Bayesian Inference for PM2.5 (Air Quality Index) in India Forecasting

by
Megha Gulati
(Dated: December 03, 2023)

Table of Contents

1. Preface

Outdoor air pollution is a major environmental health problem affecting everyone. The adverse impacts of air pollution, particularly microscopic yet potent PM2.5 particles has created urgency to develop advanced predictive models. We will try to explore Bayesian Inference and Gibbs Sampling as powerful tools in forecasting PM2.5 levels.

1.1 The Context:

WHO estimates that in 2019, some 37% of outdoor air pollution-related premature deaths were due to ischaemic heart disease and stroke, 18% and 23% of deaths were due to chronic obstructive pulmonary disease and acute lower respiratory infections respectively, and 11% of deaths were due to cancer within the respiratory tract. In this context, we will dive into the intricacies of PM2.5, recognizing its significance as a key indicator of air quality.

1.2 The Vision:

The goal is to bridge the gap between the data science techniques and environmental management. By leveraging Bayesian Inference with Gibbs Sampling, we aim to unlock the potential to model temporal dependencies, inherent uncertainities and the evolving nature of air pollution.

2. Introduction

Air pollution, a ubiquitous byproduct of industrialization and urbanization, poses a significant threat to public health and environment. The fine particular matter with a diameter of 2.5 micrometers or smaller(PM2.5) is of particular concern due to its ability to penetrate deep into the respiratory system, causing a range of health issues.

Accurate forecasting of PM2.5 levels is imperative for public health planning, environmental management and policy formulation. Since, traditional statistical methods often fall short in capturing the complex temporal dynamics and uncertainities associated with air quality data. This is where Bayesian Inference comes to rescue with Gibbs Sampling, this implementation offers a flexible and probabilistic approach to model the intricate interplay of factors influencing PM2.5 concentrations.

2.1 Motivation

This year my home town, New Delhi, India had some record breaking levels of PM2.5 levels = 600. People were afraid of going out, they were actually advised by doctors not to go out. Cricket World Cup matches were cancelled, players had to use inhalers, yes those healthy fit players had trouble breathing. Since I have dealt with being a TB patient with lung problems, the poor air quality index hindered the recovery for months. Maybe if we can employ Bayesian Inference to analyze or try to forecast the air quality index scores well in advance so that we could deploy some better prevention methods instead of trying to cure it.

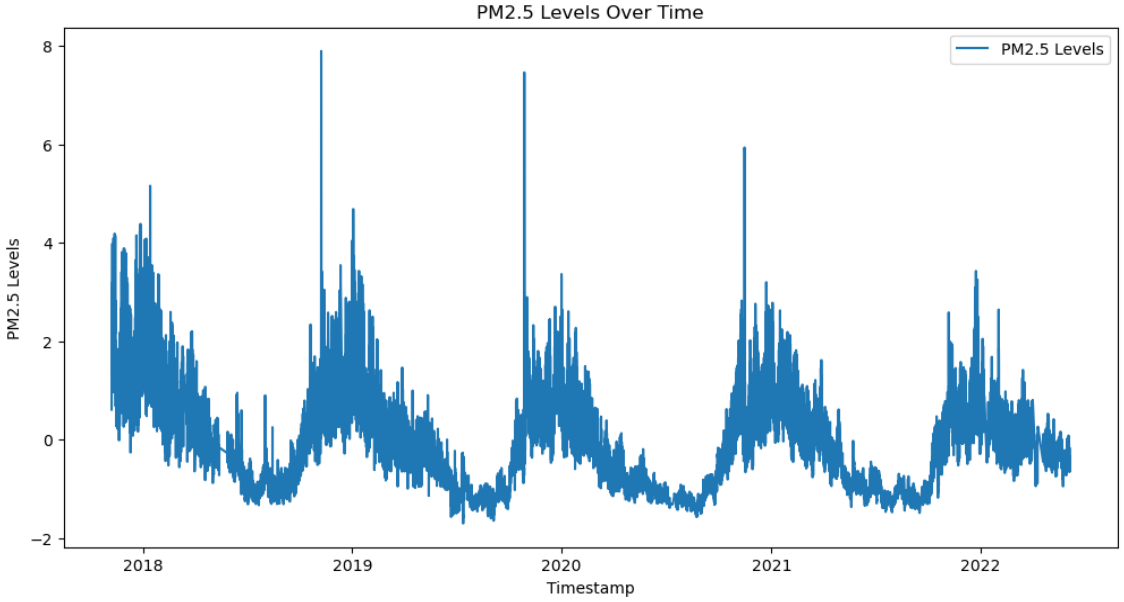
3. Data Preprocessing

The first step in our analytical approach involves picking a dataset and preprocess it before doing any analysis on it. Data preprocessing includes - handling missing values, converting timestamps, sorting the data and selecting relevant columns. A crucial aspect is scaling the PM2.5 values to ensure convergence during the Bayesian modelling phase.

Here, we have taken a data set from Kaggle - air-quality-india.csv which contains 37000 records, we performed some basic functions like -

```
def describe_data(self):
    print(self.data.describe())

def handle_missing_values(self):
    #check missing values
    sum_of_missing_values = self.data.isnull().sum()
    if sum_of_missing_values.sum() == 0:
        print("No missing values found")
    else:
        air_quality_data = self.data.dropna()
        print("Missing values found, replaced them with NA")
    print(self.data.head())

def preprocess_timestamp(self):
    self.data["Timestamp"] = pd.to_datetime(self.data["Timestamp"])

def sort_by_timestamp(self):
    self.data = self.data.sort_values(by="Timestamp")

def select_columns(self, columns):
    self.data = self.data[columns]

def scale_pm25(self):
    scaler = StandardScaler()
```

```
        self.data["PM2.5"] =
scaler.fit_transform(self.data["PM2.5"].values.reshape(-1,1))

    def plot_pm25_levels(self):
        plt.figure(figsize=(12,6))
        plt.plot(self.data["Timestamp"], self.data["PM2.5"], label="PM2.5
Levels")
        plt.title("PM2.5 Levels Over Time")
        plt.xlabel("Timestamp")
        plt.ylabel("PM2.5 Levels")
        plt.legend()
        plt.show()
```



4. Bayesian Model Specification

4.1 Model Definition:

We choose a model that reflects the temporal dependencies in PM2.5 data. The model structure includes priors and likelihood functions. The Bayesian ARIMA model is represented as -

$$Y_t = \phi Y_{t-1} + \theta \epsilon_{t-1} + \epsilon_t$$

where -

- Y_t is the PM2.5 level at time t.
- ϕ is the auto regressive coefficient.
- θ is the moving average coefficient.
- ϵ_t is the error term.

4.2 Likelihood Function:

To model the temporal nature of PM2.5 levels, we define a likelihood function. The likelihood captures the relationship between observed and latent variables, incorporating autoregressive and moving average components. In our study, the likelihood function is formulated as:

$$Y_t \sim \mathcal{N}(\phi Y_{t-1} + \theta \epsilon_{t-1}, \sigma^2)$$

where -

- Y_t is the observed PM2.5 level at time t.
- N represents Normal Distribution.
- ϕ and θ are model parameters.
- ϵ_{t-1} is the error term.
- σ^2 is the variance.

5. Prior Specification

5.1 Prior Distributions:

Prior distributions represent our beliefs about model parameters before observing the data. We carefully choose informative or uninformative priors based on our understanding of PM2.5 dynamics. These priors play a crucial role in influencing the posterior distributions and, consequently, the model's predictions.

$$\phi \sim \mathcal{N}(\mu_\phi, \sigma_\phi^2)$$

$$\theta \sim \mathcal{N}(\mu_\theta, \sigma_\theta^2)$$

$$\sigma \sim \text{HalfNormal}(\sigma_\sigma)$$

6. Gibbs Sampling Implementation

Gibbs Sampling is a Markov Chain Monte Carlo (MCMC) method used for sampling from complex probability distributions. In the context of Bayesian inference, Gibbs Sampling is employed to draw samples from the posterior distribution of model parameters. Let's break down the process with formulae:

6.1 Setup:

We have a joint posterior distribution
$P(\theta|y)$, where θ represents the parameters of interest, and y is the observed data.

6.2 Initialization:

Start with an initial guess for the parameter values $\theta_{(0)}$

6.3 Iteration:

Sample from the conditional distribution of θ_i given all other parameters and the data

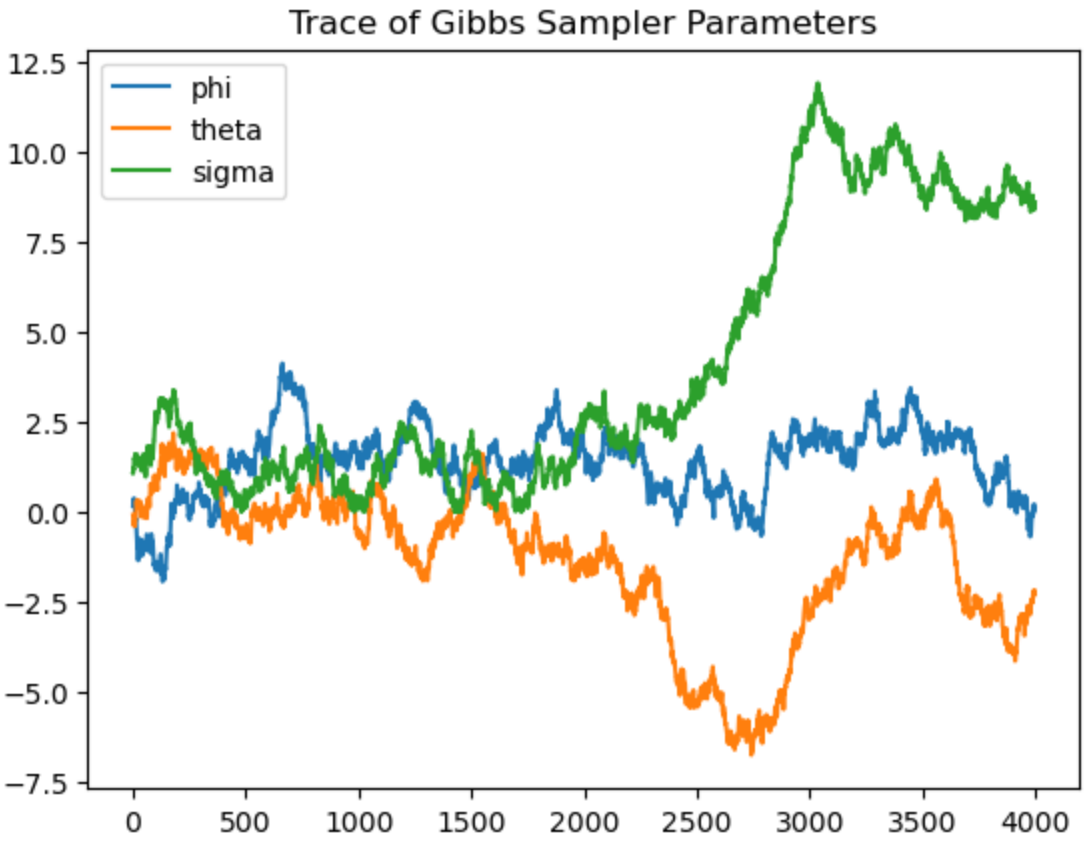
$$\theta_i^{(t+1)} \sim P(\theta_i|\theta_1^{(t)}, \theta_2^{(t)}, ..., \theta_{i-1}^{(t)}, \theta_{i+1}^{(t)}, ..., \theta_k^{(t)}, y)$$

This involves sampling from the posterior distribution of θ_i while holding all other parameters fixed.

.

6.4 Repeat:

Iterate over all parameters until convergence, discarding a burn-in period.



6.5 Gibbs Sampling for Bayesian Time Series Forecasting:

In the context of time series forecasting, let's consider a simple Bayesian model with parametersϕ, θ, and σ, representing autoregressive coefficients, moving average coefficients, and the noise standard deviation, respectively.

6.5.1 Paramterization:

Assume a Bayesian ARIMA(1,1,1) model.

$Y_t = \phi Y_{t-1} + \epsilon_t + \theta \epsilon_{t-1}$ where Y_t is the PM2.5 level at time t, ϵ_t is the noise term.

6.5.2 Posterior Distribution:

The joint posterior distribution is given by -

$$P(\phi, \theta, \sigma|Y) \propto P(Y|\phi, \theta, \sigma) \times P(\phi) \times P(\theta) \times P(\sigma)$$

6.5.3 Conditional Distributions:

For Gibbs Sampling, we need to sample from the conditional distributions of each parameter given the others.

$$P(\phi|Y, \theta, \sigma)$$
$$P(\theta|Y, \phi, \sigma)$$
$$P(\sigma|Y, \phi, \theta)$$

6.5.4 Sampling Steps:

The sampling steps for each parameter involve using their respective conditional distributions to update the parameter values.

7. Evaluation

After completing the Gibbs Sampling Implementation, the next step involves evaluating the performance of the Bayesian time series forecasting model. This evaluation phase is optional but highly recommended for understanding how well the model aligns with real-world data.

7.1 Performance Metrics:

Mean Absolute Error (MAE):

- The MAE measures the average absolute difference between the true and forecasted PM2.5 levels, providing insights into the magnitude of prediction errors.

Root Mean Squared Error (RMSE):

- The RMSE incorporates the squared differences between true and predicted values, offering a more comprehensive view of prediction accuracy

7.2 Visual Inspection:

True vs. Forecasted Plot: Plotting the true PM2.5 levels against the forecasted values over time provides a visual representation of the model's performance.

Here, we took another data set for 2023 PM2.5 levels, and in the last step we are comparing the values which are predicted vs the actual values fetched from the new data set.

7.3 Implementation:

Code Development:

- Implement the evaluation metrics using Python, leveraging libraries such as NumPy or scikit-learn for computation.

Visualization:

- Utilize Matplotlib or other visualization tools to create plots that facilitate a qualitative assessment of the model's accuracy.

8. Bibliography

Kaggle For DataSets:

Data Set

https://www.kaggle.com/code/amankumar234/air-quality-analysis-in-india