LASSO Estimated AR model for Electricity Price Forecasting Using MCMC and Shrinkage Priors

Purushottam Saha (MB2416)

as part of *Statistical Inference II Course*
M.Stat. 2^{nd} Year

under the guidance of
Prof. Kiranmoy Das

Indian Statistical Institute
Kolkata, India

Date of Submission: November 10, 2024

1 Introduction

Electricity prices in liberalized power markets such as Nord Pool exhibit strong temporal dependence and short-term fluctuations driven by demand, supply, and renewable generation. Modeling and forecasting these dynamics are essential for effective bidding and risk management.[5]

Autoregressive (AR) models are a standard approach for capturing such temporal structures. However, when higher-order lags are included to account for seasonal or short-term persistence, parameter estimation can become unstable. To address this, LASSO regularization provides a way to shrink less relevant lag coefficients toward zero, yielding a more parsimonious and interpretable model.[3][2]

In this study, we fit a Bayesian LASSO autoregressive model of order 28 (AR(28))[3] to the daily average day-ahead electricity prices from the Nord Pool market. The Bayesian hierarchical formulation allows estimation and uncertainty quantification through Markov Chain Monte Carlo (MCMC) sampling. We present posterior summaries, credible intervals, trace and density plots, and analyze the shrinkage effect through the posterior of the penalty parameter.

The aim is to identify the most influential lag effects while maintaining predictive stability and interpretability in short-term electricity price modeling.

2 Data

The dataset represents the Nord Pool (NP), i.e. the European power market of the Nordic countries, and spans from 01.01.2013 to 24.12.2018. This dataset was open-accessed by Lago et al.[2] (see for more), to standardize the Electricity Price Forecasting methodology benchmarks. The dataset contains hourly observations of day-ahead prices, the day-ahead load forecast, and the day-ahead wind generation forecast, however, only day-ahead prices are modelled as an LASSO estimated AR(28) model as part of the current project. The dataset was constructed using data freely available on the webpage of the Nordic power exchange Nord Pool [1]. The figure below displays the electricity price time series of the dataset; as can be seen, the prices are always positives, zero prices are rare, and prices spikes seldom occur.

For the sake of this analysis, the hourly data is aggregated into daily average prices, and as a standard practice[2][4], the data is centered around median and scaled appropriately by the Median Absolute Deviation (MAD). For variance stabilizing, a standard (generalized) arcsin transformation $x \mapsto log(x + \sqrt{1 + x^2})$ is applied. The transformed data is showed along with the original data as per below. Also, for the task of forecasting, the data is split into a 4:1 train test split.

3 Model Specification and Methodology

We consider the daily average day-ahead electricity price series from the Nord Pool market, denoted by $\{y_t\}_{t=1}^{T}$. To capture short-term temporal dependence, we model the series using an autoregressive model of order $P = 28$:

$$y_t = \beta_0 + \sum_{j=1}^{P} \beta_j y_{t-j} + \varepsilon_t, \quad \varepsilon_t \sim \mathcal{N}(0, \sigma^2). \tag{1}$$

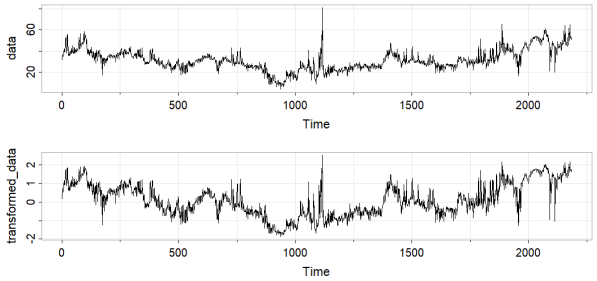


Figure 1: Average Day Ahead Prices and transformed version

To prevent overfitting and allow automatic selection of influential lags, we impose a **Bayesian LASSO prior** [3] on the autoregressive coefficients β_j. The LASSO prior corresponds to a Laplace (double-exponential) distribution, which can be expressed as a scale mixture of normals with an exponential mixing distribution, enabling Gibbs sampling in JAGS.

3.1 Hierarchical Model

The full hierarchical model can be represented as:

$$y_t \sim \mathcal{N}\left(\mu_t, \sigma^2\right), \quad \mu_t = \beta_0 + \sum_{j=1}^{P} \beta_j y_{t-j}, \tag{2}$$

$$\beta_j \mid \lambda^2, \tau_j \sim \mathcal{N}(0, \tau_j^{-1}), \quad \tau_j \sim \text{Exponential}\left(\frac{\lambda^2}{2}\right), \quad j = 1, \ldots, P, \tag{3}$$

$$\beta_0 \sim \mathcal{N}(0, 10^6), \tag{4}$$

$$\sigma^{-2} \sim \text{Gamma}(0.001, 0.001), \tag{5}$$

$$\lambda \sim \text{Gamma}(a_\lambda, b_\lambda). \tag{6}$$

Here, λ controls the overall shrinkage of the regression coefficients, while τ_j governs the local variance of each coefficient, allowing adaptive shrinkage across lags.

3.2 Posterior Inference via MCMC

The joint posterior distribution is given by:

$$p(\beta_0, \boldsymbol{\beta}, \boldsymbol{\tau}, \lambda, \sigma^2 \mid \mathbf{y}) \propto \prod_{t=1}^{T} p(y_t \mid \mu_t, \sigma^2) \prod_{j=1}^{P} p(\beta_j \mid \tau_j) p(\tau_j \mid \lambda^2) p(\lambda) p(\sigma^2) p(\beta_0). \tag{7}$$

Sampling is performed using **Markov Chain Monte Carlo (MCMC)** implemented in **JAGS**, which alternates between:

1. Sampling the regression coefficients β_j conditional on current values of other parameters,
2. Sampling the local precisions τ_j from their full conditional exponential distributions,
3. Updating the global shrinkage parameter λ from its gamma posterior,

4. Sampling σ^2 and the intercept β_0 from their respective full conditionals.

This hierarchical structure allows simultaneous estimation and variable shrinkage, yielding both posterior uncertainty quantification and automatic lag selection.

4 Results and Discussion

The Bayesian LASSO autoregressive model of order 28 was fitted to the daily average day-ahead electricity prices from the Nord Pool market using the MCMC algorithm implemented in `JAGS`. The model was run for 20,000 iterations with a burn-in of 5,000 and thinning interval of 5 to reduce autocorrelation. Convergence was assessed through trace plots and cumulative mean plots of the parameters.

4.1 Posterior Estimates and Credible Intervals

Table 1 reports the posterior means and 95% credible intervals for the intercept and autoregressive coefficients. Most of the lag coefficients are heavily shrunk towards zero, confirming the effectiveness of the Bayesian LASSO in identifying the dominant lags contributing to short-term price dynamics.

The posterior mean of the shrinkage parameter λ indicates moderate regularization, producing a sparse coefficient structure where only a few lag terms remain significant. The posterior of λ (Figure 3) shows concentration around 3, reflecting a balance between bias and variance in the fitted AR process.

4.2 MCMC Diagnostics

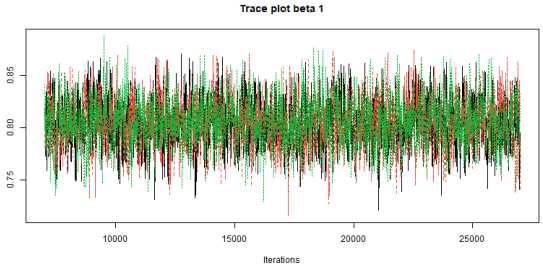
Figures 2 present MCMC trace plots and cumulative mean plots for selected parameters (see for all plots). The trace plots exhibit stable mixing and no apparent trends, suggesting convergence of the Markov chains. The cumulative mean plots approach stable values after a few thousand iterations, confirming adequate chain length.

4.3 Posterior Density and Shrinkage Behavior

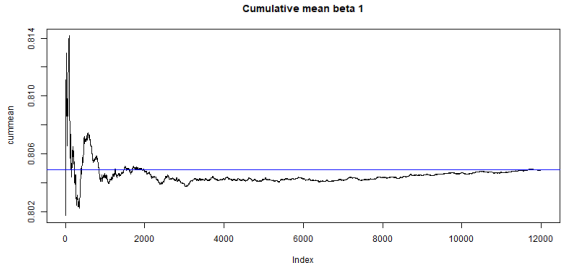
Figure 3 displays the posterior densities of selected coefficients. Most of the densities are sharply peaked around zero, with only a few showing notable spread, indicating significant predictive contribution. This sparsity pattern aligns with the Bayesian LASSO's tendency to shrink uninformative coefficients while retaining dominant lag effects.

4.4 Lag Importance and Shrinkage Frequency

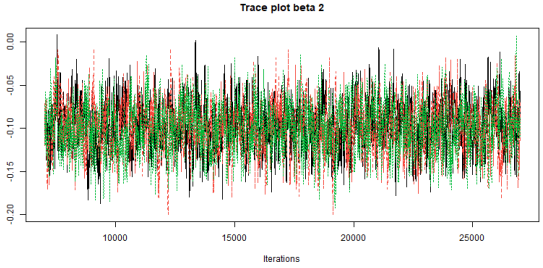
The percentage of times each lag coefficient is sampled as numerically close to zero (within a small threshold, set to 10^{-4}) across MCMC iterations is reported in Table 1. Higher percentages indicate stronger shrinkage effects, implying negligible influence of that lag on current prices. Only a small subset of recent lags (e.g., 1-4) and weekly influential lags (e.g. 7,14,15,28) retain strong predictive influence.



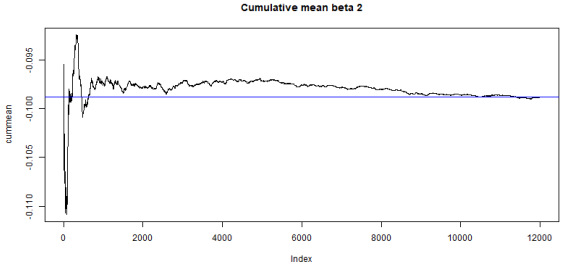
(a) Trace plot for β_1



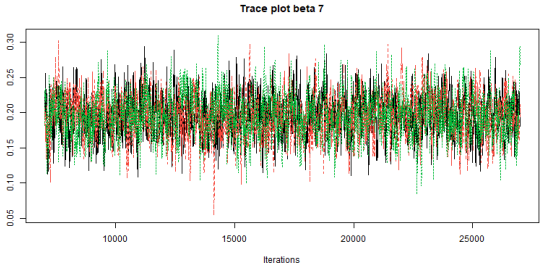
(b) Cumulative mean plot for β_1



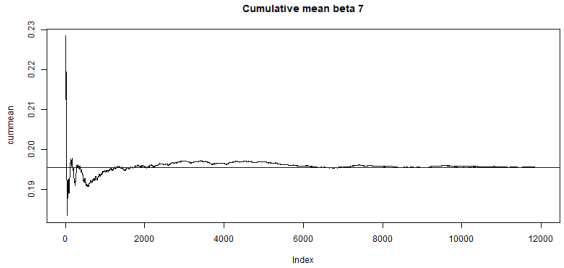
(c) Trace plot for β_2



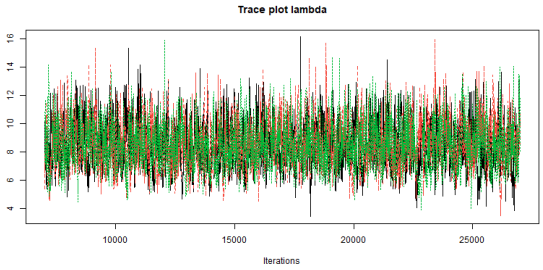
(d) Cumulative mean plot for β_2



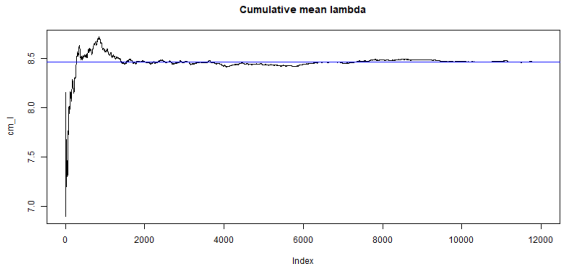
(e) Trace plot for β_7



(f) Cumulative mean plot for β_7



(g) Trace plot for λ



(h) Cumulative mean plot for λ

Figure 2: MCMC diagnostics: trace and cumulative means for (a,b) β_1, (c,d)β_2, (e,f) β_7,(g,h) λ.

Parameter	Posterior Mean	95% Credible Interval	P($\beta_j = 0$) (%)
β_1	0.80	[0.75, 0.84]	0.00
β_2	-0.10	[-0.16, -0.03]	0.00
β_3	0.11	[0.04, 0.17]	0.00
β_4	0.05	[-0.01, 0.11]	0.12
β_5	-0.07	[-0.13, -0.01]	0.03
β_6	0.07	[0.00, 0.13]	0.01
β_7	0.19	[0.13,0.25]	0.00
β_8	-0.08	[-0.14,-0.02]	0.02
β_9	-0.05	[-0.11, 0.01]	0.09
β_{10}	-0.02	[-0.08, 0.03]	0.23
β_{11}	0.03	[-0.03, 0.09]	0.17
β_{12}	0.01	[-0.04, 0.07]	0.27
β_{13}	0.02	[-0.04, 0.07]	0.30
β_{14}	0.14	[0.08, 0.20]	0.00
β_{15}	-0.15	[-0.21, -0.09]	0.00
β_{16}	0.01	[-0.05, 0.07]	0.21
β_{17}	-0.01	[-0.06, 0.05]	0.33
β_{18}	-0.01	[-0.07, 0.04]	0.22
β_{19}	-0.04	[-0.09, 0.02]	0.18
β_{20}	0.04	[-0.01, 0.10]	0.13
β_{21}	0.11	[0.05, 0.17]	0.00
β_{22}	-0.04	[-0.10, 0.01]	0.12
β_{23}	-0.03	[-0.09, 0.02]	0.12
β_{24}	0.01	[-0.05, 0.07]	0.30
β_{25}	-0.03	[-0.09, 0.02]	0.13
β_{26}	0.02	[-0.03, 0.07]	0.22
β_{27}	0.00	[-0.05, 0.06]	0.33
β_{28}	0.03	[-0.01, 0.08]	0.12
λ (shrinkage)	8.59	[5.69, 12.14]	-
σ^2	0.067	[0.063, 0.072]	-

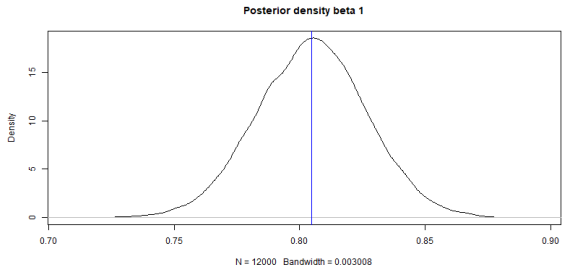
Table 1: Posterior Estimates and Credible Intervals

4.5 Prediction

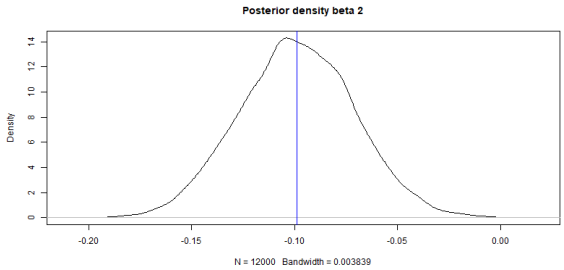
The test dataset kept aside before, was used to show the predictive performance of the model. An test MAPE of 0.55 was achieved, which is on par with the statistical state of the art models [2]. The posterior mean and credible regions along with the true values is presented below.

5 Discussion and Conclusion

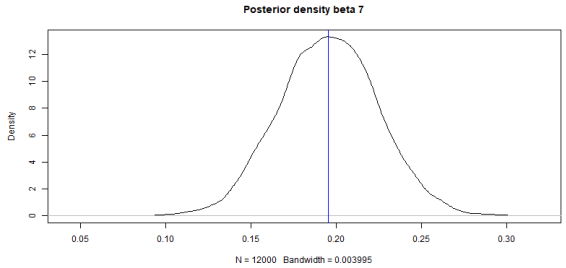
The results confirm that the Bayesian LASSO AR(28) model effectively identifies meaningful lag dependencies while maintaining parsimony. The adaptive shrinkage structure provides robust parameter estimation and reduces the risk of overfitting compared to traditional maximum-likelihood AR models. The credible intervals and MCMC diagnostics demonstrate stable convergence and interpretable posterior uncertainty. The inferred



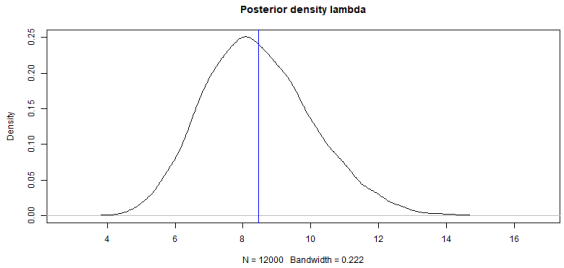
(a) Posterior density plot for β_1



(b) Posterior density plot for β_2



(c) Posterior density plot for β_7



(d) Posterior density plot for λ

Figure 3: MCMC diagnostics: Posterior density plots for (a) β_1, (b) β_2, (c) β_7, and (d) λ.

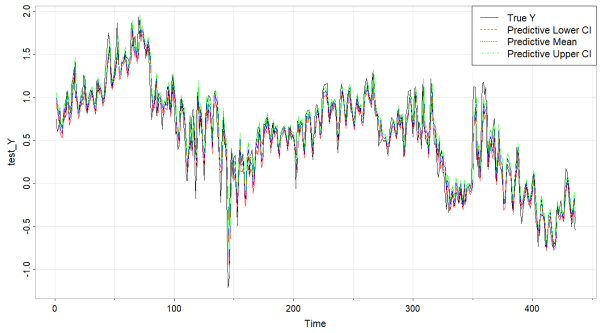


Figure 4: Posterior Mean, Credible Regions and True values for test data

lag structure suggests that short-term (1–7 day) effects and weekly effects dominate the dynamics of Nord Pool day-ahead prices, consistent with operational and weekly demand patterns in electricity markets. The results also achieve good predictive performance, satisfying the goal of the project.

6 R Code Implementation

```
# install.packages(c("rjags","coda","ggplot2","matrixStats","readr"))
library(rjags)
library(coda)
library(ggplot2)
library(matrixStats)
library(readr)
library(zoo)

# Loading data
df <- read.zoo("data//NP_reduced.csv", header = TRUE, sep = ",",
    format = "%Y-%m-%d", aggregate = mean)
data <- as.numeric(df)
dates <- index(df)
n <- length(data)
```

```
# Transforming data
median <- median(data)
mad <- mad(data)
cat("median: ",median, "\nMAD: ",mad)
scaled_data = (data-median)/mad
transformed_data = log(scaled_data+sqrt(1+scaled_data*scaled_data))
# Plotting data
par(mfrow=c(2,1))
astsa::tsplot(data)
astsa::tsplot(transformed_data)
# Buildomg lag matrix for AR(28)
p <- 28
T0 <- n - p
Y <- transformed_data[(p+1):n]
X <- matrix(NA, nrow=T0, ncol=p)
for(j in 1:p) X[,j] <- transformed_data[(p+1-j):(n-j)]
X <- as.matrix(X)
n_train = floor(n*(1/5))
test_indices = c(1:n_train)
test_X = X[test_indices,]
X = X[-test_indices,]
test_Y = Y[test_indices]
Y = Y[-test_indices]
n = n-n_train

# JAGS model (Bayesian LASSO hierarchical)
jags_model_string <- "
model {
  # Likelihood
  for(t in 1:T) {
    mu[t] <- beta0 + inprod(beta[1:P], X[t,])
    y[t] ~ dnorm(mu[t], tau)   # tau = 1/sigma2
  }

  # Priors for coefficients (scale mixture -> double exponential)
  for(j in 1:P) {
    beta[j] ~ dnorm(0, 1.0 / lambda2_tau[j])   # precision = 1/tau_j
    lambda2_tau[j] ~ dexp(lambda^2 / 2.0)      # mixing for Laplace
  }

  # Intercept
  beta0 ~ dnorm(0.0, 1.0E-6)

  # Noise
  tau ~ dgamma(0.001, 0.001)    # tau = 1/sigma2
  sigma2 <- 1.0 / tau

  # Hyperprior for lambda (penalty). gamma prior
  lambda ~ dgamma(a_lambda, b_lambda)
  # store lambda^2 for convenience
  lambda_sq <- lambda * lambda
}
"
# Data list for JAGS
data_jags <- list(
  T = nrow(X),
  P = p,
  y = as.numeric(Y),
  X = X,
  a_lambda = 1.0,
  b_lambda = 1.0
)
# initial values (function)
inits_fn <- function() {
  list(
    beta = rnorm(p,0,0.1),
    beta0 = mean(Y),
    tau = 1/var(Y),
    lambda = rgamma(1,1,1),
    lambda2_tau = rexp(p, rate = 1.0)
  )
}
# JAGS run settings
n.chains <- 3
n.adapt <- 2000
n.burn <- 5000
n.iter <- 20000
n.thin <- 5

# initialize and run
cat("Compiling JAGS model...\n")
jags_mod <- jags.model(textConnection(jags_model_string),
                       data = data_jags,
                       inits = inits_fn,
                       n.chains = n.chains,
                       n.adapt = n.adapt)

cat("Updating (burn-in) ...\n")
update(jags_mod, n.burn)

params <- c("beta0","beta","sigma2","lambda")
cat("Sampling...\n")
draws <- coda.samples(jags_mod, variable.names = params, n.iter = n.iter, thin = n.thin)

# Combine chains
mcmc_comb <- as.mcmc(do.call(rbind, draws))
# Posterior summaries
summary_stats <- summary(draws)
print(summary_stats)
```

```
# Extract beta draws matrix (samples x p)
beta_names <- paste0("beta[",1:p,"]")
beta_draws <- do.call(rbind, lapply(draws, function(ch) as.matrix(ch[,beta_names])))
beta0_draws <- do.call(rbind, lapply(draws, function(ch) as.matrix(ch[,"beta0"])))
lambda_draws <- do.call(rbind, lapply(draws, function(ch) as.matrix(ch[,"lambda"])))
sigma2_draws <- do.call(rbind, lapply(draws, function(ch) as.matrix(ch[,"sigma2"])))

# Posterior means and 95% CI for betas
post_mean_beta <- colMeans(beta_draws)
post_ci_beta <- t(apply(beta_draws, 2, quantile, probs=c(0.025,0.5,0.975)))

# Penalty lambda posterior
lambda_mean <- mean(lambda_draws)
lambda_ci <- quantile(lambda_draws, c(0.025,0.5,0.975))

# Table of % times each predictor is (near) zero
eps <- 1e-4   # threshold: |beta| < eps considered 'zero' - changeable
pct_zero <- 100 * colMeans(abs(beta_draws) < eps)

# Save summary table
beta_table <- data.frame(
  lag = 1:p,
  mean = post_mean_beta,
  ci025 = post_ci_beta[,1],
  median = post_ci_beta[,2],
  ci975 = post_ci_beta[,3],
  pct_near_zero = pct_zero
)
print(beta_table)
# Plots: trace, cumulative mean, density for each coefficient + lambda
# Trace + density for a subset (first 6 betas) and lambda
plot_dir <- "plots"
if(!dir.exists(plot_dir)) dir.create(plot_dir)

# Use coda traceplots
for(j in 1:p) {
  png(sprintf("%s/trace_beta%02d.png", plot_dir, j), width=800, height=400)
  traceplot(draws[,paste0("beta[",j,"]")], main=paste("Trace plot beta",j))
  dev.off()
  png(sprintf("%s/density_beta%02d.png", plot_dir, j), width=800, height=400)
  dens <- density(beta_draws[,j])
  plot(dens, main=paste("Posterior density beta",j))
  abline(v=mean(beta_draws[,j]), col="blue")
  dev.off()
  png(sprintf("%s/cummean_beta%02d.png", plot_dir, j), width=800, height=400)
  cm <- cumsum(beta_draws[,j]) / seq_along(beta_draws[,j])
  plot(cm, type='l', main=paste("Cumulative mean beta",j), ylab="cummean")
  abline(h=mean(beta_draws[,j]), col="blue")
  dev.off()
}
# lambda plots
png(sprintf("%s/trace_lambda.png", plot_dir), width=800, height=400)
traceplot(draws[,"lambda"], main="Trace plot lambda")
dev.off()
png(sprintf("%s/density_lambda.png", plot_dir), width=800, height=400)
plot(density(lambda_draws), main="Posterior density lambda")
abline(v=lambda_mean, col="blue")
dev.off()
png(sprintf("%s/cummean_lambda.png", plot_dir), width=800, height=400)
cm_l <- cumsum(lambda_draws)/seq_along(lambda_draws)
plot(cm_l, type='l', main="Cumulative mean lambda")
abline(h=lambda_mean, col="blue")
dev.off()

# Posterior predictive check & one-step ahead prediction (example)
# Posterior predictive for training set: compute fitted mean and predictive intervals
S <- nrow(beta_draws)
fitted_mat <- test_X %*% t(beta_draws) + matrix(rep(beta0_draws, each=nrow(test_X)), nrow=nrow(test_X), byrow=FALSE)
# but simpler: get predictive mean for y
pred_mean_y <- rowMeans(fitted_mat)
pred_ci_lower <- apply(fitted_mat, 1, quantile, 0.025)
pred_ci_upper <- apply(fitted_mat, 1, quantile, 0.975)

par(mfrow=c(1,1))
astsa::tsplot(test_Y)
lines(pred_ci_lower,col='red',lty=2)
lines(pred_mean_y,col='blue',lty=3)
lines(pred_ci_upper,col='green',lty=4)
legend("topright",
       legend = c("True Y", "Predictive Lower CI", "Predictive Mean", "Predictive Upper CI"), # Labels for the legend
       col = c("black","red", "blue","green"),
       lty = c(1, 2,3,4))

test_df <- data.frame(
  date = dates[test_indices],
  true_y = test_Y,
  fit_mean = pred_mean_y,
  fit_low = pred_ci_lower,
  fit_high = pred_ci_upper
)

# Save outputs
write.csv(beta_table, "beta_posterior_table.csv", row.names=FALSE)
save(draws, file="jags_draws.RData")
cat("Finished. Plots are in folder:", plot_dir, "\n")
```

References

[1] NordPool Spot Price Website.

[2] Lago, J., Marcjasz, G., De Schutter, B., and Weron, R. Forecasting day-ahead electricity prices: A review of state-of-the-art algorithms, best practices and an open-access benchmark. *Applied Energy 293* (2021), 116983.

[3] Park, T., and Casella, G. The bayesian lasso. *Journal of the American Statistical Association* (2008).

[4] Uniejewski, B., Weron, R., and Ziel, F. Variance stabilizing transformations for electricity spot price forecasting. *IEEE Transactions on Power Systems 33*, 2 (2018), 2219–2229.

[5] Weron, R. Electricity price forecasting: A review of the state-of-the-art with a look into the future. *International Journal of Forecasting 30* (10 2014).