University of Edinburgh

School of Mathematics

Bayesian Data Analysis, 2023/2024, Semester 2

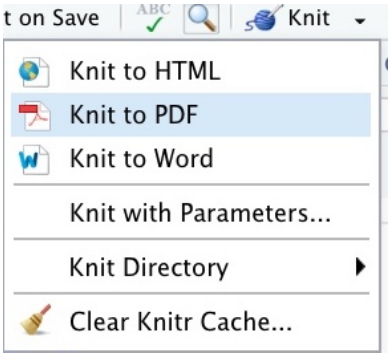
Assignment 2

IMPORTANT INFORMATION ABOUT THE ASSIGNMENT

In this paragraph, we summarize the essential information about this assignment. The format and rules for this assignment are different from your other courses, so please pay attention.

1) Deadline: The deadline for submitting your solutions to this assignment is 12 April 12:00 noon Edinburgh time.

2) Format: You will need to submit your work as 2 components: a PDF report, and your R Markdown (.Rmd) notebook (this can be in a zip file if you include additional images). There will be two separate submission systems on Learn: Gradescope for the report in PDF format, and a Learn assignment for the code in Rmd format. You need to write your solutions into this R Markdown notebook (code in R chunks and explanations in Markdown chunks), and then select Knit/Knit to PDF in RStudio to create a PDF report.



The compiled PDF needs to contain everything in this notebook, with your code sections clearly visible (not hidden), and the output of your code included. Reports without the code displayed in the PDF, or without the output of your code included in the PDF will be marked as 0, with the only feedback "Report did not meet submission requirements".

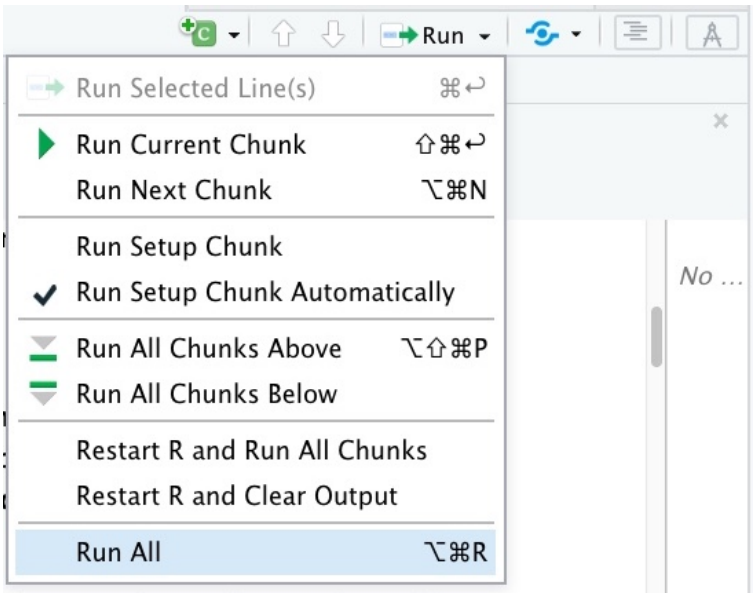
You need to upload this PDF in Gradescope submission system, and your Rmd file in the Learn assignment submission system. You will be required to tag every sub question on Gradescope.

Students who do not tag their questions will get a penalty of 10 marks deduced (out of 50).

Some key points that are different from other courses:

a) Your report needs to contain written explanation for each question that you solve, and some numbers or plots showing your results. Solutions without written explanation that clearly demonstrates that you understand what you are doing will be marked as 0 irrespectively whether the numerics are correct or not.

b) Your code has to be possible to run for all questions by the Run All in RStudio, and reproduce all of the numerics and plots in your report (up to some small randomness due to stochasticity of Monte Carlo simulations). The parts of the report that contain material that is not reproduced by the code will not be marked (i.e. the score will be 0), and the only feedback in this case will be that the results are not reproducible from the code.



c) Multiple Submissions are allowed BEFORE THE DEADLINE are allowed for both the report, and the code.
However, multiple submissions are NOT ALLOWED AFTER THE DEADLINE.
YOU WILL NOT BE ABLE TO MAKE ANY CHANGES TO YOUR SUBMISSION AFTER THE DEADLINE.
Nevertheless, if you did not submit anything before the deadline, then you can still submit your work after the deadline, but late penalties will apply. The timing of the late penalties will be determined by the time you have submitted BOTH the report, and the code (i.e. whichever was submitted later counts).

We illustrate these rules by some examples:

Alice has spent a lot of time and effort on her assignment for BDA. Unfortunately she has accidentally introduced a typo in her code in the first question, and it did not run using Run All in RStudio. - Alice will get 0 for the part of the assignments that do not run, with the only feedback "Results are not reproducible from the code".

Bob has spent a lot of time and effort on his assignment for BDA. Unfortunately he forgot to submit his code. He will get one reminder to submit his code. If he does not do it, Bob will get 0 for the whole assignment, with the only feedback "Results are not reproducible from the code, as the code was not submitted."

Charles has spent a lot of time and effort on his assignment for BDA. He has submitted both his code and report in the correct formats. However, he did not include any explanations in the report. Charles will get a 0 for the whole assignment, with the only feedback "Explanation is missing."

Denise uploaded her report to Gradescope but did not tag any questions. For this, she received a 10-mark deduction.

3) Group work: This is an INDIVIDUAL ASSIGNMENT. You can talk to your classmates to clarify questions, but you have to do your work individually and cannot copy parts from other students. Students who submit work that has not been done individually will be reported for Academic Misconduct, which can lead to severe consequences. Each question will be marked by a single instructor, and submissions will be compared by advanced software tools, so we will be able to spot students who copy.

4) Piazza: During the assignments, the instructor will change Piazza to allow messaging the

instructors only, i.e. students will not see each others messages and replies.

Only questions regarding clarification of the statement of the problems will be answered by the instructors. The instructors will not give you any information related to the solution of the problems, such questions will be simply answered as "This is not about the statement of the problem so we cannot answer your question."

THE INSTRUCTORS ARE NOT GOING TO DEBUG YOUR CODE, AND YOU ARE ASSESSED ON YOUR ABILITY TO RESOLVE ANY CODING OR TECHNICAL DIFFICULTIES THAT YOU ENCOUNTER ON YOUR OWN.

5) Office hours: There will be one office hour per week (Wednesdays 16:00-17:00) during the 2 weeks for this assignment. This is in JCMB 5413. I will be happy to discuss the course/workshop materials. However, I will only answer questions about the assignment that require clarifying the statement of the problems, and will not give you any information about the solutions.

6) Late submissions and extensions: UP TO A MAXIMUM OF 3 CALENDAR DAYS EXTENSION IS ALLOWED FOR THIS ASSIGNMENT IN THE ESC SYSTEM. You need to apply before the deadline.

If you submit your solutions on Learn before the deadline, the system will not allow you to update it even if you have received an extension. There is only 1 submission allowed after the deadline.

Students who have existing Learning Adjustments in Euclid will be allowed to have the same adjustments applied to this course as well, but they need to apply for this BEFORE THE DEADLINE on the website.

https://www.ed.ac.uk/student-administration/extensions-special-circumstances

by clicking on "Access your learning adjustment". This will be approved automatically.

Students who submit their work late will have late submission penalties applied by the ESC team automatically (this means that even if you are 1 second late because of your internet connection was slow, the penalties will still apply). The penalties are 5% of the total mark deduced for every day of delay started (i.e. one minute of delay counts for 1 day). The course instructors do not have any role in setting these penalties, we will not be able to change them.

```
rm(list = ls(all = TRUE))
#Do not delete this!
#It clears all variables to ensure reproducibility
```

```
require(INLA)
```

```
## Loading required package: INLA
```

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

```
## Warning: package 'Matrix' was built under R version 4.3.2
```

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

```
## Warning: package 'sp' was built under R version 4.3.2
```

```
## This is INLA_23.09.09 built 2023-10-16 17:29:11 UTC.
##  - See www.r-inla.org/contact-us for how to get help.
```

```
housing = read.csv("housing.csv")
# removing rows with NA's, there are only a few of these
housing = housing[complete.cases(housing), ]
# creating a new covariate
```

Figure 1: **The dataset is about the houses found in a given California district and some summary stats about them based on the 1990 census data.**

```
housing$average_bed_rooms = housing$total_bedrooms / housing$households
# add an indicator for each ocean proximity value
housing$op_near_bay = as.numeric(housing$ocean_proximity == "NEAR BAY")
housing$op_inland = as.numeric(housing$ocean_proximity == "INLAND")
housing$op_near_ocean = as.numeric(housing$ocean_proximity == "NEAR OCEAN")
housing$op_island = as.numeric(housing$ocean_proximity == "ISLAND")

head(housing)
```

```
##   longitude latitude housing_median_age total_rooms total_bedrooms population
## 1   -122.23    37.88                41         880            129        322
## 2   -122.22    37.86                21        7099           1106       2401
## 3   -122.24    37.85                52        1467            190        496
## 4   -122.25    37.85                52        1274            235        558
## 5   -122.25    37.85                52        1627            280        565
## 6   -122.25    37.85                52         919            213        413
##   households median_income median_house_value ocean_proximity average_bed_rooms
## 1        126        8.3252             452600        NEAR BAY         1.0238095
## 2       1138        8.3014             358500        NEAR BAY         0.9718805
## 3        177        7.2574             352100        NEAR BAY         1.0734463
## 4        219        5.6431             341300        NEAR BAY         1.0730594
## 5        259        3.8462             342200        NEAR BAY         1.0810811
## 6        193        4.0368             269700        NEAR BAY         1.1036269
##   op_near_bay op_inland op_near_ocean op_island
## 1           1         0             0         0
## 2           1         0             0         0
## 3           1         0             0         0
## 4           1         0             0         0
```

```
## 5           1         0             0         0
## 6           1         0             0         0
```

The covariates in the dataset are as follows:

longitude, latitude, housing_median_age (median age of houses in district), total_rooms (total rooms in all houses in district), total_bedrooms (total bedrooms in all houses in district), population (population of district), households (number of households in district), median_income (median income in district), median_house_value (median house value in district), ocean_proximity (categorical covariate about proximity of district to ocean), average_bed_rooms (average number of bedrooms of houses in district).

```
# We split the original dataset into two parts, training and test
housing.training<-housing[seq(from=1,to=nrow(housing),by=2), ]
housing.test<-housing[seq(from=2,to=nrow(housing),by=2), ]
```

Q1)[10 marks]

Fit a Bayesian Linear regression model in INLA (with Gaussian likelihood) using the housing.training dataset such that the response variable is the log(median_house_value), and the covariates in the model are as follows:

longitude, latitude, housing_median_age, log(median_income), ocean_proximity, average_bed_rooms.

Use scaled versions of the non-categorical covariates in your model.

Print out the model summary and interpret the posterior means of the regression coefficients.

Compute the DIC, NLSCPO and WAIC scores.

Check the sensitivity of your results to changing the priors.

```
# add the log / scale version of desired columns into the dataframe
housing.training$median_house_value.log = log(housing.training$median_house_value)
housing.training$longitude.std = scale(housing.training$longitude)
housing.training$latitude.std = scale(housing.training$latitude)
housing.training$housing_median_age.std = scale(housing.training$housing_median_age)
housing.training$median_income.log = log(housing.training$median_income)
housing.training$median_income.stdlog = scale(housing.training$median_income.log)
housing.training$average_bed_rooms.std = scale(housing.training$average_bed_rooms)
```

```
# define priors for sensitivity check
prec.prior1 = list(prec=list(prior="loggamma", param=c(0.1, 0.1)))
prior.beta1 = list(mean.intercept=0, prec.intercept=1e-6,
                   mean=0, prec=1e-6)

prec.prior2 = list(prec=list(prior="loggamma", param=c(0.1, 0.1)))
prior.beta2 = list(mean.intercept=10, prec.intercept=1e-4,
                   mean=0, prec=1e-4)

prec.prior3 = list(prec=list(prior="loggamma", param=c(0.1, 0.1)))
prior.beta3 = list(mean.intercept=100, prec.intercept=1e-6,
                   mean=100, prec=1e-6)

prec.prior4 = list(prec=list(prior="loggamma", param=c(1, 1)))
prior.beta4 = list(mean.intercept=0, prec.intercept=1e-6,
                   mean=0, prec=1e-6)
```

```r
# function of fitting & summaries for the fixed-effect-only Bayesian linear regression
#   model.
#   data: data for model fitting; prec.prior: prior for Gaussian precision;
#   prior.beta: prior for fixed effects;
#   sense.check: indicator of whether this run should print no summary results
bayes.ord = function(data, prec.prior, prior.beta, sense.check=F, ...) {
  # model fitting with inla
  cali.ord = inla(median_house_value.log ~ longitude.std + latitude.std +
                    housing_median_age.std + median_income.stdlog +
                    ocean_proximity + average_bed_rooms.std, data=data,
                  family="gaussian", control.fixed=prior.beta,
                  control.family=list(hyper=prec.prior),
                  control.compute=list(config=T, dic=T, cpo=T, waic=T), ...)

  # print summary results if not for sensitivity check
  if (!sense.check) {
    print(summary(cali.ord))
    print(as.data.frame(t(cbind(cali.ord$summary.fixed[, "mean", drop=F],
                                cali.ord$summary.fixed[, "sd", drop=F]))))

    cat("DIC of model 1:\t\t"); print(cali.ord$dic$dic)
    cat("NLSCPO of model 1:\t"); print(-sum(log(cali.ord$cpo$cpo)))
    cat("WAIC of model 1:\t"); print(cali.ord$waic$waic)
  }

  # return model, posterior means of fixed effects, and model scores
  list(model=cali.ord,
       post=as.data.frame(t(cali.ord$summary.fixed[, "mean", drop=F])),
       metrics=data.frame(DIC=cali.ord$dic$dic, NLSCPO=-sum(log(cali.ord$cpo$cpo)),
                          WAIC=cali.ord$waic$waic))
}

# implement model 1 on the training data based on different priors
res.ord1 = bayes.ord(housing.training, prec.prior1, prior.beta1)
```

```
##
## Call:
##    c("inla.core(formula = formula, family = family, contrasts = contrasts,
##    ", " data = data, quantiles = quantiles, E = E, offset = offset, ", "
##    scale = scale, weights = weights, Ntrials = Ntrials, strata = strata,
##    ", " lp.scale = lp.scale, link.covariates = link.covariates, verbose =
##    verbose, ", " lincomb = lincomb, selection = selection, control.compute
##    = control.compute, ", " control.predictor = control.predictor,
##    control.family = control.family, ", " control.inla = control.inla,
##    control.fixed = control.fixed, ", " control.mode = control.mode,
##    control.expert = control.expert, ", " control.hazard = control.hazard,
##    control.lincomb = control.lincomb, ", " control.update =
##    control.update, control.lp.scale = control.lp.scale, ", "
##    control.pardiso = control.pardiso, only.hyperparam = only.hyperparam,
##    ", " inla.call = inla.call, inla.arg = inla.arg, num.threads =
##    num.threads, ", " keep = keep, working.directory = working.directory,
##    silent = silent, ", " inla.mode = inla.mode, safe = FALSE, debug =
##    debug, .parent.frame = .parent.frame)" )
## Time used:
```

```
##     Pre = 0.606, Running = 1.62, Post = 0.309, Total = 2.54
## Fixed effects:
##                               mean    sd 0.025quant 0.5quant 0.975quant   mode
## (Intercept)                 12.185 0.006     12.174   12.185     12.197 12.185
## longitude.std               -0.313 0.014     -0.341   -0.313     -0.285 -0.313
## latitude.std                -0.324 0.015     -0.354   -0.324     -0.295 -0.324
## housing_median_age.std       0.028 0.004      0.020    0.028      0.035  0.028
## median_income.stdlog         0.322 0.004      0.315    0.322      0.329  0.322
## ocean_proximityINLAND       -0.310 0.012     -0.334   -0.310     -0.286 -0.310
## ocean_proximityISLAND        0.710 0.241      0.237    0.710      1.183  0.710
## ocean_proximityNEAR BAY     -0.005 0.013     -0.031   -0.005      0.021 -0.005
## ocean_proximityNEAR OCEAN   -0.014 0.011     -0.036   -0.014      0.007 -0.014
## average_bed_rooms.std        0.033 0.003      0.026    0.033      0.039  0.033
##                             kld
## (Intercept)                   0
## longitude.std                 0
## latitude.std                  0
## housing_median_age.std        0
## median_income.stdlog          0
## ocean_proximityINLAND         0
## ocean_proximityISLAND         0
## ocean_proximityNEAR BAY       0
## ocean_proximityNEAR OCEAN     0
## average_bed_rooms.std         0
##
## Model hyperparameters:
##                                         mean   sd 0.025quant 0.5quant
## Precision for the Gaussian observations 8.61 0.12       8.37     8.61
##                                         0.975quant mode
## Precision for the Gaussian observations       8.85 8.61
##
## Deviance Information Criterion (DIC) ...............: 7001.29
## Deviance Information Criterion (DIC, saturated) ....: 10216.64
## Effective number of parameters .....................: 6.07
##
## Watanabe-Akaike information criterion (WAIC) ...: 7031.83
## Effective number of parameters .................: 26.92
##
## Marginal log-Likelihood:  -3617.92
## CPO, PIT is computed
## Posterior summaries for the linear predictor and the fitted values are computed
## (Posterior marginals needs also 'control.compute=list(return.marginals.predictor=TRUE)')
##
##       (Intercept) longitude.std latitude.std housing_median_age.std
## mean 12.185095068   -0.31281905  -0.32430830            0.027554185
## sd    0.005867404    0.01433607   0.01505846            0.003659134
##      median_income.stdlog ocean_proximityINLAND ocean_proximityISLAND
## mean          0.321873358          -0.31005345             0.7097759
## sd            0.003603441           0.01219408             0.2412581
##      ocean_proximityNEAR BAY ocean_proximityNEAR OCEAN average_bed_rooms.std
## mean            -0.005295707               -0.01421109           0.032531702
## sd               0.013320786                0.01093346           0.003460332
## DIC of model 1:      [1] 7001.293
## NLSCPO of model 1:   [1] 3512.789
```

```
## WAIC of model 1: [1] 7031.828
```

```
res.ord2 = bayes.ord(housing.training, prec.prior2, prior.beta2, T)
res.ord3 = bayes.ord(housing.training, prec.prior3, prior.beta3, T)
res.ord4 = bayes.ord(housing.training, prec.prior4, prior.beta4, T)

# print posterior means of fixed effects and model scores for different implementations
posts = rbind(res.ord1[[2]], res.ord2[[2]], res.ord3[[2]], res.ord4[[2]])
metrics = rbind(res.ord1[[3]], res.ord2[[3]], res.ord3[[3]], res.ord4[[3]])
print(posts)
```

```
##       (Intercept) longitude.std latitude.std housing_median_age.std
## mean      12.1851    -0.3128191   -0.3243083             0.02755418
## mean1     12.1851    -0.3128191   -0.3243083             0.02755419
## mean2     12.1851    -0.3128191   -0.3243083             0.02755418
## mean3     12.1851    -0.3128191   -0.3243083             0.02755418
##       median_income.stdlog ocean_proximityINLAND ocean_proximityISLAND
## mean             0.3218734            -0.3100535             0.7097759
## mean1            0.3218734            -0.3100534             0.7097719
## mean2            0.3218734            -0.3100535             0.7097759
## mean3            0.3218734            -0.3100535             0.7097759
##       ocean_proximityNEAR BAY ocean_proximityNEAR OCEAN average_bed_rooms.std
## mean              -0.005295707               -0.01421109             0.0325317
## mean1             -0.005295667               -0.01421107             0.0325317
## mean2             -0.005295707               -0.01421109             0.0325317
## mean3             -0.005295707               -0.01421109             0.0325317
```

```
print(metrics)
```

```
##        DIC   NLSCPO     WAIC
## 1 7001.293 3512.789 7031.828
## 2 7001.293 3512.789 7031.828
## 3 7001.293 3512.789 7031.828
## 4 7001.326 3512.802 7031.807
```

Explanation: (Write your explanation here)

Note that all the result numbers recorded in the explanation are from a single "Run All" of the notebook. Since the knitting process will rerun the whole notebook again, the values in the explanation may differ from the output displayed (up to some small randomness).

In this question, we fit an ordinary Bayesian Linear Regression model over the California housing training dataset using `INLA`, with the log of `median_house_value` as the response value, and `longitude`, `latitude`, `housing_median_age`, log of `median_income`, `ocean_proximity`, and `average_bed_rooms` as the covariates. Among these variables, all continuous covariates are scaled and the corresponding log & scale versions of them are stored in the data frame as new columns with the suffix `.log` or `.std` in their names as the indicator. One-hot encoder for `ocean_proximity` is created for later model usage. Summaries for the model, posterior means for the fixed effects and the three model scores are explicitly printed out.

From the results we can see that the posterior means for the fixed effects are $12.185(0.006)$, $-0.313(0.014)$, $-0.324(0.015)$, $0.028(0.004)$, $0.322(0.004)$, $-0.310(0.012)$, $0.710(0.241)$, $-0.005(0.013)$, $-0.014(0.011)$, and $0.033(0.003)$, correspondingly (standard deviations, SDs, attached), and except `ocean_proximityNEAR BAY`, the posterior SDs are reasonably small relative to their means and 0 is excluded in all $\mu \pm \sigma$ intervals, which indicate reasonably accurate estimates and directions of correlation. We can have some initial insights:

- The more **average bedrooms** of houses, the higher the **median income**, and the higher the **median age of houses** in a district, the higher the median house values in the district will be.

- Generally, the districts in locations with lower **longitudes** and **latitudes** have higher median house values.
- For **ocean proximity**, districts less than 1 hour away from the ocean (`<1H OCEAN`, explained by `Intercept`) and on islands (`ISLAND`) generally have higher median house prices than others.

However, the model has relatively high **DIC**, **NLSCPO**, and **WAIC** scores (7001.29, 3512.79, and 7031.83), which synthetically evaluates the data's likelihood under the model and the model's complexity (the lower the better). This implies further possible improvements.

Regarding the **sensitivity check**, we pack the whole fit-plus-summary process in a function `bayes.ord` and apply different priors of fixed effects and the Gaussian precision. The results demonstrated an almost identical posterior means of fixed effects and model scores, which implies that the model is not very sensitive to prior changes.

Q2)[10 marks]

Update your model in Q1 to also include an rw1 random effect model for the housing_median_age, and an ar1 random effect model for log(median_income).

Print out the model summary and interpret the posterior means of the regression coefficients.

Plot the posterior means of the random effects for housing_median_age and log(median_income). The x-axis should be the covariate value (such as housing_median_age), and the y-axis should be the posterior mean of the random effect.

Compute the DIC, NLSCPO and WAIC scores.

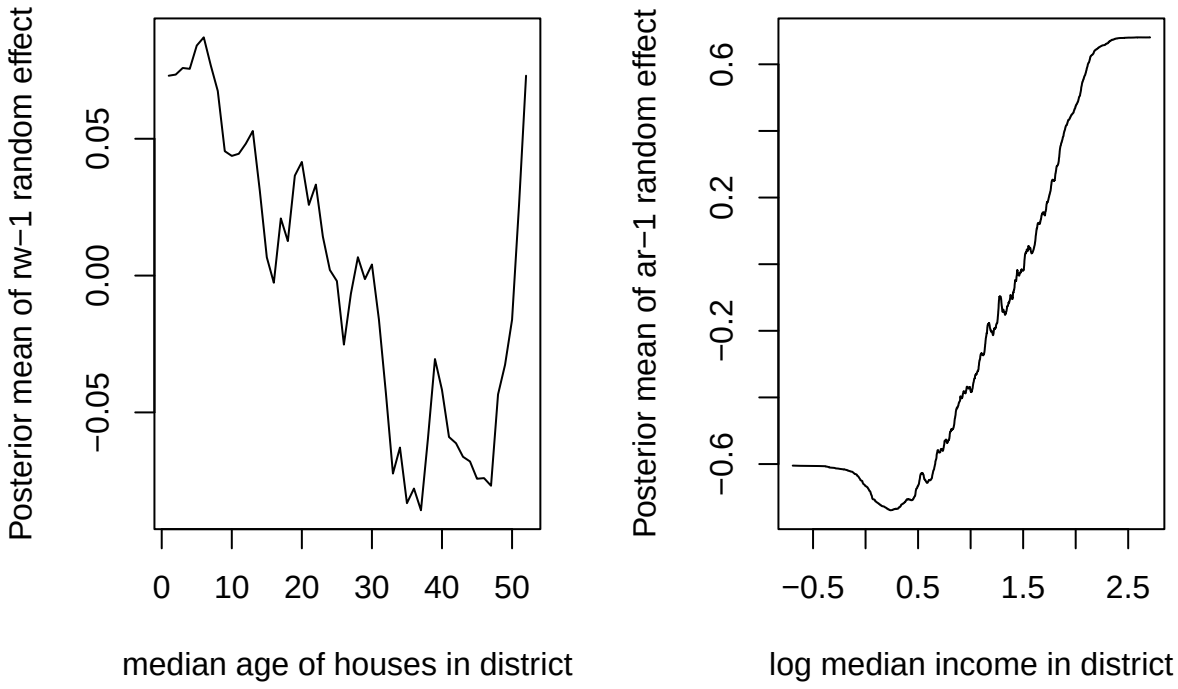
Check the sensitivity of your results to changing the priors.

```
# function of fitting & summaries for the Bayesian linear regression model with
#   2 additional random effects.
#   data: data for model fitting; prec.prior: prior for Gaussian precision;
#   prior.beta: prior for fixed effects;
#   sense.check: indicator of whether this run should print no summary results
bayes.rwar = function(data, prec.prior, prior.beta, sense.check=F, ...) {
  cali.rwar = inla(median_house_value.log ~ longitude.std + latitude.std +
                     housing_median_age.std + median_income.stdlog +
                     ocean_proximity + average_bed_rooms.std +
                     f(housing_median_age, model="rw1") +
                     f(median_income.log, model="ar1"),
                   data=data, family="gaussian",
                   control.fixed=prior.beta, control.family=list(hyper=prec.prior),
                   control.compute=list(config=T, dic=T, cpo=T, waic=T), ...)

  # print summary results if not for sensitivity check
  if (!sense.check) {
    print(summary(cali.rwar))
    print(as.data.frame(t(cbind(cali.rwar$summary.fixed[, "mean", drop=F],
                                cali.rwar$summary.fixed[, "sd", drop=F]))))

    cat("DIC of model 2:\t\t"); print(cali.rwar$dic$dic)
    cat("NLSCPO of model 2:\t"); print(-sum(log(cali.rwar$cpo$cpo)))
    cat("WAIC of model 2:\t"); print(cali.rwar$waic$waic)

    par(mfrow=c(1, 2))
    # plot the random effects versus the corresponding covariate values
    plot(cali.rwar$summary.random$housing_median_age[, c("ID", "mean")], type="l",
         xlab="median age of houses in district",
```

```
        ylab="Posterior mean of rw-1 random effect")
    plot(cali.rwar$summary.random$median_income.log[, c("ID", "mean")], type="l",
        xlab="log median income in district",
        ylab="Posterior mean of ar-1 random effect")
  }

  # return model, posterior means of fixed effects, and model scores
  list(model=cali.rwar,
       post=as.data.frame(t(cali.rwar$summary.fixed[, "mean", drop=F])),
       metrics=data.frame(DIC=cali.rwar$dic$dic, NLSCPO=-sum(log(cali.rwar$cpo$cpo)),
                          WAIC=cali.rwar$waic$waic))
}

# implement model 2 on the training data based on different priors
res.rwar1 = bayes.rwar(housing.training, prec.prior1, prior.beta1)
```

```
##
## Call:
##    c("inla.core(formula = formula, family = family, contrasts = contrasts,
##    ", " data = data, quantiles = quantiles, E = E, offset = offset, ", "
##    scale = scale, weights = weights, Ntrials = Ntrials, strata = strata,
##    ", " lp.scale = lp.scale, link.covariates = link.covariates, verbose =
##    verbose, ", " lincomb = lincomb, selection = selection, control.compute
##    = control.compute, ", " control.predictor = control.predictor,
##    control.family = control.family, ", " control.inla = control.inla,
##    control.fixed = control.fixed, ", " control.mode = control.mode,
##    control.expert = control.expert, ", " control.hazard = control.hazard,
##    control.lincomb = control.lincomb, ", " control.update =
##    control.update, control.lp.scale = control.lp.scale, ", "
##    control.pardiso = control.pardiso, only.hyperparam = only.hyperparam,
##    ", " inla.call = inla.call, inla.arg = inla.arg, num.threads =
##    num.threads, ", " keep = keep, working.directory = working.directory,
##    silent = silent, ", " inla.mode = inla.mode, safe = FALSE, debug =
##    debug, .parent.frame = .parent.frame)" )
## Time used:
##     Pre = 0.56, Running = 14.4, Post = 1.09, Total = 16.1
## Fixed effects:
##                               mean    sd 0.025quant 0.5quant 0.975quant   mode
## (Intercept)                 12.355 0.305     11.756   12.355     12.955 12.355
## longitude.std               -0.295 0.014     -0.322   -0.295     -0.268 -0.295
## latitude.std                -0.307 0.015     -0.336   -0.307     -0.279 -0.307
## housing_median_age.std       0.060 0.048     -0.034    0.060      0.154  0.060
## median_income.stdlog         0.000 0.037     -0.075    0.001      0.072  0.000
## ocean_proximityINLAND       -0.318 0.012     -0.341   -0.318     -0.295 -0.318
## ocean_proximityISLAND        0.625 0.234      0.167    0.625      1.083  0.625
## ocean_proximityNEAR BAY     -0.039 0.013     -0.065   -0.039     -0.014 -0.039
## ocean_proximityNEAR OCEAN   -0.013 0.011     -0.033   -0.013      0.008 -0.013
## average_bed_rooms.std        0.030 0.003      0.023    0.030      0.036  0.030
##                               kld
## (Intercept)                     0
## longitude.std                   0
## latitude.std                    0
## housing_median_age.std          0
## median_income.stdlog            0
```

```
## ocean_proximityINLAND       0
## ocean_proximityISLAND       0
## ocean_proximityNEAR BAY     0
## ocean_proximityNEAR OCEAN   0
## average_bed_rooms.std       0
##
## Random effects:
##   Name     Model
##     housing_median_age RW1 model
##    median_income.log AR1 model
##
## Model hyperparameters:
##                                          mean      sd 0.025quant 0.5quant
## Precision for the Gaussian observations    9.26   0.131      9.008     9.26
## Precision for housing_median_age        1666.38 662.208    735.894  1546.42
## Precision for median_income.log           33.20 308.974      0.000     1.08
## Rho for median_income.log                  1.00   0.003      0.997     1.00
##                                         0.975quant    mode
## Precision for the Gaussian observations       9.52    9.26
## Precision for housing_median_age           3298.13 1332.56
## Precision for median_income.log             244.97    0.00
## Rho for median_income.log                     1.00    1.00
##
## Deviance Information Criterion (DIC) ...............: 6339.42
## Deviance Information Criterion (DIC, saturated) ....: 10305.60
## Effective number of parameters .....................: 89.02
##
## Watanabe-Akaike information criterion (WAIC) ...: 6360.19
## Effective number of parameters .................: 104.00
##
## Marginal log-Likelihood:  -3337.98
## CPO, PIT is computed
## Posterior summaries for the linear predictor and the fitted values are computed
## (Posterior marginals needs also 'control.compute=list(return.marginals.predictor=TRUE)')
##
##      (Intercept) longitude.std latitude.std housing_median_age.std
## mean  12.3554914    -0.2949031  -0.30741324             0.05987222
## sd     0.3054934     0.0139409   0.01465017             0.04759345
##      median_income.stdlog ocean_proximityINLAND ocean_proximityISLAND
## mean        -2.687591e-05           -0.31778801             0.6252164
## sd           3.749667e-02            0.01183179             0.2336034
##      ocean_proximityNEAR BAY ocean_proximityNEAR OCEAN average_bed_rooms.std
## mean             -0.03943915               -0.01264716           0.029521134
## sd                0.01317178                0.01059847           0.003350844
## DIC of model 2:      [1] 6339.417
## NLSCPO of model 2:   [1] 3179.523
## WAIC of model 2: [1] 6360.19
```




```
res.rwar2 = bayes.rwar(housing.training, prec.prior2, prior.beta2, T)
res.rwar3 = bayes.rwar(housing.training, prec.prior3, prior.beta3, T)
res.rwar4 = bayes.rwar(housing.training, prec.prior4, prior.beta4, T)

# print posterior means of fixed effects and model scores for different implementations
posts = rbind(res.rwar1[[2]], res.rwar2[[2]], res.rwar3[[2]], res.rwar4[[2]])
metrics = rbind(res.rwar1[[3]], res.rwar2[[3]], res.rwar3[[3]], res.rwar4[[3]])
print(posts)
```

```
##       (Intercept) longitude.std latitude.std housing_median_age.std
## mean     12.35549    -0.2949031   -0.3074132             0.05987222
## mean1    12.20126    -0.3118024   -0.3253474             0.05620784
## mean2    12.20152    -0.3111488   -0.3246543             0.05605194
## mean3    12.34799    -0.2949817   -0.3074895             0.05992023
##       median_income.stdlog ocean_proximityINLAND ocean_proximityISLAND
## mean         -2.687591e-05            -0.3177880             0.6252164
## mean1         3.208848e-01            -0.3179490             0.5603624
## mean2         3.213176e-01            -0.3180265             0.5620734
## mean3         8.668204e-03            -0.3177972             0.6249261
##       ocean_proximityNEAR BAY ocean_proximityNEAR OCEAN average_bed_rooms.std
## mean              -0.03943915              -0.01264716            0.02952113
## mean1             -0.03765890              -0.01860270            0.03162582
## mean2             -0.03740053              -0.01812658            0.03143108
## mean3             -0.03946854              -0.01269476            0.02954193
```

```
print(metrics)
```

```
##        DIC   NLSCPO     WAIC
## 1 6339.417 3179.523 6360.190
## 2 6188.662 3270.052 6542.391
## 3 6787.411 3403.259 6808.891
## 4 6340.757 3180.266 6361.711
```

Explanation: (Write your explanation here)

In this question, we add a random walk (rw1) temporal random effect for `housing_median_age` and a 1st-order autoregressive (ar1) effect for `median_income.log` to the model before.

The posterior means for the fixed effect are 12.346(0.310), −0.295(0.014), −0.307(0.015), 0.060(0.048), 0.010(0.040), −0.318(0.012), 0.625(0.234), −0.039(0.013), −0.013(0.011), and 0.030(0.003), correspondingly, with reasonably small SDs and 0 excluded in $\mu \pm \sigma$ interval for most variables, which demonstrate similar results as the first model. The interpretations are also the same as question 1. For the random effects, we plot them versus their corresponding covariate values, and the plots demonstrate that apart from the fixed effect for the two covariates:

- The **rw1 temporal effect** for the median age of houses in a district generally decreases as the age of houses increases, particularly between 0 and 40 years, but changes into an increasing trend after 40 years.
- The **ar1 temporal effect** for the log of median income in a district generally decreases as the log of median income increases before 0.5, increases between 0.5 and 2.2, and stays the same afterwards.

The **DIC**, **NLSCPO**, and **WAIC** values reduce to 6340.89, 3180.42, and 6362.04, which demonstrates improvements over the first model. Given the complexity of this model is higher, the likelihood of data under this model is much higher than under the first model.

Regarding **sensitivity check**, we pack the whole fit-plus-summary process in a function `bayes.rwar` and try the same set of priors as before. The results demonstrate slight but tolerable differences in posterior means of fixed effects and model scores. Specifically, more informative priors (prior3 and prior4) with non-zero means lead to bigger differences in these results, implying that the model sensitivity to prior changes is higher than the first model with random effects included.

Q3)[10 marks]

In this question, we will use a spatial random effects model for the location.

Create a Bayesian regression model in INLA or inlabru with Gaussian likelihood using the housing.training dataset with log(median_house_value) as the response variable, and the fixed effects in the model are as follows:

longitude, latitude,

housing_median_age, $(housing\_median\_age)^2$**,**$(housing\_median\_age)^3$,$(housing\_median\_age)^4$

log(median_income), $(\log(median\_income))^2$**,** $(\log(median\_income))^3$, $(\log(median\_income))^4$ **,**

housing_median_age*log(median_income),

ocean_proximity, average_bed_rooms.

Use scaled versions of the non-categorical covariates in your model.

Include a spatial (spde2) random effect for the location (longitude, latitude), with Matern covariance. [Hint: You must create a mesh first; see the code for Lecture 7 and the solutions of Workshop 5.]

Print out the model summary and interpret the posterior means of the regression coefficients.

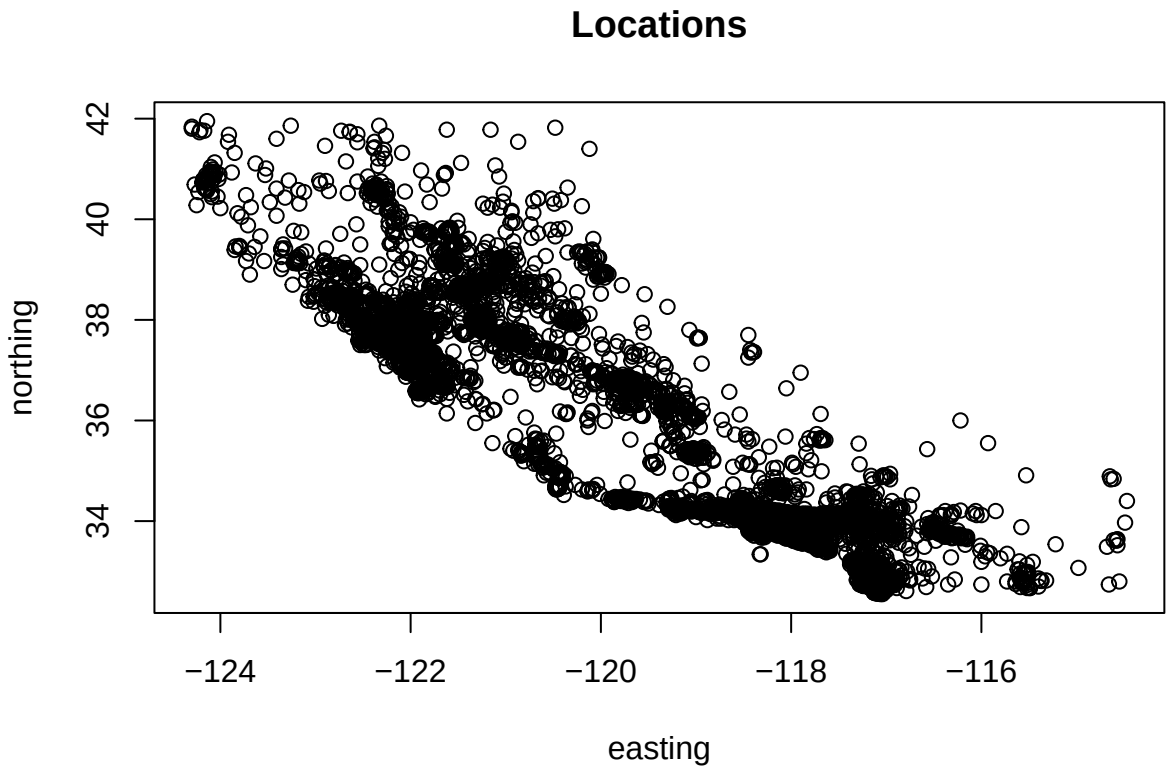
Plot the posterior mean of the spatial random effect in terms of the location.

Compute the DIC, NLSCPO and WAIC scores.

Compare the models in Q1) - Q3) in terms of DIC, NLSCPO and WAIC scores.

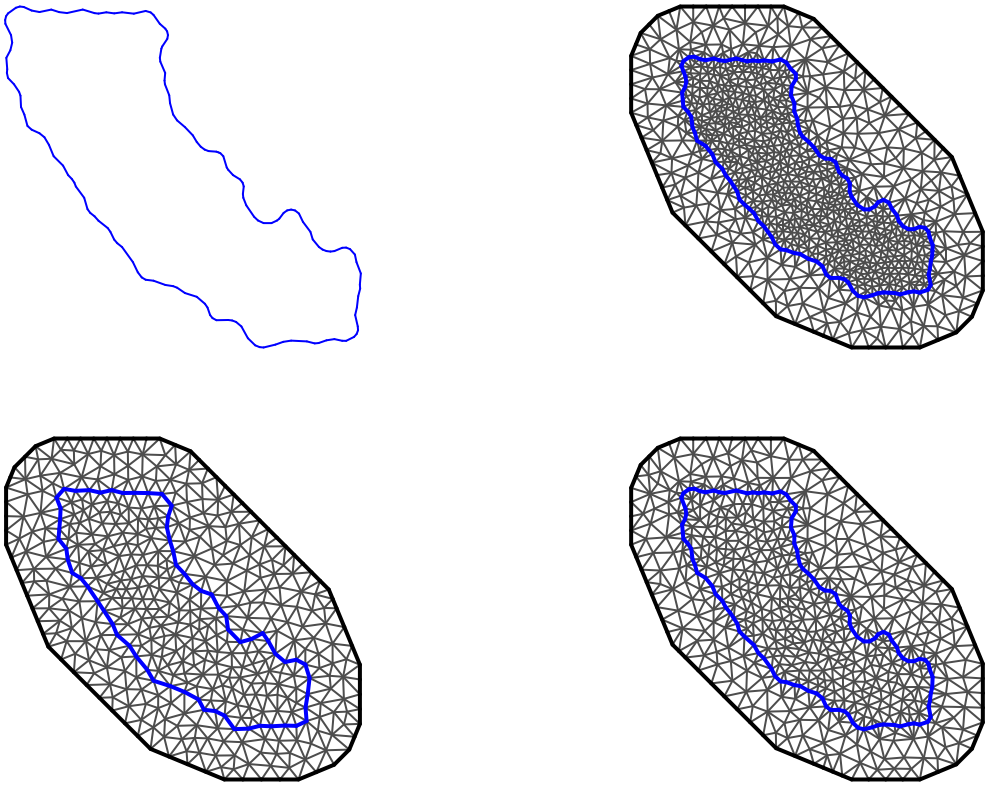
Check the sensitivity of your results to changing the priors and using a finer mesh.

```
# set Location data frame based on longitudes and latitudes & plot overview
Locations = data.frame(easting=housing.training$longitude,
                       northing=housing.training$latitude)
plot(Locations, main="Locations")
```




```
# transform Locations to sf objects
housing.training$geometry = sf::st_as_sf(Locations,
                                         coords=c("easting", "northing"))$geometry

# generate the domain based on Locations
hsdomain = inla.nonconvex.hull(
  as.matrix(Locations),
  convex=-0.03, concave=-0.05, resolution=c(100, 100))

# creating different meshes for sensitivity check
hsmesh1 = inla.mesh.2d(boundary=hsdomain, max.edge=c(0.45, 1), cutoff=0.2)
hsmesh2 = inla.mesh.2d(boundary=hsdomain, max.edge=c(0.45, 1), cutoff=0.4)
hsmesh3 = inla.mesh.2d(boundary=hsdomain, max.edge=c(0.65, 1), cutoff=0.2)

par(mfrow=c(2, 2))
par(mar=c(1, 1, 1, 1))
```

```
plot(hsdomain)
plot(hsmesh1)
plot(hsmesh2)
plot(hsmesh3)
```



```
require(inlabru)
```

```
## Loading required package: inlabru
```

```
## Warning: package 'inlabru' was built under R version 4.3.3
```

```
## Loading required package: fmesher
```

```
## Warning: package 'fmesher' was built under R version 4.3.2
```

```
require(ggplot2)
```

```
## Loading required package: ggplot2
```

```
## Warning: package 'ggplot2' was built under R version 4.3.2
```

```
require(RColorBrewer)
```

```
## Loading required package: RColorBrewer
```

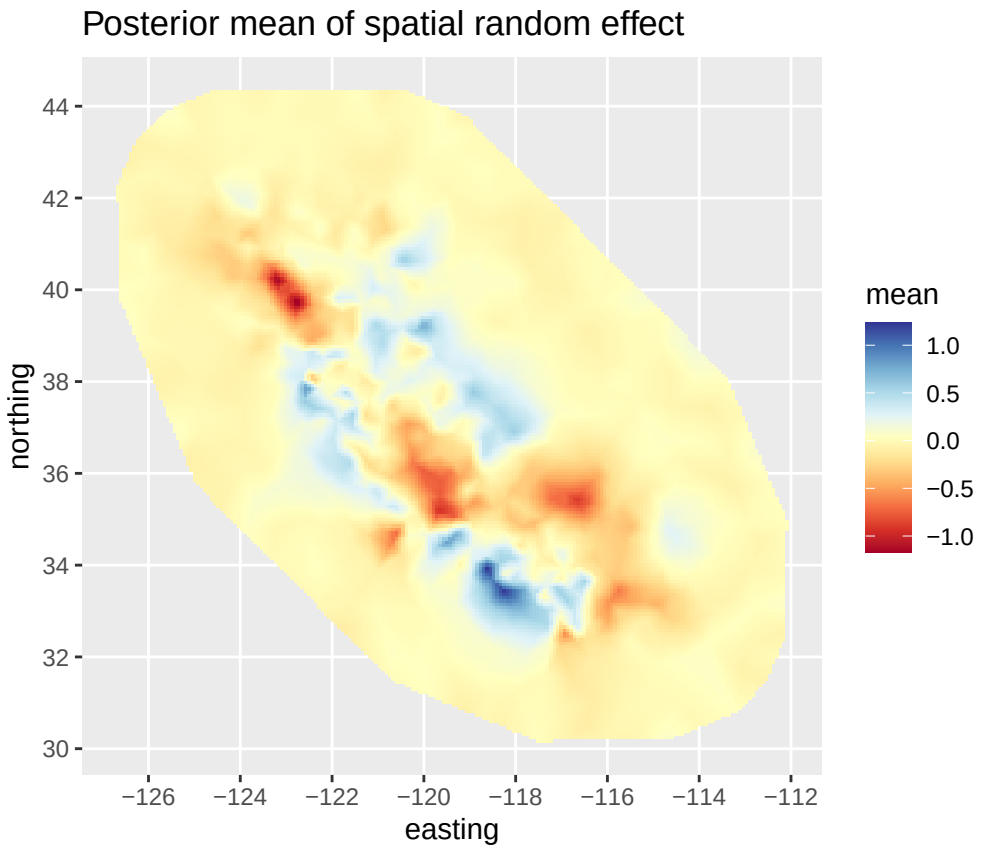
```
# function of fitting & summaries for the Bayesian linear regression model with
#   spatial random effects.
#   data: data for model fitting; hsdomain: domain for the Location values;
#   hsmesh: the mesh objects for the spatial model;
#   prec.prior: prior for Gaussian precision;
```

```
#   prior.beta: prior for fixed effects;
#   prior.range, prior.sigma: hyperparameters for the spatial model:
#   sense.check: indicator of whether this run should print no summary results
bayes.spde = function(data, hsdomain, hsmesh, prior.beta, prec.prior,
                      prior.range=c(1, 0.1), prior.sigma=c(10, 0.1),
                      sense.check=F, ...) {
  # create the spatial model object, using a PC prior for the parameters
  loc.spde = inla.spde2.pcmatern(mesh=hsmesh,
                                 prior.range=prior.range, prior.sigma=prior.sigma)

  # define the model formula & fit with inlabru
  cmp = median_house_value.log ~ floc(geometry, model=loc.spde) +
    longitude.std + latitude.std +
    fprod(housing_median_age.std * median_income.stdlog) +
    housing_median_age.std + fhma2(housing_median_age.std ^ 2) +
    fhma3(housing_median_age.std ^ 3) + fhma4(housing_median_age.std ^ 4) +
    median_income.stdlog + fmi2(median_income.stdlog ^ 2) +
    fmi3(median_income.stdlog ^ 3) + fmi4(median_income.stdlog ^ 4) +
    op_near_bay + op_inland + op_near_ocean + op_island +
    average_bed_rooms.std + Intercept(1)

  cali.spde = bru(components=cmp, data=data, family="gaussian",
                  samplers=hsdomain, domain = list(coordinates=hsmesh),
                  options=list(control.inla=list(tolerance=1e-10),
                               control.fixed=prior.beta,
                               control.family=list(hyper=prec.prior),
                               control.compute=list(config=T, cpo=T, dic=T, waic=T), ...))

  # print summary results if not for sensitivity check
  if (!sense.check) {
    print(summary(cali.spde))
    print(as.data.frame(t(cbind(cali.spde$summary.fixed[, "mean", drop=F],
                                cali.spde$summary.fixed[, "sd", drop=F]))))

    # predict based on pixels & plot the random effects versus the corresponding locations
    pix = fm_pixels(hsmesh, dims=c(200, 200))
    pred = predict(cali.spde, newdata=pix, formula=~ floc)
    plot.post.re = ggplot() +
      gg(pred, geom="tile") +
      ggtitle("Posterior mean of spatial random effect") +
      xlab("easting") + ylab("northing") +
      scale_fill_gradientn(colours=brewer.pal(11, "RdYlBu"), limits=range(pred$mean))
    print(plot.post.re)

    cat("DIC of model 3:\t\t"); print(cali.spde$dic$dic)
    cat("NLSCPO of model 3:\t"); print(-sum(log(cali.spde$cpo$cpo)))
    cat("WAIC of model 3:\t"); print(cali.spde$waic$waic)
  }

  # return model, posterior means of fixed effects, and model scores
  list(model=cali.spde,
       post=as.data.frame(t(cali.spde$summary.fixed[, "mean", drop=F])),
       metrics=data.frame(DIC=cali.spde$dic$dic, NLSCPO=-sum(log(cali.spde$cpo$cpo)),
```

```
                          WAIC=cali.spde$waic$waic))
}

# implement model 3 on the training data based on different meshes & priors
res.spde1 = bayes.spde(housing.training, hsdomain, hsmesh1, prior.beta1, prec.prior1)
```

```
## inlabru version: 2.10.1
## INLA version: 23.09.09
## Components:
## floc: main = spde(geometry), group = exchangeable(1L), replicate = iid(1L)
## longitude.std: main = linear(longitude.std), group = exchangeable(1L), replicate = iid(1L)
## latitude.std: main = linear(latitude.std), group = exchangeable(1L), replicate = iid(1L)
## fprod: main = linear(housing_median_age.std * median_income.stdlog), group = exchangeable(1L), replic
## housing_median_age.std: main = linear(housing_median_age.std), group = exchangeable(1L), replicate =
## fhma2: main = linear(housing_median_age.std^2), group = exchangeable(1L), replicate = iid(1L)
## fhma3: main = linear(housing_median_age.std^3), group = exchangeable(1L), replicate = iid(1L)
## fhma4: main = linear(housing_median_age.std^4), group = exchangeable(1L), replicate = iid(1L)
## median_income.stdlog: main = linear(median_income.stdlog), group = exchangeable(1L), replicate = iid
## fmi2: main = linear(median_income.stdlog^2), group = exchangeable(1L), replicate = iid(1L)
## fmi3: main = linear(median_income.stdlog^3), group = exchangeable(1L), replicate = iid(1L)
## fmi4: main = linear(median_income.stdlog^4), group = exchangeable(1L), replicate = iid(1L)
## op_near_bay: main = linear(op_near_bay), group = exchangeable(1L), replicate = iid(1L)
## op_inland: main = linear(op_inland), group = exchangeable(1L), replicate = iid(1L)
## op_near_ocean: main = linear(op_near_ocean), group = exchangeable(1L), replicate = iid(1L)
## op_island: main = linear(op_island), group = exchangeable(1L), replicate = iid(1L)
## average_bed_rooms.std: main = linear(average_bed_rooms.std), group = exchangeable(1L), replicate = i
## Intercept: main = linear(1), group = exchangeable(1L), replicate = iid(1L)
## Likelihoods:
##   Family: 'gaussian'
##     Data class: 'data.frame'
##     Predictor: median_house_value.log ~ .
## Time used:
##     Pre = 1.48, Running = 17.4, Post = 0.305, Total = 19.2
## Fixed effects:
##                            mean    sd 0.025quant 0.5quant 0.975quant   mode kld
## longitude.std            -0.362 0.072     -0.502   -0.362     -0.219 -0.362   0
## latitude.std             -0.367 0.073     -0.511   -0.368     -0.222 -0.368   0
## fprod                     0.020 0.003      0.015    0.020      0.026  0.020   0
## housing_median_age.std -0.060 0.006     -0.073   -0.060     -0.048 -0.060   0
## fhma2                    -0.044 0.008     -0.060   -0.044     -0.027 -0.044   0
## fhma3                     0.019 0.002      0.015    0.019      0.024  0.019   0
## fhma4                     0.018 0.002      0.014    0.018      0.023  0.018   0
## median_income.stdlog      0.323 0.005      0.314    0.323      0.332  0.323   0
## fmi2                      0.040 0.004      0.033    0.040      0.047  0.040   0
## fmi3                     -0.015 0.001     -0.017   -0.015     -0.013 -0.015   0
## fmi4                     -0.002 0.000     -0.002   -0.002     -0.001 -0.002   0
## op_near_bay               0.088 0.022      0.044    0.088      0.131  0.088   0
## op_inland                 0.022 0.021     -0.019    0.022      0.064  0.022   0
## op_near_ocean             0.077 0.016      0.046    0.077      0.108  0.077   0
## op_island                 0.003 0.283     -0.554    0.004      0.554  0.004   0
## average_bed_rooms.std     0.016 0.003      0.009    0.016      0.022  0.016   0
## Intercept                11.821 0.078     11.667   11.821     11.975 11.821   0
##
## Random effects:
```

```
##   Name     Model
##     floc SPDE2 model
##
## Model hyperparameters:
##                                          mean    sd 0.025quant 0.5quant
## Precision for the Gaussian observations 14.379 0.205     13.978   14.379
## Range for floc                           0.956 0.137      0.723    0.944
## Stdev for floc                           0.376 0.029      0.323    0.375
##                                         0.975quant   mode
## Precision for the Gaussian observations     14.784 14.380
## Range for floc                               1.262  0.914
## Stdev for floc                               0.438  0.371
##
## Deviance Information Criterion (DIC) ...............: 2011.11
## Deviance Information Criterion (DIC, saturated) ....: 10470.37
## Effective number of parameters .....................: 253.82
##
## Watanabe-Akaike information criterion (WAIC) ...: 1988.29
## Effective number of parameters .................: 228.14
##
## Marginal log-Likelihood:  -1346.22
## CPO, PIT is computed
## Posterior summaries for the linear predictor and the fitted values are computed
## (Posterior marginals needs also 'control.compute=list(return.marginals.predictor=TRUE)')
##
##      longitude.std latitude.std       fprod housing_median_age.std        fhma2
## mean   -0.36175409  -0.36733176 0.020341691           -0.060266255 -0.043896012
## sd      0.07169014   0.07309913 0.002806518            0.006253863  0.008405838
##            fhma3       fhma4 median_income.stdlog        fmi2          fmi3
## mean 0.019337449 0.018471752          0.323018919 0.039978221 -0.0149098046
## sd   0.002386632 0.002366737          0.004727441 0.003586741  0.0009917896
##               fmi4 op_near_bay  op_inland op_near_ocean   op_island
## mean -0.0016374624  0.08774117 0.02208334    0.07698446 0.003312652
## sd    0.0004317253  0.02227385 0.02118880    0.01582755 0.282536090
##      average_bed_rooms.std   Intercept
## mean           0.015906123 11.82097708
## sd             0.003337223  0.07786567
```




```
## DIC of model 3:      [1] 2011.113
## NLSCPO of model 3:   [1] 1017.097
## WAIC of model 3: [1] 1988.29
```

```
res.spde2 = bayes.spde(housing.training, hsdomain, hsmesh2, prior.beta1, prec.prior1,
                       sense.check=T)
res.spde3 = bayes.spde(housing.training, hsdomain, hsmesh3, prior.beta1, prec.prior1,
                       sense.check=T)
res.spde4 = bayes.spde(housing.training, hsdomain, hsmesh1, prior.beta2, prec.prior2,
                       sense.check=T)
res.spde5 = bayes.spde(housing.training, hsdomain, hsmesh1, prior.beta1, prec.prior1,
                       prior.range=c(10, 0.1), sense.check=T)
res.spde6 = bayes.spde(housing.training, hsdomain, hsmesh1, prior.beta1, prec.prior1,
                       prior.range=c(1, 0.1), prior.sigma=c(200, 0.1), sense.check=T)

# print posterior means of fixed effects and model scores for different implementations
posts = rbind(res.spde1[[2]], res.spde2[[2]], res.spde3[[2]], res.spde4[[2]],
              res.spde5[[2]], res.spde6[[2]])
metrics = rbind(res.spde1[[3]], res.spde2[[3]], res.spde3[[3]], res.spde4[[3]],
                res.spde5[[3]], res.spde6[[3]])
print(posts)
```

```
##       longitude.std latitude.std      fprod housing_median_age.std       fhma2
## mean    -0.36175409   -0.3673318 0.02034169            -0.06026625 -0.04389601
## mean1   -0.38898450   -0.3924078 0.02922529            -0.04958292 -0.03033565
## mean2   -0.37084791   -0.3931289 0.01863410            -0.06026224 -0.04099683
## mean3   -0.36176221   -0.3673375 0.02034153            -0.06026699 -0.04389508
```

```
## mean4   -0.33810086   -0.3478001 0.02042910            -0.06037788 -0.04379022
## mean5    0.06666222    0.1410868 0.02046872            -0.06040515 -0.04369212
##            fhma3      fhma4 median_income.stdlog       fmi2        fmi3
## mean  0.01933745 0.01847175            0.3230189 0.03997822 -0.01490980
## mean1 0.02108136 0.01724702            0.3240885 0.04222652 -0.01344957
## mean2 0.02038452 0.01837174            0.3260994 0.04486200 -0.01605128
## mean3 0.01933805 0.01847174            0.3230210 0.03997891 -0.01490967
## mean4 0.01941497 0.01851413            0.3230534 0.04016257 -0.01489233
## mean5 0.01944955 0.01854085            0.3230670 0.04026696 -0.01488436
##               fmi4 op_near_bay  op_inland op_near_ocean    op_island
## mean  -0.001637462  0.08774117 0.02208334   0.076984464  0.003312652
## mean1 -0.001453330  0.01129141 0.01028470   0.003542253  0.360023939
## mean2 -0.002279994  0.11523360 0.07866621   0.079253744 -0.135838365
## mean3 -0.001637583  0.08774286 0.02207697   0.076979991  0.003593152
## mean4 -0.001649479  0.08741910 0.02612842   0.073447594 -0.150436471
## mean5 -0.001653906  0.08713245 0.02897696   0.070693653 -0.402255030
##       average_bed_rooms.std Intercept
## mean             0.01590612 11.820977
## mean1            0.01669992 11.904199
## mean2            0.01691588 11.816411
## mean3            0.01590506 11.820998
## mean4            0.01571492 11.827367
## mean5            0.01573032  9.050038
```

```
print(metrics)
```

```
##        DIC   NLSCPO     WAIC
## 1 2011.113 1017.097 1988.290
## 2 3271.271 1642.927 3254.986
## 3 2271.270 1149.348 2260.587
## 4 2011.131 1017.107 1988.335
## 5 2015.858 1019.713 1993.061
## 6 2022.356 1022.895 1997.260
```

Explanation: (Write your explanation here)

In this question, we include the spatial random effect for location (longitude, latitude) with Matern covariance, power terms of `housing_median_age` and `median_income.log` up to order 4, and the interaction term of `housing_median_age` and `median_income.log`. Meshes are created in advance based on the locations in the dataset.

The posterior means for the fixed effect are $-0.362(0.072)$, $-0.367(0.073)$, $0.020(0.003)$, $-0.060(0.006)$, $-0.044(0.008)$, $0.019(0.002)$, $0.018(0.002)$, $0.323(0.005)$, $0.040(0.004)$, $-0.015(0.001)$, $-0.002(0.000)$, $0.088(0.022)$, $0.022(0.021)$, $0.077(0.016)$, $0.003(0.283)$, $0.016(0.003)$, and $11.821(0.078)$, correspondingly, with reasonably small sds and 0 excluded in $\mu \pm \sigma$ interval for most variables except `longitude.log`, `latitude.log`, and `Intercept`. The results demonstrate some reversed effects:

- The posterior means of `housing_median_age` and its square are negative, implying that increasing the **median age of houses** in a district will result in a reduced median house value in the district; Those of the power of 3 and 4 have reversed effects.
- The posterior means of `median_income.log` and its square are positive, implying that increasing the **median income** in a district will result in an increased median house value in the district; Those of the power of 3 and 4 have reversed effects.
- **Ocean proximity** does not display significant effects on median house prices, since a considerable amount of the fixed effects are absorbed into the spatial random effects.

For the random effects, we plot them on a heat map versus their locations based on longitudes and altitudes, and the plot demonstrates that apart from the fixed effects of `longitude` and `latitude`, the **spatial random effect** is generally higher in the southwest part of California, corresponding to the locations along the coastline, and northeastern state. In comparison, the spatial random effect tends to be lower in inland areas including the northern and southeastern part of California, especially in the centre of the state, which may be due to some sociological issues.

The **DIC**, **NLSCPO**, and **WAIC** scores of the spatial model (2011.10, 1017.08, and 1988.26) are significantly reduced compared to the former two models, which makes it the best model among all. Based on the three sets of scores, the first model is the worst model among all with the highest DIC, NLSCPO, and WAIC scores, and the second model is ranked the second.

Regarding **sensitivity check**, we pack the whole fit-plus-summary process in a function `bayes.spde` and try two types of priors and different meshes.

- For the meshes, `max.edge` represents the largest allowed triangle edge length in the inner domain and outer extension, and `cutoff` represents the minimum distance allowed between points. We tune the `max.edge` and `cutoff` arguments to generate different meshes displayed above (`res.spde1`, `res.spde2`, and `res.spde3`). The results demonstrate considerable differences in posterior means of fixed effects and model scores, with the altered meshes generating worse fitting results.
- `prior.range` and `prior.sigma` represent the hyperparameters needed to be specified for the spatial model, i.e. the lower tail quantile and probability for the spatial range, ρ_0 and p_ρ, and the upper tail quantile and probability for the marginal standard deviation, σ_0 and α_σ. We alter the prior value of ρ_0 and σ_0 for sensitivity check (`res.spde1`, `res.spde5`, and `res.spde6`). The results demonstrate slight but tolerable differences between the posterior means and model scores with different ρ_0, and not much differences given different σ_0. However, it is generally wise to set σ_0 similar to the response variable changing scale to obtain stable results.
- For different uninformative priors of fixed effects and the Gaussian precision, the sensitivity is as low as before (`res.spde1` and `res.spde4`).

Q4)[10 marks]

In this question, we will evaluate the predictive performance of these models.

Do the following two tests for all 3 models.

First, compute the posterior mean of the log(median_house_value) for the districts in the training dataset housing.training. Compute the median absolute difference between the posterior means of the log(median_house_value) and its true values on the training dataset. This can be done by including the posterior means in an array v , the true values in an array t, and computing $\text{median}(|v - t|)$.

Second, evaluate the log(median_house_value) 's posterior predictive means on the test dataset housing.test. Compute the median absolute difference between the log(median_house_value) 's posterior predictive mean and its true value on the test dataset.

Discuss the results.

```
# obtain the fitted model from above
cali.ord = res.ord1[[1]]; cali.rwar = res.rwar1[[1]]; cali.spde = res.spde1[[1]]

# retrieve the true response values from training data
y.train = housing.training$median_house_value.log

# calculate the median absolute differences between posterior predictive means &
#   true values for all 3 models
median(abs(cali.ord$summary.fitted.values$mean - y.train))
```

```
## [1] 0.2094229
```

```
median(abs(cali.rwar$summary.fitted.values$mean - y.train))
```

```
## [1] 0.1995396
```

```
median(abs(cali.spde$summary.fitted.values$mean[1: nrow(housing.training)] - y.train))
```

```
## [1] 0.1506088
```

```
# function for scaling data (A) based on the mean & sd of aother data (B)
scaleAbyB = function(A, B) {
  (A - mean(B)) / sd(B)
}

# retrieve the true response values from test data
y.test = log(housing.test$median_house_value)

# add the log / scale version of desired columns into the dataframe for
#   testing data & set the response values as NA for prediction
housing.test$median_house_value.log = NA
housing.test$longitude.std = scaleAbyB(housing.test$longitude,
                                       housing.training$longitude)
housing.test$latitude.std = scaleAbyB(housing.test$latitude,
                                      housing.training$latitude)
housing.test$housing_median_age.std = scaleAbyB(housing.test$housing_median_age,
                                                housing.training$housing_median_age)
housing.test$median_income.log = log(housing.test$median_income)
housing.test$median_income.stdlog = scaleAbyB(housing.test$median_income.log,
                                              housing.training$median_income.log)
housing.test$average_bed_rooms.std = scaleAbyB(housing.test$average_bed_rooms,
                                               housing.training$average_bed_rooms)

# set Location data frame & transform Locations to sf objects for testing data
Locations = data.frame(easting=housing.test$longitude,
                       northing=housing.test$latitude)
housing.test$geometry = sf::st_as_sf(Locations,
                                     coords=c("easting", "northing"))$geometry

# generate the dataset for prediction
housing.pred = rbind(housing.training, housing.test)
```

```
# fit the 3 models again based on the prediction data & retrieve the fitted models
cali.ord.pred = bayes.ord(housing.pred, prec.prior1, prior.beta1, T,
                          control.predictor=list(compute=T))[[1]]
cali.rwar.pred = bayes.rwar(housing.pred, prec.prior1, prior.beta1, T,
                            control.predictor=list(compute=T))[[1]]
cali.spde.pred = bayes.spde(housing.pred, hsdomain, hsmesh1, prior.beta1,
                            prec.prior1, sense.check=T,
                            control.predictor=list(compute=T))[[1]]

# calculate the median absolute differences between posterior predictive means &
#   true values for all 3 models
N.train = nrow(housing.training)
N = nrow(housing.pred)
```

```r
median(abs(cali.ord.pred$summary.fitted.values$mean[(N.train + 1): N] - y.test))
```

```
## [1] 0.208305
```

```r
median(abs(cali.rwar.pred$summary.fitted.values$mean[(N.train + 1): N] - y.test))
```

```
## [1] 0.2032275
```

```r
median(abs(cali.spde.pred$summary.fitted.values$mean[(N.train + 1): N] - y.test))
```

```
## [1] 0.1524755
```

Explanation: (Write your explanation here)

In this question, we first withdraw the posterior mean of the log of median house values in districts under the fitted model based on `housing.training`. The median absolute differences between them and the true values are 0.209, 0.200, and 0.151, indicating that in terms of fitting performances, the spatial model is the best and the model with only fixed effects is the worst among all.

Subsequently, we modify the test data under the same scale as the training data, set the response value column in the testing data as `NA`, and merge the two datasets for prediction. The median absolute differences between the predicted values and the true values are 0.208, 0.203, and 0.152, indicating that in terms of prediction performances, the spatial model and the fixed-effect-model are again the best and the worst model among all, correspondingly.

Q5)[10 marks] Perform posterior predictive checks (using replicates) on all 3 models Q1-Q3 fitted on the housing.training dataset. Choose your test functions to provide insight into the model. Discuss the results.

```r
require(fBasics)
```

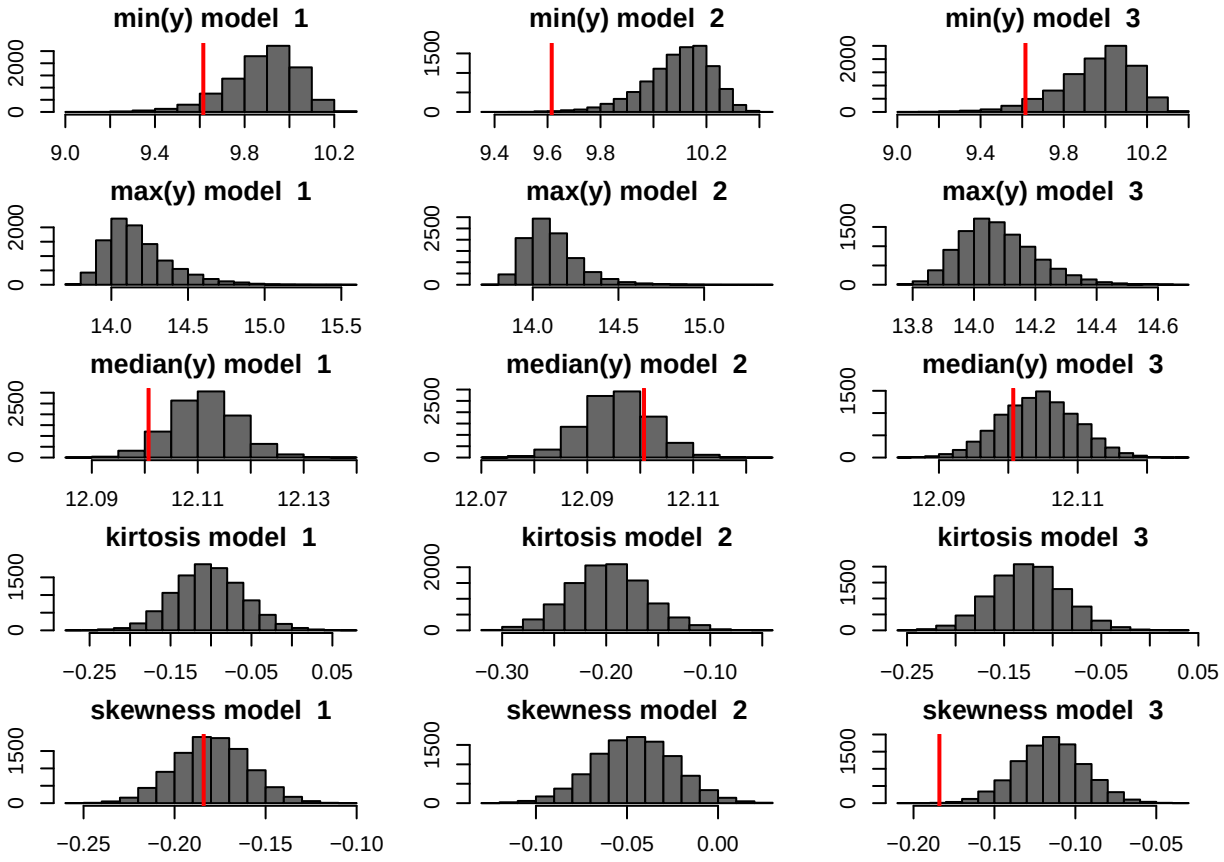
```
## Loading required package: fBasics
```

```
## Warning: package 'fBasics' was built under R version 4.3.2
```

```r
# function for replicate data checks, calculating the five statistics and plot
#   model: the model about to check; data: the observed data
#   ind: model index; nbsamp: sample times
model.rep.check = function(model, data, ind, nbsamp=10000) {
  N.train = nrow(data)

  # sampling process of the replicate data
  model.samps = inla.posterior.sample(nbsamp, model)

  # calculate the replicate data from the model
  y.rep = unlist(lapply(model.samps, function(x) x$latent[1: N.train]))
  sigma.rep = 1 / sqrt(unlist(lapply(model.samps, function(x) x$hyperpar[1])))

  eps.rep = rnorm(N.train * nbsamp, 0, rep(sigma.rep, each=N.train))
  y.rep = matrix(y.rep + eps.rep, nrow=N.train)

  # the original data for log median house value
  y.ori = data$median_house_value.log

  # replicate data checks, calculate the five statistics and plot
  y.min = apply(y.rep, 2, min)
  y.max = apply(y.rep, 2, max)
  y.med = apply(y.rep, 2, median)
  y.kurt = apply(y.rep, 2, kurtosis)
```

```
  y.skew = apply(y.rep, 2, skewness)

  hist(y.min, col="gray40", main=paste("min(y) model ", ind))
  abline(v=min(y.ori), col="red", lwd=2)
  hist(y.max, col="gray40", main=paste("max(y) model ", ind))
  abline(v=max(y.ori), col="red", lwd=2)
  hist(y.med, col="gray40",main=paste("median(y) model ", ind))
  abline(v=median(y.ori), col="red", lwd=2)
  hist(y.kurt, col="gray40", main=paste("kirtosis model ", ind))
  abline(v=kurtosis(y.ori), col="red", lwd=2)
  hist(y.skew, col="gray40", main=paste("skewness model ", ind))
  abline(v=skewness(y.ori), col="red", lwd=2)
}
```

```
# posterior predictive checks for log median house value on the 3 models
par(mfcol=c(5, 3))
par(mar=c(2, 2, 2, 2))
model.rep.check(cali.ord, housing.training, 1)
model.rep.check(cali.rwar, housing.training, 2)
model.rep.check(cali.spde, housing.training, 3)
```



Explanation: (Write your explanation here)

In this question, we perform posterior predictive checks on the observed data, which means reproducing the data based on our model and comparing the relative statistics of the replicated and original data, thus checking the reasonability of our model. Here, we choose minimum, maximum, median, kurtosis, and skewness as the comparison statistics.

From the plots, we can see that the posterior distributions of the median generally meet those of the real-world data. Regarding minimum and skewness, model 1 generally fits well, but not for the other models. As for maximum and kurtosis, all three models fail to meet the true value. This implies that there may exist some limitations for our models in failing to meet some of the test functions of the response value.