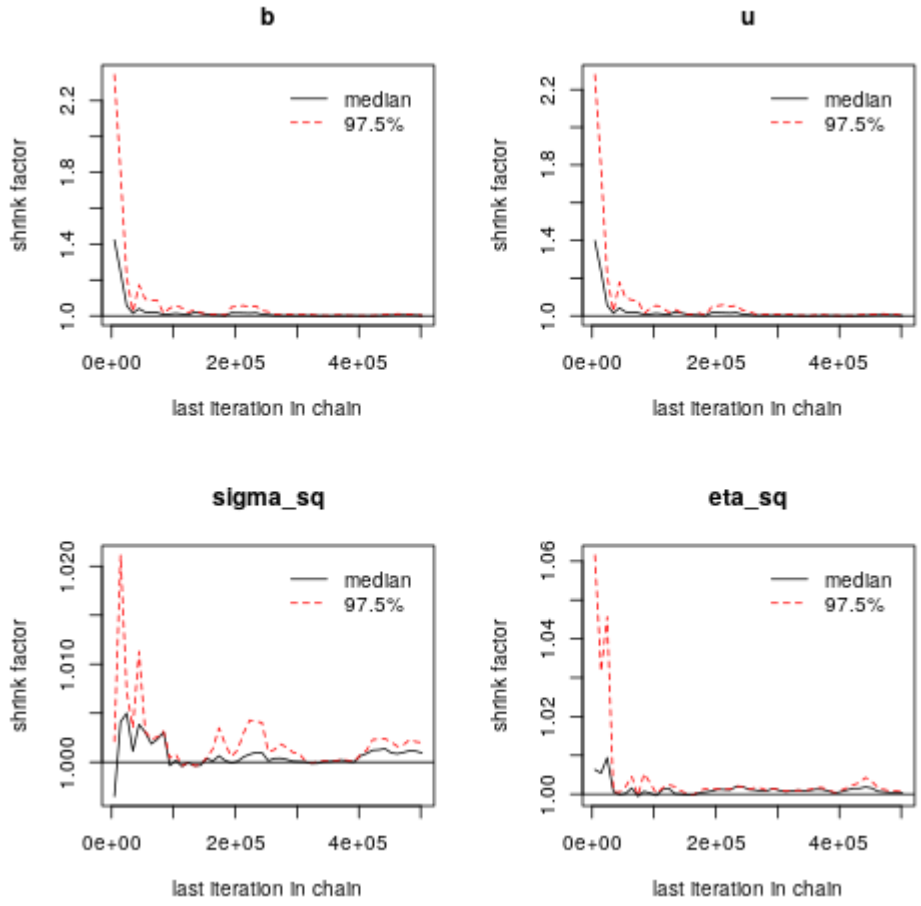
The Gelman-Rubin diagnostics can be computed using the function `gelman.diag` and the Gelman-Rubin plots can be produced using the function `gelman.plot`. We can see that the estimated Gelman-Rubin statistic $\hat{R}$ and the upper confidence interval for $\hat{R}$ of all these four parameters and the multivariate extension of $\hat{R}$ are all equal to 1, which indicates good mixing of the chain.

```
gelman.diag(sample2)
```

```
Potential scale reduction factors:

         Point est. Upper C.I.
b                 1          1
u                 1          1
sigma_sq          1          1
```

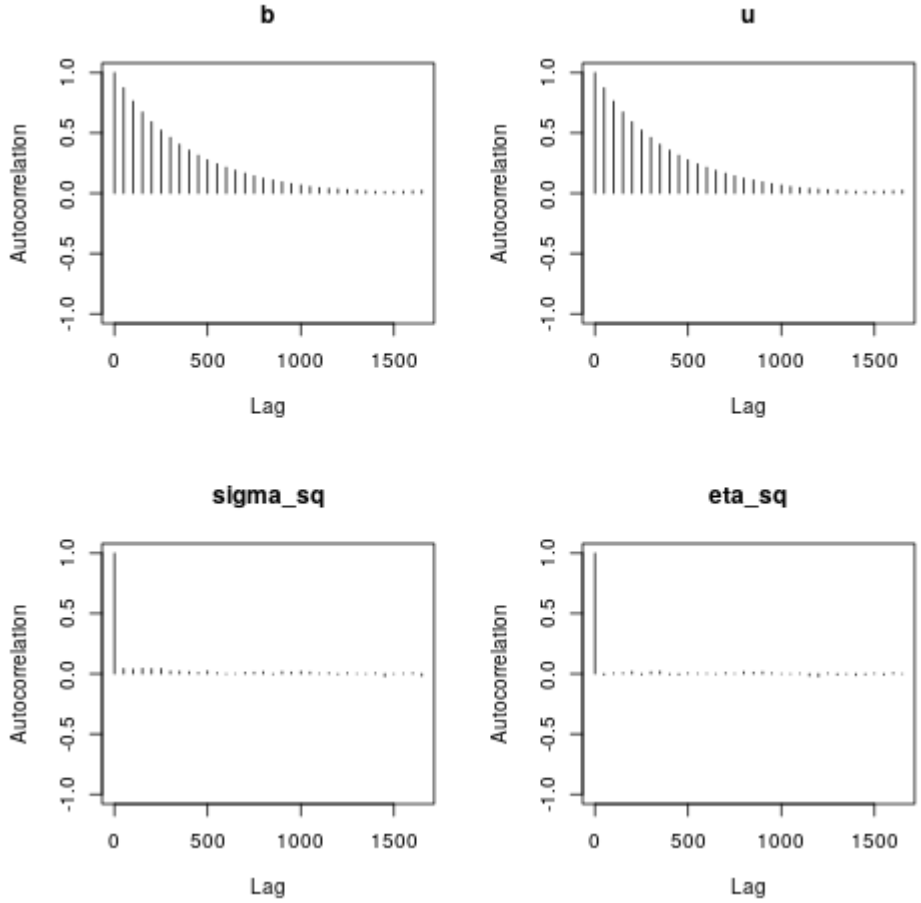
```
eta_sq            1          1

Multivariate psrf

1
```

```
par(mfrow = c(2, 2))
gelman.plot(sample2)
```



The autocorrelation functions of the parameters b, u, σ^2 and η^2 can be plotted using the function `autocorr.plot` and the effective sizes of them can be obtained using the function `effectiveSize`. I use the first chain of these parameters as a representative to analyse their autocorrelation functions, effective sample sizes and the quality of mixing and convergence of the chains for them. We can see from the autocorrelation plots that the autocorrelation functions of σ^2 and η^2 decrease rapidly, especially η^2. However, the decreases of the autocorrelation of b and u are all slow. The effective sample sizes of b and u are more than 600, and that of σ^2 is about 6626, and that of η^2 is almost 10000. To sum up, the mixings of the chain for the parameters η^2 and σ^2 are good, but the mixings of the chain for b and u are very slow. Because the chains for b and u are highly correlated, I have run 500000 iterations and perform a thinning of 50 to ensure their effective sample sizes are at least 500.

```
sample3 = sample2[[1]]
par(mfrow = c(2, 2))
autocorr.plot(sample3)
```




```
effectiveSize(sample3)
```

b
643.33137926734
u
641.170721496752
sigma_sq
6625.98369906281
eta_sq
9999.99999999997

Question (d)

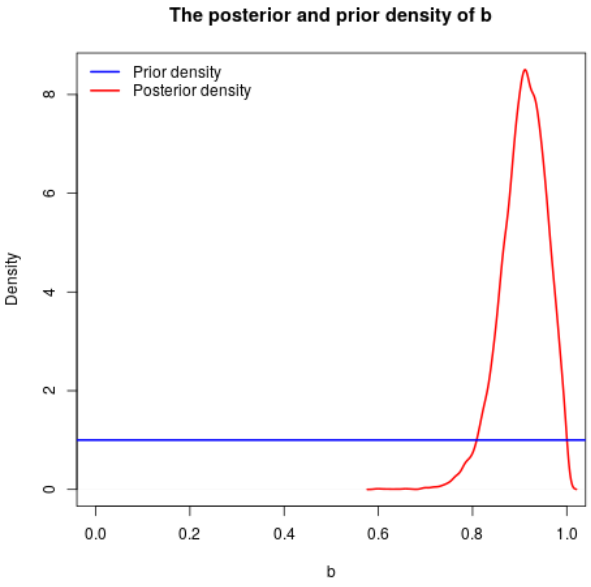
The summary statistics for the four parameters b, u, σ^2 and η^2 are printed out as follows.

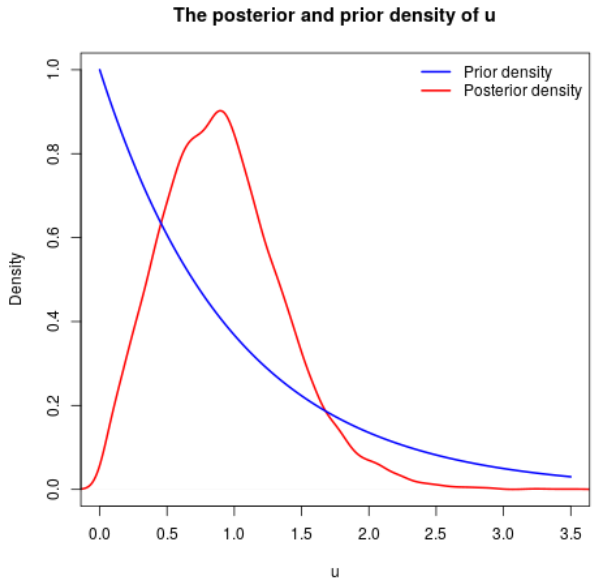
```
summary(sample2)

Iterations = 2050:502000
Thinning interval = 50
Number of chains = 2
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
b        0.90719 0.04844 0.0003425      0.0013324
u        0.92440 0.45916 0.0032467      0.0126037
sigma_sq 0.03589 0.01678 0.0001187      0.0001437
eta_sq   0.03800 0.01781 0.0001260      0.0001276
```

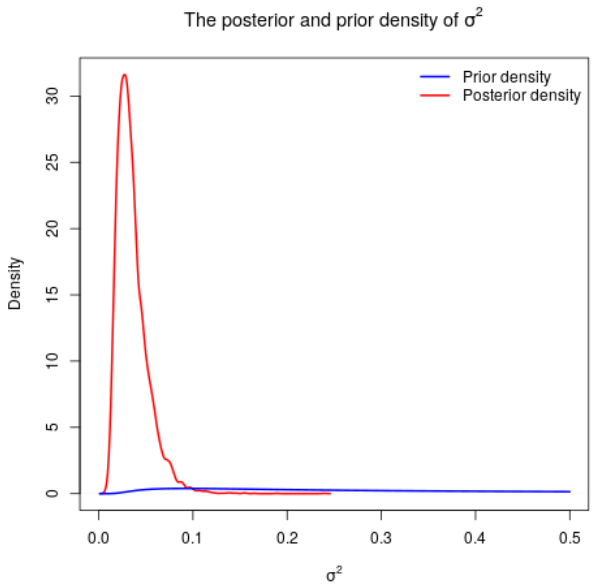
```
2. Quantiles for each variable:

             2.5%     25%     50%     75%   97.5%
b         0.80043 0.87775 0.91094 0.94230 0.98819
u         0.15985 0.59161 0.88853 1.20387 1.93226
sigma_sq 0.01468 0.02424 0.03226 0.04334 0.07835
eta_sq    0.01577 0.02572 0.03404 0.04576 0.08340
```

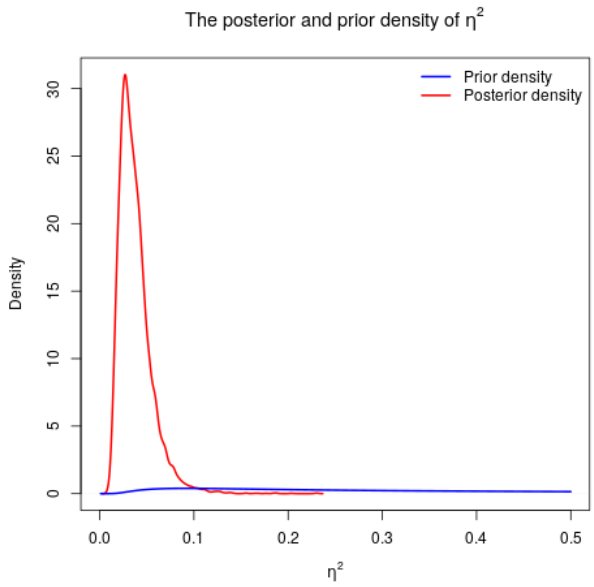
Now, let me plot the posterior and prior densities of these parameters respectively and compare the posterior density with the prior density for each of them.

```
# The posterior density and the prior density of the parameter b
plot(density(sample3[, 'b']), col = 'red', lwd = 2, xlab = 'b', ylab= 'Density',
     xlim = c(0, 1), main = 'The posterior and prior density of b')
abline(h = 1, lw = 2, col = 'blue')
legend('topleft', legend = c('Prior density', 'Posterior density'), col = c('blue', 'red'),
       lwd = 2, bty = 'n')

# The posterior density and the prior density of the parameter u
plot(density(sample3[, 'u']), col = 'red', lwd = 2, xlab = 'u', ylab= 'Density',
     xlim = c(0, 3.5), ylim = c(0, 1), main = 'The posterior and prior density of u')
u_seq = seq(0, 3.5, 0.01)
lines(u_seq, exp(-u_seq), col = 'blue', lwd = 2)
legend('topright', legend = c('Prior density', 'Posterior density'), col = c('blue', 'red'),
       lwd = 2, bty = 'n')

# The posterior density and the prior density of the parameter sigma^2
plot(density(sample3[, 'sigma_sq']), col = 'red', lwd = 2, xlab = expression(sigma^2), ylab= 'Density',
     xlim = c(0, 0.5), main = expression(paste('The posterior and prior density of ', sigma^2)))
inv_gamma_den = function(x, a, b){(b^a)*(x^(-a-1))*exp(-b/x)/gamma(a)}
sigma2_seq = seq(0, 0.5, 0.001)
lines(sigma2_seq, inv_gamma_den(sigma2_seq, a = 0.1, b = 0.1), col = 'blue', lwd = 2)
legend('topright', legend = c('Prior density', 'Posterior density'), col = c('blue', 'red'),
       lwd = 2, bty = 'n')

# The posterior density and the prior density of the parameter eta^2
plot(density(sample3[, 'eta_sq']), col = 'red', lwd = 2, xlab = expression(eta^2), ylab= 'Density',
     xlim = c(0, 0.5), main = expression(paste('The posterior and prior density of ', eta^2)))
eta2_seq = seq(0, 0.5, 0.001)
lines(eta2_seq, inv_gamma_den(eta2_seq, a = 0.1, b = 0.1), col = 'blue', lwd = 2)
legend('topright', legend = c('Prior density', 'Posterior density'), col = c('blue', 'red'),
       lwd = 2, bty = 'n')
```









From these figures, we can see that the posterior distributions of these four parameters all have huge differences with their prior distributions. For the parameter b, we have adopted a uniform prior on (0, 1) in the beginning, but the posterior of it concentrates on (0.8, 1) and its posterior mean is 0.907. For the parameter u, the prior of it is an exponential distribution with the rate parameter being equal to 1. The posterior distribution of it is a kind of left-skewed distribution and its posterior mean is 0.924. The priors of the parameters σ^2 and η^2 are all inverse Gamma distribution with parameters (0.1, 0.1) which can be viewed as a non-informative prior on $(0, +\infty)$. The posterior distributions of σ^2 and η^2 are very similar. They are all extremely concentrated on the region (0, 0.1). The posterior means of them are also very close, which are 0.036 and 0.038 respectively.

Question (e)

To perform a prior sensitivity analysis, we need to alter the prior parameters of the JAGS model and rerun the Markov chain for the model parameters b, u, σ^2 and η^2. In order to improve the efficiency of the code, I implement a function called `prior_analysis` which is specifically used to conduct prior sensitivity analyses. It takes all the parameters of the JAGS model constructed in (b) and some arguments of the function `coda.samples` as its arguments. Finally, this function will return the posterior samples of all the four parameters, the MCMC output of the function `coda.samples`. The default values of the arguments of this function are all the same as the parameters used to run the model in (b). In each sampling from the posteriors of these parameters, I will run the model for 500000 iterations and perform a thinning of 50 after a burn-in period of 2000 iterations, and I will run only one chain each time.

```
# Construct a function to perform the sensitivity analyses to avoid repetitive code
prior_analysis = function(mu0 = log(2500), prec0 = 1, low = 0, up = 1, lambda = 1, a1 = 0.1, b1 = 0.1,
                          a2 = 0.1, b2 = 0.1, n.chain = 1, n.adapt = 2000, n.iter = 500000, thin = 50){
  data.prior_ana = list(y = log(Count), n = n, mu0 = mu0, prec0 = prec0, low = low, up = up,
                        lambda = lambda, a1 = a1, b1 = b1, a2 = a2, b2 = b2)
  model.prior_ana = jags.model(textConnection(model_string), data = data.prior_ana,
                               n.chain = n.chain, n.adapt = n.adapt, quiet = TRUE)
  sample.prior_ana = coda.samples(model.prior_ana, c('b', 'u', 'sigma_sq', 'eta_sq'),
                                  n.iter = n.iter, thin = thin)
  return(sample.prior_ana)
}
```

For the prior of the parameter b, I think it is reasonable to let its support interval start from 0. Therefore, I fix the lower bound of the support interval of the uniform distribution to be 1 and change the upper bound of it to be 1, 2 and 4, and then observe the effect on the posterior of b. We can see from the summary statistics of b that there is no evident effect on the posterior of b when changing the upper bound of the support interval.

```
# The prior sensitivity analysis for the parameter b
summary(prior_analysis(low = 0, up = 1))

Iterations = 2050:502000
Thinning interval = 50
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
b        0.90694 0.04999 0.0004999      0.0020206
eta_sq   0.03822 0.01829 0.0001829      0.0001829
sigma_sq 0.03609 0.01752 0.0001752      0.0002236
u        0.92634 0.47397 0.0047397      0.0191096

2. Quantiles for each variable:

            2.5%     25%     50%     75%   97.5%
b        0.79321 0.87694 0.91103 0.94305 0.98882
eta_sq   0.01575 0.02585 0.03436 0.04591 0.08436
sigma_sq 0.01451 0.02428 0.03211 0.04357 0.07996
u        0.15067 0.58433 0.88823 1.20767 1.99856

summary(prior_analysis(low = 0, up = 2))

Iterations = 2050:502000
Thinning interval = 50
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
b        0.90858 0.05102 0.0005102      0.0020778
eta_sq   0.03852 0.01829 0.0001829      0.0001829
sigma_sq 0.03633 0.01726 0.0001726      0.0002347
u        0.91180 0.48301 0.0048301      0.0195526

2. Quantiles for each variable:

            2.5%     25%     50%     75%   97.5%
b        0.79549 0.87736 0.91266 0.94564 0.99261
eta_sq   0.01611 0.02595 0.03445 0.04599 0.08576
sigma_sq 0.01454 0.02421 0.03247 0.04396 0.07971
u        0.11783 0.56357 0.87348 1.20783 1.99253

summary(prior_analysis(low = 0, up = 4))

Iterations = 2050:502000
Thinning interval = 50
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
```

```
b        0.90685 0.04941 0.0004941      0.0018742
eta_sq   0.03804 0.01813 0.0001813      0.0001847
sigma_sq 0.03584 0.01705 0.0001705      0.0002086
u        0.92814 0.46762 0.0046762      0.0176822

2. Quantiles for each variable:

            2.5%     25%     50%     75%   97.5%
b        0.80029 0.87579 0.90981 0.94264 0.99083
eta_sq   0.01547 0.02582 0.03405 0.04553 0.08402
sigma_sq 0.01449 0.02418 0.03219 0.04309 0.07831
u        0.13663 0.58965 0.89850 1.21974 1.93624
```

To conduct prior sensitivity analysis for u, I totally use four values 0.5, 1, 2 and 4 to be the rate parameter of the exponential distribution, the prior distribution of u. From the results, we can see that the posterior mean and the posterior standard deviation of u all decrease when we increase the value of the rate parameter of the prior of u. The 95% credible interval of u would also be narrowed as the increase of the rate parameter.

```
# The prior sensitivity analysis for the parameter u
summary(prior_analysis(lambda = 0.5))

Iterations = 2050:502000
Thinning interval = 50
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
b        0.89630 0.05287 0.0005287      0.0022208
eta_sq   0.03836 0.01812 0.0001812      0.0001917
sigma_sq 0.03684 0.01802 0.0001802      0.0002792
u        1.02757 0.50120 0.0050120      0.0210080

2. Quantiles for each variable:

            2.5%     25%     50%     75%   97.5%
b        0.77964 0.86383 0.90058 0.93357 0.98599
eta_sq   0.01580 0.02595 0.03435 0.04615 0.08345
sigma_sq 0.01483 0.02455 0.03284 0.04402 0.08323
u        0.17994 0.67496 0.98471 1.33406 2.13077

summary(prior_analysis(lambda = 1))

Iterations = 2050:502000
Thinning interval = 50
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
b        0.90670 0.04793 0.0004793      0.0018718
eta_sq   0.03825 0.01842 0.0001842      0.0001842
sigma_sq 0.03580 0.01716 0.0001716      0.0002164
u        0.92911 0.45431 0.0045431      0.0177441

2. Quantiles for each variable:

            2.5%     25%     50%     75%   97.5%
b        0.80268 0.87763 0.91076 0.94068 0.98780
eta_sq   0.01556 0.02592 0.03419 0.04584 0.08425
sigma_sq 0.01451 0.02395 0.03193 0.04334 0.08077
```

```
u         0.15710 0.60747 0.88922 1.20152 1.92532

summary(prior_analysis(lambda = 2))

Iterations = 2050:502000
Thinning interval = 50
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
b         0.92973 0.04230 0.0004230      0.0014866
eta_sq    0.03783 0.01782 0.0001782      0.0001876
sigma_sq  0.03484 0.01613 0.0001613      0.0001870
u         0.71057 0.40109 0.0040109      0.0140766

2. Quantiles for each variable:

            2.5%     25%     50%     75%   97.5%
b         0.83935 0.90335 0.93391 0.96182 0.99469
eta_sq    0.01567 0.02582 0.03406 0.04530 0.08338
sigma_sq  0.01447 0.02377 0.03141 0.04197 0.07495
u         0.09506 0.40304 0.67298 0.95883 1.56949

summary(prior_analysis(lambda = 4))

Iterations = 2050:502000
Thinning interval = 50
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
b         0.95345 0.03307 0.0003307      0.0009703
eta_sq    0.03727 0.01754 0.0001754      0.0001721
sigma_sq  0.03514 0.01658 0.0001658      0.0001729
u         0.48424 0.31421 0.0031421      0.0094499

2. Quantiles for each variable:

            2.5%     25%     50%     75%   97.5%
b         0.87527 0.93342 0.95885 0.97983 0.99759
eta_sq    0.01572 0.02545 0.03361 0.04446 0.08087
sigma_sq  0.01455 0.02384 0.03169 0.04200 0.07663
u         0.06010 0.23442 0.43378 0.67382 1.22135
```

The priors of the parameters σ^2 and η^2 are all inverse gamma distribution. To conduct prior sensitivity analysis for these two parameters, I test both symmetric parameter pairs (0.01, 0.01), (0.1, 0.1) and (0.5, 0.5) and non-symmetric parameter pairs (0.1, 0.5) and (0.5, 0.1). From the results of σ^2, I find that the rate parameter `b1` (the second parameter) of the inverse gamma distribution has a significant influence on the posterior of σ^2 but the influence of the shape parameter `a1` (the first parameter) is very weak. The larger the rate parameter `b1`, the larger the posterior mean and posterior standard deviation of σ^2 would be and the wider the 95% credible interval of it will become. The results of the parameter η^2 are completely similar. The larger the rate parameter `b2`, the larger the posterior mean and posterior standard deviation of η^2 would be and the wider the 95% credible interval of it will become. The shape parameter `a2` also has very weak influence on the posterior of η^2.

```
# The prior sensitivity analysis for the parameter sigma^2
summary(prior_analysis(a1 = 0.01, b1 = 0.01))
```

```
Iterations = 2050:502000
Thinning interval = 50
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
b        0.92302 0.03252 0.0003252      0.0014681
eta_sq   0.03845 0.01609 0.0001609      0.0001649
sigma_sq 0.01163 0.00855 0.0000855      0.0001244
u        0.77400 0.30848 0.0030848      0.0140652

2. Quantiles for each variable:

             2.5%      25%      50%     75%   97.5%
b        0.854188 0.903317 0.924178 0.94512 0.98357
eta_sq   0.017454 0.027383 0.035096 0.04564 0.07860
sigma_sq 0.002783 0.006075 0.009336 0.01454 0.03393
u        0.203142 0.563634 0.762617 0.96122 1.42194

summary(prior_analysis(a1 = 0.1, b1 = 0.1))

Iterations = 2050:502000
Thinning interval = 50
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
b        0.90446 0.05328 0.0005328      0.0022184
eta_sq   0.03813 0.01769 0.0001769      0.0001933
sigma_sq 0.03640 0.01760 0.0001760      0.0002961
u        0.95035 0.50426 0.0050426      0.0208658

2. Quantiles for each variable:

            2.5%     25%     50%     75%   97.5%
b        0.78545 0.87523 0.90922 0.94154 0.98762
eta_sq   0.01589 0.02600 0.03431 0.04587 0.08197
sigma_sq 0.01483 0.02444 0.03258 0.04373 0.08172
u        0.15986 0.59978 0.90776 1.22639 2.06663

summary(prior_analysis(a1 = 0.5, b1 = 0.5))

Iterations = 2050:502000
Thinning interval = 50
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
b        0.88360 0.06903 0.0006903      0.0023261
eta_sq   0.04482 0.02433 0.0002433      0.0002395
sigma_sq 0.09361 0.03623 0.0003623      0.0004521
u        1.14591 0.65419 0.0065419      0.0225442

2. Quantiles for each variable:

            2.5%     25%     50%    75%  97.5%
b        0.72511 0.84128 0.89241 0.9343 0.9897
eta_sq   0.01670 0.02866 0.03906 0.0538 0.1077
sigma_sq 0.04602 0.06866 0.08596 0.1102 0.1823
u        0.13631 0.66496 1.06665 1.5411 2.6407
```

```
summary(prior_analysis(a1 = 0.1, b1 = 0.5))

Iterations = 2050:502000
Thinning interval = 50
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
b        0.87863 0.07214 0.0007214      0.0024926
eta_sq   0.04514 0.02465 0.0002465      0.0002465
sigma_sq 0.09984 0.04003 0.0004003      0.0005097
u        1.19319 0.68312 0.0068312      0.0235211

2. Quantiles for each variable:

            2.5%     25%     50%     75%  97.5%
b        0.71304 0.83450 0.88675 0.93289 0.9894
eta_sq   0.01641 0.02869 0.03927 0.05452 0.1093
sigma_sq 0.04804 0.07244 0.09168 0.11786 0.1993
u        0.13986 0.67943 1.11552 1.61024 2.7568

summary(prior_analysis(a1 = 0.5, b1 = 0.1))

Iterations = 2050:502000
Thinning interval = 50
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
b        0.90761 0.04812 0.0004812      0.0019881
eta_sq   0.03790 0.01742 0.0001742      0.0001742
sigma_sq 0.03361 0.01568 0.0001568      0.0001772
u        0.92055 0.45624 0.0045624      0.0187968

2. Quantiles for each variable:

            2.5%     25%     50%     75%   97.5%
b        0.80268 0.87614 0.91151 0.94354 0.98835
eta_sq   0.01583 0.02597 0.03393 0.04560 0.08209
sigma_sq 0.01397 0.02284 0.03024 0.04043 0.07315
u        0.15575 0.57997 0.88377 1.21678 1.91379

# The prior sensitivity analysis for the parameter eta^2
summary(prior_analysis(a2 = 0.01, b2 = 0.01))

Iterations = 2050:502000
Thinning interval = 50
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
b        0.90802 0.049032 4.903e-04      1.862e-03
eta_sq   0.01469 0.009998 9.998e-05      9.998e-05
sigma_sq 0.03713 0.015777 1.578e-04      1.981e-04
u        0.91722 0.464547 4.645e-03      1.756e-02

2. Quantiles for each variable:
```

```
             2.5%      25%     50%     75%   97.5%
b        0.796240 0.878708 0.91302 0.94344 0.98773
eta_sq   0.003501 0.007933 0.01221 0.01853 0.04068
sigma_sq 0.015956 0.026147 0.03404 0.04454 0.07541
u        0.158338 0.583046 0.87281 1.19215 1.97551

summary(prior_analysis(a2 = 0.1, b2 = 0.1))

Iterations = 2050:502000
Thinning interval = 50
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
b        0.90582 0.04813 0.0004813      0.0018349
eta_sq   0.03793 0.01788 0.0001788      0.0001788
sigma_sq 0.03584 0.01681 0.0001681      0.0002122
u        0.93755 0.45582 0.0045582      0.0173397

2. Quantiles for each variable:

            2.5%     25%     50%     75%   97.5%
b        0.80069 0.87606 0.90830 0.94050 0.98760
eta_sq   0.01544 0.02557 0.03415 0.04575 0.08255
sigma_sq 0.01454 0.02403 0.03233 0.04321 0.07871
u        0.15645 0.60785 0.91334 1.22011 1.93519

summary(prior_analysis(a2 = 0.5, b2 = 0.5))

Iterations = 2050:502000
Thinning interval = 50
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
b        0.89338 0.05895 0.0005895      0.0025823
eta_sq   0.09670 0.03820 0.0003820      0.0004395
sigma_sq 0.04202 0.02212 0.0002212      0.0003150
u        1.05492 0.55880 0.0055880      0.0244339

2. Quantiles for each variable:

            2.5%     25%     50%     75%   97.5%
b        0.75668 0.86001 0.89942 0.93558 0.98688
eta_sq   0.04613 0.07043 0.08889 0.11374 0.19241
sigma_sq 0.01593 0.02681 0.03679 0.05116 0.09934
u        0.16632 0.65669 0.99465 1.37291 2.35449

summary(prior_analysis(a2 = 0.1, b2 = 0.5))

Iterations = 2050:502000
Thinning interval = 50
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
b        0.89643 0.05653 0.0005653      0.0022492
eta_sq   0.10104 0.04024 0.0004024      0.0004024
sigma_sq 0.04256 0.02356 0.0002356      0.0002946
u        1.02601 0.53609 0.0053609      0.0213058
```

```
2. Quantiles for each variable:

             2.5%     25%     50%     75%  97.5%
b         0.77111 0.86116 0.90228 0.93811 0.9883
eta_sq    0.04822 0.07330 0.09261 0.11966 0.2017
sigma_sq  0.01543 0.02693 0.03711 0.05156 0.1019
u         0.15601 0.62897 0.97169 1.35796 2.2168

summary(prior_analysis(a2 = 0.5, b2 = 0.1))

Iterations = 2050:502000
Thinning interval = 50
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
b         0.90651 0.04944 0.0004944      0.0019869
eta_sq    0.03525 0.01613 0.0001613      0.0001707
sigma_sq  0.03580 0.01716 0.0001716      0.0002221
u         0.93087 0.46807 0.0046807      0.0188773

2. Quantiles for each variable:

             2.5%     25%     50%     75%   97.5%
b         0.79703 0.87664 0.91072 0.94251 0.98850
eta_sq    0.01474 0.02416 0.03182 0.04235 0.07599
sigma_sq  0.01470 0.02415 0.03188 0.04285 0.07997
u         0.15523 0.59159 0.89027 1.21093 1.96634
```

Question (f)

For this question, we are required to update the model to compute the posterior distribution of the log population sizes x_t every year from 1951 to 2050. We do not need to change the BUGS code, and instead we need to update the data set that we use to run the JAGS model and create a new JAGS model for this question. At first, let me update our response variable `Count`. For years beyond 1997, I denote `NA` for the values of the observed log population sizes y_t to create a new response variable for this question, which is denoted as `Count.new`.

```
# Initialize some variables
n.new = 100
# Construct the new response variable
Year.new = 1951:2050; Count.new = rep(NA, n.new)
for (k in Year.new){
  if (k %in% graywhales[, 'Year']){
    ind = which(graywhales[, 'Year'] == k)
    Count.new[(k - Year.new[1] + 1)] = graywhales[, 'Count'][ind]
  }
}
```

Then, I am going to run the new JAGS model and sample from the posterior distribution of the log population size x_t in each year. I will use the new response variable `Count.new` and the new length of period `n.new=100` to run the JAGS model and keep all the parameters for the prior distributions of the model parameters the same as in question (b). After acquiring the new JAGS model, I will take samples from the posterior distributions of the log population sizes x_t in the years 1951-2050 using the function `coda.samples`. Because I only need the samples from the posterior of x_t $(1 \leq t \leq 100)$, I only need to include the variable name `x` in the argument `variable.names`.

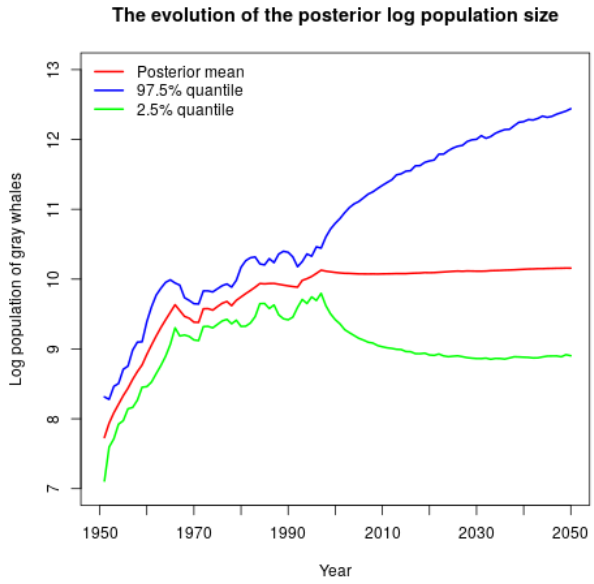
```
# Update the model in (b)
data.new = list(y = log(Count.new), n = n.new, mu0 = log(2500), prec0 = 1, low = 0, up = 1,
                lambda = 1, a1 = 0.1, b1 = 0.1, a2 = 0.1, b2 = 0.1)
model.new = jags.model(textConnection(model_string), data = data.new, n.adapt = 2000)
sample.new = coda.samples(model.new, variable.names = c('x'), n.iter = 500000, thin = 50)

Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 24
   Unobserved stochastic nodes: 180
   Total graph size: 415

Initializing model
```

In the output of the code, the 24 observed stochastic nodes still represent the 24 observed log population sizes y_t that are not missing. The 180 unobserved stochastic nodes consist of the 76 missing y_t, and the 100 log population sizes x_t (include the states of the year 1951 and from the year 1998 to 2050), and the four parameters b, u, σ^2, η^2.

Now, I will plot the evolution of the posterior mean of the log population sizes x_t, along with two other curves corresponding to the [2.5%, 97.5%] credible interval of x_t, from 1951 to 2050. I will extract the posterior mean, 2.5% quantile and 97.5% quantile of the posterior sample of x_t (`sample.new`) from the summary statistics of `sample.new`, using the commands `summary(sample.new)[[1]][, 'Mean']`, `summary(sample.new)[[2]][, '2.5%']` and `summary(sample.new)[[2]][, '97.5%']` respectively. Then, the evolution of the posterior mean together with the [2.5%, 97.5%] credible interval of x_t can be plotted. We can see that since about the year 1998, the 95% credible interval of x_t begins to become much wider than the previous years and the credible interval continues to widen after 1998. It is because we have no available data beyond the year 1997 and so the uncertainty about x_t will become greater and greater after 1997.

```
# Plot the evolution of the log population sizes
plot(Year.new, summary(sample.new)[[1]][, 'Mean'], type = 'l', xlab = 'Year',
     ylab = 'Log population of gray whales', xlim = c(1950, 2050), ylim = c(7, 13), xaxt = 'n',
     main = 'The evolution of the posterior log population size', col = 'red', lwd = 2)
lines(Year.new, summary(sample.new)[[2]][, '2.5%'], col = 'green', lwd = 2)
lines(Year.new, summary(sample.new)[[2]][, '97.5%'], col = 'blue', lwd = 2)
axis(side = 1, at = seq(1950, 2050, 10))
legend('topleft', legend = c('Posterior mean', '97.5% quantile', '2.5% quantile'),
       col = c('red', 'blue', 'green'), lwd = 2, bty = 'n')
```

Note: The graph is in the next page.



Finally, for each sample from the posterior of x_t, I compute the minimum value of x_t from the year 1951 to 2050 and judge whether it is smaller than log(100). Then, I count the number of the samples in which the minimum value of x_t is less than log(100). Finally, I divide this number by the total number of samples, and then the posterior probability that the population of graywhales becomes smaller than 100 at any year from 1951 to 2050 is obtained. From the result, we can see that this probability is usually equal to 0 (sometimes 1/10000). The population of graywhales in the year 1952 is already 2894 and the prior mean of the initial x_t in 1951 is also already log(2500), and the population of graywhales shows a trend of continuous growth. Thus, it is almost impossible that the population of graywhales becomes smaller than 100 from the year 1951, so the result that we obtain is sensible.

```
# The posterior probability that the population of gray whales becomes smaller than 100
# at any year from 1951 until the end of 2050
nsamp.new = nrow(sample.new[[1]])
xt.min = numeric(nsamp.new)
for(l in 1:nsamp.new){
    xt.min[l] = min(sample.new[[1]][l, ])
}
prob = sum(xt.min <= log(100))/nsamp.new
prob
```

```
0
```

Question (g)

For this question, I will use the posterior sample of the parameters η^2 and x_t $(1 \leq t \leq 47)$ generated in question (b). The size of this sample is 10000, so the number of replicate observations that we will create for each year would also be 10000. To create replicate observations from the posterior predictive distribution, we need to first extract the posterior sample of the log population size x_t in each year, and then add some Gaussian noise to this sample since our likelihood model is $y_t \sim N(x_t, \eta^2)$. Because we need to evaluate the `min`, `max`, and `median` functions only on the observations that are included in the data set, I introduce a variable named `index` in the code to indicate the location of the observations that are not missing in the original data set. Since η^2 is also a parameter from the model that is different for each sample, we also need to extract η^2 from the posterior sample `sample` obtained in question (b) and then add the corresponding noise.

```
sample.mat = as.matrix(sample[[1]][, 1:51])
nsamp = nrow(sample.mat)
nobs = sum(!is.na(Count))
index = which(!is.na(Count))
count.rep = matrix(0, nrow = nobs, ncol = nsamp)
row.names(count.rep) = index
for(l in 1:nsamp){
  for (k in index){
    ind = which(index == k)
    count.rep[ind, l] = sample.mat[l, k+4] + rnorm(1, mean = 0, sd = sqrt(sample.mat[l, 'eta_sq']))
  }
}
```

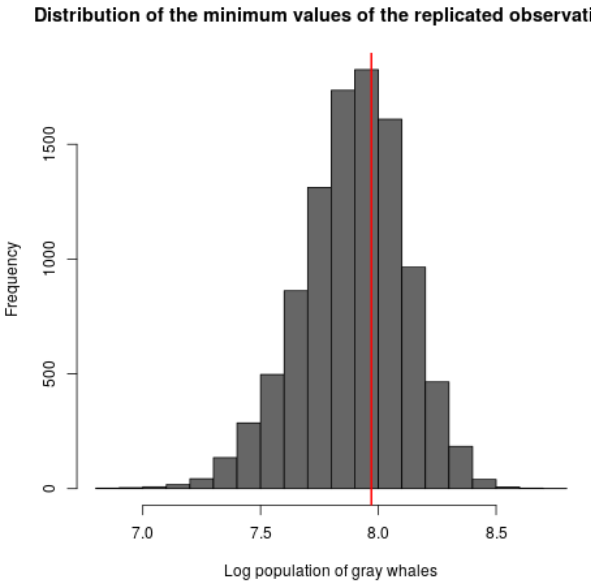
After obtaining the replicate observations from the posterior predictive distribution, I respectively compute the minimum, maximum, and median for the replicated observations of the years in which the observations of the original data set are not missing in each sample.

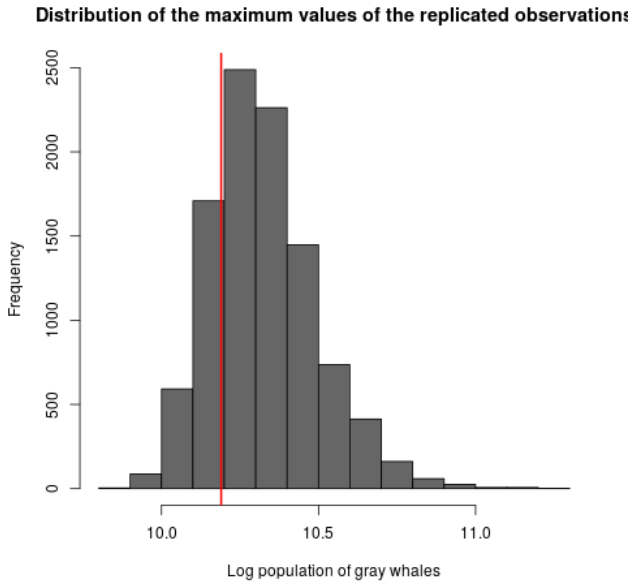
```
count.rep.min = sapply(1:nsamp, function(l){min(count.rep[, l])})
count.rep.max = sapply(1:nsamp, function(l){max(count.rep[, l])})
count.rep.median = sapply(1:nsamp, function(l){median(count.rep[, l])})
```

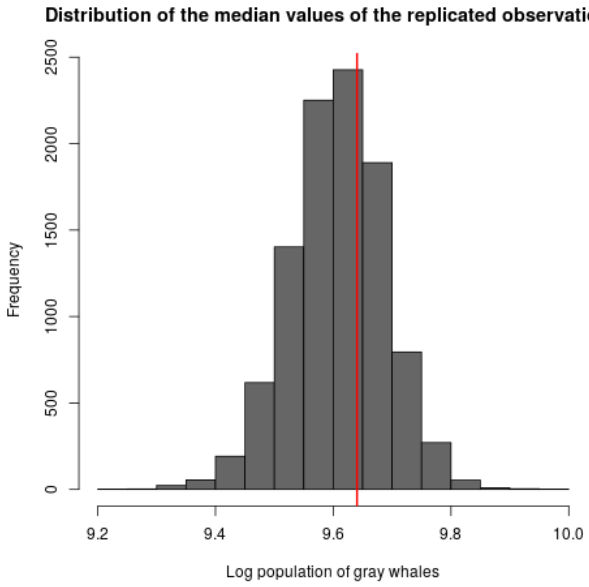
Finally, the histograms for these quantities together with a line that shows the value of the function considered on the actual data set can be plotted. These histograms quantify the variability of the three test statistics, the minimum, maximum and median. The red lines which correspond to the values of these statistics based on the actual data set can be viewed as special members among the results based on replicate observations. From the results, we can see, except that the red line in the histogram of the maximum values of the replicate observations slightly deviates from the center of the graph, the red lines in the other two histograms are all very close to the centers of the graphs. Therefore, the samples from the posterior predictive distribution of the log population of graywhales look sensible. This indicates the suitability of the model and good mixing and convergence of the chains for the parameters η^2 and x_t $(1 \leq t \leq 47)$.

```
hist(count.rep.min, col = "gray40", xlab = 'Log population of gray whales',
     main = 'Distribution of the minimum values of the replicated observations')
abline(v = min(log(graywhales[, 'Count']), na.rm = TRUE), col = "red", lwd = 2)

hist(count.rep.max, col = "gray40", xlab = 'Log population of gray whales',
     main = 'Distribution of the maximum values of the replicated observations')
abline(v = max(log(graywhales[, 'Count']), na.rm = TRUE), col = "red", lwd = 2)

hist(count.rep.median, col = "gray40", xlab = 'Log population of gray whales',
     main = 'Distribution of the median values of the replicated observations')
abline(v = median(log(graywhales[, 'Count']), na.rm = TRUE), col = "red", lwd = 2)
```

Note: The histograms are in the next page.


Distribution of the minimum values of the replicated observations
Frequency
0
500
1000
1500
7.0
7.5
8.0
8.5
Log population of gray whales


Distribution of the maximum values of the replicated observations
Frequency
0
500
1000
1500
2000
2500
10.0
10.5
11.0
Log population of gray whales


Distribution of the median values of the replicated observations
Frequency
0
500
1000
1500
2000
2500
9.2
9.4
9.6
9.8
10.0
Log population of gray whales

Problem 2 - Property Price Data

Question (a)

To begin with, for the data set `house`, I center and scale all of the covariates and remain the response variable unchanged. The acquired new data set is saved as the variable `house2`.

```
# Centering and scaling all of the covariates
house2 = data.frame(
  'X1.transaction.date' = (house[[2]] - mean(house[[2]]))/sd(house[[2]]),
  'X2.house.age' = (house[[3]] - mean(house[[3]]))/sd(house[[3]]),
  'X3.distance.to.the.nearest.MRT.station' = (house[[4]] - mean(house[[4]]))/sd(house[[4]]),
  'X4.number.of.convenience.stores' = (house[[5]] - mean(house[[5]]))/sd(house[[5]]),
  'X5.latitude' = (house[[6]] - mean(house[[6]]))/sd(house[[6]]),
  'X6.longitude' = (house[[7]] - mean(house[[7]]))/sd(house[[7]]),
  'Y.house.price.of.unit.area' = house[[8]]
)
```

Then, I fit the standard linear regression model as required using the function `lm`.

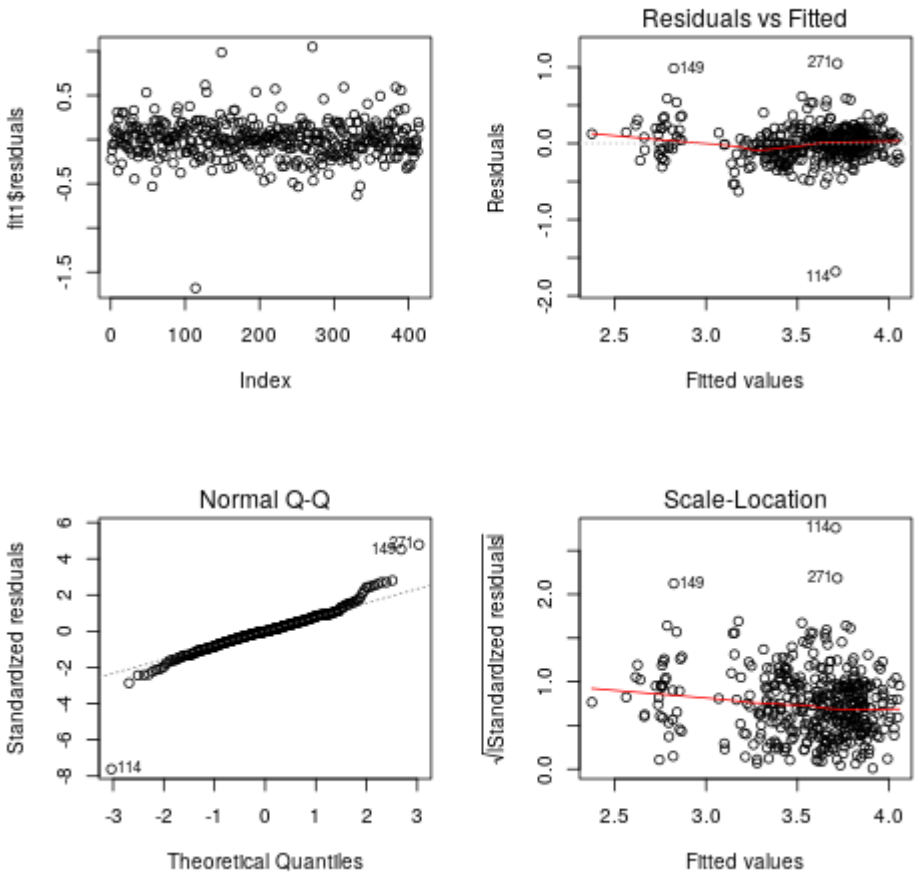
```
# Fit the standard linear regression model (lm)
fit1 = lm(log(Y.house.price.of.unit.area) ~ X1.transaction.date + X2.house.age +
          X3.distance.to.the.nearest.MRT.station + X4.number.of.convenience.stores +
          X5.latitude + X6.longitude, data = house2)
```

The summary statistics of the fit of this model are shown below. We can see that the estimate of the intercept is 3.5667, and the estimates of the regression coefficients of all the covariates are 0.0382, -0.0794, -0.1836, 0.0817, 0.0983, 0.0057 respectively, and the residual standard error is 0.2216.

```
summary(fit1)

Call:
lm(formula = log(Y.house.price.of.unit.area) ~ X1.transaction.date +
    X2.house.age + X3.distance.to.the.nearest.MRT.station + X4.number.of.convenience.stores +
    X5.latitude + X6.longitude, data = house2)

Residuals:
     Min       1Q   Median       3Q      Max
-1.68095 -0.11498 -0.00267  0.11540  1.04849

Coefficients:
                                        Estimate Std. Error t value Pr(>|t|)
(Intercept)                             3.566695   0.010892 327.447  < 2e-16
X1.transaction.date                     0.038198   0.010985   3.477 0.000561
X2.house.age                           -0.079373   0.010983  -7.227 2.46e-12
X3.distance.to.the.nearest.MRT.station -0.183625   0.022675  -8.098 6.54e-15
X4.number.of.convenience.stores         0.081731   0.013868   5.894 7.94e-09
X5.latitude                             0.098348   0.013839   7.107 5.36e-12
X6.longitude                            0.005659   0.018656   0.303 0.761766

(Intercept)                            ***
X1.transaction.date                    ***
X2.house.age                           ***
X3.distance.to.the.nearest.MRT.station ***
X4.number.of.convenience.stores        ***
X5.latitude                            ***
X6.longitude
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.2216 on 407 degrees of freedom
Multiple R-squared:  0.6857, Adjusted R-squared:  0.6811
F-statistic:   148 on 6 and 407 DF,  p-value: < 2.2e-16
```

Next, let me create the residual plots and perform residual analyses to evaluate the quality of the fit.

```
plot(fit1$residuals)
plot(fit1)
```



The first, second and fourth graphs are the plots of the residuals versus index and versus the fitted values and the scaled residuals versus the fitted values respectively. In both the second and fourth graphs, the points are obviously gathered as two clusters, and the variance of the cluster on the right is apparently larger than the variance of the cluster on the left. Moreover, there are also some obvious outliers in the first, second and fourth graphs, e.g. the points labelled as 114, 149 and 271. Therefore, the linearity and constant variance assumptions of the model may not be satisfied. The third graph is the Normal QQ-plot. We can see, at the both ends of the graph, the sample quantiles of the residuals do not match the theoretical quantiles well, and the points labelled as 114, 149 and 271 are also outliers in this graph. Thus, the normality assumption of this model cannot be satisfied well. Hence, to sum up, the quality of the fit of this linear model is not good.

Question (b)

For this question, I will fit the same linear regression model as question (a) using INLA. Firstly, let me set the priors for both the precision and all of the regression coefficients (include the intercept), and save the information of these two kinds of priors in the variables `prec.prior` and `prior.beta` respectively. As

internally the logarithm of the precision $\theta = \log(\tau)$ is stored, it is equivalent to specify a Log-Gamma prior on θ with the same parameters.

```
# The variables for specifying prior information
prec.prior = list(prec = list(prior = 'loggamma', param = c(0.1, 0.1)))
prior.beta = list(mean.intercept = 0, prec.intercept = 0.000001, mean = 0, prec = 0.000001)
```

Then, let me fit the model using INLA. Because we are required to fit the same model as in (a), I also use the same data set `house2` as in (a) where the covariates have been centered and scaled. Furthermore, in the following questions (d), (e) and (f), because we are asked to compare several different models, we should keep the factors not concerned consistent as far as possible, so I also use the data set with the covariates being centered and scaled. Besides, centering and scaling the covariates is helpful to setting the priors and speeds up convergence.

```
# Fit the linear regression model using INLA
fit.inla = inla(log(Y.house.price.of.unit.area) ~ X1.transaction.date + X2.house.age +
                X3.distance.to.the.nearest.MRT.station + X4.number.of.convenience.stores +
                X5.latitude + X6.longitude, data = house2, family = 'gaussian',
                control.family = list(hyper = prec.prior), control.fixed = prior.beta,
                control.compute = list(config = TRUE, cpo = TRUE, dic = TRUE),
                control.predictor = list(compute = TRUE))
```

The summary statistics of the fit are printed out as follows.

```
summary(fit.inla)

Call:
   c("inla(formula = log(Y.house.price.of.unit.area) ~ X1.transaction.date
   + ", " X2.house.age + X3.distance.to.the.nearest.MRT.station +
   X4.number.of.convenience.stores + ", " X5.latitude + X6.longitude,
   family = \"gaussian\", data = house2, ", " control.compute =
   list(config = TRUE, cpo = TRUE, dic = TRUE), ", " control.predictor =
   list(compute = TRUE), control.family = list(hyper = prec.prior), ", "
   control.fixed = prior.beta)")
Time used:
    Pre = 0.494, Running = 0.66, Post = 0.0996, Total = 1.25
Fixed effects:
                                          mean    sd 0.025quant 0.5quant
(Intercept)                              3.567 0.011      3.545    3.567
X1.transaction.date                      0.038 0.011      0.016    0.038
X2.house.age                            -0.079 0.011     -0.101   -0.079
X3.distance.to.the.nearest.MRT.station -0.184 0.023     -0.228   -0.184
X4.number.of.convenience.stores          0.082 0.014      0.054    0.082
X5.latitude                              0.098 0.014      0.071    0.098
X6.longitude                             0.006 0.019     -0.031    0.006
                                       0.975quant   mode kld
(Intercept)                                 3.588  3.567   0
X1.transaction.date                         0.060  0.038   0
X2.house.age                               -0.058 -0.079   0
X3.distance.to.the.nearest.MRT.station     -0.139 -0.184   0
X4.number.of.convenience.stores             0.109  0.082   0
X5.latitude                                 0.126  0.098   0
X6.longitude                                0.043  0.006   0

Model hyperparameters:
                                           mean   sd 0.025quant 0.5quant
Precision for the Gaussian observations 20.17 1.41      17.50    20.14
                                        0.975quant  mode
Precision for the Gaussian observations      23.03 20.07

Expected number of effective parameters(stdev): 7.05(0.004)
Number of equivalent replicates : 58.72
```

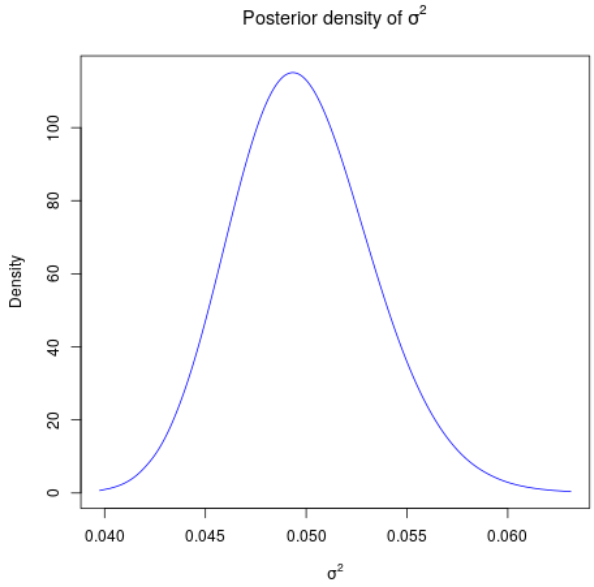
```
Deviance Information Criterion (DIC) ...............: -63.45
Deviance Information Criterion (DIC, saturated) ....: 419.37
Effective number of parameters .....................: 8.14

Marginal log-Likelihood:  -45.08
CPO and PIT are computed

Posterior marginals for the linear predictor and
 the fitted values are computed
```

We can extract the posterior marginals of the hyperparameters of the model via the element `marginals.hyperpar` in the `inla` object, i.e. using the command `fit.inla$marginals.hyperpar`. In order to obtain the posterior marginals of the variance parameter σ^2, we need first extract the posterior marginals of the precision parameter τ and then use the function `inla.tmarginal` to transform it. Then, the posterior mean of σ^2 can be calculated and we can also use the `plot` function to plot the posterior density for σ^2.

```
marginal.tau = fit.inla$marginals.hyperpar[[1]]
marginal.sigma2 = inla.tmarginal(function(tau){1/tau}, marginal.tau)
mean(marginal.sigma2)
```

```
0.0498341915851792
```

```
plot(marginal.sigma2, type = 'l', xlab = expression(sigma^2), ylab = 'Density',
     col = 'blue', main = expression(paste('Posterior density of ', sigma^2)))
```



To compute the model assessment criteria in INLA, we should set `control.compute = list(cpo = TRUE, dic = TRUE)` at first. The negative sum log CPO (NSLCPO) values for this model can be computed using the command `sum(log(fit.inla$cpo$cpo))`, and the value of the Deviance information criterion (DIC) for this model can be accessed using the command `fit.robust$dic$dic`. The posterior mean of the fitted values can be obtained using the command `fit.inla$summary.fitted.values$mean` from an INLA object, and then the standard deviation of the mean residuals can be calculated.

```
cat('NSLCPO: ', -sum(log(fit.inla$cpo$cpo)), '\n')
cat('DIC: ', fit.inla$dic$dic)
```

```
NSLCPO:  -28.77816
DIC:  -63.4466
```

```
sd(log(house2$Y.house.price.of.unit.area) - fit.inla$summary.fitted.values$mean)
```

```
0.219984903115561
```

Looking at the summary statistics of the fit using INLA with those of the fit of the standard linear model, I find that the posterior means and standard deviations of the coefficients and the intercept of the INLA fit are very similar with the estimates and standard deviations of the coefficients and the intercept of the standard linear model fit. However, the interpretation of these results is completely different. The model fit using INLA belongs to Bayesian methods, while the fit of the standard linear model is a frequentist approach. The posterior distribution of σ^2 looks like a normal distribution with mean 0.0498. The negative sum log CPO (NSLCPO) values and the Deviance information criterion (DIC) for this model are -28.7782 and -63.4466 respectively. The standard deviation of the mean residuals is 0.2200, and the square of it is very close to the posterior mean of σ^2.

Question (c)

To begin with, I take 10000 posterior samples from our linear regression model using the function `inla.posterior.sample` and extract samples from the posterior distribution of σ^2 using the function `inla.posterior.sample.eval`. Then, I compute the fitted values of this model. Because for this linear regression model, the linear predictor η_i is equal to the mean of the observation μ_i, we can extract the fitted values directly by extracting the linear predictors using the command `inla.posterior.sample.eval(function(...){Predictor}, samples)`, but we need to set `control.predictor = list(compute = TRUE)` in the beginning.

```
# Compute the fitted values and extract posterior samples of sigma^2
nsamp = 10000
ndata = nrow(house2)
samples = inla.posterior.sample(n = nsamp, result = fit.inla)
sigma_sq = 1/inla.posterior.sample.eval(function(...){theta}, samples)
fittedvalues = inla.posterior.sample.eval(function(...){Predictor}, samples)
```

Now, let me compute the studentized residuals for the Bayesian regression model. The studentized residuals are defined by

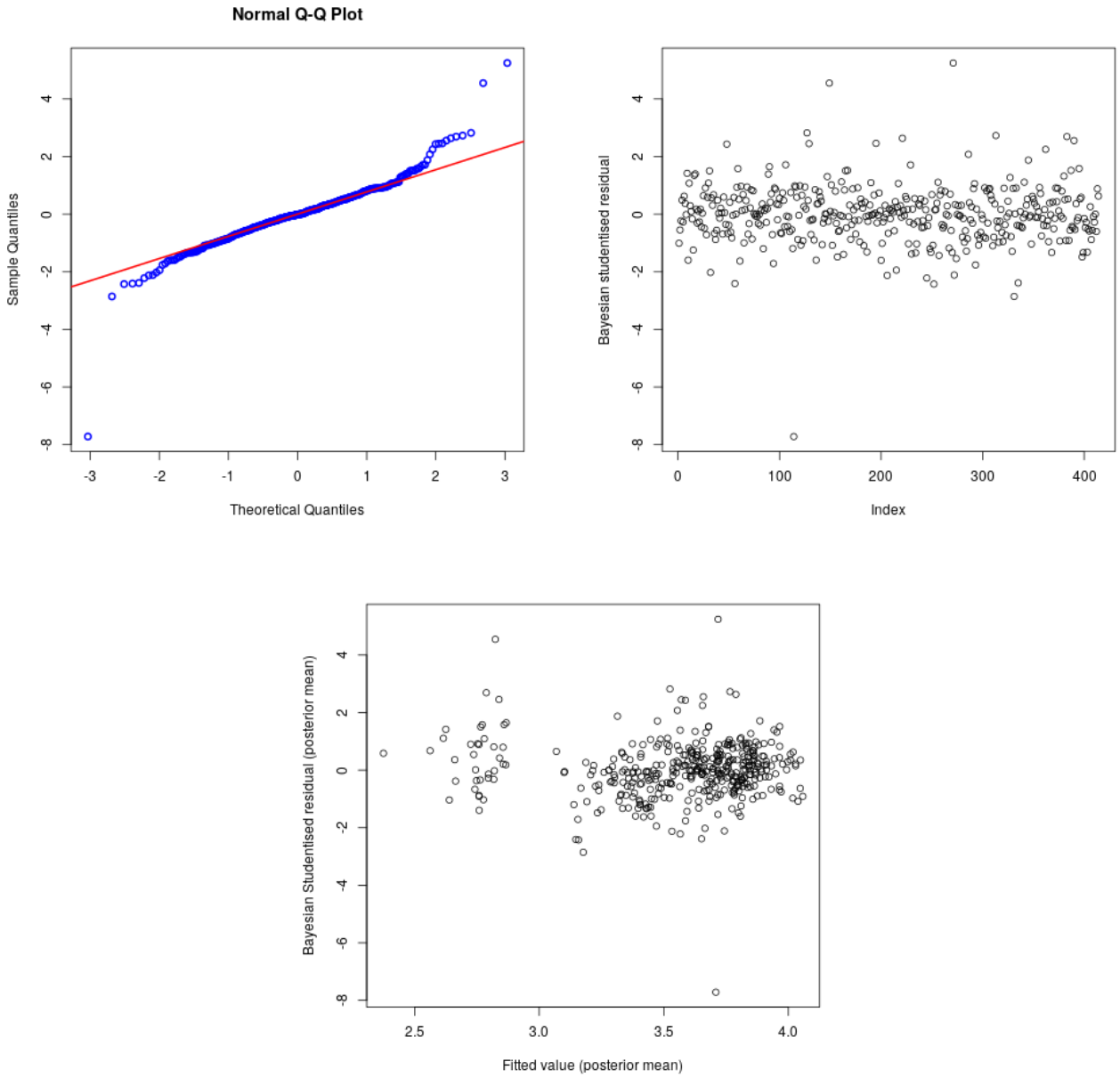
$$\widehat{\varepsilon_i} = \frac{y_i - \widehat{y_i}}{\widehat{\sigma} * \sqrt{(1 - h_{ii})}}, \qquad i = 1, \dots, n,$$

where $\widehat{y_i}$ are the fitted values of this model, y_i are the true values of the response variable $(i = 1, \dots, n)$, $\widehat{\sigma}$ is from the posterior distribution of σ^2 and h_{ii} is the i-th diagonal entry of the hat matrix $H = \boldsymbol{x}(\boldsymbol{x}^{\boldsymbol{T}}\boldsymbol{x})^{-1}\boldsymbol{x}^{\boldsymbol{T}}$. By this formula, the studentized residuals can be calculated. In the Bayesian context, the estimates $\widehat{\beta_j}$ and $\widehat{\sigma}$ are replaced by distributions and we adopt the posterior mean of the residual to create single-valued (studentized) residuals.

```
# Compute the studentised residuals
student_resi_mean = rep(0, ndata)
X = cbind(rep(1, ndata), house2[[2]], house2[[3]], house2[[4]], house2[[5]],
          house2[[6]], house2[[7]])
H = X%*%solve((t(X)%*%X))%*%t(X)
for (i in 1:ndata){
  student_resi = rep(0, nsamp)
  for(l in 1:nsamp){
    student_resi[l] = (log(house2$Y.house.price.of.unit.area[i])- fittedvalues[i, l])/
                      (sqrt(sigma_sq[l]*(1-diag(H)[i])))
  }
  student_resi_mean[i] = mean(student_resi)
}
```

After these preparation work, now we can perform the Q-Q plot on the studentized residuals, and plot the studentized residuals versus their index and versus the posterior mean of the fitted values respectively.

```
# Create the plots
qqnorm(student_resi_mean, col = 'blue', lwd = 2)
qqline(student_resi_mean, col = 'red', lwd = 2)
plot(seq_along(student_resi_mean), student_resi_mean, xlab = 'Index',
     ylab = 'Bayesian studentised residual')
fittedvalues_mean = rep(0, ndata)
for (i in 1:ndata){
  fittedvalues_mean[i] = mean(fittedvalues[i, ])
}
plot(fittedvalues_mean, student_resi_mean, xlab = 'Fitted value (posterior mean)',
     ylab = 'Bayesian Studentised residual (posterior mean)')
```



These residual plots are very similar with the residual plots of the standard linear regression model in question (a). We are fitting the same model but from different point of view. We use the frequentist approach in (a) and we adopt the Bayesian method in (b). We can see there are also some obvious outliers in all the three graphs (apparently the same nodes as in (a)). The points in the third graph are also gathered in two clusters and it seems that the variance of the cluster on the right is larger than that of the cluster

on the left. Moreover, at the both ends of the QQ-plot, the sample quantiles also do not match the theorical quantiles well. Therefore, to sum up, the normality, linearity and constant variance assumptions of the model may not be satisfied. Hence, from the analyses conducted so far, this linear model may not be so suitable for predicting the response variable, the logarithm of house price per unit area.

Question (d)

In this question, we are going to fit a robust linear regression model in the same settings as in question (b). I will replace the normal likelihood of the observations with the likelihood of the t-distribution, a fat-tailed distribution. This can be done by specifying `family = 't'`. For the degrees of freedom parameter ν of the t-distribution, I place the same prior as the one used for the precision on it, namely, the Gamma prior with parameters (0.1, 0.1). The prior information of the hyperparameters of this model is stored by the variable `prior.t`, and the priors for the regression coefficients (include the intercept) are the same as in (b) which are still specified by the variable `prior.beta`.

```
# Fit the robust regression model
prior.t = list(prec = list(prior = "loggamma", param = c(0.1, 0.1)),
               dof = list(prior = "loggamma", param = c(0.1, 0.1)))
prior.beta = list(mean.intercept = 0, prec.intercept = 0.000001, mean = 0, prec = 0.000001)
fit.robust = inla(log(Y.house.price.of.unit.area) ~ X1.transaction.date + X2.house.age +
                  X3.distance.to.the.nearest.MRT.station + X4.number.of.convenience.stores +
                  X5.latitude + X6.longitude, data = house2, family = 't',
                  control.family = list(hyper = prior.t), control.fixed = prior.beta,
                  control.compute = list(cpo = TRUE, dic = TRUE))
```

The summary statistics of the robust linear model are printed out below.

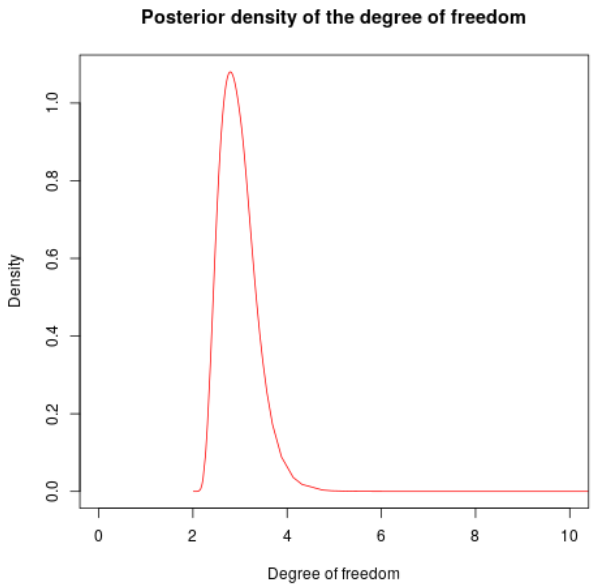
```
summary(fit.robust)

Call:
   c("inla(formula = log(Y.house.price.of.unit.area) ~ X1.transaction.date
   + ", " X2.house.age + X3.distance.to.the.nearest.MRT.station +
   X4.number.of.convenience.stores + ", " X5.latitude + X6.longitude,
   family = \"t\", data = house2, ", " control.compute = list(cpo = TRUE,
   dic = TRUE), control.family = list(hyper = prior.t), ", " control.fixed
   = prior.beta)")
Time used:
    Pre = 0.454, Running = 7.15, Post = 0.114, Total = 7.72
Fixed effects:
                                          mean    sd 0.025quant 0.5quant
(Intercept)                              3.564 0.009      3.547    3.564
X1.transaction.date                      0.023 0.008      0.007    0.023
X2.house.age                            -0.089 0.009     -0.106   -0.089
X3.distance.to.the.nearest.MRT.station -0.226 0.020     -0.265   -0.226
X4.number.of.convenience.stores          0.073 0.012      0.050    0.073
X5.latitude                              0.089 0.012      0.067    0.089
X6.longitude                            -0.011 0.014     -0.038   -0.011
                                        0.975quant   mode kld
(Intercept)                                  3.580  3.564   0
X1.transaction.date                          0.040  0.023   0
X2.house.age                                -0.071 -0.089   0
X3.distance.to.the.nearest.MRT.station      -0.186 -0.226   0
X4.number.of.convenience.stores              0.097  0.073   0
X5.latitude                                  0.112  0.089   0
X6.longitude                                 0.017 -0.011   0

Model hyperparameters:
                                           mean    sd 0.025quant 0.5quant
precision for the student-t observations 16.92 3.647      10.37    16.77
degrees of freedom for student-t          2.97 0.391       2.37     2.91
                                         0.975quant  mode
```

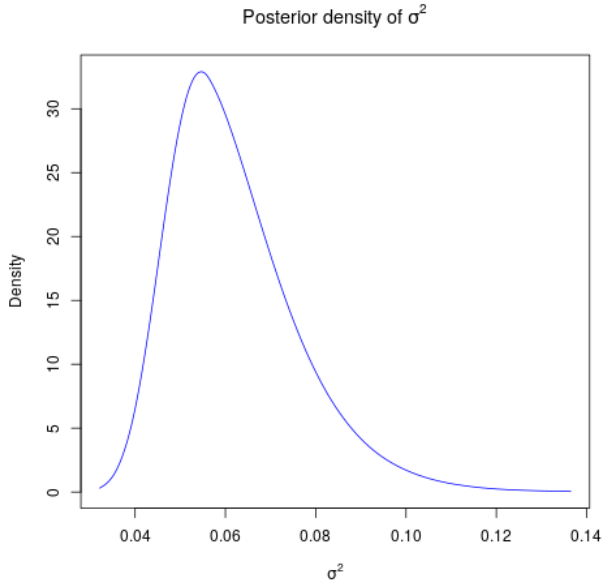
```
precision for the student-t observations      24.56 16.57
degrees of freedom for student-t               3.88  2.79

Expected number of effective parameters(stdev): 7.08(0.006)
Number of equivalent replicates : 58.43

Deviance Information Criterion (DIC) ...............: -156.51
Deviance Information Criterion (DIC, saturated) ....: 461.57
Effective number of parameters .....................: 9.41

Marginal log-Likelihood:  -1.03
CPO and PIT are computed

Posterior marginals for the linear predictor and
 the fitted values are computed
```

Compared with the summary statistics of the model in (b), there are evident differences but some terms are also very similar, e.g. the means, quantiles and modes of the posterior distributions of the intercepts for these two models are very close.

Now, we can plot the posterior distribution of the degrees of freedom parameter ν and the variance parameter σ^2, compute the negative sum log CPO values (NSLCPO) and the Deviance information criterion (DIC) for this model and compute the standard deviation of the mean residuals. The degrees of freedom parameter ν is a hyperparameter of the model and it can be extracted using the command `fit.robust$marginals.hyperpar[[2]]`.

```
summary(fit.robust)
deg_of_free = fit.robust$marginals.hyperpar[[2]]
plot(deg_of_free, type = 'l', xlab = 'Degree of freedom', ylab = 'Density',
     xlim = c(0, 10), col = 'red', main = 'Posterior density of the degree of freedom')
robu.margin.tau = fit.robust$marginals.hyperpar[[1]]
robu.margin.sigma_sq = inla.tmarginal(function(tau){1/tau}, robu.margin.tau)
plot(robu.margin.sigma_sq, type = 'l', xlab = expression(sigma^2), ylab = 'Density',
     col = 'blue', main = expression(paste('Posterior density of ', sigma^2)))
```






```
cat('NSLCPO: ', -sum(log(fit.robust$cpo$cpo)), '\n')
cat('DIC: ', fit.robust$dic$dic)
```

```
NSLCPO:  -77.70485
DIC:  -156.506
```

```
sd(log(house2$Y.house.price.of.unit.area) - fit.robust$summary.fitted.values$mean)
```

```
0.222038438240945
```

The NSLCPO, the DIC and the standard deviation of the mean residuals for this model are -77.7049, -156.506 and 0.2220 respectively. Compared with the model in question (b), the NSLCPO and DIC of the robust model are all significantly smaller than the normal linear model, so the robust model fits the relation between the response variable and the covariates much better than the model in (b). However, the standard deviation of the mean residuals for this model is slightly larger than that of the model in (b). Because the residuals of this model are assumed to have a t-distribution and this is a fat-tailed distribution, the variance of the mean residuals would be a little larger.

In the standard linear model, the normality assumption of the error terms renders the model quite sensitive to outliers. Also, from the plots of the studentized residuals in question (c), we can see there are some obvious outliers. Thus, the normal linear model in (b) may not be a good model to explain the relationship between the response and the covariates. The t-distribution is more robust to cope with outliers, and from the analyses so far, the robust model in this question is a much better model.

Question (e)

We have treated the number of stores variable as a numerical variable so far, and we are going to fit a linear regression model with INLA that uses this as a categorical variable in this question. Firstly, I use the function `as.factor` to convert this variable into a categorical variable. Moreover, I keep all the other covariates the same as in (b) and (d), which are still centered and scaled.

```
house3 = house2
house3$X4.number.of.convenience.stores = as.factor(house$X4.number.of.convenience.stores)
```

I will implement both standard and robust linear regression (with Student-t likelihood) for this model. Firstly, let me fit the standard normal linear regression for this model.

```
# Implement the standard linear regression using INLA
fit.cate.stan = inla(log(Y.house.price.of.unit.area) ~ X1.transaction.date + X2.house.age +
                     X3.distance.to.the.nearest.MRT.station + X4.number.of.convenience.stores +
                     X5.latitude + X6.longitude, data = house3, family = 'gaussian',
                     control.family = list(hyper = prec.prior), control.fixed = prior.beta,
                     control.compute = list(cpo = TRUE, dic = TRUE))
```

The summary statistics, NSLCPO and DIC values and the standard deviation of the mean residuals for this model are printed out as follows.

```
summary(fit.cate.stan)
```

```
Call:
   c("inla(formula = log(Y.house.price.of.unit.area) ~ X1.transaction.date
   + ", " X2.house.age + X3.distance.to.the.nearest.MRT.station +
   X4.number.of.convenience.stores + ", " X5.latitude + X6.longitude,
   family = \"gaussian\", data = house3, ", " control.compute = list(cpo =
   TRUE, dic = TRUE), control.family = list(hyper = prec.prior), ", "
   control.fixed = prior.beta)")
Time used:
    Pre = 0.54, Running = 0.701, Post = 0.0849, Total = 1.33
Fixed effects:
                                       mean    sd 0.025quant 0.5quant
(Intercept)                           3.474 0.032      3.411    3.474
X1.transaction.date                   0.041 0.011      0.019    0.041
```

```
X2.house.age                            -0.078 0.011     -0.101   -0.078
X3.distance.to.the.nearest.MRT.station -0.192 0.023     -0.238   -0.192
X4.number.of.convenience.stores1        0.054 0.043     -0.030    0.054
X4.number.of.convenience.stores2       -0.032 0.056     -0.142   -0.032
X4.number.of.convenience.stores3       -0.039 0.046     -0.130   -0.039
X4.number.of.convenience.stores4        0.046 0.053     -0.058    0.046
X4.number.of.convenience.stores5        0.151 0.045      0.061    0.151
X4.number.of.convenience.stores6        0.181 0.052      0.079    0.181
X4.number.of.convenience.stores7        0.181 0.054      0.074    0.181
X4.number.of.convenience.stores8        0.224 0.056      0.115    0.224
X4.number.of.convenience.stores9        0.243 0.059      0.128    0.243
X4.number.of.convenience.stores10       0.185 0.081      0.026    0.185
X5.latitude                             0.110 0.015      0.081    0.110
X6.longitude                           -0.020 0.020     -0.059   -0.020
                                       0.975quant   mode kld
(Intercept)                                 3.537  3.474   0
X1.transaction.date                         0.063  0.041   0
X2.house.age                               -0.056 -0.078   0
X3.distance.to.the.nearest.MRT.station     -0.146 -0.192   0
X4.number.of.convenience.stores1            0.139  0.054   0
X4.number.of.convenience.stores2            0.077 -0.032   0
X4.number.of.convenience.stores3            0.052 -0.039   0
X4.number.of.convenience.stores4            0.150  0.046   0
X4.number.of.convenience.stores5            0.240  0.151   0
X4.number.of.convenience.stores6            0.282  0.181   0
X4.number.of.convenience.stores7            0.287  0.181   0
X4.number.of.convenience.stores8            0.334  0.224   0
X4.number.of.convenience.stores9            0.357  0.243   0
X4.number.of.convenience.stores10           0.345  0.185   0
X5.latitude                                 0.139  0.110   0
X6.longitude                                0.019 -0.020   0

Model hyperparameters:
                                          mean   sd 0.025quant 0.5quant
Precision for the Gaussian observations 20.54 1.45      17.79    20.51
                                       0.975quant  mode
Precision for the Gaussian observations      23.49 20.44

Expected number of effective parameters(stdev): 16.05(0.004)
Number of equivalent replicates : 25.79

Deviance Information Criterion (DIC) ...............: -62.05
Deviance Information Criterion (DIC, saturated) ....: 428.35
Effective number of parameters .....................: 17.17

Marginal log-Likelihood:  -125.68
CPO and PIT are computed

Posterior marginals for the linear predictor and
 the fitted values are computed

cat('NSLCPO: ', -sum(log(fit.cate.stan$cpo$cpo)), '\n')
cat('DIC: ', fit.cate.stan$dic$dic)

NSLCPO:  -28.02162
DIC:  -62.05112

sd(log(house3$Y.house.price.of.unit.area) - fit.cate.stan$summary.fitted.values$mean)

0.215512718261944
```

From the results, we can see that the NSLCPO, DIC and the standard deviation of the mean residuals for this model are -28.0216, -62.0511 and 0.2155 respectively. When compared with the model fitted in (b), the values of NSLCPO and DIC of this model become slightly larger while the standard deviation of the mean residuals for this model becomes slightly smaller. In a word, the quality of this model is comparable with the model in (b) and there is no evident improvement.

Then, let me fit the robust linear regression for this model.

```
# Implement the robust linear regression using INLA
fit.cate.robu = inla(log(Y.house.price.of.unit.area) ~ X1.transaction.date + X2.house.age +
                     X3.distance.to.the.nearest.MRT.station + X4.number.of.convenience.stores +
                     X5.latitude + X6.longitude, data = house3, family = 't',
                     control.family = list(hyper = prior.t), control.fixed = prior.beta,
                     control.compute = list(cpo = TRUE, dic = TRUE))
```

The summary statistics, NSLCPO and DIC values and the standard deviation of the mean residuals for the robust model are printed out in the following.

```
summary(fit.cate.robu)

Call:
   c("inla(formula = log(Y.house.price.of.unit.area) ~ X1.transaction.date
   + ", " X2.house.age + X3.distance.to.the.nearest.MRT.station +
   X4.number.of.convenience.stores + ", " X5.latitude + X6.longitude,
   family = \"t\", data = house3, ", " control.compute = list(cpo = TRUE,
   dic = TRUE), control.family = list(hyper = prior.t), ", " control.fixed
   = prior.beta)")
Time used:
    Pre = 0.518, Running = 7.55, Post = 0.0854, Total = 8.15
Fixed effects:
                                       mean    sd 0.025quant 0.5quant
(Intercept)                           3.496 0.032      3.432    3.496
X1.transaction.date                   0.025 0.008      0.009    0.025
X2.house.age                         -0.086 0.009     -0.105   -0.086
X3.distance.to.the.nearest.MRT.station -0.223 0.022     -0.266   -0.223
X4.number.of.convenience.stores1       0.029 0.040     -0.051    0.029
X4.number.of.convenience.stores2      -0.048 0.046     -0.138   -0.048
X4.number.of.convenience.stores3      -0.055 0.039     -0.130   -0.055
X4.number.of.convenience.stores4       0.001 0.045     -0.086    0.000
X4.number.of.convenience.stores5       0.133 0.041      0.052    0.133
X4.number.of.convenience.stores6       0.174 0.047      0.082    0.174
X4.number.of.convenience.stores7       0.159 0.049      0.063    0.159
X4.number.of.convenience.stores8       0.194 0.048      0.100    0.194
X4.number.of.convenience.stores9       0.183 0.051      0.084    0.183
X4.number.of.convenience.stores10      0.162 0.059      0.046    0.162
X5.latitude                            0.104 0.013      0.078    0.103
X6.longitude                          -0.036 0.015     -0.066   -0.036
                                      0.975quant   mode kld
(Intercept)                                3.559  3.496   0
X1.transaction.date                        0.042  0.025   0
X2.house.age                              -0.068 -0.086   0
X3.distance.to.the.nearest.MRT.station    -0.180 -0.223   0
X4.number.of.convenience.stores1           0.106  0.030   0
X4.number.of.convenience.stores2           0.042 -0.048   0
X4.number.of.convenience.stores3           0.022 -0.055   0
X4.number.of.convenience.stores4           0.091 -0.001   0
X4.number.of.convenience.stores5           0.214  0.133   0
X4.number.of.convenience.stores6           0.266  0.174   0
X4.number.of.convenience.stores7           0.254  0.159   0
X4.number.of.convenience.stores8           0.290  0.193   0
X4.number.of.convenience.stores9           0.283  0.184   0
X4.number.of.convenience.stores10          0.279  0.162   0
X5.latitude                                0.131  0.103   0
X6.longitude                              -0.007 -0.036   0

Model hyperparameters:
                                        mean    sd 0.025quant 0.5quant
precision for the student-t observations 17.51 3.867      10.59    17.34
degrees of freedom for student-t          2.96 0.399       2.36     2.90
                                        0.975quant  mode
precision for the student-t observations      25.65 17.10
```

```
degrees of freedom for student-t               3.89  2.78

Expected number of effective parameters(stdev): 16.09(0.006)
Number of equivalent replicates : 25.73

Deviance Information Criterion (DIC) ...............: -170.54
Deviance Information Criterion (DIC, saturated) ....: 542.45
Effective number of parameters .....................: 18.58

Marginal log-Likelihood:  -76.90
CPO and PIT are computed

Posterior marginals for the linear predictor and
 the fitted values are computed
```

```
cat('NSLCPO: ', -sum(log(fit.cate.robu$cpo$cpo)), '\n')
cat('DIC: ', fit.cate.robu$dic$dic)
```

```
NSLCPO:  -84.15377
DIC:  -170.5371
```

```
sd(log(house3$Y.house.price.of.unit.area) - fit.cate.robu$summary.fitted.values$mean)
```

```
0.217257631884471
```

The values of NSLCPO, DIC and the standard deviation of the mean residuals for this model are -84.1538, -170.5371 and 0.2173 respectively. All of them are smaller than those of the robust linear regression model in question (d). Therefore, after converting the number of convenience stores variable into a categorical variable, the model has an evident improvement and it is better than the model of question (d) in fitting the relation between the response and the covariates.

Comparing the standard and robust linear regression model in question (e), we can see that the values of NSLCPO and DIC of the robust linear model are remarkably smaller than those of the standard linear model. Hence, in this question, the robust linear regression model is much better than the standard linear model and it is significantly more suitable to fit our data in which there are some outliers. The standard deviation of the mean residuals of the robust model is slightly larger than that of standard model. It is because, in the robust model, the residuals are assumed to have t-distribution which is a fat-tailed distribution, and so the mean residuals would have a larger variance than the standard normal linear model. The t-distribution is more robust when tackling outliers and it reduces the impact of outliers on the fit of the model.

Question (f)

In this question, we are going to modify the regression models (both the standard and the robust model) in question (e) by adding interaction terms. We aim to find a model that improves upon the previous models in terms of accuracy (i.e. the mean residuals have smaller standard deviation). After many trials, I find that for the standard linear regression model, adding the interaction terms `X1.transaction.date*X2.house.age` and `X3.distance.to.the.nearest.MRT.station*X5.latitude` can achieve our goal. On the other hand, for the robust linear regression model, adding the interaction term `X1.transaction.date*X2.house.age` can improve the accuracy.

Firstly, let me fit the standard linear regression model with interaction terms illustrated above.

```
# Implement the standard linear regression with interaction terms
fit.inter.stan=inla(log(Y.house.price.of.unit.area) ~ X1.transaction.date + X2.house.age +
                    X3.distance.to.the.nearest.MRT.station + X4.number.of.convenience.stores +
```

```
                    X5.latitude + X6.longitude + I(X1.transaction.date*X2.house.age) +
                    I(X3.distance.to.the.nearest.MRT.station*X5.latitude),
                    data = house3, family = 'gaussian', control.family = list(hyper = prec.prior),
                    control.fixed = prior.beta, control.compute = list(cpo = TRUE, dic = TRUE))
```

The summary statistics, the values of NSLCPO and DIC and the standard deviation of the mean residuals are printed out below.

```
summary(fit.inter.stan)

Call:
   c("inla(formula = log(Y.house.price.of.unit.area) ~ X1.transaction.date
   + ", " X2.house.age + X3.distance.to.the.nearest.MRT.station +
   X4.number.of.convenience.stores + ", " X5.latitude + X6.longitude +
   I(X1.transaction.date * X2.house.age) + ", "
   I(X3.distance.to.the.nearest.MRT.station * X5.latitude), ", " family =
   \"gaussian\", data = house3, control.compute = list(cpo = TRUE, ", "
   dic = TRUE), control.family = list(hyper = prec.prior), ", "
   control.fixed = prior.beta)")
Time used:
    Pre = 0.615, Running = 0.716, Post = 0.076, Total = 1.41
Fixed effects:
                                                    mean    sd 0.025quant
(Intercept)                                        3.449 0.032      3.386
X1.transaction.date                                0.043 0.011      0.022
X2.house.age                                      -0.080 0.011     -0.102
X3.distance.to.the.nearest.MRT.station            -0.223 0.025     -0.272
X4.number.of.convenience.stores1                   0.060 0.042     -0.023
X4.number.of.convenience.stores2                  -0.005 0.055     -0.113
X4.number.of.convenience.stores3                   0.005 0.047     -0.087
X4.number.of.convenience.stores4                   0.055 0.052     -0.048
X4.number.of.convenience.stores5                   0.139 0.045      0.051
X4.number.of.convenience.stores6                   0.176 0.051      0.076
X4.number.of.convenience.stores7                   0.171 0.053      0.066
X4.number.of.convenience.stores8                   0.207 0.055      0.099
X4.number.of.convenience.stores9                   0.215 0.058      0.100
X4.number.of.convenience.stores10                  0.135 0.081     -0.025
X5.latitude                                        0.129 0.015      0.099
X6.longitude                                      -0.008 0.020     -0.047
I(X1.transaction.date * X2.house.age)              0.021 0.011     -0.001
I(X3.distance.to.the.nearest.MRT.station * X5.latitude) -0.041 0.011     -0.063
                                                  0.5quant 0.975quant
(Intercept)                                          3.449      3.512
X1.transaction.date                                  0.043      0.064
X2.house.age                                        -0.080     -0.058
X3.distance.to.the.nearest.MRT.station              -0.223     -0.174
X4.number.of.convenience.stores1                     0.060      0.144
X4.number.of.convenience.stores2                    -0.005      0.103
X4.number.of.convenience.stores3                     0.005      0.097
X4.number.of.convenience.stores4                     0.055      0.158
X4.number.of.convenience.stores5                     0.139      0.227
X4.number.of.convenience.stores6                     0.176      0.277
X4.number.of.convenience.stores7                     0.171      0.276
X4.number.of.convenience.stores8                     0.207      0.315
X4.number.of.convenience.stores9                     0.215      0.329
X4.number.of.convenience.stores10                    0.135      0.294
X5.latitude                                          0.129      0.159
X6.longitude                                        -0.008      0.031
I(X1.transaction.date * X2.house.age)                0.021      0.042
I(X3.distance.to.the.nearest.MRT.station * X5.latitude)   -0.041     -0.018
                                                    mode kld
(Intercept)                                        3.449   0
X1.transaction.date                                0.043   0
X2.house.age                                      -0.080   0
X3.distance.to.the.nearest.MRT.station            -0.223   0
X4.number.of.convenience.stores1                   0.060   0
X4.number.of.convenience.stores2                  -0.005   0
```

```
X4.number.of.convenience.stores3                          0.005   0
X4.number.of.convenience.stores4                          0.055   0
X4.number.of.convenience.stores5                          0.139   0
X4.number.of.convenience.stores6                          0.176   0
X4.number.of.convenience.stores7                          0.171   0
X4.number.of.convenience.stores8                          0.207   0
X4.number.of.convenience.stores9                          0.215   0
X4.number.of.convenience.stores10                         0.135   0
X5.latitude                                               0.129   0
X6.longitude                                             -0.008   0
I(X1.transaction.date * X2.house.age)                     0.021   0
I(X3.distance.to.the.nearest.MRT.station * X5.latitude) -0.041   0

Model hyperparameters:
                                        mean   sd 0.025quant 0.5quant
Precision for the Gaussian observations 21.29 1.51      18.43    21.25
                                        0.975quant  mode
Precision for the Gaussian observations      24.36 21.18

Expected number of effective parameters(stdev): 18.05(0.004)
Number of equivalent replicates : 22.93

Deviance Information Criterion (DIC) ...............: -75.00
Deviance Information Criterion (DIC, saturated) ....: 430.24
Effective number of parameters .....................: 19.18

Marginal log-Likelihood:  -140.37
CPO and PIT are computed

Posterior marginals for the linear predictor and
 the fitted values are computed

cat('NSLCPO: ', -sum(log(fit.inter.stan$cpo$cpo)), '\n')
cat('DIC: ', fit.inter.stan$dic$dic)

NSLCPO:  -33.73235
DIC:  -74.99679

sd(log(house3$Y.house.price.of.unit.area) - fit.inter.stan$summary.fitted.values$mean)

0.211113379608135
```

The standard deviation of the mean residuals for this model is 0.2111 and it is smaller than that of the standard linear model in (e). Moreover, we can see that the values of NSLCPO and DIC of this model are -33.7324 and -74.9968 respectively and they are all evidently smaller than those of the standard linear models in (e) and (d). Therefore, adding these intersection terms obviously improves the standard linear model in (e) and it is more suitable to depict the relation between the response and the covariates.

Then, let me fit the robust linear regression model with the intersection term `X1.transaction.date * X2.house.age`

```
# Implement the robust linear regression with interaction terms
fit.inter.robu = inla(log(Y.house.price.of.unit.area) ~ X1.transaction.date + X2.house.age +
                      X3.distance.to.the.nearest.MRT.station + X4.number.of.convenience.stores +
                      X5.latitude + X6.longitude + I(X1.transaction.date*X2.house.age),
                      data = house3, family = 't', control.family = list(hyper = prior.t),
                      control.fixed = prior.beta, control.compute = list(cpo = TRUE, dic = TRUE))
```

The summary statistics, the values of NSLCPO and DIC and the standard deviation of the mean residuals are printed out as follows.

```
summary(fit.inter.robu)
```

```
Call:
   c("inla(formula = log(Y.house.price.of.unit.area) ~ X1.transaction.date
   + ", " X2.house.age + X3.distance.to.the.nearest.MRT.station +
   X4.number.of.convenience.stores + ", " X5.latitude + X6.longitude +
   I(X1.transaction.date * X2.house.age), ", " family = \"t\", data =
   house3, control.compute = list(cpo = TRUE, ", " dic = TRUE),
   control.family = list(hyper = prior.t), ", " control.fixed =
   prior.beta)")
Time used:
    Pre = 0.78, Running = 7.73, Post = 0.0798, Total = 8.59
Fixed effects:
                                     mean    sd 0.025quant 0.5quant
(Intercept)                         3.495 0.032      3.431    3.495
X1.transaction.date                 0.026 0.009      0.009    0.026
X2.house.age                       -0.086 0.010     -0.105   -0.086
X3.distance.to.the.nearest.MRT.station -0.222 0.022     -0.265   -0.222
X4.number.of.convenience.stores1    0.030 0.040     -0.050    0.030
X4.number.of.convenience.stores2   -0.047 0.046     -0.137   -0.047
X4.number.of.convenience.stores3   -0.054 0.039     -0.129   -0.054
X4.number.of.convenience.stores4    0.003 0.045     -0.084    0.002
X4.number.of.convenience.stores5    0.134 0.041      0.053    0.134
X4.number.of.convenience.stores6    0.175 0.047      0.083    0.176
X4.number.of.convenience.stores7    0.159 0.049      0.063    0.159
X4.number.of.convenience.stores8    0.195 0.049      0.100    0.194
X4.number.of.convenience.stores9    0.185 0.051      0.085    0.185
X4.number.of.convenience.stores10   0.162 0.059      0.046    0.161
X5.latitude                         0.104 0.013      0.079    0.104
X6.longitude                       -0.036 0.015     -0.066   -0.036
I(X1.transaction.date * X2.house.age)   0.004 0.009     -0.014    0.004
                                   0.975quant   mode kld
(Intercept)                             3.558  3.496   0
X1.transaction.date                     0.043  0.025   0
X2.house.age                           -0.067 -0.086   0
X3.distance.to.the.nearest.MRT.station -0.179 -0.222   0
X4.number.of.convenience.stores1        0.107  0.031   0
X4.number.of.convenience.stores2        0.043 -0.048   0
X4.number.of.convenience.stores3        0.023 -0.054   0
X4.number.of.convenience.stores4        0.093  0.001   0
X4.number.of.convenience.stores5        0.215  0.134   0
X4.number.of.convenience.stores6        0.267  0.176   0
X4.number.of.convenience.stores7        0.254  0.159   0
X4.number.of.convenience.stores8        0.291  0.194   0
X4.number.of.convenience.stores9        0.284  0.185   0
X4.number.of.convenience.stores10       0.279  0.161   0
X5.latitude                             0.131  0.103   0
X6.longitude                           -0.006 -0.036   0
I(X1.transaction.date * X2.house.age)   0.021  0.003   0

Model hyperparameters:
                                          mean    sd 0.025quant 0.5quant
precision for the student-t observations 17.79 3.853      10.91    17.60
degrees of freedom for student-t          2.98 0.401       2.38     2.92
                                         0.975quant  mode
precision for the student-t observations      25.91 17.31
degrees of freedom for student-t               3.92  2.80

Expected number of effective parameters(stdev): 17.09(0.006)
Number of equivalent replicates : 24.23

Deviance Information Criterion (DIC) ...............: -168.68
Deviance Information Criterion (DIC, saturated) ....: 436.34
Effective number of parameters .....................: 19.64

Marginal log-Likelihood:  -88.52
CPO and PIT are computed

Posterior marginals for the linear predictor and
 the fitted values are computed
```

```
cat('NSLCPO: ', -sum(log(fit.inter.robu$cpo$cpo)), '\n')
cat('DIC: ', fit.inter.robu$dic$dic)

NSLCPO:  -83.12028
DIC:  -168.6773

sd(log(house3$Y.house.price.of.unit.area) - fit.inter.robu$summary.fitted.values$mean)

0.21684566997635
```

The standard deviation of the mean residuals for this model is 0.2168 and it is faintly smaller than that of the robust linear model in (e). However, the values of NSLCPO and DIC of this model, which are -83.1203 and -168.6773 respectively, are all slightly larger than those of the robust model of (e). Therefore, except a little promotion on the accuracy, the quality of this model has no evident improvement after adding the intersection term to this model.

Then, we can also compare the standard and robust linear regression model with intersection terms in this question. It is completely similar with the situation in the previous question. Although the standard deviation of the mean residuals of the robust linear model is slightly larger than that of the standard linear model, the values of NSLCPO and DIC of the robust model are remarkably smaller than those of the standard model. Hence, all the analyses conducted so far have proved that the robust linear regression model is much better than the standard linear model for our data.

Furthermore, it is worth noting that the quality of the robust linear model does not have evident improvement after the addition of the intersection term. Unlike the standard linear model whose quality will have much improvement after adding the intersection term `X3.distance.to.the.nearest.MRT.station * X5.latitude`, the quality of the robust model will become very bad if this item is added to it. I have tried many combinations of the intersection items for the robust model, but I find that only the intersection term `X1.transaction.date * X2.house.age` can bring some improvements on the accuracy of this model and all the other situations that I have tried will make this model become worse. Hence, it may not be very suitable to add intersection terms to the robust linear model.

Question (g)

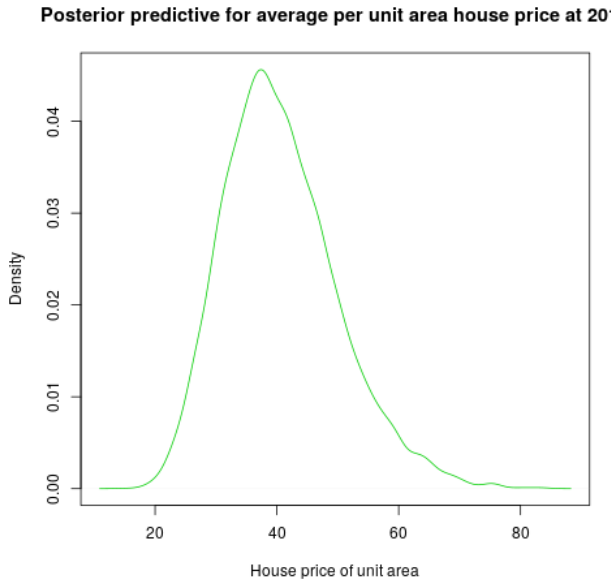
For this question, I am going to compute and plot the posterior predictive density of the average per unit area house price averaged among all of the houses in the dataset at different transaction dates using the model of question (b).

I have implemented a function called `post.pred.inla` to perform such computation. This function can be used to sample from the posterior predictive distribution of the average per unit area house price at any given transaction date. it takes three arguments, where the first argument `data` takes the data set of house transaction information as input, the second argument `transaction.date` refers to the date that the transaction happens and the third argument `nsamp` refers to the size of the sample that we want to take from the posterior predictive distribution of the average per unit area house price. The default value of the argument `data` is the original data set `house` read from the file `Real_estate.csv` and the default of the argument `nsamp` is 10000. This function finally returns the sample of size `nsamp` from the posterior predictive distribution of the average per unit area house price. The covariate `house.age` would be updated according to the transaction date, and all the other covariates "distance from nearest MRT", "number of convenience stores", "latitude", and "longitude" stay the same during computation. In the function `post.pred.inla`, before fitting the linear regression model using INLA, I would still center and scale the covariates in the data set at first.

```
post.pred.inla = function(data = house, transaction.date, nsamp = 10000){
  No = nrow(data) + 1
  house.age = mean(data$X2.house.age - (data$X1.transaction.date - transaction.date))
  distance.to.the.nearest.MRT.station = mean(data$X3.distance.to.the.nearest.MRT.station)
  number.of.convenience.stores = mean(data$X4.number.of.convenience.stores)
  latitude = mean(data$X5.latitude)
  longitude = mean(data$X6.longitude)
  house.aver = data.frame(No = No, X1.transaction.date = transaction.date, X2.house.age = house.age,
                          X3.distance.to.the.nearest.MRT.station = distance.to.the.nearest.MRT.station,
                          X4.number.of.convenience.stores = number.of.convenience.stores,
                          X5.latitude = latitude, X6.longitude = longitude, Y.house.price.of.unit.area
= NA)
  house.pred = rbind(data, house.aver)
  house.pred.adjust = data.frame(
  'X1.transaction.date' = (house.pred[[2]] - mean(house.pred[[2]]))/sd(house.pred[[2]]),
  'X2.house.age' = (house.pred[[3]] - mean(house.pred[[3]]))/sd(house.pred[[3]]),
  'X3.distance.to.the.nearest.MRT.station' = (house.pred[[4]] - mean(house.pred[[4]]))/sd(house.pred[[4
]]),
  'X4.number.of.convenience.stores' = (house.pred[[5]] - mean(house.pred[[5]]))/sd(house.pred[[5]]),
  'X5.latitude' = (house.pred[[6]] - mean(house.pred[[6]]))/sd(house.pred[[6]]),
  'X6.longitude' = (house.pred[[7]] - mean(house.pred[[7]]))/sd(house.pred[[7]]),
  'Y.house.price.of.unit.area' = house.pred[[8]]
  )

  prec.prior = list(prec = list(prior = 'loggamma', param = c(0.1, 0.1)))
  prior.beta = list(mean.intercept = 0, prec.intercept = 0.000001, mean = 0, prec = 0.000001)
  fit.pred.inla = inla(log(Y.house.price.of.unit.area) ~ X1.transaction.date + X2.house.age +
                       X3.distance.to.the.nearest.MRT.station + X4.number.of.convenience.stores +
                       X5.latitude + X6.longitude, data = house.pred.adjust, family = 'gaussian',
                       control.family = list(hyper = prec.prior), control.fixed = prior.beta,
                       control.compute = list(config = TRUE), control.predictor = list(compute = TRUE))
  samples = inla.posterior.sample(n = nsamp, result = fit.pred.inla, selection = list(Predictor = No))
  predictors = inla.posterior.sample.eval(function(...){Predictor}, samples)
  sigma = 1/sqrt(inla.posterior.sample.eval(function(...){theta}, samples))
  sam.post.pred = exp(predictors + rnorm(n = nsamp, mean = 0, sd = sigma))
  return(sam.post.pred)
}
```

Now, I am going to take samples (10000 samples in each case) from the posterior predictive distribution of the average per unit area house price averaged among all of the houses in the data set at the transaction dates 2014.0, 2015.0 and 2016.0 using the function `post.pred.inla` that I implement. Then, I will plot the posterior predictive density of the average per unit area house price using these samples and compute the posterior means for all the 3 cases. The results are shown in the following.

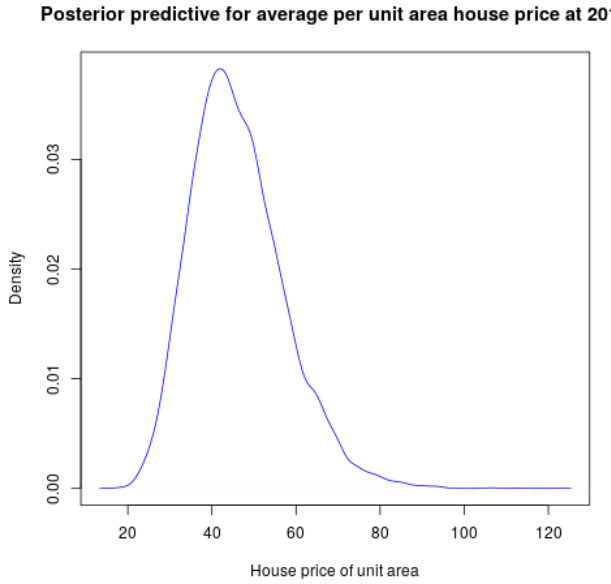
```
# The posterior predictive distribution of the average per unit area house price at the transaction dat
e 2014.0
sam.post.pred2014 = post.pred.inla(transaction.date = 2014)
plot(density(sam.post.pred2014), xlab = 'House price of unit area', ylab = 'Density', col = 'green3',
     main = 'Posterior predictive for average per unit area house price at 2014')
```




```
mean(sam.post.pred2014)
```

```
40.6268848510242
```

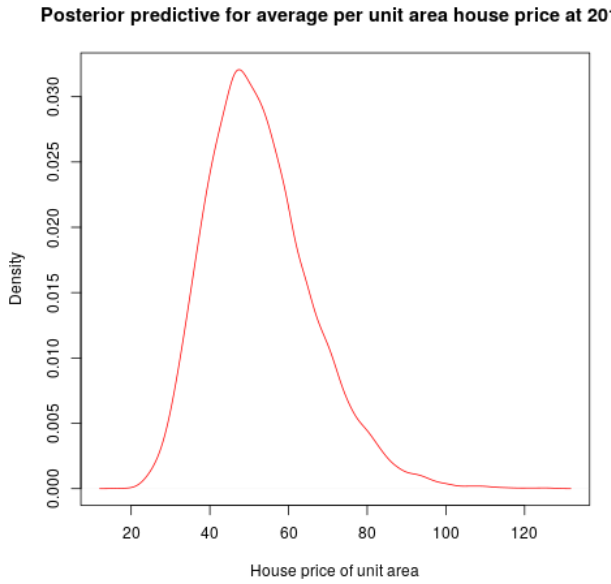
```
# The posterior predictive distribution of the average per unit area house price at the transaction dat
e 2015.0
sam.post.pred2015 = post.pred.inla(transaction.date = 2015)
plot(density(sam.post.pred2015), xlab = 'House price of unit area', ylab = 'Density', col = 'blue',
     main = 'Posterior predictive for average per unit area house price at 2015')
```




```
mean(sam.post.pred2015)
```

```
46.1837908920769
```

```
# The posterior predictive distribution of the average per unit area house price at the transaction dat
e 2016.0
sam.post.pred2016 = post.pred.inla(transaction.date = 2016)
plot(density(sam.post.pred2016), xlab = 'House price of unit area', ylab = 'Density', col = 'red',
     main = 'Posterior predictive for average per unit area house price at 2016')
```




```
mean(sam.post.pred2016)
```

```
52.7669348221116
```

The posterior means of the average per unit area house price at the transaction dates 2014.0, 2015.0 and 2016.0 are 40.6269, 46.1838 and 52.7669 respectively. From the results, we can see that the posterior mean of the average per unit area house price has a trend of increasing year by year, which means the posterior predictive density of the average per unit area house price has a tendency of right moving year by year.