Bayesian Calibration of Numerical Model Output by an R package: spCalibration

ChenYW

2020-04-08

Contents

Setup

```
knitr::opts_chunk$set(echo = TRUE, message = FALSE, warning = FALSE)
# , class.source = "BK"
rm(list=ls())
suppressMessages(Sys.setlocale("LC_TIME", "English"))
```

```
## [1] "English_United States.1252"
```

1 R package and data

1.1 Load R package: spCalibration

```
######################################################################
#                   load  created R package: spCalibration
######################################################################
suppressMessages(library(spCalibration))
```

```
## Number of platforms: 1
## - platform: NVIDIA Corporation: OpenCL 1.2 CUDA 10.1.0
##   - context device index: 0
##     - GeForce RTX 2080 Ti
## checked all devices
## completed initialization
```

```
gcQuiet(FALSE, verbose=TRUE)
```

```
##           used  (Mb) gc trigger  (Mb) max used  (Mb)
## Ncells 3476143 185.7    6631857 354.2  4988472 266.5
## Vcells 5336462  40.8   10146329  77.5  7379176  56.3
```

Note: Our all data used in this file was encapsulated in "`spCalibration`" along with the code.

1.2 Data layout

```
######################################################################
#                       Fundamental dataset from pollution
######################################################################
Model_Base_Tab[, c(15, 4:6, 17:18, 7:8, 12, 19:20)]
```

```
##        SITEID CMAQ_ID     LON    LAT      LON_X   LAT_Y  DATE_TIME YEAR_MONTH
##     1:      1    5724 114.542 38.031 -242926.92 4243177 2015-06-01     201506
##     2:      1    5724 114.542 38.031 -242926.92 4243177 2015-06-02     201506
##     3:      1    5724 114.542 38.031 -242926.92 4243177 2015-06-03     201506
##     4:      1    5724 114.542 38.031 -242926.92 4243177 2015-06-04     201506
##     5:      1    5724 114.542 38.031 -242926.92 4243177 2015-06-05     201506
##    ---
## 12508:     76    7932 116.650 40.410  -38955.09 4492669 2016-01-27     201601
## 12509:     76    7932 116.650 40.410  -38955.09 4492669 2016-01-28     201601
## 12510:     76    7932 116.650 40.410  -38955.09 4492669 2016-01-29     201601
## 12511:     76    7932 116.650 40.410  -38955.09 4492669 2016-01-30     201601
```

```
## 12512:     76    7932 116.650 40.410  -38955.09 4492669 2016-01-31     201601
##        REAL_PM25 CMAQ_PM25 CMAQ_PM25_30
##     1:  87.20833 53.104367    26.146596
##     2:  44.04167 47.539544    23.730816
##     3:  59.45833 53.139121    29.578908
##     4:  75.04167 39.686555    21.140597
##     5: 100.25000 37.311844    18.116450
##    ---
## 12508: 110.20833 42.721178    55.522024
## 12509: 119.37500 12.222126    16.034347
## 12510:  86.25000 37.876560    42.432469
## 12511:  64.91667 15.412478    19.596098
## 12512:  15.62500  4.373301     6.221695
```

```
######################################################################
#                   the coordinates of monitoring station
######################################################################
head(Site)
```

```
##    SITEID         CITY  STATION_NAME     LON    LAT     LON_X   LAT_Y
## 1:      1 Shijiazhuang Shijigongyuan 114.542 38.031 -242926.9 4243177
## 2:      2 Shijiazhuang  Xinangaojiao 114.467 38.012 -249721.6 4241671
## 3:      4 Shijiazhuang Zhigongyiyuan 114.455 38.051 -250376.2 4246107
## 4:      5 Shijiazhuang Renminhuitang 114.521 38.052 -244560.1 4245683
## 5:      6 Shijiazhuang Xibeishuiyuan 114.502 38.140 -245328.9 4255626
## 6:      7 Shijiazhuang      Gaoxinqu 114.605 38.040 -237292.7 4243672
```

```
######################################################################
#                           dataset of CMAQ
######################################################################
CMAQ_PM25[, c(1:3, 8)]
```

```
##          CMAQ_ID  DATE_TIME CMAQ_PM25 YEAR_MONTH
##      1:     4619 2015-06-01  9.824296     201506
##      2:     4619 2015-06-02 10.246346     201506
##      3:     4619 2015-06-03 17.661194     201506
##      4:     4619 2015-06-04  4.019376     201506
##      5:     4619 2015-06-05  3.203634     201506
##     ---
## 459812:    11438 2016-01-27 86.914974     201601
## 459813:    11438 2016-01-28 20.297575     201601
## 459814:    11438 2016-01-29 20.376099     201601
## 459815:    11438 2016-01-30 34.814252     201601
## 459816:    11438 2016-01-31 41.928738     201601
```

```
######################################################################
#                    the coordinates of CMAQ lattice
######################################################################
cmaq_site
```

```
##       CMAQ_ID      LON      LAT     LON_X   LAT_Y         CITY   ID
##    1:    4619 113.5025 36.71028 -349353.9 4105037       Handan    1
```

```
##    2:    4638 113.5744 38.30076 -325010.6 4281459 Shijiazhuang    2
##    3:    4639 113.5783 38.38442 -323712.1 4290736 Shijiazhuang    3
##    4:    4640 113.5822 38.46806 -322412.8 4300010 Shijiazhuang    4
##    5:    4641 113.5860 38.55172 -321112.5 4309285 Shijiazhuang    5
##   ---
## 2495:   11199 119.5909 40.25370  210047.3 4461493  Qinhuangdao 2495
## 2496:   11317 119.6677 39.99614  215508.6 4432649  Qinhuangdao 2496
## 2497:   11318 119.6783 40.07928  216765.3 4441846  Qinhuangdao 2497
## 2498:   11319 119.6890 40.16242  218020.3 4451043  Qinhuangdao 2498
## 2499:   11438 119.7762 39.98793  224742.0 4431396  Qinhuangdao 2499
```

1.3 Create grids by CMAQ or INLA

```
######################################################################
#            A. Create directly grids  by CMAQ lattices
######################################################################
# temp <- cmaq_site
# coordinates(temp) = ~ LON_X + LAT_Y
# coords <- temp@coords
# Neighbor_order <- dnearneigh(coords, 0, 1e4)
# # View(Neighbor_order)
# adjacent.matrix <- matrix(0, nrow(coords), nrow(coords))
# for(i in 1:nrow(coords))
# {
#   for( j in 1: length(Neighbor_order[[i]]))
#   {
#     Ad.num <- Neighbor_order[[i]][j]
#     adjacent.matrix[i, Ad.num] = -1
#   }
# }
# grid <- list(grid.coords = cmaq_site,
#                adjacent.matrix = adjacent.matrix)
######################################################################
#                   B. Create grids  by INLA
######################################################################

grid <- ProduceGrid(Site,     # Monitoring stations data
                    max.edge = c(0.3, 0.7),  # max.edge, offset and cutoff: see INLA
                    offset = c(0.4, 0.6),
                    cutoff = 0.1,
                    col = "black",
                    size = 1)
grid$plot.grid
```

1.4 Mapping matrix: H

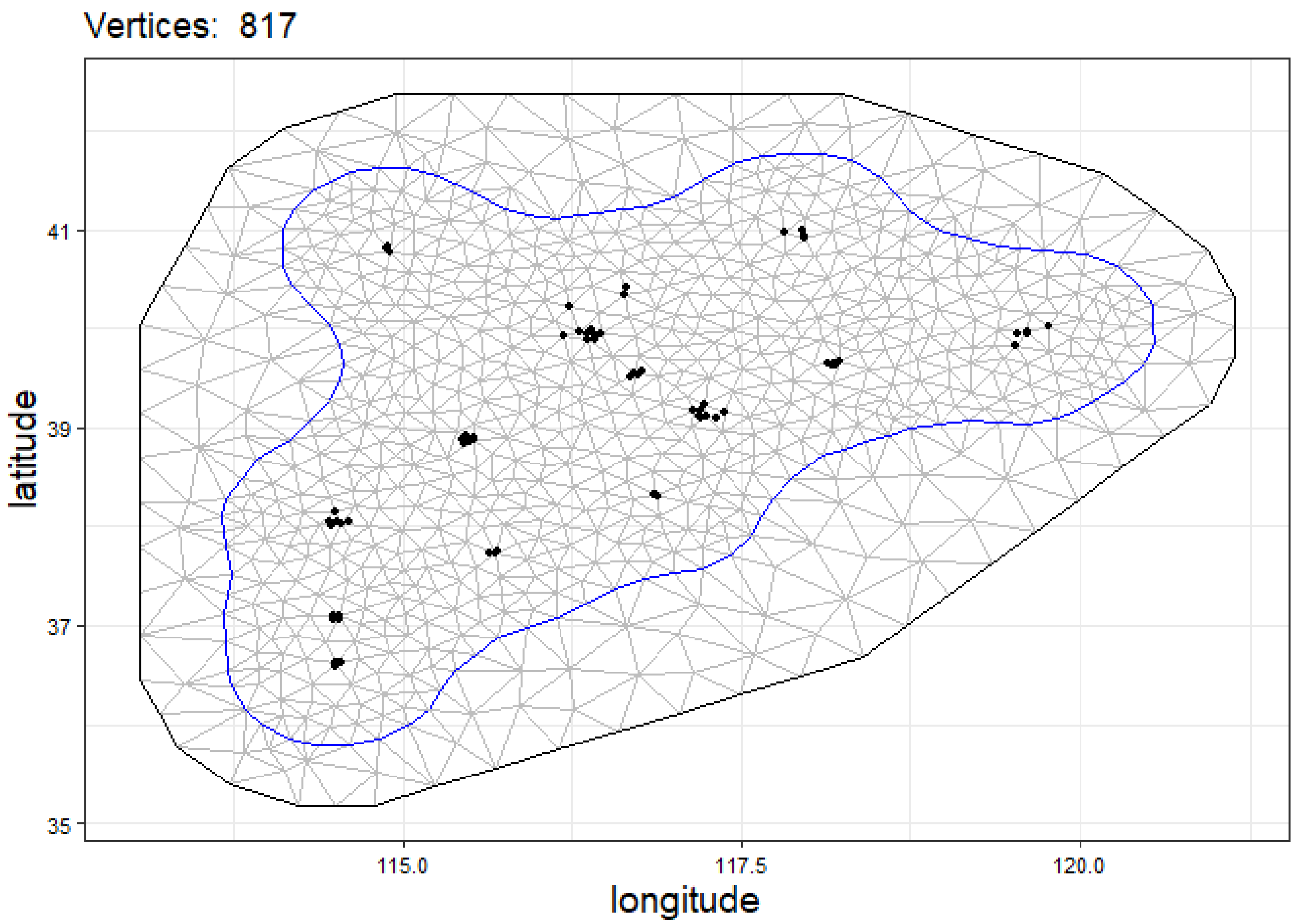

Figure 1.1: The irregular division of spatial domain by INLA

```
Data_Str <- ProduceHmatrix(grid, Site, threshold = 1e5)
# save(grid, file = "./data/BaseTable/Data_Str.Rda")
# load("./data/BaseTable/Data_Str.Rda")
######################################################################
```

1.5 Initialize parameter

1.5.1 Prior

```
######################################################################
#                         initialize  parameters
######################################################################
{
  ####################################################################
  # true.para <- list(beta = rep(NA, 2), nugget.tau2 = NA,
  #                   theta = c(NA, NA), k0 = NA, k = NA)
  ####################################################################
  #####################################################################
  #                          prior distribution
  #####################################################################
  prior <- list(
                beta = list(mu = c(0, 1), Sigma2 = 1e5*diag(2))
                , nugget.tau2 = list(a = 2, b = 1)
                , theta1 = list(mu = 0.005, Sigma2 =  1e5)
              )
}
```

1.5.2 Importance sampling

```
IS.size <- 100
 # rinvgamma(IS.size, shape = 0.5, rate = 1)
IS <- list(theta2_random_sample = runif(IS.size, 1e-3, 5)
            , K_random_sample = runif(IS.size, 1e-3, 15)
            , K0_random_sample = runif(IS.size, 1e-3, 15)
            , G_theta2_weight = rep(1/IS.size, IS.size)
            , G_k_weight = rep(1/IS.size, IS.size)
            , G_k0_weight = rep(1/IS.size, IS.size)
            , IS.size = IS.size
            , Thresh = c(1, 1, 1)
           )
```

1.5.3 Initialization

```
initial.para <- list(
                   beta = list(E_beta = c(2, 0.5), Sigma2 = diag(2))
                             , theta1 = list(E_theta1 = 0.005, Sigma2 = 2)
                             , k = list(E_k = 5, a = 2, b = 1)
                             , k0 = list(E_k0 = 5, a = 2, b = 1)
```

```
                              , theta2 = list(mu = c(1))
                              , tau2 = list(E_tau2 = 1, a = 2, b = 1)
                            )
```

2 Data truncation

```
######################################################################
#                              Data truncation
######################################################################
YearMonth <- c(201511, 201512, 201601)  # 201506,201507,201508 # 201511, 201512, 201601
Yts_Xts <- ParYtsXts(Model_Base_Tab, YearMonth = YearMonth)
```

3 Model cross validation

```
######################################################################
#                            leave one city out
######################################################################
city_num = 1
GSD = Test_Train_Fun(Site, Data_Str, Yts_Xts, city_num)
```

```
##
##
## Test city:  Baoding ...
```

```
Train = GSD$Train
Test =  GSD$Test
Train$Y_ts = sqrt(Train$Y_ts)
Train$X_ts = sqrt(Train$X_ts)
Test$X_ts = sqrt(Test$X_ts)
```

```
######################################################################
#                            fit model
######################################################################
# CV <- spVBEnKs(data = Train,
#                prior = prior,
#                IS = IS,
#                para = initial.para,
#                true.para = NULL,
#                parallel = TRUE,
#                verbose = TRUE,
#                verbose.VB = TRUE,
#                Ensemble.size = 50,
#                cs = 0.4,
#                ct = 1,
#                Remove.Cores.Count = 2,
#                N.Chunk = 1,
#                itMax = 1e2,
#                tol.vb = 1e-3,
```

```r
#                tol.real = 1e-3)
load( "./data/Generate_Data/Test/CV.Rdata")
```

```r
######################################################################
#                            output
######################################################################
print(CV)
```

```
##
## Call: spVBEnKs(data = Train, prior = prior, IS = IS, para = initial.para,
##     true.para = NULL, parallel = TRUE, verbose = TRUE, verbose.VB = TRUE,
##     Ensemble.size = 50, cs = 0.4, ct = 1, Remove.Cores.Count = 2,
##     N.Chunk = 1, itMax = 100, tol.vb = 0.01, tol.real = 0.001)
##
## Iterations: 7
##
## Log-likelihood: 181408.598
##
## Parameter estimation:
##       beta0 beta1 tau_2 theta1 theta2      k    k0 iter
## True:    NA    NA    NA     NA     NA     NA    NA    0
## Init: 2.000 0.500 1.000  0.005  1.000  5.000 5.000    0
## Esti: 1.457 0.627 1.021  0.011  0.871 11.039 5.106    7
##
## (...output suppressed due to large dimension!)
```

3.1 Prediction

```r
######################################################################
#                          prediction
######################################################################
P <- predict(CV, test = Test, site = Site, method = c("ensemble"),
             transf = "square", ncol.layout = 2)

P$error
```

```
## $Diff.Station.RMSE
##      34      35      36      37      38      39
##  70.524  62.764 101.934  64.384  59.967  56.861
##
## $Total.Error
##   Pearson.before Pearson.after   RMSE      MB     NMB    NME
## 1          0.628         0.817 71.038 -23.675 -14.873 31.831
```

```r
P$ensemble.plot
```

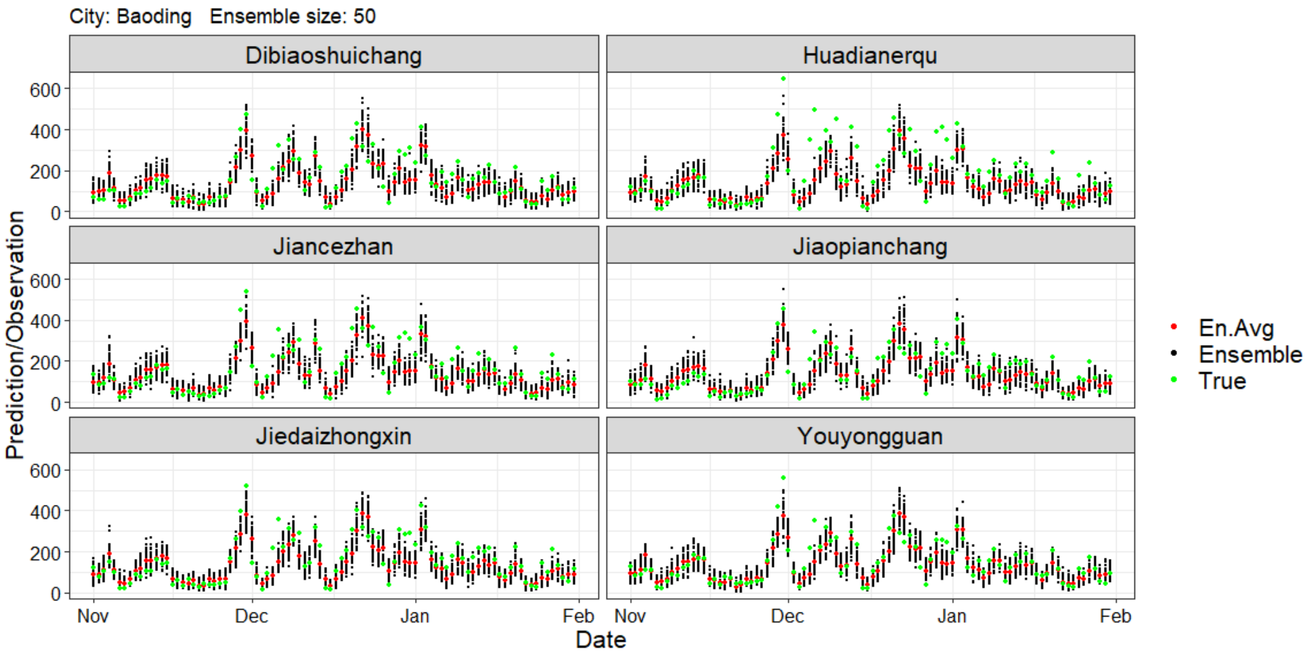

Figure 3.1: Prediction by ensemble

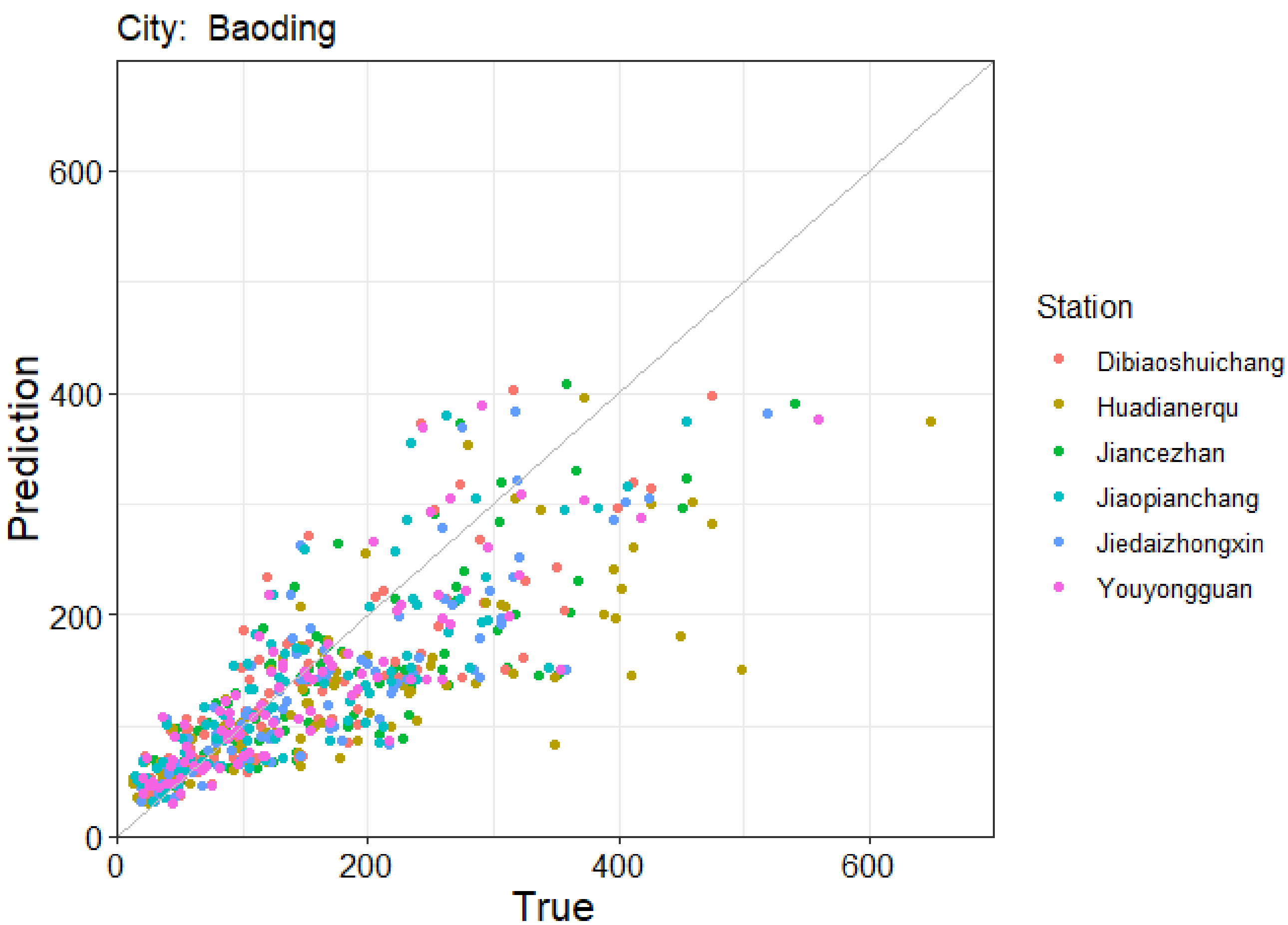

Figure 3.2: Prediction by ensemble mean

```
P$mean.plot
```

3.2 The distributions of parameters

```
##########################################################################
#                          the distributions of parameters
##########################################################################
P <- plot(CV, n = 500, title.size = 30, text.size = 20, line.size = 1)
P$Beta.plot
```

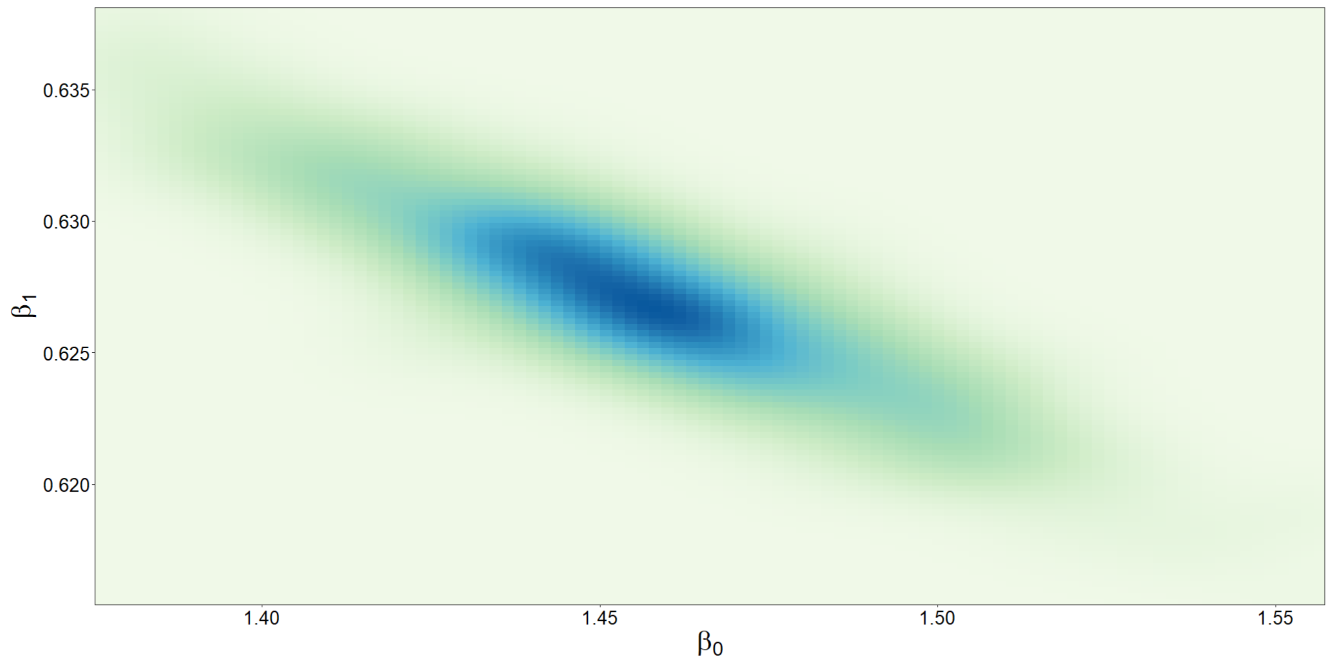

Figure 3.3: The distribution of Beta parameter

```
P$Close.Dist
```

```
P$NoClose.Dist
```

4 Complete dataset

```
##########################################################################
#                              Complete dataset
##########################################################################
Total_Data <- list(
                n = dim(Yts_Xts$Y_ts)[2]
                , Nt = dim(Yts_Xts$Y_ts)[1]
                , N.BAUs = Data_Str$N.BAUs
                , Y_ts = Yts_Xts$Y_ts
                , X_ts = Yts_Xts$X_ts
                , BAUs.Dist = Data_Str$BAUs.Dist
                , Adj.Mat = Data_Str$G
```

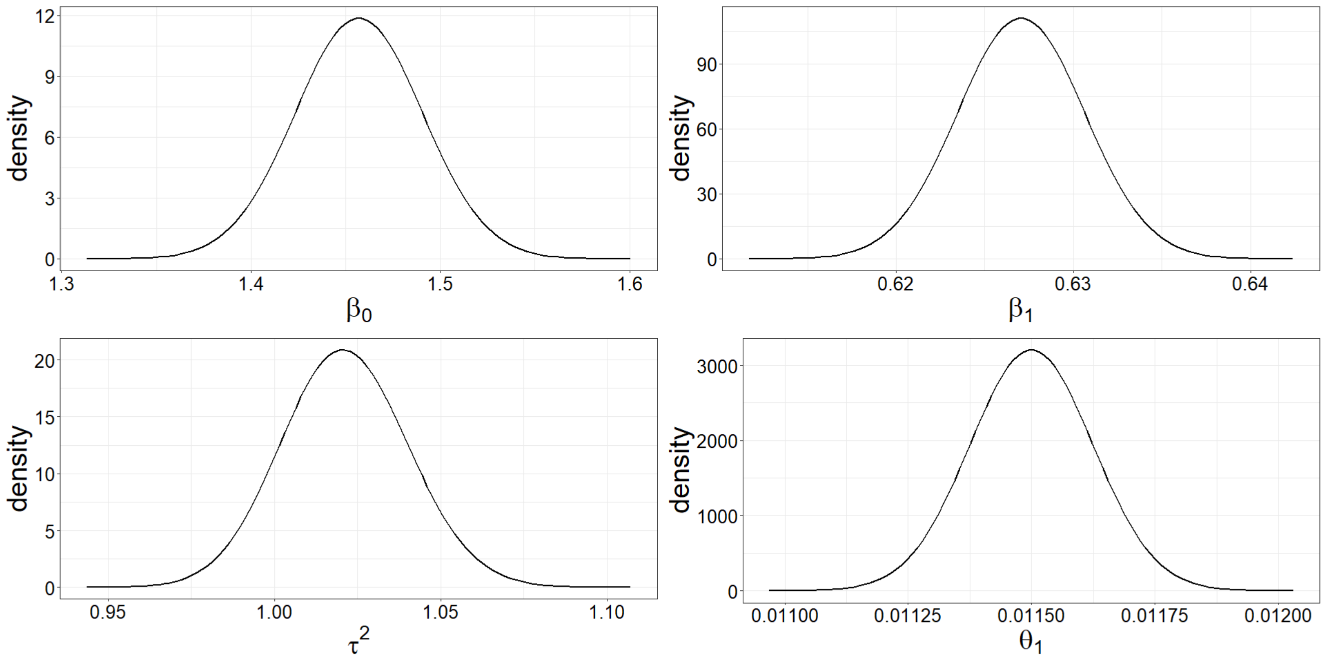

Figure 3.4: The distributions of parameters of close form

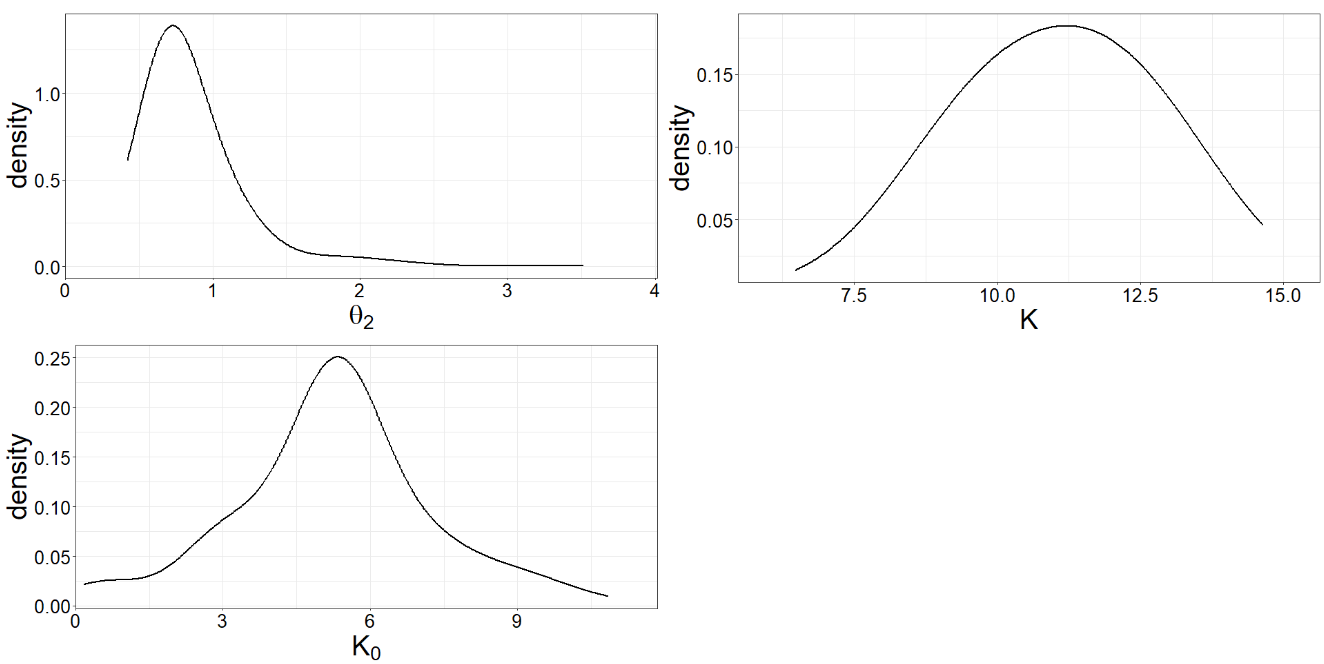

Figure 3.5: The distributions of parameters of no close form

```r
               , Hs = Data_Str$Hs
             )
Total_Data$Y_ts = sqrt(Total_Data$Y_ts)
Total_Data$X_ts = sqrt(Total_Data$X_ts)
```

```r
######################################################################
#                           fit model and prediction
######################################################################
# Total <- spVBEnKs(data = Total_Data,
#                   prior = prior,
#                   IS = IS,
#                   para = initial.para,
#                   true.para = NULL,
#                   parallel = TRUE,
#                   verbose = TRUE,
#                   verbose.VB = TRUE,
#                   Ensemble.size = 50,
#                   cs = 0.4,
#                   ct = 1,
#                   Remove.Cores.Count = 2,
#                   N.Chunk = 1,
#                   itMax = 1e2,
#                   tol.vb = 1e-3,
#                   tol.real = 1e-3)
load("./data/Generate_Data/Total/Total.Rdata")
print(Total)
```

```
##
## Call: spVBEnKs(data = Total_Data, prior = prior, IS = IS, para = initial.para,
##     true.para = NULL, parallel = TRUE, verbose = TRUE, verbose.VB = TRUE,
##     Ensemble.size = 50, cs = 0.4, ct = 1, Remove.Cores.Count = 2,
##     N.Chunk = 1, itMax = 100, tol.vb = 0.01, tol.real = 0.001)
##
## Iterations: 7
##
## Log-likelihood: 180458.524
##
## Parameter estimation:
##       beta0 beta1 tau_2 theta1 theta2      k    k0 iter
## True:    NA    NA    NA     NA     NA     NA    NA    0
## Init: 2.000  0.50 1.000  0.005  1.000  5.000 5.000    0
## Esti: 1.455  0.62 1.049  0.012  0.871 10.938 5.103    7
##
## (...output suppressed due to large dimension!)
```

4.1 The distributions of parameters

```r
######################################################################
#                         the distributions of parameters
```

```
############################################################################
P <- plot(Total, n = 500, title.size = 30, text.size = 20, line.size = 1)
P$Beta.plot
```

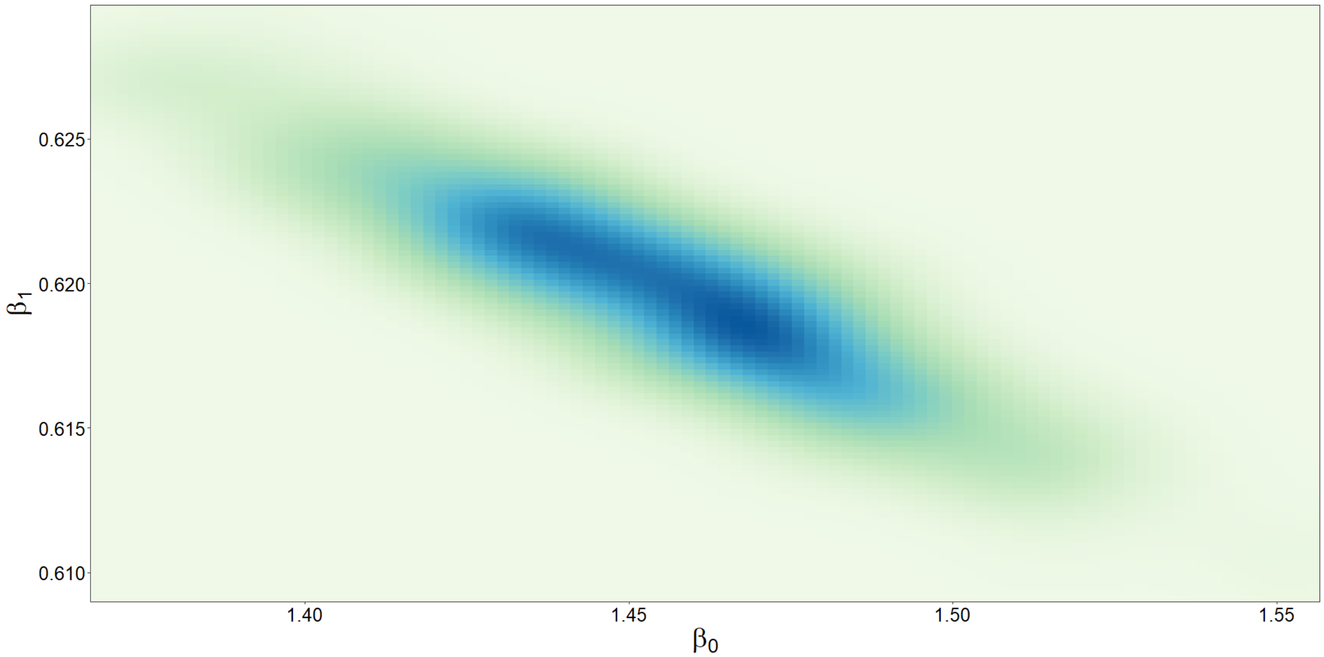

Figure 4.1: The distribution of Beta parameter

```
P$Close.Dist
```

```
P$NoClose.Dist
```

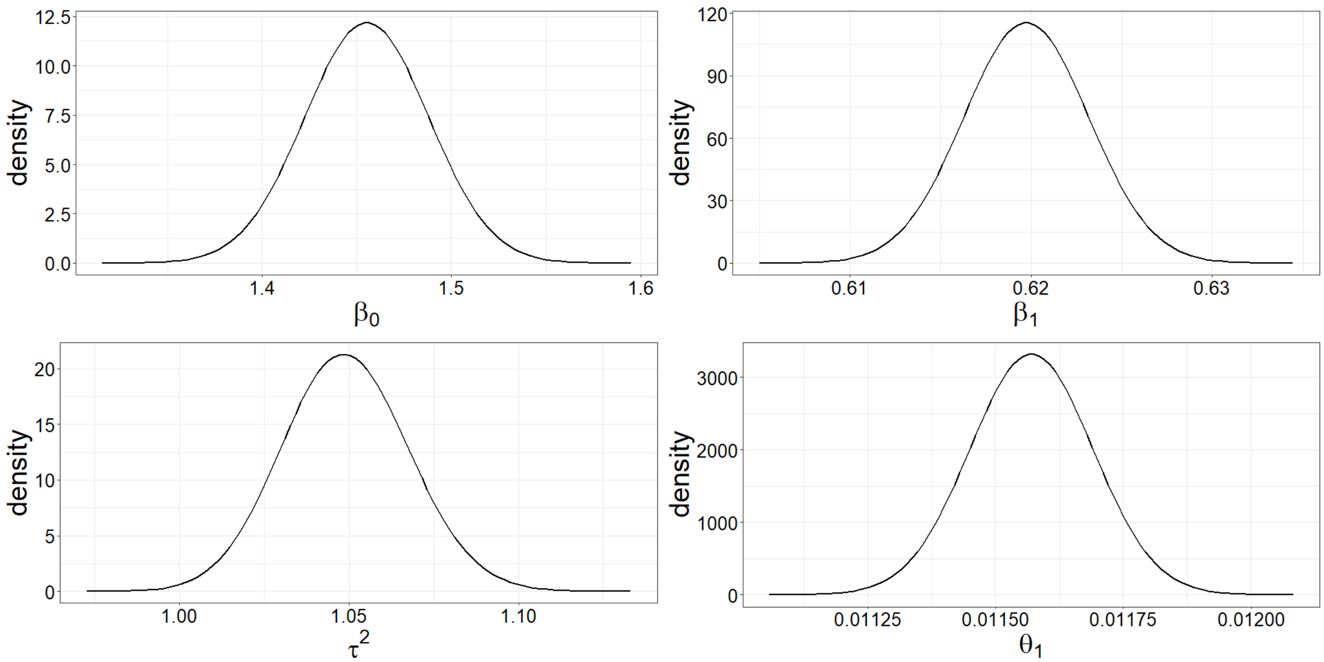

Figure 4.2: The distributions of parameters of close form

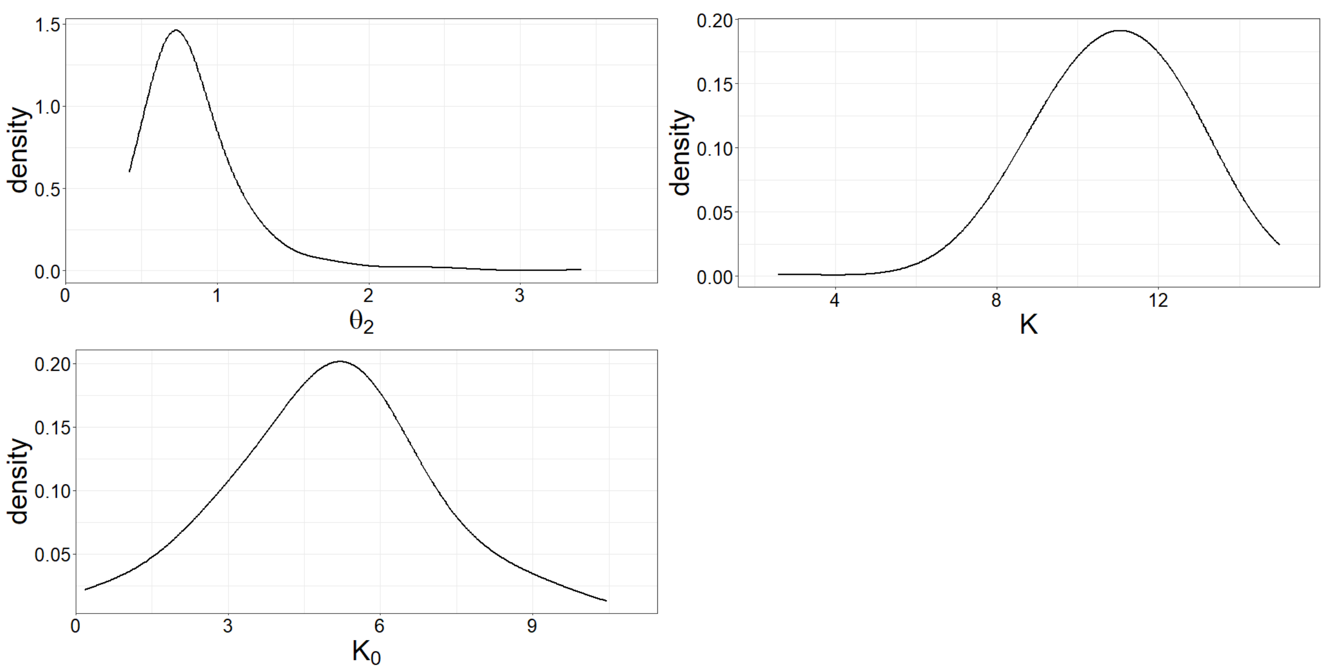

Figure 4.3: The distributions of parameters of no close form