PM2.5 Data Analysis

Ze Qian Wu (1002192254)

Feb 23, 2019

Question 1

Exploratory Data Analysis and Linear Regression

The aim of the project is to investigate the relationship between particulate matter of less than 2.5 microns in diameter (PM2.5) and a satellite data product the provides a mis-calibrated estimate of PM2.5. The data set contains information measurements for PM2.5 and the satellite product at a number of locations.

```
library(ggplot2)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(INLA)
```

```
## Loading required package: Matrix
```

```
## Loading required package: sp
```

```
## This is INLA_18.07.12 built 2018-07-12 11:07:12 UTC.
## See www.r-inla.org/contact-us for how to get help.
## To enable PARDISO sparse library; see inla.pardiso()
```

```
library(tidyr)
```

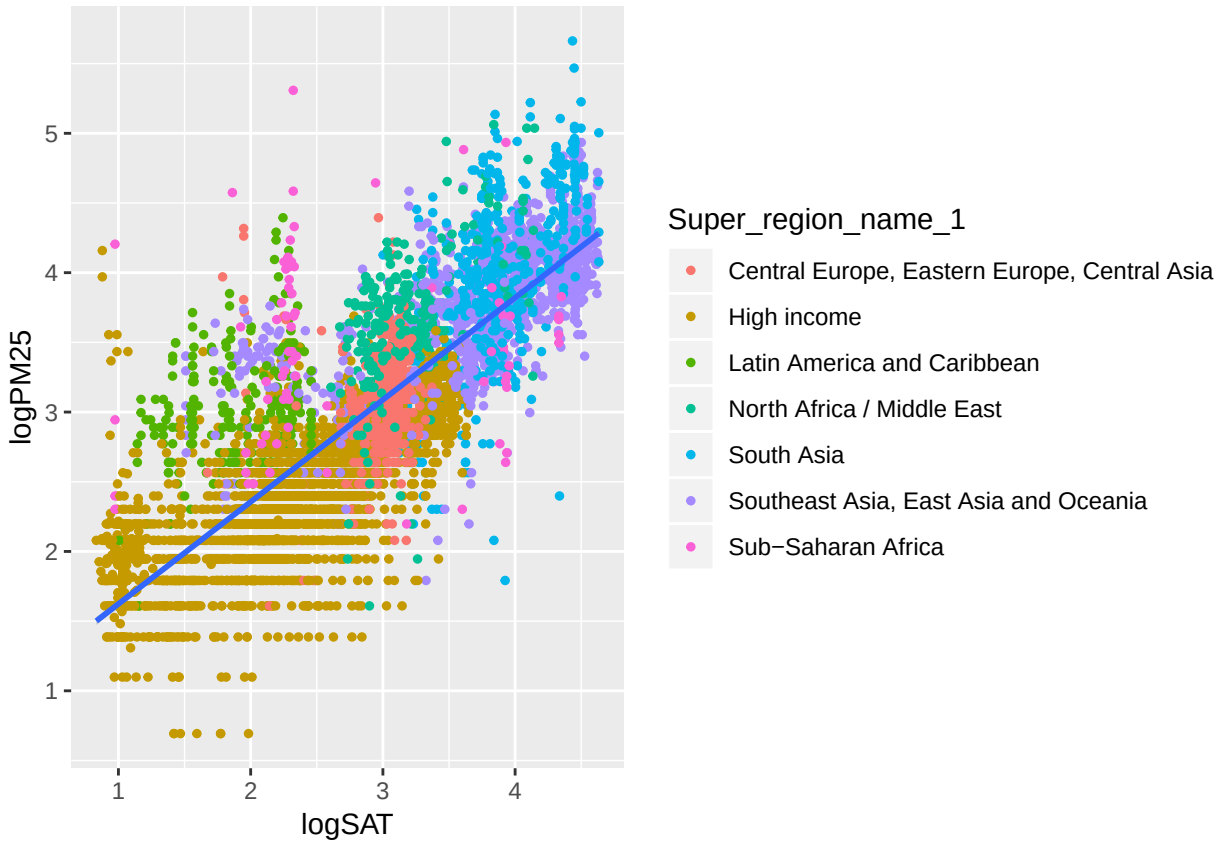
```
##
## Attaching package: 'tidyr'
```

```
## The following object is masked from 'package:Matrix':
##
##     expand
```

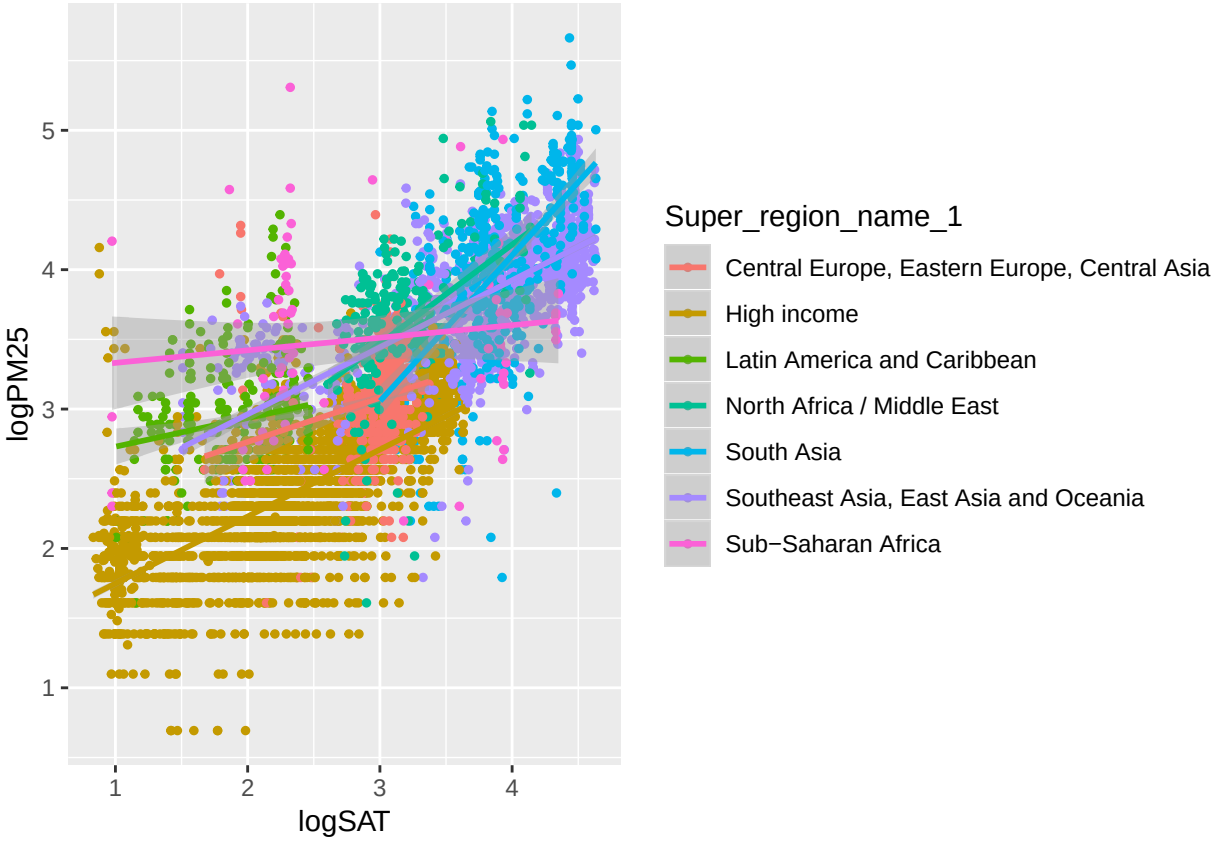
```
load("/Users/Claire/Desktop/assignment1.RData")
```

We first illustrate the method using the following data:

```
ggplot(data = assignment_1, mapping = aes(x = logSAT, y = logPM25)) +
  geom_point(size = 1, mapping = aes(x = logSAT, y = logPM25, colour = Super_region_name_1)) + stat_smo
```




```
ggplot(data = assignment_1, mapping = aes(x = logSAT, y = logPM25, colour = Super_region_name_1)) + geo
```



Based on the gg-plots above, we can conclude that

1) there are no visible outliers presented in the graph as all of the points lie in between [1,5], which is not too small and also not too large.
2) for each individual super region, the line indicates the relationship between logSAT and logPM25 are all positive

By graphing out (log)PM2.5 vs. (log)SAT, it is presented that all the points lie in a relatively straight line and therefore there is a linear relationship between the two variables. In this case, it is appropriate to use linear regression model to further explore the relationship between (log)PM2.5 and (log)SAT.

```
fit_total <- lm(logPM25 ~ logSAT + 1, data = assignment_1)
summary(fit_total)
```

```
##
## Call:
## lm(formula = logPM25 ~ logSAT + 1, data = assignment_1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.97320 -0.28996 -0.06591  0.26351  2.71659
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.891078   0.020005   44.54   <2e-16 ***
## logSAT      0.732294   0.006682  109.59   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 0.4641 on 6001 degrees of freedom
## Multiple R-squared:  0.6668, Adjusted R-squared:  0.6668
## F-statistic: 1.201e+04 on 1 and 6001 DF,  p-value: < 2.2e-16
```

```
output <- data.frame(super_region = unique(assignment_1$Super_region_name_1), R2 = rep(NA,7))
dat <- data.frame(logPM25 = assignment_1$logPM25, logSAT = assignment_1$logSAT, super_region = assignmer
for(i in 1:7) {
  dat_local <- dat %>% filter(super_region == output$super_region[i])
  output$R2[i] <- summary(lm(logPM25 ~ logSAT, data = dat_local))$r.squared
}
print(output, digits=2)
```

```
##                                   super_region    R2
## 1                                   South Asia 0.389
## 2 Central Europe, Eastern Europe, Central Asia 0.033
## 3                                  High income 0.478
## 4                    North Africa / Middle East 0.289
## 5                  Latin America and Caribbean  0.032
## 6        Southeast Asia, East Asia and Oceania 0.472
## 7                            Sub-Saharan Africa 0.015
```

According to the table that indicates all the values of R^2, we can see that the value we obtained from fitting one line to all the data is greater than when we separate regions. In particular, the values of Central Europe, Eastern Europe, Central Asia(0.033);Latin America and Caribbean(0.032)and Sub-Saharan Africa(0.015) are extremely small comparing to the other categories. Based on the R^2 values, considering all the regions separately generates a less strong correlation than fitting all the regions into one category.

Now we want to define our own prior. In this case, the penalizd complexity(pc) prior is chosen in INLA. We use the residuals of the standard deviation to set the scale of the pc prior.

```
res_sd <- sd(residuals(fit_total))
prior_spatial <- list(prec = list(prior = "pc.prec", param = c(5*res_sd, 0.01)),phi = list(prior = "pc"
prior_iid <- list(prec = list(prior = "pc.prec", param = c(5*res_sd, 0.01)))
formula1 <- logPM25 ~ 1 + logSAT + f(Super_region_name_1, model = "iid", hyper = prior_iid) + f(country_

formula1
```

```
## logPM25 ~ 1 + logSAT + f(Super_region_name_1, model = "iid",
##     hyper = prior_iid) + f(country_code_1, model = "bym2", graph = "/Users/Claire/Desktop/world.adj"
##     hyper = prior_spatial)
```

```
result1 <- inla(formula1, family = "gaussian", data = assignment_1, control.fixed = list(mean = 1,prec =
```

```
## Warning in inla.model.properties.generic(inla.trim.family(model), (mm[names(mm) == : Model 'bym2' in
##   Use this model with extra care!!! Further warnings are disabled.
```

```
summary(result1)
```

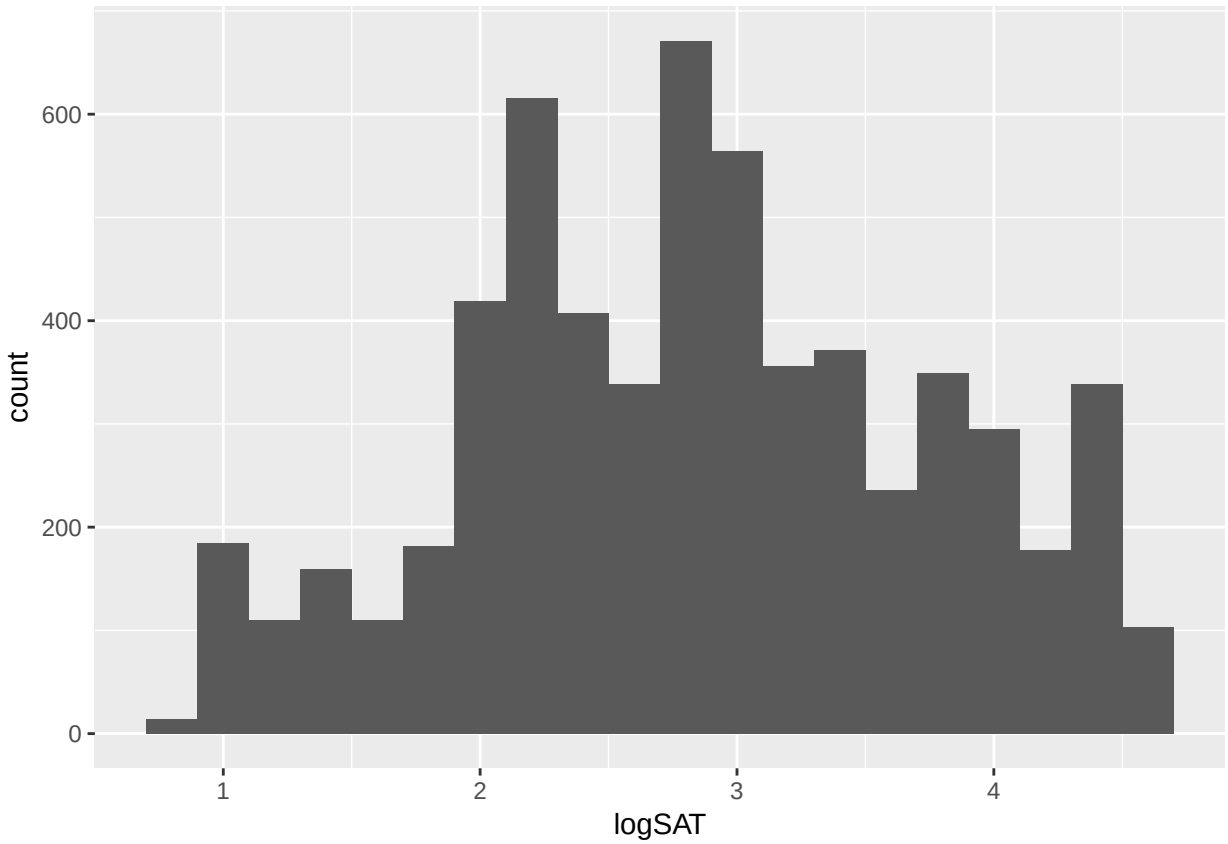
```
##
## Call:
## c("inla(formula = formula1, family = \"gaussian\", data = assignment_1, ",  "    control.compute = li
##
## Time used:
##  Pre-processing    Running inla Post-processing           Total
##         10.7803         15.2310          0.6641         26.6754
##
## Fixed effects:
##               mean     sd 0.025quant 0.5quant 0.975quant   mode   kld
```

```
## (Intercept) 1.9015 0.1533     1.5953   1.9004     2.2129 1.8983 1e-04
## logSAT      0.4923 0.0115     0.4697   0.4923     0.5149 0.4923 0e+00
##
## Random effects:
## Name   Model
##  Super_region_name_1   IID model
## country_code_1   BYM2 model
##
## Model hyperparameters:
##                                          mean     sd 0.025quant 0.5quant
## Precision for the Gaussian observations 9.6378 0.1776     9.2933   9.6361
## Precision for Super_region_name_1       9.7557 6.8233     2.1455   8.0431
## Precision for country_code_1            7.6853 1.9004     4.8705   7.3633
## Phi for country_code_1                  0.3541 0.1865     0.0545   0.3414
##                                         0.975quant   mode
## Precision for the Gaussian observations     9.9921 9.6329
## Precision for Super_region_name_1          27.4968 5.2667
## Precision for country_code_1               12.2621 6.7257
## Phi for country_code_1                      0.7302 0.2544
##
## Expected number of effective parameters(std dev): 93.80(1.85)
## Number of equivalent replicates : 64.00
##
## Marginal log-Likelihood:  -1819.61
## CPO and PIT are computed
##
## Posterior marginals for linear predictor and fitted values computed
```

Question 2

Bayesian Model

```
ggplot(data = assignment_1, aes(x = logSAT)) +geom_histogram(bins = 20)
```



By looking at the histogram, we can conclude that (log)SAT lies in the range of [0,5], and therefore the value of beta should not be much greater than 2. This finding is consistent with the data generated from the prior models.

```
library(rstanarm)
```

```
## Loading required package: Rcpp
## rstanarm (Version 2.18.2, packaged: 2018-11-08 22:19:38 UTC)
## - Do not expect the default priors to remain the same in future rstanarm versions.
## Thus, R scripts should specify priors explicitly, even if they are just the defaults.
## - For execution on a local, multicore CPU with excess RAM we recommend calling
## options(mc.cores = parallel::detectCores())
## - Plotting theme set to bayesplot::theme_default().
```

```
fit <- stan_glm(formula = logPM25 ~ 1 + logSAT, data = assignment_1, family = gaussian(),prior_intercep
```

```
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0.000682 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 6.82 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
```

```
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 5.36026 seconds (Warm-up)
## Chain 1:                4.41486 seconds (Sampling)
## Chain 1:                9.77513 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 0.0004 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 4 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 5.02778 seconds (Warm-up)
## Chain 2:                4.74574 seconds (Sampling)
## Chain 2:                9.77352 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 0.000507 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 5.07 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
```

```
## Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 5.45472 seconds (Warm-up)
## Chain 3:                4.34732 seconds (Sampling)
## Chain 3:                9.80204 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 0.000619 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 6.19 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 5.50653 seconds (Warm-up)
## Chain 4:                4.43525 seconds (Sampling)
## Chain 4:                9.94177 seconds (Total)
## Chain 4:
```

```
summary(fit)
```

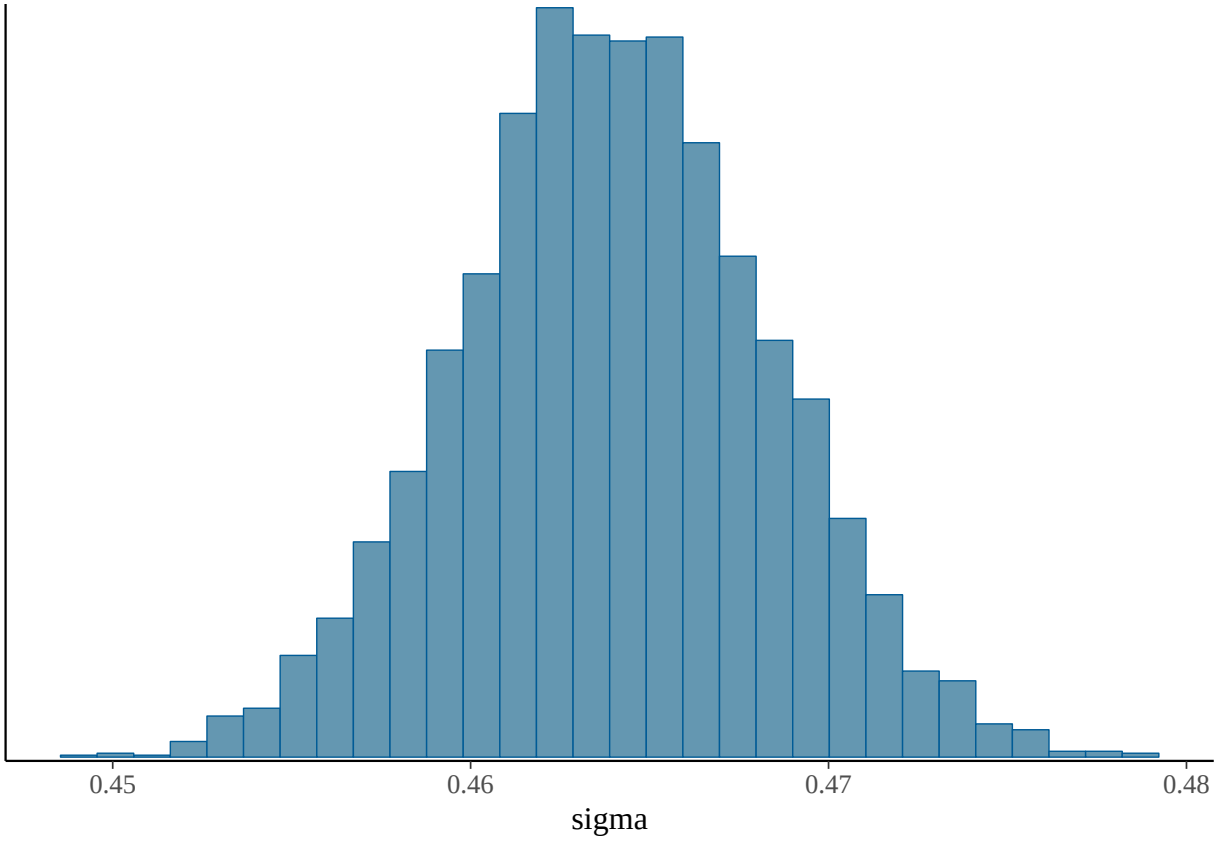
```
##
## Model Info:
##
##  function:     stan_glm
##  family:       gaussian [identity]
##  formula:      logPM25 ~ 1 + logSAT
##  algorithm:    sampling
##  priors:       see help('prior_summary')
##  sample:       4000 (posterior sample size)
##  observations: 6003
##  predictors:   2
##
## Estimates:
##                 mean    sd      2.5%    25%     50%     75%     97.5%
## (Intercept)       0.9     0.0     0.9     0.9     0.9     0.9     0.9
```

```
## logSAT            0.7     0.0     0.7     0.7     0.7     0.7     0.7
## sigma             0.5     0.0     0.5     0.5     0.5     0.5     0.5
## mean_PPD          3.0     0.0     3.0     3.0     3.0     3.0     3.0
## log-posterior -3914.7     1.3 -3918.1 -3915.3 -3914.4 -3913.8 -3913.3
##
## Diagnostics:
##               mcse Rhat n_eff
## (Intercept)   0.0  1.0  4272
## logSAT        0.0  1.0  4478
## sigma         0.0  1.0  1979
## mean_PPD      0.0  1.0  3174
## log-posterior 0.0  1.0  1363
##
## For each parameter, mcse is Monte Carlo standard error, n_eff is a crude measure of effective sample
```

We now lot a histogram with the sigma value we generated from above.

```
plot(fit, pars="sigma", "hist")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
formula2 = logPM25 ~ 1 + logSAT +
  f(Super_region_name_1,model="iid", hyper = prior_iid) +
  f(country_code_1, model = 'iid', graph = '/Users/Claire/Desktop/world.adj', hyper = prior_iid)
result2 = inla(formula2, family="gaussian", data = assignment_1, control.fixed = list(mean = 1, prec=1/1
summary(result2)
```

```
##
## Call:
```

```
## c("inla(formula = formula2, family = \"gaussian\", data = assignment_1, ",  "    control.compute = li
##
## Time used:
##  Pre-processing    Running inla Post-processing           Total
##          1.3750          6.4069          0.5966          8.3785
##
## Fixed effects:
##               mean     sd 0.025quant 0.5quant 0.975quant   mode kld
## (Intercept) 1.8901 0.1591     1.5715   1.8892     2.2133 1.8873   0
## logSAT      0.4923 0.0115     0.4697   0.4923     0.5148 0.4923   0
##
## Random effects:
## Name   Model
##  Super_region_name_1   IID model
## country_code_1   IID model
##
## Model hyperparameters:
##                                          mean     sd 0.025quant 0.5quant
## Precision for the Gaussian observations 9.638 0.1776      9.291    9.637
## Precision for Super_region_name_1       8.489 5.4910      1.994    7.200
## Precision for country_code_1            8.715 1.4970      6.113    8.600
##                                         0.975quant  mode
## Precision for the Gaussian observations       9.99 9.638
## Precision for Super_region_name_1            22.69 4.919
## Precision for country_code_1                 11.99 8.384
##
## Expected number of effective parameters(std dev): 94.24(1.726)
## Number of equivalent replicates : 63.70
##
## Marginal log-Likelihood:  -1896.25
## CPO and PIT are computed
##
## Posterior marginals for linear predictor and fitted values computed
```

```
formula3 <- logPM25 ~ 1 + logSAT
result3 <- inla(formula3, family = "gaussian", data = assignment_1, control.fixed = list(mean = 1, prec
summary(result3)
```

```
##
## Call:
## c("inla(formula = formula3, family = \"gaussian\", data = assignment_1, ",  "    control.compute = li
##
## Time used:
##  Pre-processing    Running inla Post-processing           Total
##          0.8655          3.3893          0.5488          4.8035
##
## Fixed effects:
##               mean     sd 0.025quant 0.5quant 0.975quant   mode kld
## (Intercept) 0.8910 0.0200     0.8517   0.8910     0.9303 0.8910   0
## logSAT      0.7323 0.0067     0.7192   0.7323     0.7454 0.7323   0
##
## The model has no random effects
##
## Model hyperparameters:
##                                          mean     sd 0.025quant 0.5quant
```

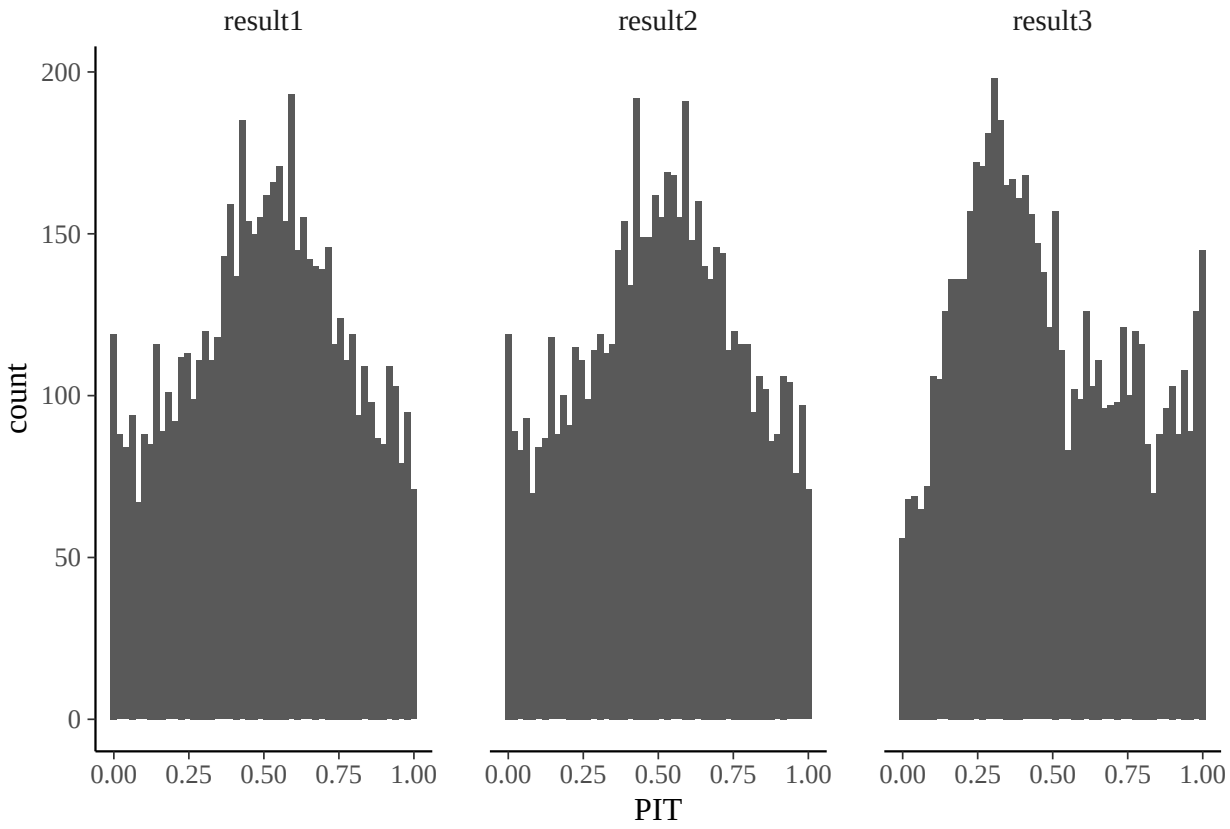
```
## Precision for the Gaussian observations 4.645 0.0836      4.482    4.644
##                                         0.975quant  mode
## Precision for the Gaussian observations      4.811 4.643
##
## Expected number of effective parameters(std dev): 2.171(0.0032)
## Number of equivalent replicates : 2764.84
##
## Marginal log-Likelihood:  -3931.55
## CPO and PIT are computed
##
## Posterior marginals for linear predictor and fitted values computed
```

```
cpo1 <- result1$cpo
cpo2 <- result2$cpo
cpo3 <- result3$cpo
sum(cpo1$failure, cpo2$failure, cpo3$failure)
```

```
## [1] 0
```

Based on the sum we get from calculating the number of failures of CPO/PIT, the output we get is 0. We can now graph out the PIT histograms for the three models

```
PIT_data <- tibble(result1 = cpo1$pit, result2 = cpo2$pit,result3 = cpo3$pit)
PIT_data <- PIT_data %>% gather(key = "Model", value = "PIT") %>% mutate(Model = as.factor(Model))
ggplot(data = PIT_data, mapping=aes(x = PIT)) +
  geom_histogram(bins = 50) +facet_wrap(.~Model)
```



From the histograms above,we can see that the full model and the spatial iid model look almost identical and they are more of a bell-shape, whereas the baseline model seems to be more left-tailed. Comparing the three models, the full model and the spatial iid model are more normal and uniform than the baseline model.

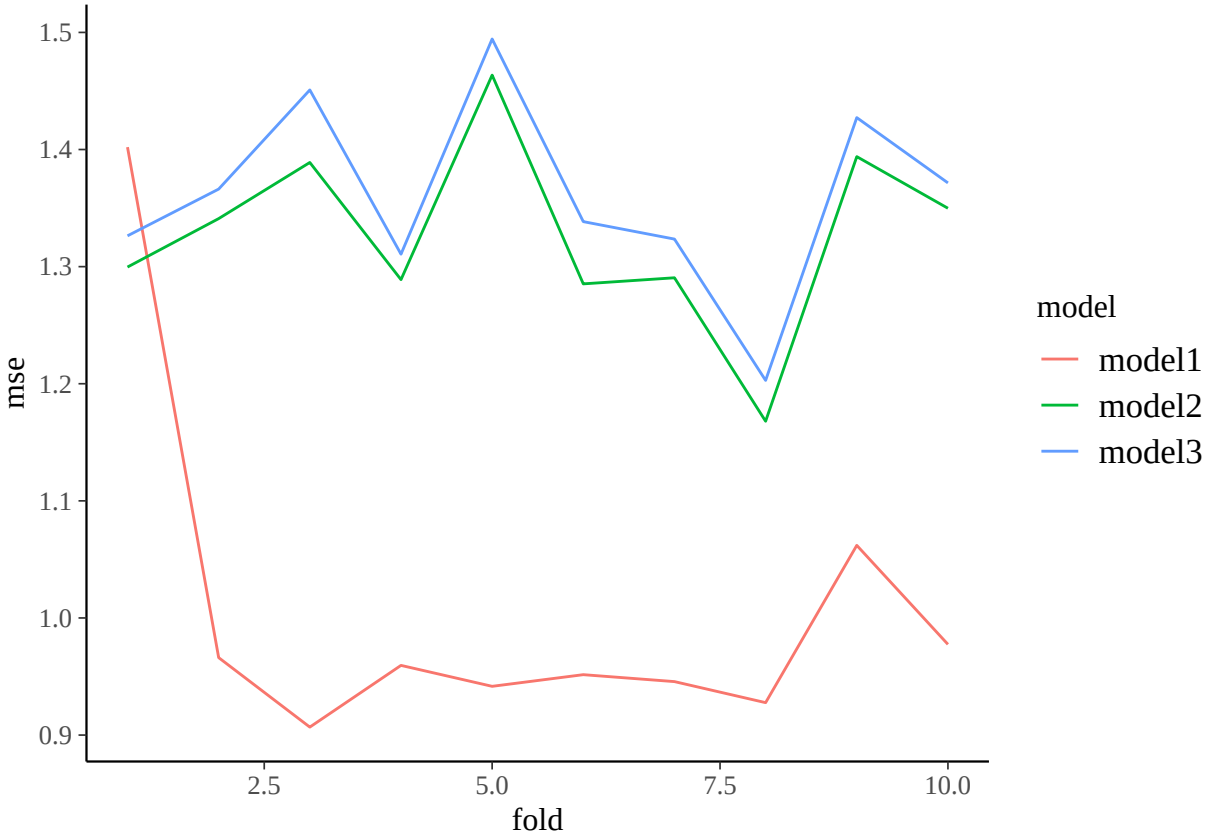
Question 3

Cross-Validation

To approach this question,we can split the data into 3 different training sets and perform a k-fold cross validation. In this case, 10-fold cross-validation is devised to evaluate the models.

```
df_valid <- data.frame(matrix(ncol = 3, nrow = 0))
x <- c("model", "fold", "mse")
colnames(df_valid) <- x

dt <- assignment_1[sample(nrow(assignment_1)),]

#Create 10 equally-sized folds
folds <- cut(seq(1,nrow(dt)),breaks=10,labels=FALSE)

for(i in 1:10){
    dt_copy <- dt
    testIndexes <- which(folds==i,arr.ind=TRUE)
    testData <- dt_copy[testIndexes, ]
    dt_copy[testIndexes, "logSAT"] <- NA

    # model1
    formula1.pred = logPM25 ~ 1 + logSAT
    result1.pred <- inla(formula1.pred, family = "gaussian", data = dt_copy, control.fixed = list(mean =
    predict_value1 <- result1.pred$summary.fitted.values$mean[1:length(testData[,1])]
    mse_model1 <- mean((testData$logSAT - predict_value1)^2)

    df_valid[nrow(df_valid) + 1,] <- list("model1",i, mse_model1)

    # model2
    formula2.pred = logPM25 ~ 1 + logSAT + f(Super_region_name_1,model="iid", hyper = prior_iid) +
f(country_code_1, model = 'iid', graph = '/Users/Claire/Desktop/world.adj', hyper = prior_iid)
    result2.pred = inla(formula2.pred, family="gaussian", data = assignment_1,control.fixed = list(mean
    predict_value2 <- result2.pred$summary.fitted.values$mean[1:length(testData[,1])]
    mse_model2 <- mean((testData$logSAT - predict_value2)^2)

    df_valid[nrow(df_valid) + 1,] <- list("model2",i, mse_model2)

    # model3
    formula3.pred <- logPM25 ~ 1 + logSAT
    result3.pred <- inla(formula3.pred, family = "gaussian", data = assignment_1, control.fixed = list(m

    predict_value3 <- result3.pred$summary.fitted.values$mean[1:length(testData[,1])]
    mse_model3 <- mean((testData$logSAT - predict_value3)^2)

    df_valid[nrow(df_valid) + 1,] <- list("model3",i, mse_model3)
}
```

```
ggplot(data = df_valid, mapping = aes(x = fold, y = mse, colour = model)) + geom_line()
```



By using Cross-Validation, the MSE of model 1, which corresponds to the baseline model, is significantly smaller than model 2&3 in the 10 folds of sampling. On the other hand, there shows a similar trend between the full model and the spatial iid model, which is consistent with the previous histograms.

Therefore, we conclude that the baseline model is relatively worse comparing to the other two models as its less stable.