Stat 447 Final Project

Team: Manav Doshi

Github: https://github.com/Manav535/Bayesian-Time-Series-Analysis/tree/5cb26b8019a2d5384450bfbefb05305bfb9818bc

Introduction

Accurate forecasting of equity market indices is a central task in financial econometrics, with direct implications for portfolio management, risk assessment, and algorithmic trading. The NIFTY50 index, India's benchmark stock-market index (analogous to the S&P500 in the U.S.), aggregates the performance of fifty of the largest Indian securities, serving as a barometer of India's economic health. Traditional time-series approaches often analyze the index in isolation, overlooking valuable information contained in related sectoral indices. However, sector indices (e.g., Information Technology, Financial Services, Healthcare) typically exhibit correlated dynamics with the broader market, especially during periods of sector-specific shocks or macroeconomic regime changes. The key methodological challenge we address is developing forecasting models that can effectively capture these interrelationships while handling the non-stationary nature of financial time series. This project specifically focuses on time series and state space models as frameworks for this prediction task.

Literature Review

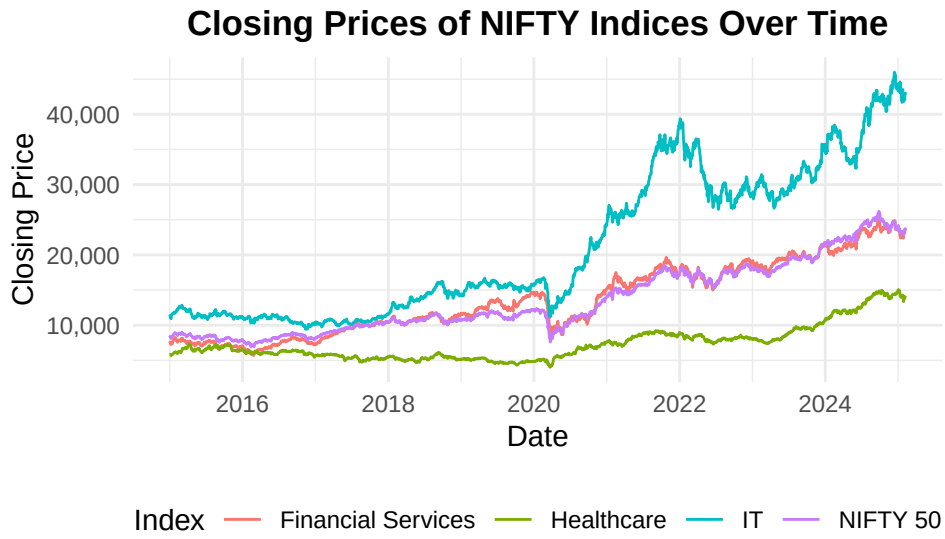
Financial time series modeling has evolved significantly to address the challenges of market data. Hendershott and Menkveld (2014) demonstrated how state-space models can decompose financial variables into unobserved components, highlighting their advantage in handling missing values through Kalman filtering - a technique we employ in our comparison of methods. For markets with distinct regimes, Azzouzi and Nabney's work on Switching State Space Models (SSSMs) provides insight into how markets transition between approximately stationary states, particularly relevant for analyzing the market during periods of volatility. Traditional ARIMA models - which serve as our benchmark - remain fundamental despite their limitations. Kane et al. (2014) showed that while ARIMA offers interpretability, it often underperforms newer approaches in capturing the non-linear relationships common in financial markets. These methodological approaches inform our comparative analysis between hierarchical Bayesian state-space modeling, Kalman filtering, and ARIMA forecasting for predicting NIFTY50 movements using both the index itself and related sectoral indices.

Our Approach

Based on this methodological foundation, we develop a hierarchical Bayesian state-space model that jointly leverages the NIFTY50 and multiple sector indices to improve forecast accuracy and quantify predictive uncertainty. We benchmark this approach against standard ARIMA forecasts and Kalman filtering, demonstrating how the Bayesian framework accommodates parameter sharing across sectors, produces full posterior predictive distributions, and enables rigorous uncertainty quantification. Our implementation uses Stan for MCMC sampling, with assessment through posterior diagnostics ($\hat{R}$, effective sample size, E-BFMI, divergences) and out-of-sample performance metrics (RMSE, MAE, MAPE). We aim to forecast the next 60 trading-day (long enough to matter for strategic decisions but short enough to avoid regime shifts) closing prices of the NIFTY50 index using daily closing prices of the NIFTY50 itself and three sectoral indices: Information Technology (IT), Financial Services (FIN), and Healthcare (HEA).

Exploratory Plot of the NIFTY50 and Sector Indices

First, we plot the closing prices of the NIFTY50 and the three sector indices over time to get a feel for their joint dynamics and any broad trends or structural breaks.



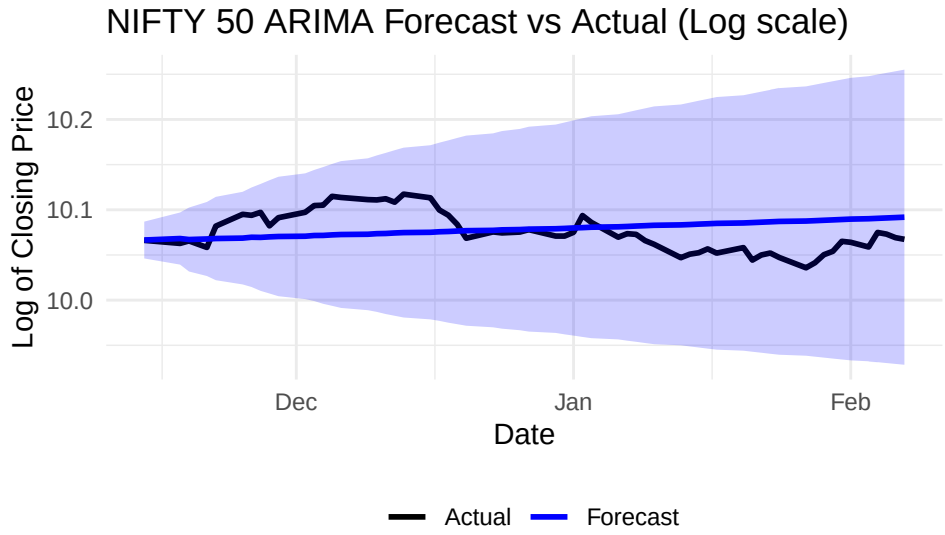
We see that all series trend upward, with occasional spikes and drawdowns—especially a big fall around 2020, and a spike in 2022. This non-stationarity motivates differencing before classical ARIMA modeling, and a latent-state approach in our Bayesian model.

Benchmark: ARIMA(5,1,5) with Drift

To establish a classical benchmark, we fit an ARIMA model to the log-series of the NIFTY50. We use the log-series as it is standard practice while using financial data to use log-returns. An augmented Dickey–Fuller test on the log series returned $p \approx 0.083$, so we difference once to achieve stationarity ($p<0.01$).

The training set required an order of (5,1,5) with a small positive drift. We then forecast 60 trading days ahead. On the original-scale, the metrics we obtained are: **RMSE**: 643.42, **MAE**: 553.40, **MAPE**: 2.33%

These metrics suggest reasonable in-sample fit, but the forecast plot shows essentially a straight-line continuation of the last observed value—one of the known pitfalls of ARIMA when no strong cyclical pattern is detected.



Because ARIMA struggles to capture structural trends here, we next turn to a Bayesian state-space model.

Bayesian State-Space Model

We posit a latent state s_t for the NIFTY50 log-price that evolves autoregressively, with contemporaneous "shock" inputs from the first differences of the sector indices. Formally:

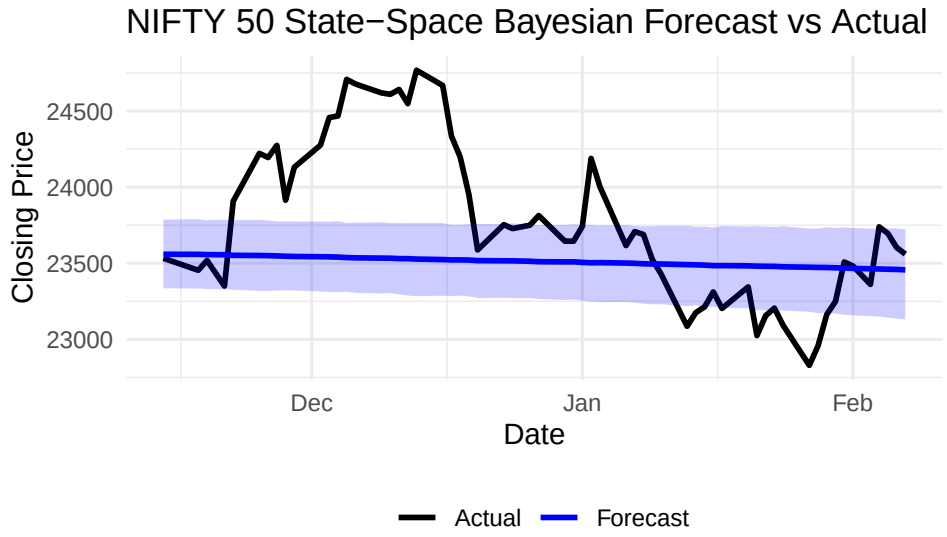
$$s_t \sim N\left(\phi s_{t-1} + \sum_{k=1}^{K} \beta_k (y_{\text{sector},t-1,k} - y_{\text{sector},t-2,k}), \sigma^2_{\text{state}}\right)$$
$$y_{\text{main},t} \sim N(s_t, \sigma^2_{\text{obs}})$$

We choose the priors as:

$$\begin{aligned} s_1 &\sim N(y_{\text{main},1}, 0.1) \\ \sigma_{\text{obs}}, \sigma_{\text{state}} &\sim \text{Exp}(10) \\ B_k &\sim N(0, 0.5) \\ \phi &\sim N(0.9, 0.1) \end{aligned}$$

To address potential non-stationarity, first differences were taken for the sector indices within the Stan model, allowing the latent state to capture long-term trends while sector differences represent short-term market shocks. The main index (y_main) was left undifferenced, as its trends are modeled through a latent state process with autoregressive dynamics. This modeling choice aligns with common practices in Bayesian state-space modeling, where the state component evolves independently and captures underlying structural trends, while avoiding unnecessary preprocessing.

The prior for the variances being selected as Exp(10) reflects the belief that the markets do not change drastically in one day, and hence have some form of stability. It is also noteworthy to remember that we are dealing in the log-scale, and hence the values will be smaller in general. The other prior choice regarding ϕ and β reflects our belief in high persistence and modest sector influence. Several alternatives were tested with these resulting in the best relative results. We run 4 chains, 10,000 iterations with half of them as warmup.



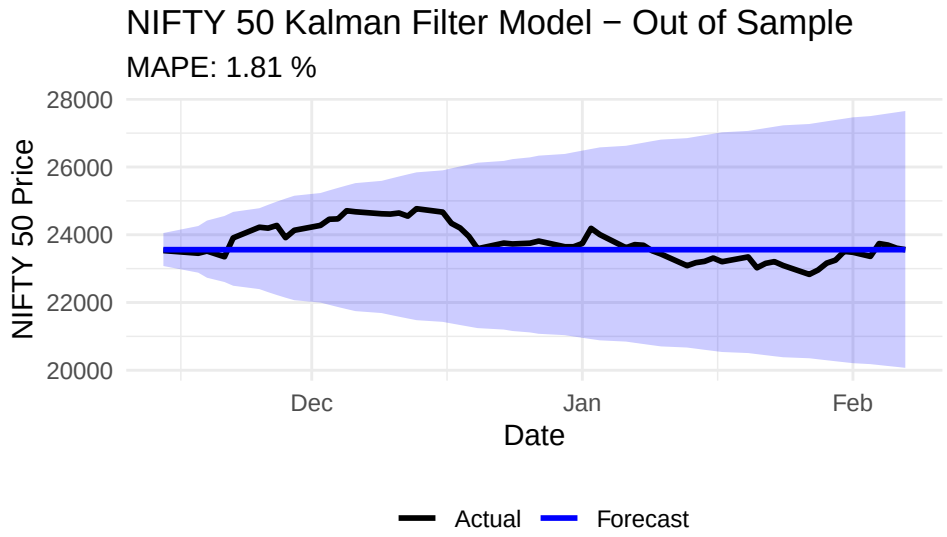
Bayesian Model Diagnostics and Forecast

Posterior diagnostics revealed mixing issues with low E-BFMI values (<0.2), low effective sample sizes for non-phi parameters, and elevated R-hat values for some parameters, all suggesting poor mixing and potential issues with the posterior geometry. Although, the trace plots (Appendix) show moderate to good mixing and stationarity across all four chains overall for all parameters. The phi parameter is consistently near 1, suggesting strong persistence in the latent state but warranting caution due to its proximity to the upper bound. Other MCMC diagnostics appear stable, with no signs of divergence or tree depth saturation. In addition, the out-of-sample forecasts achieved have the accuracy metrics of:

RMSE: 561.29, **MAE**: 436.87, **MAPE**: 1.81%. Although metrics slightly improve over ARIMA, the forecast again appears nearly flat due to the highly persistent latent component.

Kalman Filtering on NIFTY50

Finally, we apply a robust Kalman filter (via the `dlm` package) to the log-series. Kalman filtering has the advantage of exact recursive estimation and fast computation, with well-understood Gaussian updates.



The model selection routine chose a simple random-walk model as the best one:

$$s_t = s_{t-1} + \eta_t, \quad \eta_t \sim N(0, W)$$
$$y_t = s_t + \epsilon_t, \quad \epsilon_t \sim N(0, V)$$

This yielded: **RMSE**: 556.49, **MAE**: 434.01, **MAPE**: 1.81%

Residual diagnostics (Appendix) show patternless residuals centered at zero, roughly Gaussian in the QQ-plot, indicating good in-sample fit. The in-sample fit (Appendix) is also supported by the graph which shows the fitted values closely following the actual values. However, the out-of-sample forecast is again a flat line, reflecting the random-walk.

Conclusion

All 3 approaches—ARIMA, Bayesian MCMC, and Kalman filtering— are producing similar flat forecasts, even though they yield a low MAPE (<3%), indicating a good fit, albeit with MCMC and Kalman filtering having slightly lower MAPE than the ARIMA model.

The MCMC convergence issues (evidenced by low E-BFMI and ESS) might improve with more iterations, more chains, or refined tuning parameters. However, this would come at a significant computational cost, which is a practical constraint in the current setup.

Given these limitations and the similarity of forecasts across methods, our models may be missing regime-switching dynamics or longer-term dependencies in the data. The flat forecasts suggest market efficiency during this period, where past price information alone is insufficient for prediction.

Additionally, it's possible that the forecasting window selected is inherently stable or lacks significant variability — meaning that, due to coincidence or bad luck, the data in this period offers no strong patterns for the models to learn from.

Future work should explore incorporating exogenous economic variables, longer lag structures, or explicit regime-switching mechanisms to better capture market dynamics.

Citations

Azzouzi, Mehdi & Nabney, Ian. (1999). Modelling Financial Time Series with Switching State Space Models.

Hendershott, T., & Menkveld, A. J. (2014). Price pressures. *Journal of Financial Economics, 114*(3), 405–423.

Kane, M.J., Price, N., Scotch, M. *et al.* (2014) Comparison of ARIMA and Random Forest time series models for prediction of avian influenza H5N1 outbreaks. *BMC Bioinformatics* 15, 276.

Appendix

R code for the MCMC method:

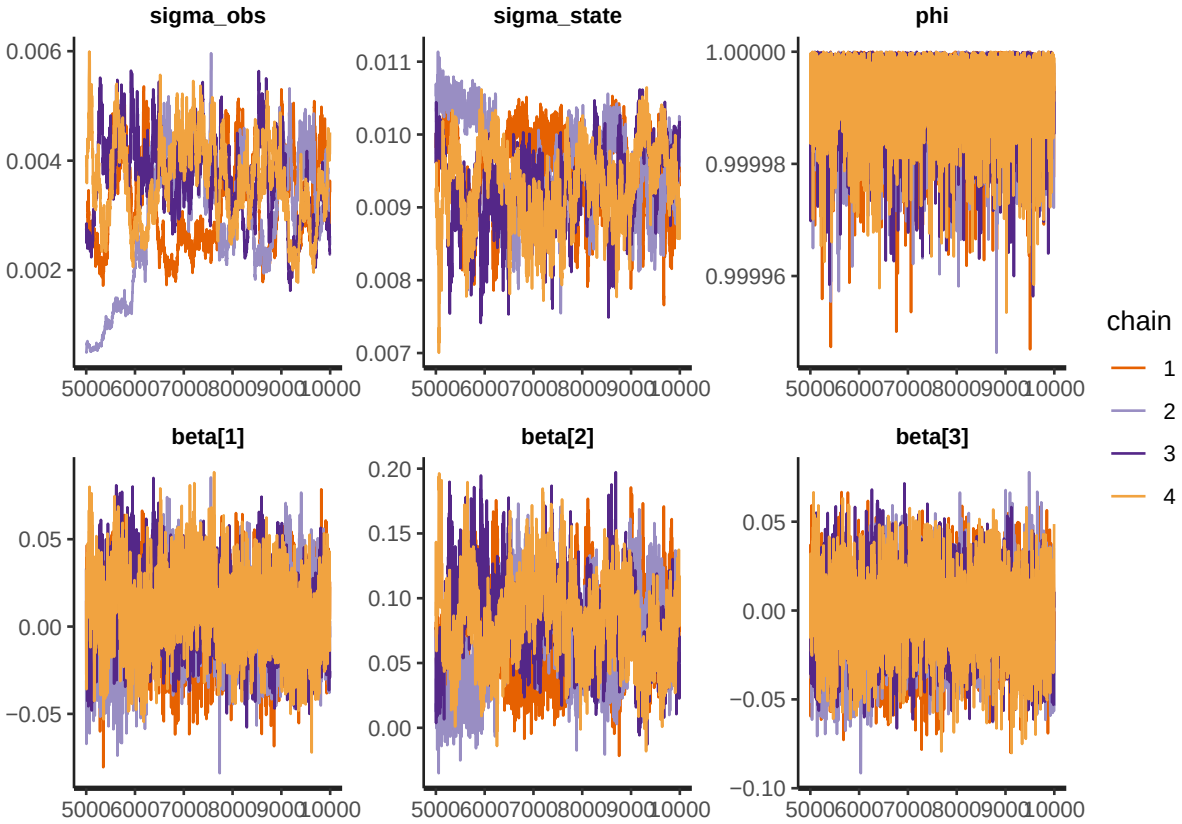
```
stan_code <- "
data {
  int<lower=0> N;                  // Number of time points (training)
  int<lower=0> H;                  // Forecast horizon
  int<lower=0> K;                  // Number of sector indices
  vector[N] y_main;                // Log-transformed NIFTY 50 (training)
  matrix[N, K] y_sectors;          // Log-transformed sector indices (training)
}
parameters {
  // State-space parameters
  real<lower=0> sigma_obs;         // Observation noise
  real<lower=0> sigma_state;       // State evolution noise
  vector[N] states;                // Latent states

  // Sector influence parameters
  vector[K] beta;                  // Sector coefficients

  // AR parameter for state evolution
  real<lower=-1, upper=1> phi;     // Persistence parameter
}
model {
  // Priors
  sigma_obs ~ exponential(10);      // Tighter prior for stability
  sigma_state ~ exponential(10);    // Tighter prior for stability
  phi ~ normal(0.9, 0.1);          // Prior centered at high persistence
  beta ~ normal(0, 0.5);           // Sector influence priors

  // Initial state prior
  states[1] ~ normal(y_main[1], 0.1);

  // State evolution with sector influences
  // Start from t=3 to avoid indexing issues when calculating differences
  for (t in 3:N) {
    real mu = phi * states[t-1];

    // Add sector influences (using first differences)
    for (k in 1:K) {
      mu += beta[k] * (y_sectors[t-1, k] - y_sectors[t-2, k]);
    }

    states[t] ~ normal(mu, sigma_state);
```

```
  }

  // Special case for t=2 without using differences
  states[2] ~ normal(phi * states[1], sigma_state);

  // Observation equation
  y_main ~ normal(states, sigma_obs);
}
generated quantities {
  // Forecast future states and observations
  vector[H] future_states;
  vector[H] forecasts;

  // Initialize with last state and sector changes
  future_states[1] = phi * states[N];

  // Add sector influences for first forecast
  if (N >= 2) {
    for (k in 1:K) {
      future_states[1] += beta[k] * (y_sectors[N, k] - y_sectors[N-1, k]);
    }
  }

  // Generate first forecast
  forecasts[1] = normal_rng(future_states[1], sigma_obs);

  // Generate remaining forecasts (assuming sector changes=0 for simplicity)
  for (h in 2:H) {
    future_states[h] = phi * future_states[h-1];
    forecasts[h] = normal_rng(future_states[h], sigma_obs);
  }
}
"

# Compile the model
efficient_state_space_model <- stan_model(model_code = stan_code)
```

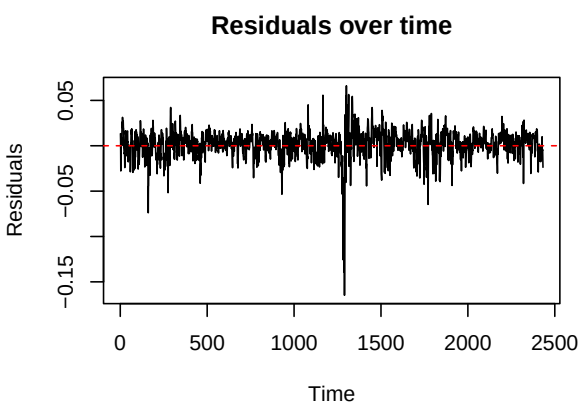
```
set.seed(535)
# Prepare the data for the state-space model
# Make sure to calculate differences for sector indices
stan_ss_data <- list(
  N = length(training_dates),
  H = forecast_horizon,
  K = 3,  # Number of sector indices
  y_main = train_y_main,
  y_sectors = as.matrix(train_y_sectors)
)

# Fit the state-space model with more efficient settings
ss_fit <- rstan::sampling(
  efficient_state_space_model,
  data = stan_ss_data,
  chains = 4,
  iter = 10000,  # Reduced iterations for speed
```

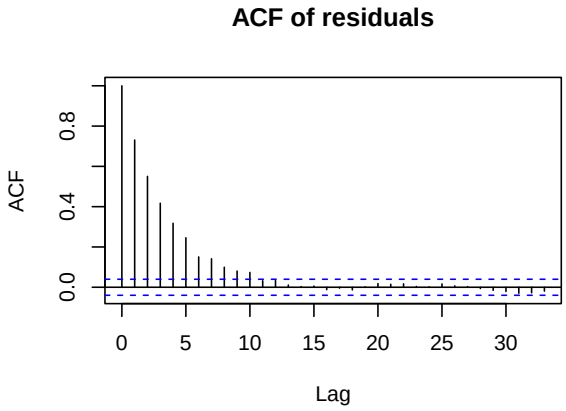
```
  warmup = 5000,
  thin = 2,      # Thinning to reduce memory usage
  control = list(
    adapt_delta = 0.9,
    max_treedepth = 10
  )
)
```

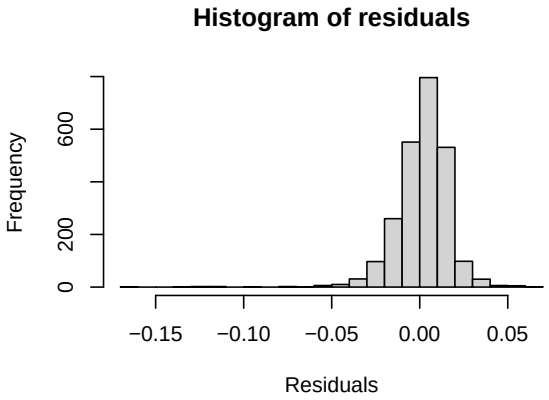
MCMC Trace Plots

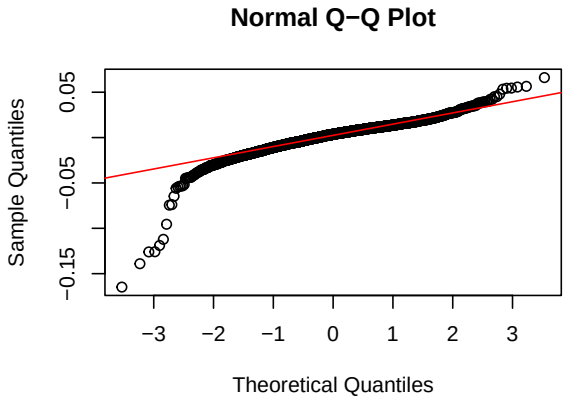


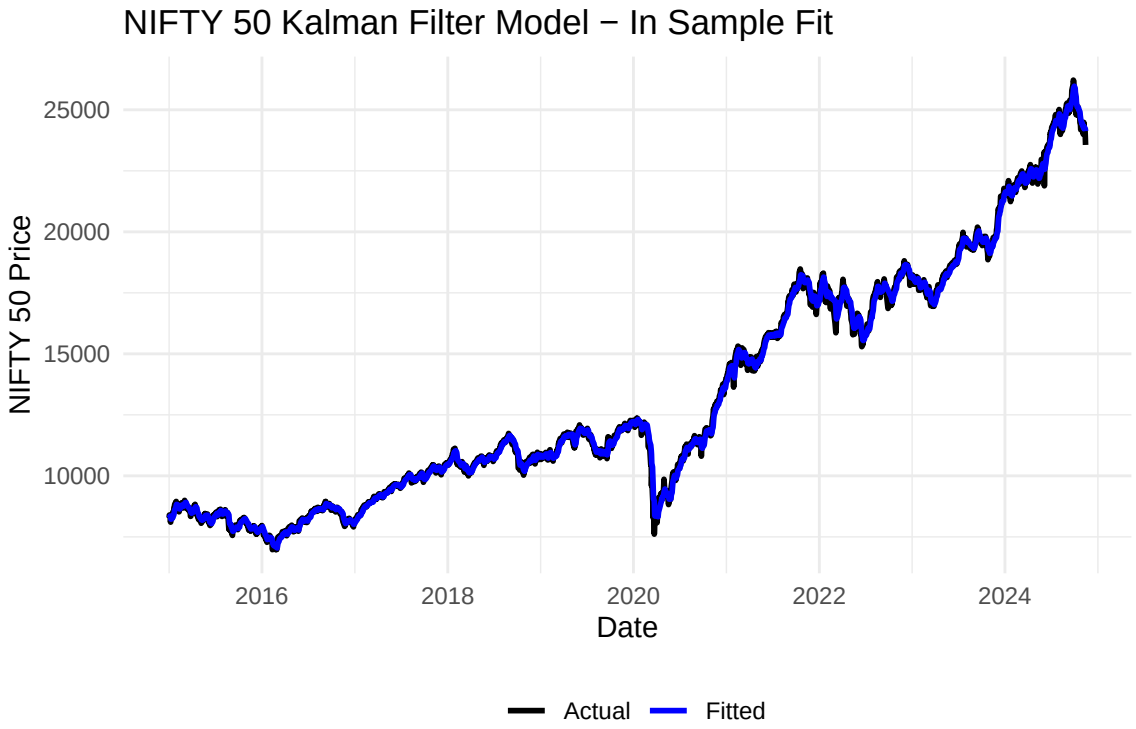
Kalman Filter Graphs


Residuals over time
Residuals
0.05
−0.05
−0.15
0
500
1000
1500
2000
2500
Time


ACF of residuals
ACF
0.8
0.4
0.0
0
5
10
15
20
25
30
Lag


Histogram of residuals
Frequency
600
200
0
−0.15
−0.10
−0.05
0.00
0.05
Residuals


Normal Q−Q Plot
Sample Quantiles
0.05
−0.05
−0.15
−3
−2
−1
0
1
2
3
Theoretical Quantiles


NIFTY 50 Kalman Filter Model − In Sample Fit
NIFTY 50 Price
25000
20000
15000
10000
2016
2018
2020
2022
2024
Date
Actual
Fitted