High Integrity Systems Project Time Series Analysis

Assignment 4

Mehjabeen Jahangeer Khan

Extended Kalman Filter (EKF)

The **Extended Kalman Filter (EKF)** is an extension of the traditional Kalman Filter (KF) designed to handle nonlinear systems. While the standard Kalman Filter assumes linear relationships between the state and measurements, the EKF is used when the system dynamics or measurement model are nonlinear [1].

Key Concepts

- **Nonlinearity**: The EKF deals with nonlinear systems where the process model (system dynamics) and measurement model are not linear functions [2]. In such cases, the EKF approximates these nonlinear functions with linear ones using a first-order Taylor series expansion (i.e., using Jacobian matrices).
- **State Prediction and Update**: Like the standard Kalman Filter, the EKF also follows the prediction and update steps. However, since the system is nonlinear, the prediction and update steps involve linearization.

Steps of the EKF

1. Prediction Step

- Predict the next state based on the previous state estimate and the nonlinear process model.
- Compute the Jacobian matrix of the process model to linearize it.

Predicted state:

$$\hat{x}_k^- = f(\hat{x}_{k-1}, u_{k-1})$$

where $f(\hat{x}_{k-1}, u_{k-1})$ is the nonlinear state transition function, and u_{k-1} is the control input.

- Predict the covariance matrix, which reflects the uncertainty about the predicted state.

Predicted covariance:

$$P_k^- = F_{k-1} P_{k-1} F_{k-1}^T + Q$$

where F_{k-1} is the Jacobian matrix of the state transition function, and Q is the process noise covariance.

2. Update Step

- Compute the Kalman gain based on the predicted covariance and the Jacobian matrix of the measurement model.
- Use the Kalman gain to update the predicted state estimate based on the new measurement.

Measurement prediction:

$$\hat{z}_k = h(\hat{x}_k^-)$$

where $h(\hat{x}_k^-)$ is the nonlinear measurement function, and $\hat{z}_k$ is the predicted measurement.

- Compute the innovation (the difference between the actual measurement and the predicted measurement).

Innovation:

$$y_k = z_k - \hat{z}_k$$

where z_k is the actual measurement.

- Compute the Kalman gain K_k and update the state estimate:

$$K_k = P_k^- H_k^T \left(H_k P_k^- H_k^T + R\right)^{-1}$$

where H_k is the Jacobian of the measurement function, and R is the measurement noise covariance.

Updated state estimate:

$$\hat{x}_k = \hat{x}_k^- + K_k y_k$$

- Update the covariance estimate:

$$P_k = (I - K_k H_k) P_k^-$$

Important Matrices in EKF

- **Jacobian of the process model** F_k: Linearized approximation of the state transition function.
- **Jacobian of the measurement model** H_k: Linearized approximation of the measurement function.
- **State vector** $\hat{x}_k$: The estimated state at time step k.
- **Covariance matrix** P_k: Represents the uncertainty in the state estimate at time step k.

Applications of EKF

- **Robotics**: Used in localization and mapping (SLAM), where both motion models and sensor models are nonlinear.
- **Navigation**: For GPS and inertial navigation systems where the dynamics are nonlinear.
- **Control Systems**: Used for estimating the state of systems with nonlinear dynamics (e.g., aircraft control, autonomous vehicles).

Limitations

- **Linearization Errors**: Since the EKF linearizes the nonlinear models at each step, if the system is highly nonlinear, the linear approximation may be inaccurate.
- **Computational Complexity**: The EKF requires computation of Jacobians at each step, which may be computationally expensive for large-scale systems.

Summary of chapter 10 "Introduction to Global Forecasting Models"

What Are GFMs?

Unlike Local Forecasting Models (LFMs), which train separate models for each time series, Global Forecasting Models (GFMs) use a single model to forecast multiple time series simultaneously. This approach assumes that all time series in a dataset share a common data-generating process or have related attributes, enabling shared learning [3].

Why Use GFMs?

- **Efficiency**: GFMs reduce the computational and engineering complexity of managing multiple models, especially for large datasets.
- **Accuracy**: They leverage the aggregated history of multiple time series, enabling better generalization and cross-learning.

Advantages of GFMs Over LFMs

Improved Data Utilization

GFMs increase the "width" of datasets by using data from all series, enabling a data-driven approach even for time series with limited history. LFMs, on

the other hand, often rely on strong assumptions like seasonality or trend to compensate for small data sizes.

Cross-Learning

GFMs share patterns across time series, benefiting those with limited data by learning from others with similar attributes. For example, new products or households without historical data can still achieve accurate forecasts by borrowing patterns from related series.

Multi-task Learning

GFMs treat forecasting for each time series as a separate task but train them together, enabling the model to learn shared features like seasonality. This acts as a regularizer, preventing overfitting by focusing on commonalities across series.

Engineering Simplicity

LFMs become impractical when dealing with thousands or millions of time series, requiring significant resources for training and managing. GFMs streamline this process by training one model, reducing deployment, monitoring, and retraining overhead.

Drawbacks of GFMs

Assumption of Common Data-Generating Process

GFMs assume relatedness among time series, which may not always hold true, leading to underfitting for highly diverse datasets.

Struggles with Unrelated Series

For datasets with minimal similarity, GFMs might fail. However, some studies suggest meta-learning capabilities in GFMs may help even for unrelated series.

Limited Interpretability

GFMs are less transparent than simpler LFMs, requiring advanced techniques for understanding their decisions.

Strategies to Improve GFMs

Feature Engineering

- **Lag Features**: Adding more lag terms (historical data points) enhances the model's memory of past events. Seasonal lags are particularly effective but must avoid overfitting.
- **Rolling Features**: Use rolling averages to condense historical data into meaningful statistics like mean or maximum values.
- **EWMA Features**: Incorporate exponentially weighted moving averages to introduce infinite memory into finite-memory models.

Incorporating Meta-Features

Meta-features (e.g., product IDs, categories) distinguish between time series and improve forecasting for heterogeneous datasets.

Encoding Techniques

- **Ordinal Encoding**: Assigns numerical codes to categories but risks introducing unintended ordinal relationships.
- **One-Hot Encoding**: Creates binary columns for each category but can become sparse with high-cardinality features.
- **LightGBM's Native Encoding**: Gradient-boosting models like LightGBM and CatBoost handle categorical features directly and efficiently.

Dataset Partitioning

Dividing datasets into smaller subsets improves model performance by focusing on more homogenous groups:

- **Judgmental Partitioning**: Uses domain knowledge or meta-features for grouping (e.g., based on demographics).
- **Algorithmic Partitioning**: Employs clustering techniques (e.g., t-SNE or Dynamic Time Warping) to identify groups based on similarity.

Hyperparameter Tuning

GFMs allow for extensive hyperparameter optimization due to shorter training times. Methods like **Bayesian Optimization** are computationally efficient, compared to grid or random search.

Performance Analysis

Efficiency Gains

Training a GFM on 150 households took 57 seconds compared to 8 minutes for LFMs, achieving 777% faster training with an 8.78% decrease in Mean Absolute Error (MAE).

Partitioning Results

Algorithmic clustering outperformed judgmental methods and random partitioning, demonstrating the importance of grouping similar series.

Hyperparameter Optimization

Bayesian optimization concentrated computational effort in promising regions, yielding better results than grid or random search.

Interpretability

Challenges

GFMs are less inherently interpretable than simple statistical models like ARIMA.

Post-Hoc Techniques

Methods like **Shapley Values**, **permutation importance**, and **LIME** explain model predictions by analyzing feature impacts.

Key Takeaways

- **Superiority of GFMs**: GFMs often outperform LFMs in terms of accuracy, scalability, and computational efficiency, particularly as the number of time series grows.
- **Flexibility**: Incorporating meta-features and advanced encoding allows GFMs to handle diverse datasets effectively.
- **Future Directions**: Transition to deep learning for time series forecasting, with GFMs serving as a robust baseline for machine learning-based approaches.

Conclusion

The chapter positions GFMs as a powerful paradigm in modern time series forecasting, offering practical solutions to common challenges. With proper tuning and enhancements, GFMs provide a scalable and accurate alternative to traditional forecasting methods, paving the way for more advanced approaches in the field.

References

[1] R. E. Kalman, *A New Approach to Linear Filtering and Prediction Problems.* ASME, 1960, vol. 82, no. 1.

[2] J. K. Julier and J. K. Uhlmann, "Unscented filtering and nonlinear estimation," *Proceedings of the IEEE*, vol. 92, no. 3, pp. 401–422, 2004.

[3] M. Joseph, *Modern Time Series Forecasting with Python: Explore industry-ready time series forecasting using modern machine learning and deep learning.* Packt Publishing Ltd, 2022.