# Bayesian Modeling of Daily Peak Electricity Demand

Akshay Umashankar* Barış Bilir† Hakeoung Hannah Lee ‡

April 25, 2023

**Keywords:** Bayesian Statistics, Time Series Analysis, Electricity Demand

## 1 Introduction

Power usage and weather patterns have become a recurring theme for the Austin area. Severe winter storms and intense heat have driven the Austin electric grid to its limit. Electricity demand is a crucial component of any power system, and grid managers must accurately forecast this demand to maintain system reliability and prevent blackouts. In Texas, a state with a swiftly expanding population and economy, modeling the daily peak electricity demand is especially crucial. According to the U.S. Energy Information Administration (EIA), Texas is the largest electricity-producing state and the largest consumer of electricity in the United States (EIA, 2022). Texas' electricity demand is expected to continue to grow in the coming years, primarily driven by the state's population and economic growth (EIA, 2022). To meet this increasing demand, Texas needs to ensure its electricity grid is capable of handling the daily peak demand, which typically occurs during the summer months when air conditioning usage is at its highest (EIA, 2022). Texas experiences significant fluctuations in electricity demand throughout the year due to its warm summers and large industrial and residential energy consumers. Accurately forecasting this peak demand is critical for grid operators to ensure system reliability and prevent power outages.

Guo et al. (2017) showed that their machine learning algorithms can be effective in short-term electricity demand forecasting. Chan et al. (2019) developed a forecast model for energy demand in Texas, which could be useful in predicting the daily peak electricity demand. Additionally, the Texas Comptroller of Public Accounts (2021) states that understanding the state's energy consumption is necessary for making informed policy decisions.

Overall, understanding these demand patterns and accurately predicting demand becomes increasingly essential as the state continues to grow in order to maintain a resilient and reliable power system. By modeling daily peak electricity demand, Texas is able to make informed decisions regarding infrastructure investments, energy policies, and grid management strategies in order to maintain a reliable and efficient electricity supply for all of its residents and businesses.

This project looks at weather and power usage data from the South Central Texas weather zone covering the Austin-San Antonio area over a 2 year period (2021 and 2022). The data was collected through the Southern Regional Climate Center and a picture of Texas weather zones are shown in Figure 1. The

*Educational Psychology, akshayumashankar38@gmail.com, au3692.
†Operations Research & Industrial Engineering, baris.bilir@austin.utexas.edu, bb39649.
‡STEM Education, hklee@utexas.edu, hl29272.

power usage data is included in Figure 2 and shows the daily peak electricity demand between 2002 and 2022. The seasonality of energy demand is clear to see.

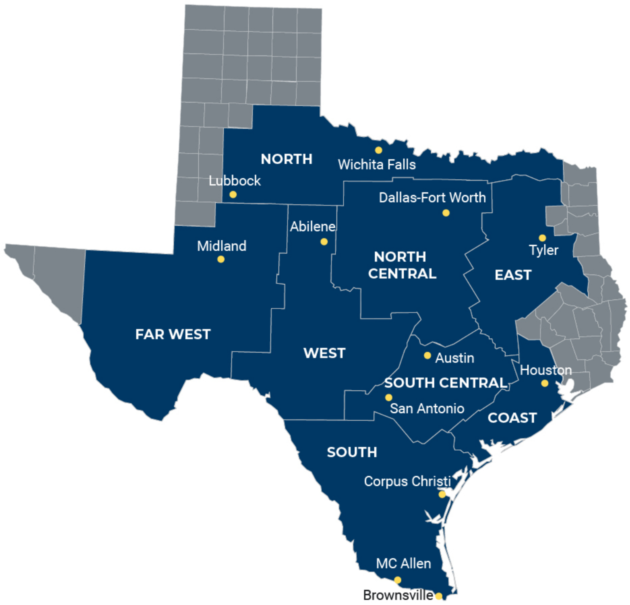


Figure 1: Texas Weather Zones.

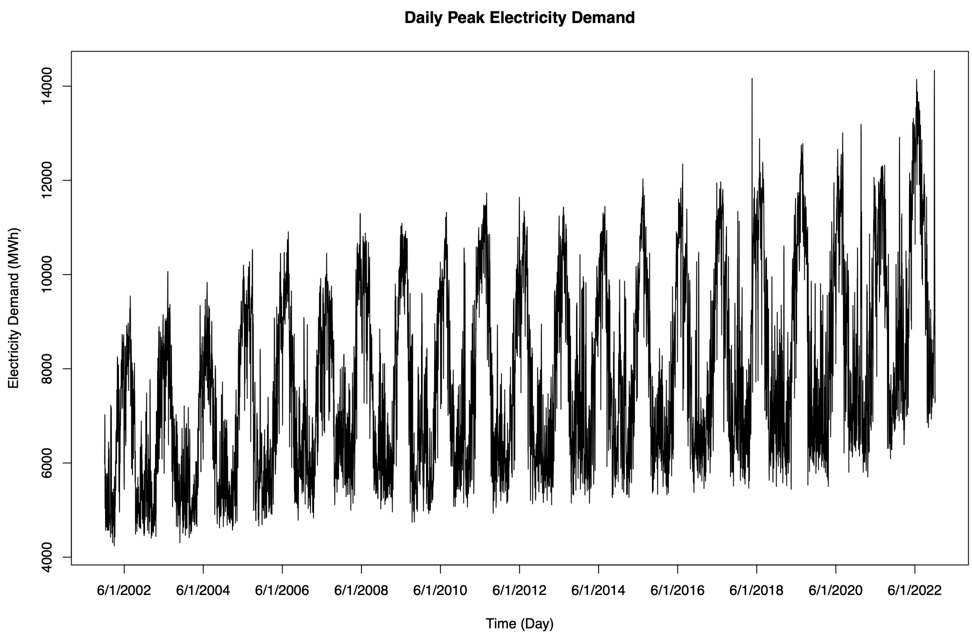


Figure 2: Demand Data.

## 2 Time Series Models

The complex and dynamic nature of the electricity demand patterns in Texas highlights the need for a time series model that can capture the trend that reflects the daily peak electricity demand. In our analysis, we used the electricity demand data for the Austin - San Antonio region over the years 2021 and 2022. To model the daily peak electricity demand, we considered two time series models: (i) hierarchical AR(1) and (ii) ARMA(1,1). Before proceeding to the details of the models, we highlight some features of the data set. While modeling the daily peak demand for South Central Texas, we first look for trend and seasonality in our data set. As can be seen in Figure 3, the data set has a clear linear trend.

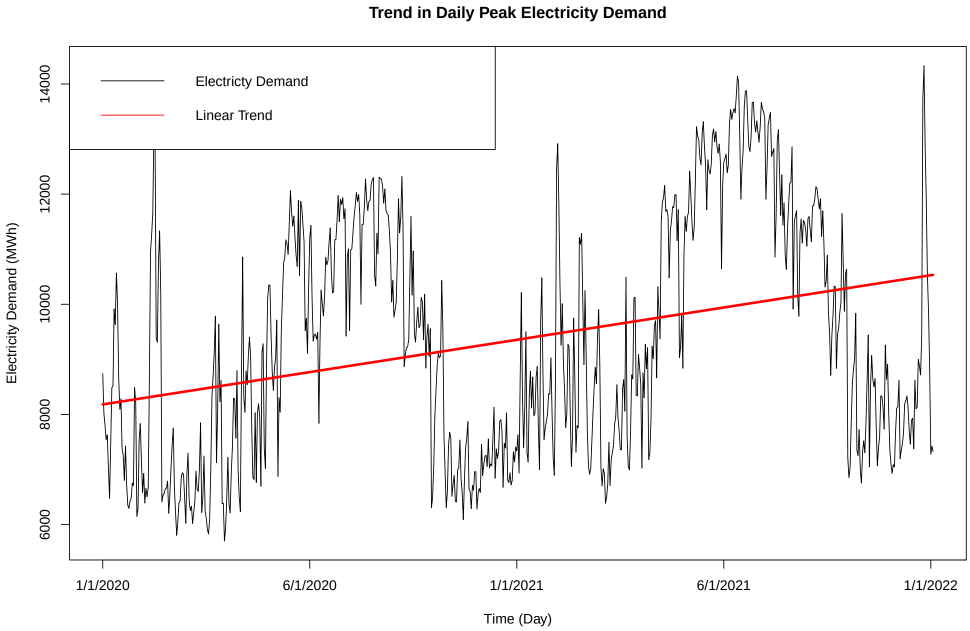


Figure 3: Linear trend in the electricity demand data.

To detect the seasonal component in the time series, after removing the trend by running a linear regression, we take the Fast Fourier Transform (FFT) of the autocorrelation function, which is also referred to as spectral density or periodgoram in the literature. The periodogram is illustrated in Figure 4. The periodogram is symmetric around 0.5, which corresponds to the Nyquist frequency. Furthermore, the periodogram is in linear scale, hence the higher freqeuncies' amplitudes are lower as expected. The four predominant frequencies are 0.0027, 0.0054, 0.0082 and 0.1428. If we take the reciprocals of these frequencies, we see that we observe seasonal patterns at 365 days, 182 days (or 6 months), 121 days (or 3 months) and 7 days (or one week).

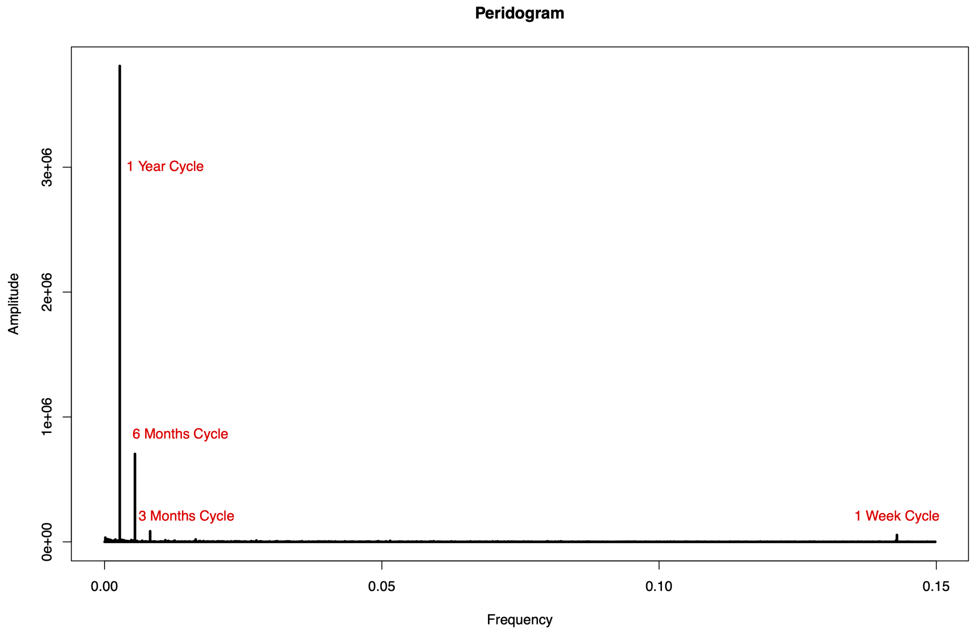


Figure 4: Seasonality captured in the electricity demand data using harmonic regression.

In our models, we take a hierarchical approach to remove trend and seasonality. In particular, we removed the trend and seasonality in electricity demand data by replacing our response variable $y_t$ with $y_t - X\beta$, where $\beta$ is a variable and $X$ is a matrix with first column is ones, second column is time, and the last eight columns are $sin(2\pi\omega t)$ and $cos(2\pi\omega t)$ for each row $t$ and $\omega \in \{0.0027, 0.0054, 0.0082, 0.1428\}$. To be more precise, the design matrix has the following form:

$$X = \begin{bmatrix} 1 & 1 & \sin\big(2\pi(0.0027)1\big) & \cos\big(2\pi(0.0027)1\big) & \dots & \sin\big(2\pi(0.1428)1\big) & \cos\big(2\pi(0.1428)1\big) \\ 1 & 2 & \sin\big(2\pi(0.0027)2\big) & \cos\big(2\pi(0.0027)2\big) & \dots & \sin\big(2\pi(0.1428)2\big) & \cos\big(2\pi(0.1428)2\big) \\ 1 & 3 & \sin\big(2\pi(0.0027)3\big) & \cos\big(2\pi(0.0027)3\big) & \dots & \sin\big(2\pi(0.1428)3\big) & \cos\big(2\pi(0.1428)3\big) \\ & & & & \vdots & & \\ 1 & T & \sin\big(2\pi(0.0027)T\big) & \cos\big(2\pi(0.0027)T\big) & \dots & \sin\big(2\pi(0.1428)T\big) & \cos\big(2\pi(0.1428)T\big) \end{bmatrix}$$

To illustrate how we captured seasonality, after removing the linear trend, by running the harmonic regression, we can clearly see the covariates capture the seasonal patterns, as can be seen in Figure 5.

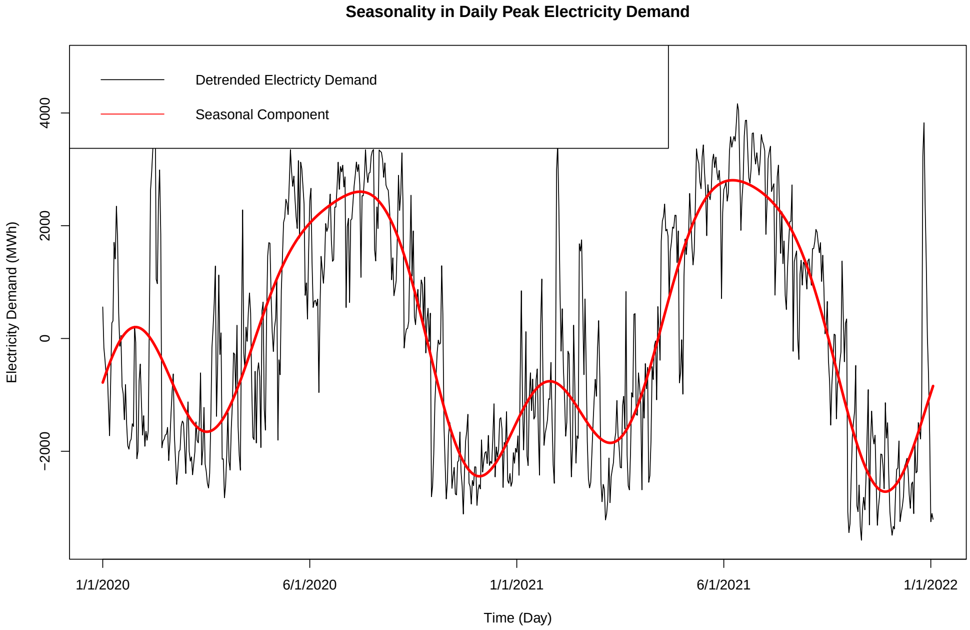


Figure 5: Seasonality captured in the electricity demand data using harmonic regression.

We explicitly modeled the linear trend and the seasonality through hierarchical modeling. In the remainder of this section, we present the details of the models and results after fitting the models with MCMC, calculating their DICs and potential scale reduction factors.

## 2.1 Hierarchical AR(1) Model

The hierarchical AR(1) model is as follows:

$$
\begin{aligned}
&y_t \sim N\big((x_t - \rho x_{t-1})'\beta + \rho y_{t-1}, \sigma^2\big) \quad \text{for } t = 2, \ldots, T, \\
&\beta \sim N(\mu_\beta, \Sigma_\beta) \\
&\sigma^2 \sim IG\big(a_0, b_0\big) \\
&\rho \sim \text{Beta}(a_1, b_1)
\end{aligned}
$$

### 2.2 ARMA(1,1) Model

The hierarchical ARMA(1) model is as follows:

$$
\begin{aligned}
&y_t \sim N\big((x_t - \rho x_{t-1})^{'}\beta + \rho y_{t-1} + \theta\epsilon_{t-1}, \sigma^2\big) \quad \text{for } t = 2, \ldots, T, \\
&\epsilon_t = \begin{cases} y_t - (x_t - \rho x_{t-1})^{'}\beta + \rho y_{t-1} & \text{for } t = 2, \ldots, T, \\ y_1 - x_1^{'}\beta & \text{for } t = 1 \end{cases} \\
&\beta \sim N(\mu_\beta, \Sigma_\beta) \\
&\sigma^2 \sim IG\big(\mu_0, \sigma_0^2\big) \\
&\rho \sim N(\mu_1, \sigma_1^2) \\
&\theta \sim N(\mu_2, \sigma_2^2)
\end{aligned}
$$

# 3 Analysis and Conclusion

In our implementation, we use hyperparameters for the AR(1) and ARMA(1,1), given in Table 2 and Table 3, respectively. For both cases, we burn-in first 100,000 samples and sampled 500,000 samples from the posterior distributions. While choosing our hyperparameters for beta, we run harmonic regression and choose MLE estimates as our hyperparameters. For $\rho$ in AR we set the hyperparameters such that the prior is uniform over $[0, 1]$. For ARMA, choose hyperparameters for the priors such that the prior distribution over $\rho$ and $\theta$ is $N(0, 2)$. It is noteworthy that $\rho$ and $\theta$ should be between -1 and 1 to get a stationary process. Both variables' posteriors with support in this range, so we didn't need to update our choice of hyperparameters.

| Parameter | Value |
|---|---|
| $a_0$ | 1 |
| $b_0$ | 1 |
| $a_1$ | 1 |
| $b_1$ | 1 |
| $\mu_\beta$ | (8.98153, 0.00037, -0.20623, 0.06325, 0.02293, 0.00962, -0.05738, 0.09979, 0.06735, -0.00103) |
| $\sigma_\beta$ | (2.5e+03, 1.0e+02, 2.5e+03, 2.5e+03, 2.5e+03, 2.5e+03, 2.5e+03, 2.5e+03, 2.5e+03, 2.5e+03) |

Table 1: Hyperparameters for AR(1) Model

| Parameter | Value |
|---|---|
| $\mu_0$ 1 | |
| $\sigma_0^2$ | 1 |
| $\mu_1$ | 0 |
| $\sigma_1^2$ | 2 |
| $\mu_2$ | 0 |
| $\sigma_2^2$ | 2 |
| $\theta$ | 2 |
| $\mu_\beta$ | (8.98153, 0.00037, -0.20623, 0.06325, 0.02293, 0.00962, -0.05738, 0.09979, 0.06735, -0.00103) |
| $\sigma_\beta$ | (2.5e+03, 1.0e+02, 2.5e+03, 2.5e+03, 2.5e+03, 2.5e+03, 2.5e+03, 2.5e+03, 2.5e+03, 2.5e+03) |

Table 2: Hyperparameters for AR(1) Model

For AR(1) and ARMA(1,1) models posterior distribuion of the variables are given in Figures 18, 19 and Figures **??**, **??**, respectively in Appendix B. It is noteworthy that the posterior distribution of the moving average term $\theta$ is centered around zero with very small variance. Hence, there is no reason to use ARMA(1,1) over AR(1) model for our data set.

The 95% confidence intervals for the variables in AR(1) and ARMA(1,1) models are given in Table 3 and Table 4, respectively. As indicated before, the moving average term $\theta$ in the ARMA(1,1) model is not significantly different than zero since the credible region includes zero. Furthermore, the harmonic cosine term that corresponds to 1-week cycle is not significantly different than zero.

| | Lower C.I. | Upper C.I. |
|---|---|---|
| $\beta_0$ | 8.97908 | 9.03086 |
| $\beta_1$ | 0.0003728116 | 0.0004979121 |
| $\beta_2$ | -0.2098177 | -0.1767726 |
| $\beta_3$ | 0.05966574 | 0.09252899 |
| $\beta_4$ | 0.01943025 | 0.05200362 |
| $\beta_5$ | 0.009797032 | 0.021325121 |
| $\beta_6$ | -0.05743003 | -0.02132014 |
| $\beta_7$ | 0.09945181 | 0.13310670 |
| $\beta_8$ | 0.06674786 | 0.09971223 |
| $\beta_9$ | -0.00116527 | 0.01037404 |
| $\rho$ | 0.7201444 | 0.7746391 |
| $\tau$ | 89.49215 | 97.45367 |

Table 3: AR(1) Credible Regions

| | Lower C.I. | Upper C.I. |
|---|---|---|
| $\beta_0$ | 8.97908 | 9.03086 |
| $\beta_1$ | 0.0003728116 | 0.0004979121 |
| $\beta_2$ | -0.2098177 | -0.1767726 |
| $\beta_3$ | 0.05966574 | 0.09252899 |
| $\beta_4$ | 0.01943025 | 0.05200362 |
| $\beta_5$ | 0.009797032 | 0.021325121 |
| $\beta_6$ | -0.05743003 | -0.02132014 |
| $\beta_7$ | 0.09945181 | 0.13310670 |
| $\beta_8$ | 0.06674786 | 0.09971223 |
| $\beta_9$ | -0.00116527 | 0.01037404 |
| $\rho$ | 0.7195274 | 0.7749183 |
| $\tau$ | 89.41014 | 97.41156 |
| $\theta$ | -0.001742537 | 0.010311335 |

Table 4: ARMA(1,1) Credible Regions

The potential scale reduction factors for AR(1) and ARMA(1,1) are reported in Table 5 and 6. We run 5 chains with length 500,000 to get the potential scale factors. Recall, when potential scale factors are close to 1, it indicates that the MCMC algorithm converged. The results indicate that the MCMC algoirthms for AR(1) and ARMA(1,1) model are converged.

| | Point Est. | Upper C.I. |
|---|---|---|
| $\beta_0$ | 1.00 | 1.00 |
| $\beta_1$ | 1.00 | 1.00 |
| $\beta_2$ | 1.00 | 1.00 |
| $\beta_3$ | 1.00 | 1.00 |
| $\beta_4$ | 1.00 | 1.00 |
| $\beta_5$ | 1.00 | 1.00 |
| $\beta_6$ | 1.00 | 1.00 |
| $\beta_7$ | 1.00 | 1.00 |
| $\beta_8$ | 1.00 | 1.00 |
| $\beta_9$ | 1.00 | 1.00 |
| $\rho$ | 1.00 | 1.00 |
| $\tau$ | 1.00 | 1.00 |

Table 5: AR - Multivariate PSRF - 1.00

| | Point Est. | Upper C.I. |
|---|---|---|
| $\beta_0$ | 1.01 | 1.01 |
| $\beta_1$ | 1.00 | 1.00 |
| $\beta_2$ | 1.00 | 1.00 |
| $\beta_3$ | 1.00 | 1.00 |
| $\beta_4$ | 1.00 | 1.00 |
| $\beta_5$ | 1.00 | 1.00 |
| $\beta_6$ | 1.00 | 1.00 |
| $\beta_7$ | 1.00 | 1.00 |
| $\beta_8$ | 1.00 | 1.00 |
| $\beta_9$ | 1.00 | 1.00 |
| $\rho$ | 1.00 | 1.00 |
| $\tau$ | 1.00 | 1.00 |
| $\theta$ | 1.01 | 1.02 |

Table 6: ARMA - Multivariate PSRF - 1.01

The DIC for both models are reported in Table 7. AR(1) model has lower DIC, and consequently we would choose AR(1) over ARMA(1,1).

| Model | $\overline{D}$ | pD | DIC |
|---|---|---|---|
| AR(1) | -1386 | 12.15 | -1373.85 |
| ARMA(1,1) | -1385 | 12.90 | -1372.10 |

Table 7: DIC Calculations for Both Models

Finally, we simulate the posterior predictive distribution to get the Credible intervals over our time series data and predict the daily peak electricity demand for the year 2023. For AR(1) and ARMA(1,1), the Figures 6, 7 are given below. Notice that the credible regions for the ARMA(1,1) model is very wide unlike AR(1) model. The reason might be the fact that the moving average therm $\theta$ is very close to zero and as it can be seen from the trace plot for $\theta$ in Appendix B, there are some convergence issues for $\theta$ in the ARMA(1,1) model. The trace plots for the other variables are robust enough for this model.

In conclusion, modeling the daily peak electricity demand in Texas is critical to ensure a reliable and efficient power system. As the state's population and economy continue to grow, understanding the patterns and accurately forecasting peak demand becomes increasingly important for maintaining a stable electricity supply. Through the use of our model, Texas may make informed decisions about infrastructure investments, energy policies, and grid management strategies. By doing so, the state can continue to meet the increasing demand for electricity and ensure that all its residents and businesses have access to a reliable and affordable power supply.

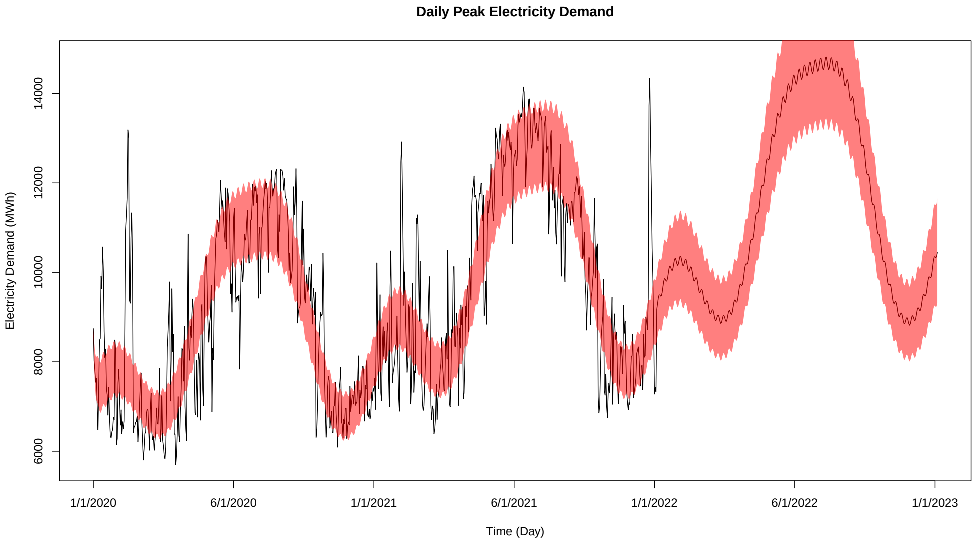


Figure 6: Credible intervals and forecast for AR(1) model.

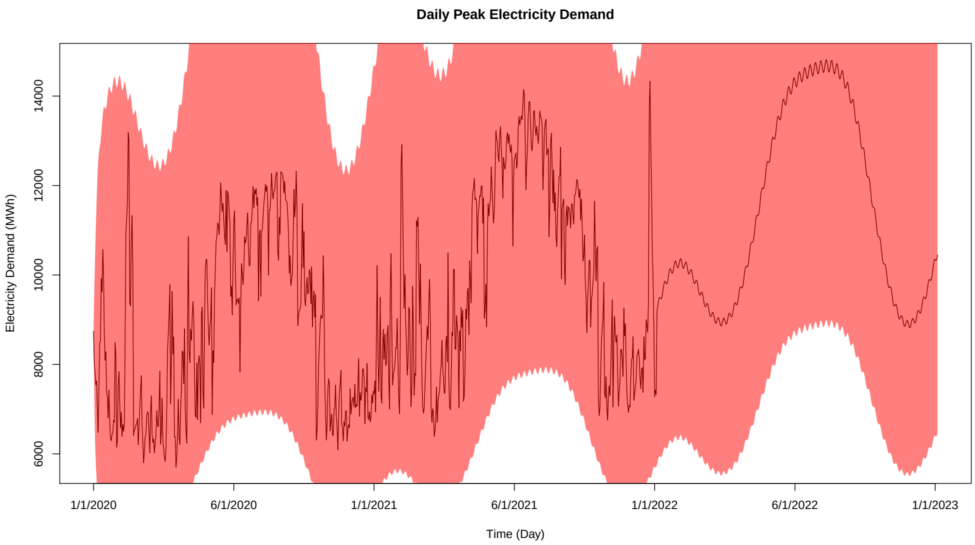


Figure 7: Credible intervals and forecast for ARMA(1,1) model.

# References

Chan, W. L., Islam, S., Sumner, D. A. (2019). A forecast model for energy demand in Texas. Energy, 167, 630-639. doi: 10.1016/j.energy.2018.11.009

Guo, J., Kim, Y. M., Lee, S. Y., Akin, S. (2017). Short-term electricity demand forecasting using machine learning algorithms: An experimental study. Energies, 10(5), 659. doi: 10.3390/en10050659

Texas Comptroller of Public Accounts. (2021). Texas power: Electricity in Texas. Retrieved from https://comptroller.texas.gov/about/policies/energy.php

U.S. Energy Information Administration. (2022, February 2). Texas State Profile and Energy Estimates. Retrieved from https://www.eia.gov/state/?sid=TX

# Appendix A: Trace Plots

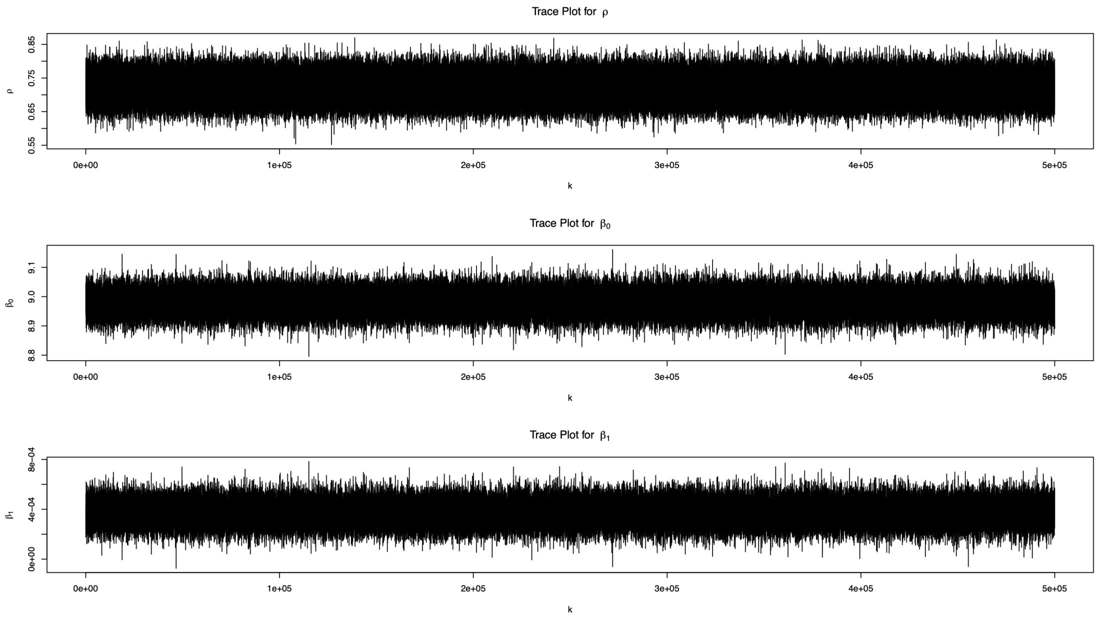


Figure 8: AR Trace Plot

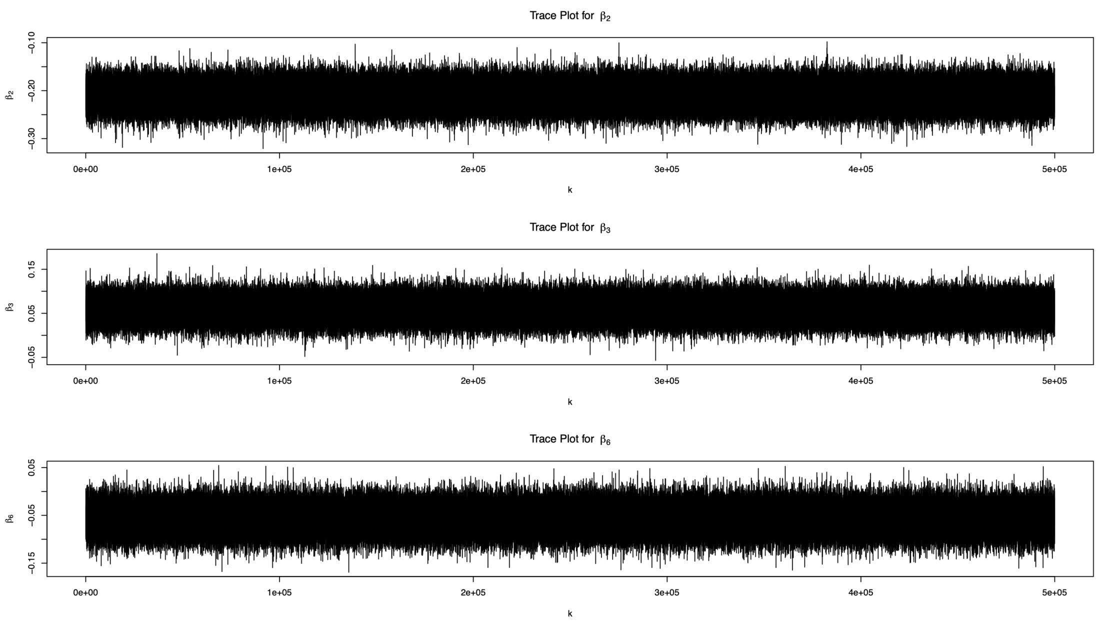


Figure 9: AR Trace Plot

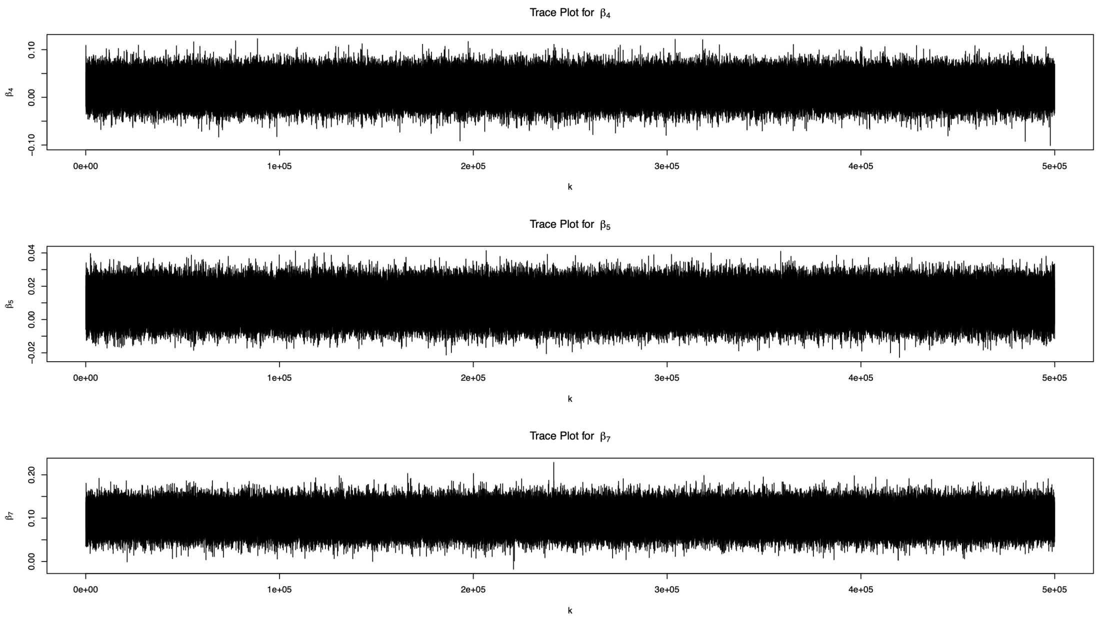


Figure 10: AR Trace Plot

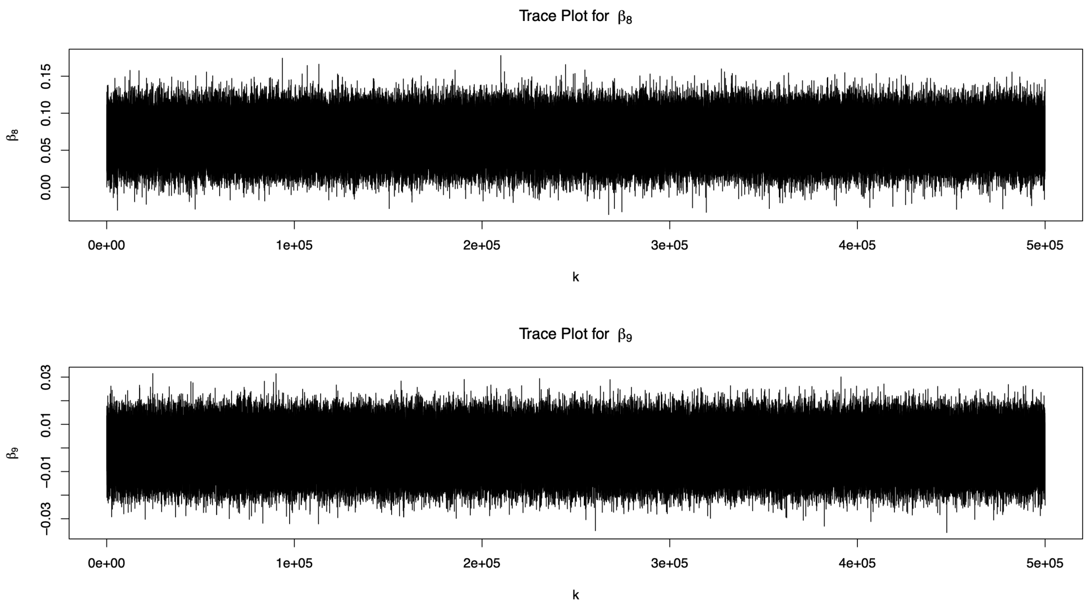


Figure 11: AR Trace Plot

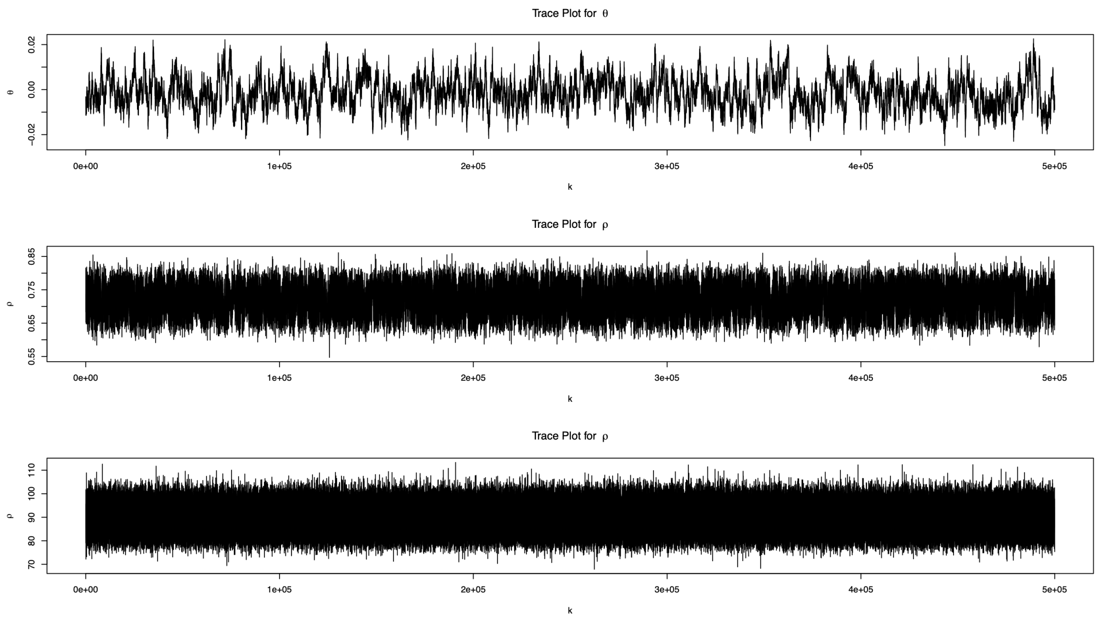


Figure 12: AR Trace Plot

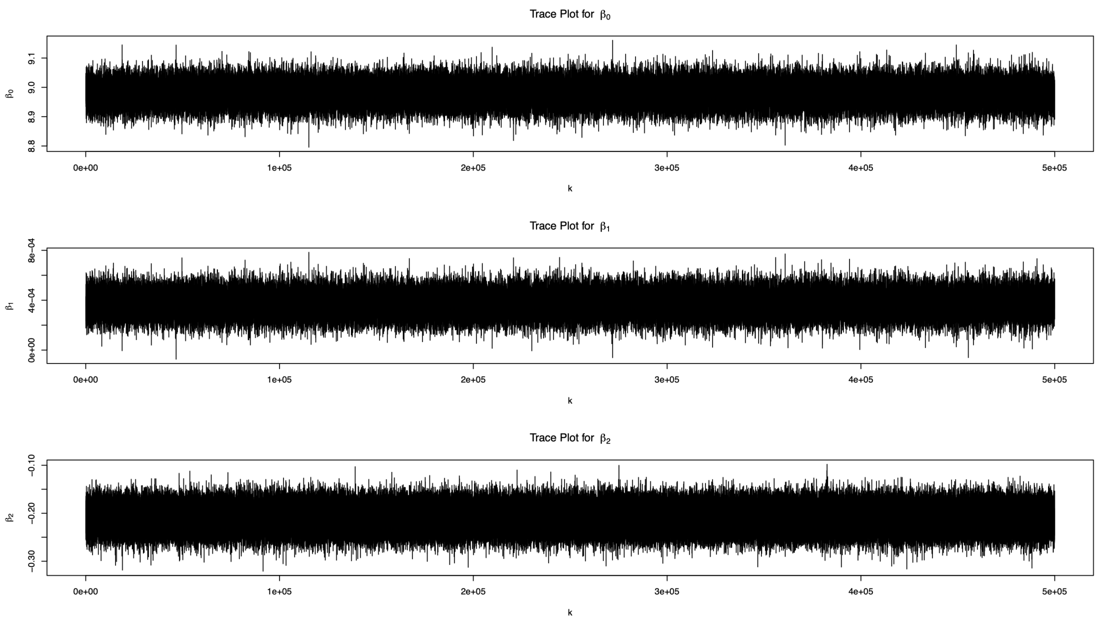


Figure 13: AR Trace Plot

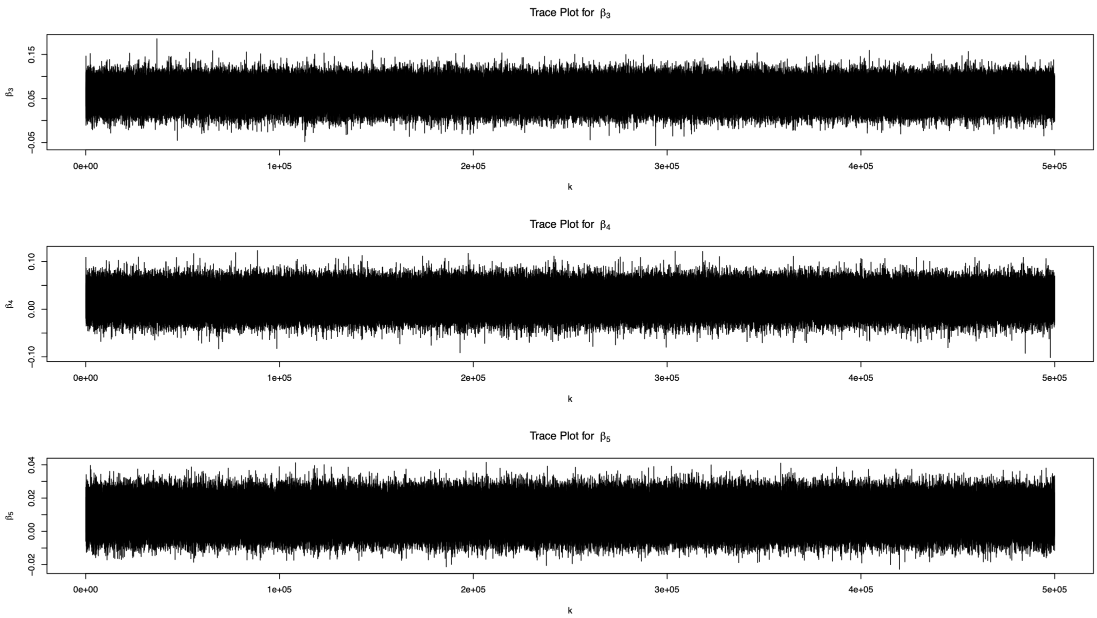


Figure 14: AR Trace Plot

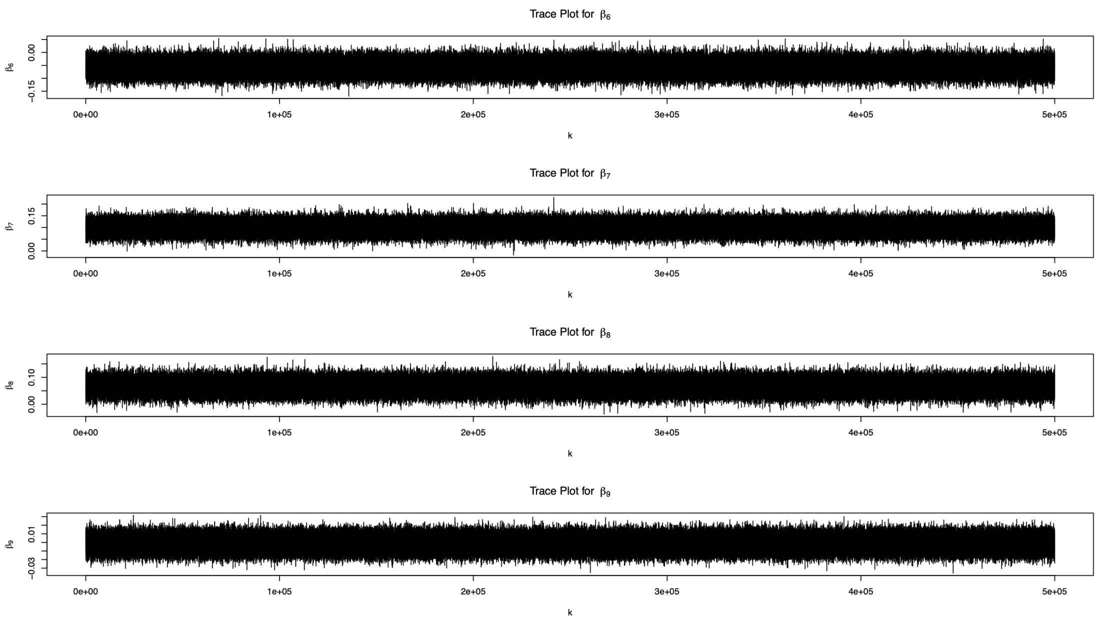


Figure 15: AR Trace Plot

# Appendix B

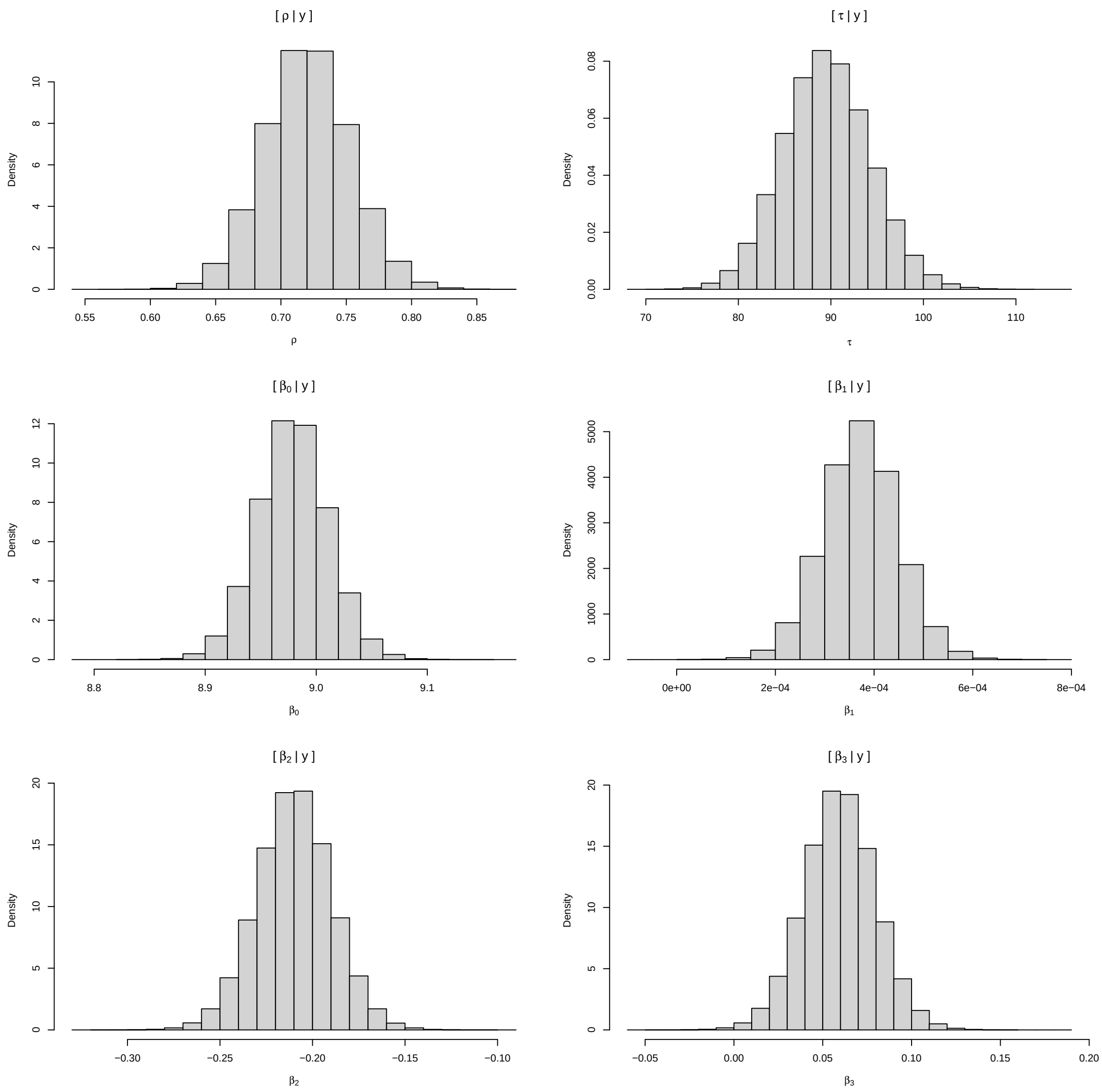


Figure 16: AR Posterior 1

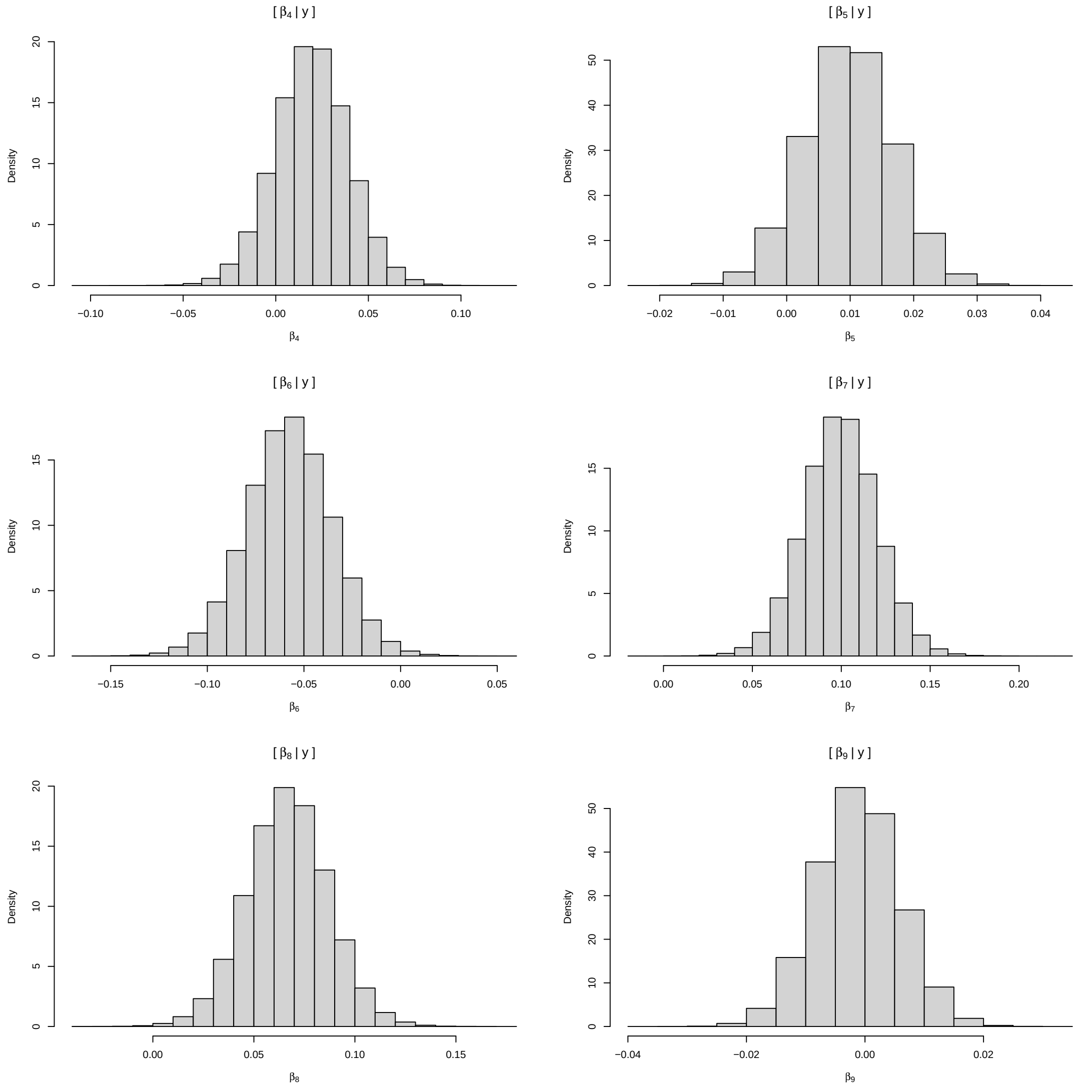


Figure 17: AR Posterior 2

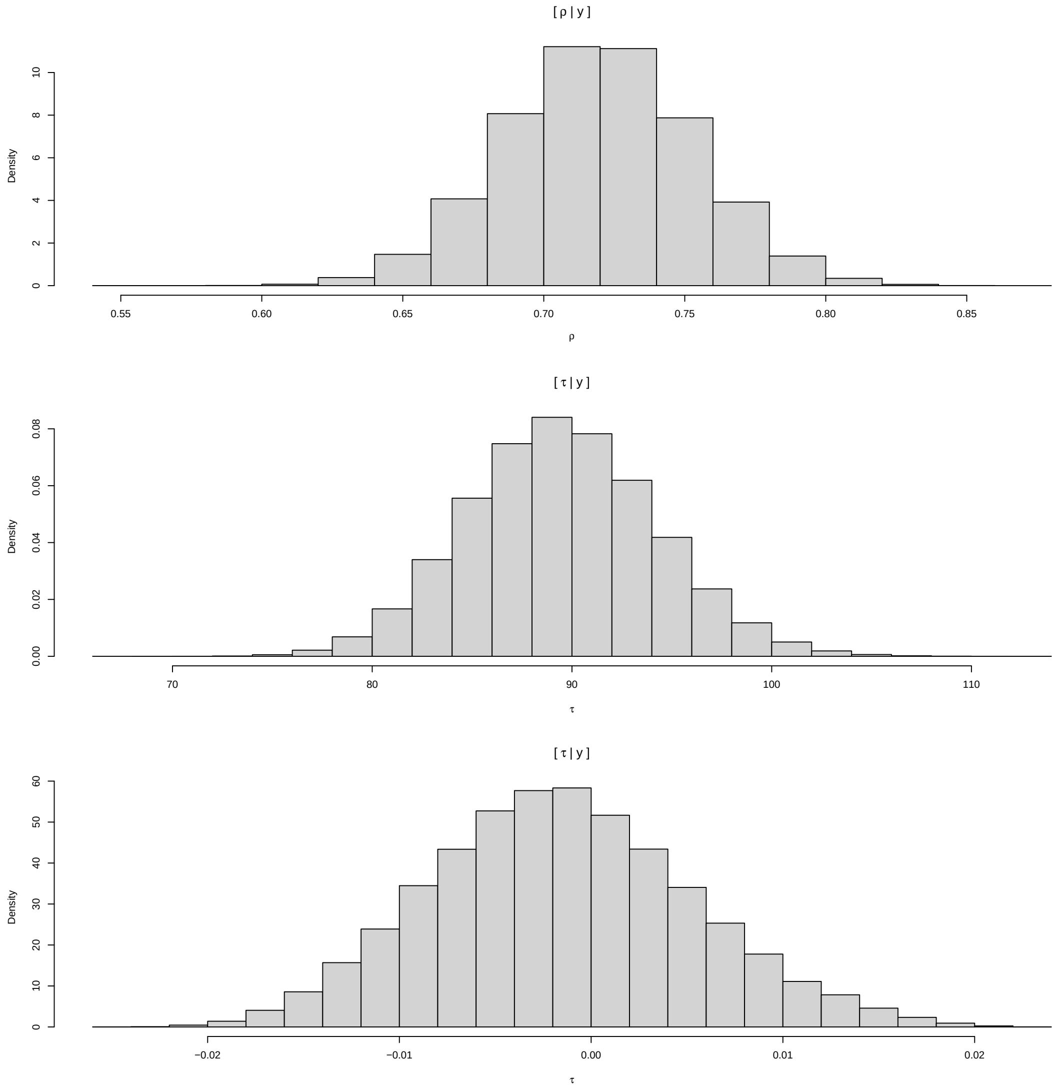


Figure 18: AR Posterior 1

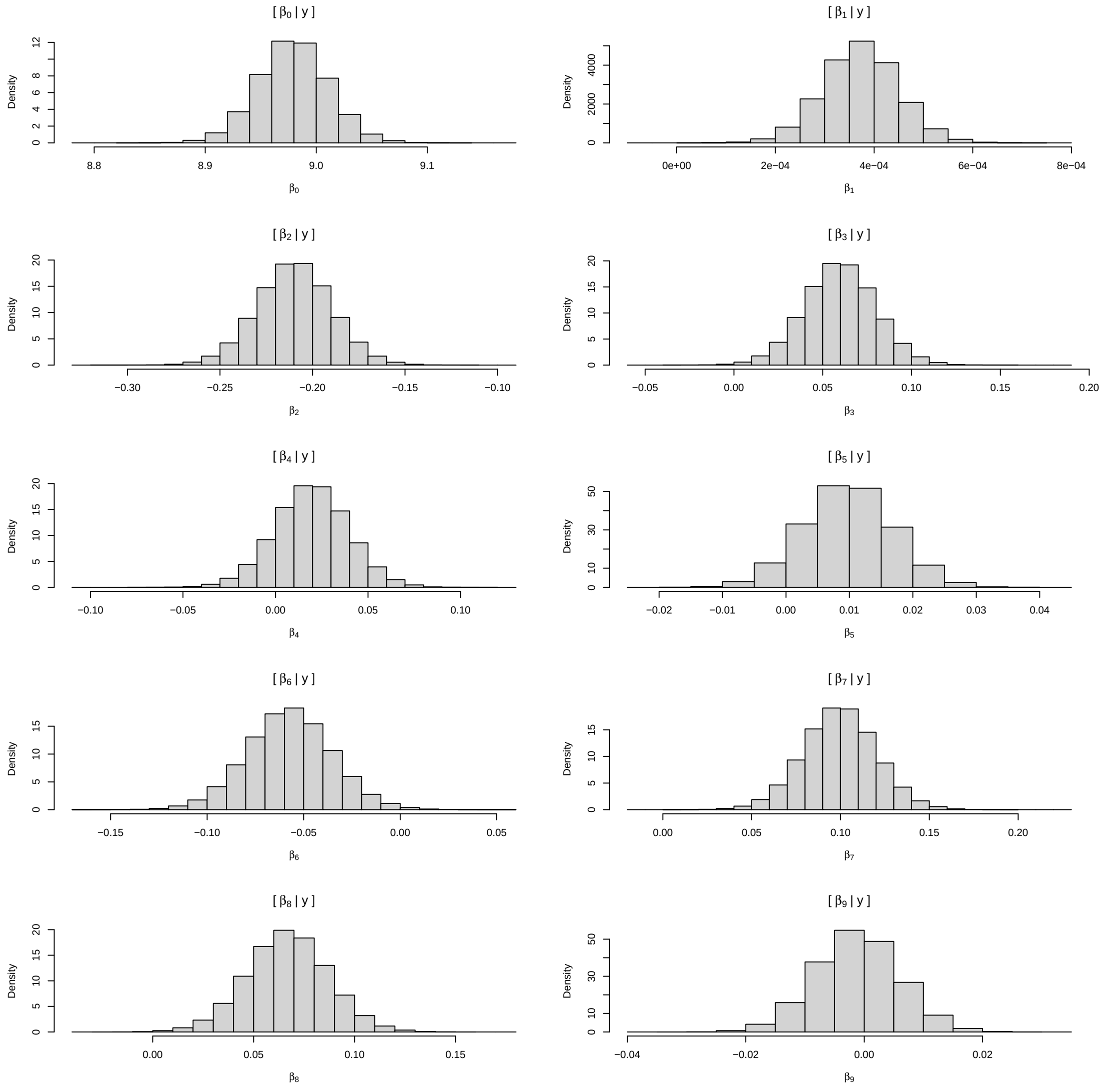


Figure 19: AR Posterior 2

# Appendix C: R Script

You can find a brief description of each file below:

- *AR.R*: This file runs AR(1) model.
- *ARMA.R*: This file runs ARMA(1,1) model.
- *model1.R*: This file contains the JAGS code for AR(1) model.
- *model2.R*: This file contains the code for ARMA(1,1) model.

- *DIC_AR.R*: This file contains the code for computing the DIC for AR(1) model.
- *DIC_ARMA.R*: This file contains the code for computing the DIC for ARMA(1,1) model.
- *psrf_AR.R* This file contains the code for computing the psrf for AR(1) model.
- *psrf_ARMA.R* This file contains the code for computing the psrf for ARMA(1,1) model.
- *AR_predictive.R* This file contains the code for get samples from the predictive distribution of the AR(1) model.
- *ARMA_predictive.R* This file contains the code for get samples from the predictive distribution of the ARMA(1,1) model.

## AR.R

```
setwd("/home1/09315/bilir/projects/bayesian_demand_reduced")

model <- ’model1’

library(coda)
library(rjags)

# parameters
n.mcmc = 500000 # number of posterior samples
n.burn=round(.2*n.mcmc) # burn=in period
n.chains=1 # number of chains
year.pred <- 1 #numer of years to predict

# source jags model
model.file <- paste0(’utils/’,model, ".R")
source(model.file)

# build model graph
m.out<-jags.model(mod,
                  data=data,
                  n.chains=n.chains,
                  n.adapt=0)

# include burn-in period
update(m.out, n.burn)

# get mcmc samples
mcmc.out <- jags.samples(m.out,params, n.mcmc)

save.file <- paste0("/work2/09315/bilir/", model, "_reduced.rds")
saveRDS(mcmc.out, file=save.file)
```

## ARMA.R

```
setwd("/home1/09315/bilir/projects/bayesian_demand_reduced")

model <- 'model2'

library(coda)
library(rjags)

# parameters
n.mcmc = 500000 # number of posterior samples
n.burn=round(.2*n.mcmc) # burn=in period
n.chains=1 # number of chains
year.pred <- 1  #numer of years to predict

# source jags model
model.file <- paste0('utils/',model, ".R")
source(model.file)

# build model graph
m.out<-jags.model(mod,
                 data=data,
                 n.chains=n.chains,
                 n.adapt=0)

# include burn-in period
update(m.out, n.burn)

# get mcmc samples
mcmc.out <- jags.samples(m.out,params, n.mcmc)

save.file <- paste0("/work2/09315/bilir/", model, "_reduced.rds")
saveRDS(mcmc.out, file=save.file)
```

### model1.R

```
# Hierarchical AR(1) model
m.jags <- "
    model{
      # likelihood
      for(t in 2:T.index){
        y[t] ~ dnorm( (X[t,] - rho*X[t-1,])%*%beta + rho*y[t-1], tau)
      }

      # priors
      for (j in 1:p){
        beta[j] ~ dnorm(mu.beta[j],tau.beta[j])
      }
      tau ~ dgamma(a.0,b.0)
      rho ~ dbeta(a.1,b.1)
    }
  "

```

```
# read data
data <- read.csv('/work2/09315/bilir/projects_data/bayesian_demand/bayesian_
    project.csv')

# dependent, independent variables
y <- data$DEMAND
y <- y[(length(y)-365*2):length(y)]
y <- log(y)

T.index <- length(y)
T.pred <- T.index + 365*year.pred

w <- c(0.00273794, 0.00547588, 0.00821382, 0.1428944)
X <- matrix(1, nrow=T.index, ncol = 10)
X[,1] <- rep(1, T.index)
X[,2] <- 1:T.index
for(i in 1:length(w)){
  col.init <- 2*(1:T.index)*w[i]
  col.cos <- sapply(col.init, FUN = cospi)
  col.sin <- sapply(col.init, FUN = sinpi)
  X[,i+2] <- col.cos
  X[,i+6] <- col.sin
}

harmonic.X <- data.frame('y'=y, 'trend'=X[,2],
                         'cos.w1'=X[,3], 'cos.w2'=X[,4],
                         'cos.w3'=X[,5], 'cos.w4'=X[,6],
                         'sin.w1'=X[,7], 'sin.w2'=X[,8],
                         'sin.w3'=X[,9], 'sin.w4'=X[,10])

# solve mle for betas
harmonic.fit <- lm(y ~ trend + cos.w1 + cos.w2 + cos.w3 + cos.w4 +
                     sin.w1 + sin.w2 + sin.w3 + sin.w4, data=harmonic.X)

# hyperparameters
mu.beta <- as.vector(harmonic.fit$coefficients)
tau.beta <- rep(2, 10)
a.0 <- 1
b.0 <- 1
a.1 <- 1
b.1 <- 1

# parameters
p <- dim(X)[2]
# read model
mod<-textConnection(m.jags)

data=list('y'=y,
          'X'=X,
          'p'=p,
          'T.index'=T.index,
          'mu.beta'=mu.beta,
          'tau.beta'=tau.beta,
```

```
          'a.0'=a.0,
          'b.0'=b.0,
          'a.1'=a.1,
          'b.1'=b.1)

params <- c('rho', 'beta', 'tau')
```

### model2.R

```
# Hierarchical ARMA(1,1) model
m.jags <- "
  model{
  # likelihood
  eps[1] <- y[1] - X[1,]%*%beta
  for(t in 2:T.index){
  y[t] ~ dnorm( (X[t,] - rho*X[t-1,])%*%beta + rho*y[t-1] + theta*eps[t-1], tau)
  eps[t] <- y[t] - (X[t,] - rho*X[t-1,])%*%beta + rho*y[t-1] + theta*eps[t-1]
  }

  # priors
  for (j in 1:p){
  beta[j] ~ dnorm(mu.beta[j],tau.beta[j])
  }
  tau ~ dgamma(a.0,b.0)
  rho ~ dnorm(rho.mu,rho.tau)
  theta ~ dnorm(theta.mu,theta.tau)
  }
"

# read data
data <- read.csv('/work2/09315/bilir/projects_data/bayesian_demand/bayesian_project.csv')

# dependent, independent variables
y <- data$DEMAND
y <- y[(length(y) -365*2): length(y)]
y <- log(y)

T.index <- length(y)
T.pred <- T.index + 365*year.pred

w <- c(0.00273794, 0.00547588, 0.00821382, 0.1428944)
X <- matrix(1, nrow=T.index, ncol = 10)
X[,1] <- rep(1, T.index)
X[,2] <- 1:T.index
for(i in 1:length(w)){
  col.init <- 2*(1:T.index)*w[i]
  col.cos <- sapply(col.init, FUN = cospi)
  col.sin <- sapply(col.init, FUN = sinpi)
  X[,i+2] <- col.cos
```

```
  X[,i+6] <- col.sin
}

harmonic.X <- data.frame('y'=y, 'trend'=X[,2],
                         'cos.w1'=X[,3], 'cos.w2'=X[,4],
                         'cos.w3'=X[,5], 'cos.w4'=X[,6],
                         'sin.w1'=X[,7], 'sin.w2'=X[,8],
                         'sin.w3'=X[,9], 'sin.w4'=X[,10])

X.tilde <- matrix(1, nrow=T.pred, ncol = 10)
X.tilde[,1] <- rep(1, T.pred)
X.tilde[,2] <- 1:T.pred
for(i in 1:length(w)){
  col.init <- 2*(1:T.pred)*w[i]
  col.cos <- sapply(col.init, FUN = cospi)
  col.sin <- sapply(col.init, FUN = sinpi)
  X.tilde[,i+2] <- col.cos
  X.tilde[,i+6] <- col.sin
}

# solve mle for betas
harmonic.fit <- lm(y ~ trend + cos.w1 + cos.w2 + cos.w3 + cos.w4 +
                     sin.w1 + sin.w2 + sin.w3 + sin.w4, data=harmonic.X)

# hyperparameters
mu.beta <- as.vector(harmonic.fit$coefficients)
tau.beta <- rep(2,10)
a.0 = 1
b.0 =1
rho.mu <- 0
rho.tau <- 1/2
theta.mu <- 0
theta.tau <- 1/2

# parameters
p <- dim(X)[2]
# read model
mod<-textConnection(m.jags)

data=list('y'=y,
          'X'=X,
          'p'=p,
          'T.index'=T.index,
          'mu.beta'=mu.beta,
          'a.0'=a.0,
          'b.0'=b.0,
          'tau.beta'=tau.beta,
          'rho.mu'=rho.mu,
          'rho.tau'=rho.tau,
          'theta.mu'=theta.mu,
          'theta.tau'=theta.tau)

params <- c('rho', 'theta', 'beta', 'tau')
```

**DIC_AR.R**

```r
setwd("/home1/09315/bilir/projects/bayesian_demand_reduced")

model <- 'model1'

library(coda)
library(rjags)

# parameters
n.mcmc = 100000 # number of posterior samples
n.burn=round(.2*n.mcmc) # burn=in period
n.chains=2 # number of chains
year.pred <- 5 #numer of years to predict

# source jags model
model.file <- paste0("utils/model1.R")

source(model.file)

# build model graph
m.out<-jags.model(mod,
                  data=data,
                  n.chains=n.chains,
                  n.adapt=0)

# include burn-in period
update(m.out, n.burn)

cat('AR(1)', '\n')
# get dic
model.dic <- dic.samples(m.out, n.iter=n.mcmc,type='pD')
print(model.dic)
```

**DIC_ARMA.R**

```r
setwd("/home1/09315/bilir/projects/bayesian_demand_reduced")

model <- 'model2'

library(coda)
library(rjags)

# parameters
n.mcmc = 100000 # number of posterior samples
n.burn=round(.2*n.mcmc) # burn=in period
n.chains=2 # number of chains
year.pred <- 5 #numer of years to predict

# source jags model
model.file <- paste0("utils/model2.R")
```

```

source(model.file)

# build model graph
m.out<-jags.model(mod,
                  data=data,
                  n.chains=n.chains,
                  n.adapt=0)

# include burn-in period
update(m.out, n.burn)

cat('ARMA', '\n')
# get dic
model.dic <- dic.samples(m.out, n.iter=n.mcmc,type='pD')
print(model.dic)
```

## psrf_AR.R

```
setwd("/home1/09315/bilir/projects/bayesian_demand_reduced")

model <- 'model1'

library(coda)
library(rjags)

# parameters
n.mcmc = 100000 # number of posterior samples
n.burn=round(.2*n.mcmc) # burn=in period
n.chains=5 # number of chains
year.pred <- 5 #numer of years to predict

# source jags model
model.file <- paste0("utils/model1.R")

source(model.file)

# build model graph


cat('AR', '\n')

# build model graph
m.out<-jags.model(mod,
                  data=data,
                  n.chains=n.chains,
                  n.adapt=0)

# include burn-in period
update(m.out, n.burn)

```

```
posterior.samples <- coda.samples(m.out,params, n.mcmc)
psrf <- gelman.diag(posterior.samples)
print(psrf)
```

### psrf_ARMA.R

```
setwd("/home1/09315/bilir/projects/bayesian_demand_reduced")

model <- 'model2'

library(coda)
library(rjags)

# parameters
n.mcmc = 500000 # number of posterior samples
n.burn=round(.2*n.mcmc) # burn=in period
n.chains=5 # number of chains
year.pred <- 5 #numer of years to predict

# source jags model
model.file <- paste0("utils/model2.R")

source(model.file)

# build model graph


cat('ARMA', '\n')

# build model graph
m.out<-jags.model(mod,
                  data=data,
                  n.chains=n.chains,
                  n.adapt=0)

# include burn-in period
update(m.out, n.burn)

posterior.samples <- coda.samples(m.out,params, n.mcmc)
psrf <- gelman.diag(posterior.samples)
print(psrf)
```

### psrf_ARMA.R

```
setwd("/home1/09315/bilir/projects/bayesian_demand_reduced")

model <- 'model2'

library(coda)
```

```
library(rjags)

# parameters
n.mcmc = 500000 # number of posterior samples
n.burn=round(.2*n.mcmc) # burn=in period
n.chains=5 # number of chains
year.pred <- 5 #numer of years to predict

# source jags model
model.file <- paste0("utils/model2.R")

source(model.file)

# build model graph


cat('ARMA', '\n')

# build model graph
m.out<-jags.model(mod,
                  data=data,
                  n.chains=n.chains,
                  n.adapt=0)

# include burn-in period
update(m.out, n.burn)

posterior.samples <- coda.samples(m.out,params, n.mcmc)
psrf <- gelman.diag(posterior.samples)
print(psrf)
```

## psrf_ARMA.R

```
setwd("/home1/09315/bilir/projects/bayesian_demand_reduced")

model <- 'model2'

library(coda)
library(rjags)

# parameters
n.mcmc = 500000 # number of posterior samples
n.burn=round(.2*n.mcmc) # burn=in period
n.chains=5 # number of chains
year.pred <- 5 #numer of years to predict

# source jags model
model.file <- paste0("utils/model2.R")

source(model.file)

```

```
# build model graph


cat('ARMA', '\n')

# build model graph
m.out<-jags.model(mod,
                  data=data,
                  n.chains=n.chains,
                  n.adapt=0)

# include burn-in period
update(m.out, n.burn)

posterior.samples <- coda.samples(m.out,params, n.mcmc)
psrf <- gelman.diag(posterior.samples)
print(psrf)
```

## AR_predictive.R

```
# read data
model <- 'model1'
file.name <- paste0('/work2/09315/bilir/AR.rds')
mcmc.out <- readRDS(file.name)

beta <- t(as.matrix(mcmc.out$beta[,,1]))
rho <- as.vector(mcmc.out$rho[,,1])
s2 <- 1/as.vector(mcmc.out$tau[,,1])

remove(mcmc.out)


T.data <- 731
T.pred <- T.data + 365*1
n.mcmc <- length(rho)

w <- c(0.00273794, 0.00547588, 0.00821382, 0.1428944)
X.tilde <- matrix(1, nrow=T.pred, ncol = 10)
X.tilde[,1] <- rep(1, T.pred)
X.tilde[,2] <- 1:T.pred
for(i in 1:length(w)){
  col.init <- 2*(1:T.pred)*w[i]
  col.cos <- sapply(col.init, FUN = cospi)
  col.sin <- sapply(col.init, FUN = sinpi)
  X.tilde[,i+2] <- col.cos
  X.tilde[,i+6] <- col.sin
}

y.hat <- matrix(0, T.pred, n.mcmc)

for(i in 1:n.mcmc){
```

```
  y.pred <- rep(0,T.pred)
  y.pred[1] <- 9.075823
  for(t in 2:T.pred){
    y.pred[t] =rnorm(1,(X.tilde[t,] - rho[i]*X.tilde[t-1,])%*%beta[i,] + rho[i]*y.pred[t-1],s2[i])
  }
  y.hat[,i] <- exp(y.pred)
  if(i %% 10000 == 0) cat(i, ’ ’)
}

save.file <- paste0(’/work2/09315/bilir/AR_predictive.rds’)
saveRDS(y.hat, file=save.file)
```

### ARMA_predictive.R

```
# read data
model <- ’model2’
file.name <- paste0(’/work2/09315/bilir/ARMA.rds’)
mcmc.out <- readRDS(file.name)

beta <- t(as.matrix(mcmc.out$beta[,,1]))
rho <- as.vector(mcmc.out$rho[,,1])
s2 <- 1/as.vector(mcmc.out$tau[,,1])
theta <- as.vector(mcmc.out$theta[,,1])

remove(mcmc.out)


T.data <- 731
T.pred <- T.data + 365*1
n.mcmc <- length(rho)

w <- c(0.00273794, 0.00547588, 0.00821382, 0.1428944)
X.tilde <- matrix(1, nrow=T.pred, ncol = 10)
X.tilde[,1] <- rep(1, T.pred)
X.tilde[,2] <- 1:T.pred
for(i in 1:length(w)){
  col.init <- 2*(1:T.pred)*w[i]
  col.cos <- sapply(col.init, FUN = cospi)
  col.sin <- sapply(col.init, FUN = sinpi)
  X.tilde[,i+2] <- col.cos
  X.tilde[,i+6] <- col.sin
}

y.hat <- matrix(0, T.pred, n.mcmc)

for(i in 1:n.mcmc){
  y.pred <- rep(0,T.pred)
  y.pred[1] <- 9.075823
  for(t in 2:T.pred){
    if(t%%10000==0){cat(t," ")}
```

```
    if(t == 2){
      eps <-   y.pred[1] - X.tilde[1,]%*%beta[i,]
    }else{
      eps <-  y.pred[t-1] - (X.tilde[t-1,] - rho[i]*X.tilde[t-2,])%*%beta[i,]
           - rho[i]*y.pred[t-2] - theta[i]*eps
    }
    y.pred[t] =rnorm(1,(X.tilde[t,] - rho[i]*X.tilde[t-1,])%*%beta[i,] +
                          rho[i]*y.pred[t-1] + theta[i]*eps,s2[i])
    }
  y.hat[,i] <- exp(y.pred)
  if(i %% 10000 == 0) cat(i, ' ')
}

save.file <- paste0('/work2/09315/bilir/ARMA_predictive.rds')
saveRDS(y.hat, file=save.file)
```

# Author Contributions

- BB contributed to the general idea of the project, creating the model, and writing up the document and code.
- HL contributed to preparing the dataset, writing up the document, and reviewing the document and code.
- AU contributed to preparing the dataset, creating plots, writing up the document, and reviewing the document and code.