TU Dortmund

Case Studies

Project 1: Forecasting electrical power consumption using individual load profiles

Lecturers:
Dr. Christine Müller
Dr. Mirko Jakubzik

Author: Mohaiminul Islam

Group number: 9

Group members: Zihad Hossain, Zakia Zaynab

August 12, 2025

Contents

Introduction

Energy consumption forecasting at the household level plays a vital role in demand-side management and grid stability, particularly in the context of energy-efficient technologies such as heat pumps. This project investigates short-term power consumption forecasting using multivariate time series data from 33 households equipped with heat pumps. Specifically, we develop and compare dynamic filtering models to forecast electricity usage, leveraging both individual and cluster-based load profile information.

The analysis begins with the preprocessing and smoothing of raw power consumption data into daily curves. Subsequently, 30 households are randomly selected to model the influence of household identity, weekday, and seasonal effects on daily load profiles. These estimated effects are then used to perform clustering, and the remaining three test households are classified accordingly. Two filtering techniques—Kalman filtering and particle filtering—are then applied to forecast the next-hour consumption for each of the test households, both with and without the inclusion of cluster-level profile information. The study ultimately evaluates the forecasting benefit of incorporating exogenous load profiles and compares the performance of the two filtering methods under various smoothing and input configurations.

Section 2 introduces the dataset and outlines the specific objectives of the study. Section 3 outlines the statistical methods used for the subsequent analysis. In Section 4, these models are applied to extract insights and to generate forecasts, followed by a comparison of prediction errors. Finally, Section 5 summarizes the findings, discusses their implications, and proposes potential directions for future work.

1 Problem statement

1.1 Dataset and Preprocessing

The dataset analyzed in this report is provided by the instructors of the "Case Studies" course at TU Dortmund University during the summer term 2023/24. The data set consists of 33 individual CSV files, each representing a single household with 15-minute interval readings. For this study, only the `HAUSHALT_TOT` variable is retained, while auxiliary columns such as `PUMPE_TOT` and `TEMPERATURE:TOTAL` are excluded.

All files were combined into a single DataFrame, with household identifiers extracted from filenames and stored as a categorical variable. Timestamps were parsed into datetime format, and relevant temporal features were engineered: `day` (integer, day of year), `hour` (integer), `weekday` (categorical), `is_weekday` (binary categorical), and `season` (continuous). The `season` variable was designed to capture annual periodicity using a sinusoidal transformation of the day index:

$$\text{season}_t = \sin\left(\frac{2\pi}{365} \cdot \text{day}_t - (31 + 28 + 21)\right)$$

The data was then aggregated at the hourly level per household and day, computing the mean of `HAUSHALT_TOT` for each group. Thus, the original 15-minute interval data was converted into hourly resolution for all subsequent analysis.

Two smoothing strategies are applied to generate structured daily load profiles, as formally defined in Section 3. These smoothed series were pivoted to wide-format daily curves with 24 hourly values each.Missing values were evaluated after smoothing, which is shown in fig 1,are addressed through linear interpolation.

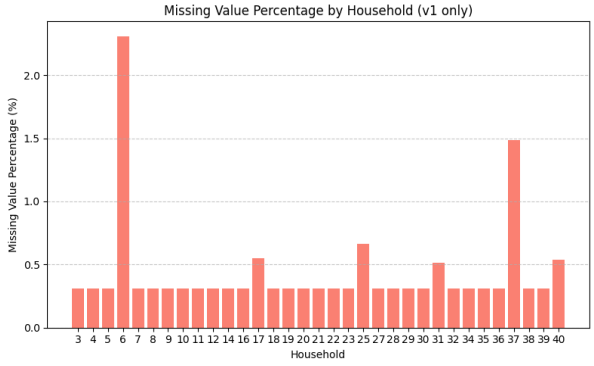


Figure 1: percentage of Missing values per household after smoothing

The resulting dataset contains two complete sets of smoothed daily profiles for each household, enriched with temporal and categorical metadata. These profiles serve as the basis for clustering, classification, and one-hour-ahead forecasting using dynamic filtering models described in subsequent sections.

1.2 Project Objectives

The goal of this project is to forecast short-term household electricity consumption using Kalman and particle filters, with and without exogenous load profile inputs.The analysis begins with MANOVA to assess the effects of weekday, season, and household on smoothed load profiles. Model selection is based on AIC. Estimated model parameters are then used to cluster households with similar patterns, forming the basis for representative load profiles.

Three test households are classified into these clusters, and forecasts are generated using both filters under different input configurations. Forecast accuracy is evaluated using RMSE.

2 Statistical methods

2.1 Moving Average Smoothing

The Moving Average (MA) method is a classical technique in time series analysis used to smooth short-term fluctuations and reveal underlying trends. Given a univariate time series $\{y_t\}_{t=1}^{T}$, the observed data is assumed to follow the model:

$$y_t = f(t) + \varepsilon_t,$$

where $f(t)$ denotes a smooth deterministic trend and ε_t is a stochastic noise component with zero mean. A symmetric moving average of window size $2k+1$ estimates the trend as:

$$\hat{f}(t) = \frac{1}{2k+1} \sum_{j=-k}^{k} y_{t+j}, \quad \text{for } t = k+1, \ldots, T-k. \tag{1}$$

This estimator relies on the assumption that adjacent observations are locally correlated. Increasing k results in a smoother estimate but may introduce greater bias, particularly near local extrema. Under the assumption that $\varepsilon_t \sim$ i.i.d. $(0, \sigma^2)$, the approximate bias and variance are given by:

$$\mathbb{E}[\hat{f}(t)] - f(t) \approx \frac{1}{2} f''(t) k(k+1), \quad \mathrm{Var}[\hat{f}(t)] \approx \frac{\sigma^2}{2k+1}.$$

Hence, the choice of window width involves a trade-off between smoothness (bias) and fidelity (variance) in the trend estimate.

Hyndman (2011)

2.2 Multivariate Analysis of Variance (MANOVA)

To examine whether group membership significantly affects multiple response variables jointly, we employed Multivariate Analysis of Variance (MANOVA). This technique generalizes the univariate ANOVA framework to accommodate multiple dependent variables, while accounting for potential correlations among them. The validity of MANOVA relies on several assumptions: multivariate normality, homogeneity of covariance across groups, and absence of multicollinearity among the dependent variables.

Let $\mathbf{Y} \in \mathbb{R}^{n \times p}$ denote the data matrix, where n is the number of observations and p is the number of response variables. Suppose the data are divided into k groups according to a categorical factor G. The null hypothesis of MANOVA is that the mean vectors of the groups are equal:

$$H_0 : \boldsymbol{\mu}_1 = \boldsymbol{\mu}_2 = \cdots = \boldsymbol{\mu}_k,$$

where $\boldsymbol{\mu}_i$ is the population mean vector of the i-th group. The alternative hypothesis posits that at least one group mean vector differs.

To test this, the total variability in $\mathbf{Y}$ is decomposed into between-group and within-group components. The variability between-groups is given by:

$$\mathbf{H} = \sum_{i=1}^{k} n_i (\bar{\mathbf{Y}}_i - \bar{\mathbf{Y}})(\bar{\mathbf{Y}}_i - \bar{\mathbf{Y}})^\top,$$

where n_i is the number of observations in group i, $\bar{\mathbf{Y}}_i$ is the mean vector of group i, and $\bar{\mathbf{Y}}$ is the overall mean vector. The within-group variation is defined as:

$$\mathbf{E} = \sum_{i=1}^{k} \sum_{j=1}^{n_i} (\mathbf{Y}_{ij} - \bar{\mathbf{Y}}_i)(\mathbf{Y}_{ij} - \bar{\mathbf{Y}}_i)^\top,$$

where $\mathbf{Y}_{ij}$ represents the j-th observation in group i.

The test statistic employed is Wilks' Lambda, denoted by:

$$\Lambda = \frac{\det(\mathbf{E})}{\det(\mathbf{H} + \mathbf{E})}.$$

Wilks' Lambda measures the proportion of total variance not accounted for by group differences. Smaller values of Λ indicate that a greater proportion of the total variance is explained by the group factor, thereby providing stronger evidence against the null hypothesis.(French et al., 2008)

The test statistic Λ is transformed into an approximate F-statistic, and the null hypothesis H_0 is rejected if the resulting p-value is less than the significance level α, typically $\alpha = 0.05$.

2.3 Multivaaite parameter estimation

To estimate the joint effects of predictors on multiple dependent variables, we utilize the multivariate linear regression model. Let $\mathbf{Y} \in \mathbb{R}^{n \times p}$ denote the response matrix consisting of p dependent variables observed across n units, and let $\mathbf{X} \in \mathbb{R}^{n \times k}$ be the design matrix of explanatory variables, assumed to be common to all response variables.

The multivariate model can first be expressed as a system of p separate univariate regressions:

$$\mathbf{y}_j = \mathbf{X}\boldsymbol{\beta}_j + \boldsymbol{\varepsilon}_j, \quad \text{for } j = 1, \ldots, p,$$

where $\mathbf{y}_j \in \mathbb{R}^n$ is the j-th response vector, $\boldsymbol{\beta}_j \in \mathbb{R}^k$ is the corresponding vector of regression coefficients, and $\boldsymbol{\varepsilon}_j \in \mathbb{R}^n$ denotes the associated residuals.

Combining these into matrix form, the model becomes:

$$\mathbf{Y} = \mathbf{X}\mathbf{B} + \mathbf{E},$$

where $\mathbf{B} \in \mathbb{R}^{k \times p}$ is the matrix whose columns are the individual coefficient vectors $\boldsymbol{\beta}_j$, and $\mathbf{E} \in \mathbb{R}^{n \times p}$ is the matrix of residuals.

The coefficient matrix $\mathbf{B}$ is estimated by minimizing the total sum of squared residuals across all responses, defined via the Frobenius norm:

$$\hat{\mathbf{B}} = \arg\min_{\mathbf{B}} \|\mathbf{Y} - \mathbf{X}\mathbf{B}\|_F^2.$$

The solution to this least squares problem, assuming $\mathbf{X}^\top\mathbf{X}$ is invertible, is given by:

$$\hat{\mathbf{B}} = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{Y}.$$

Using this estimator, the fitted values are obtained by $\hat{\mathbf{Y}} = \mathbf{X}\hat{\mathbf{B}}$, and the residual matrix is computed as $\hat{\mathbf{E}} = \mathbf{Y} - \hat{\mathbf{Y}}$. The estimation was performed via a routine equivalent to the `_multivariate_ols_fit` function, which directly applies this matrix-based OLS method. (Helwig, 2017, pp. 44–51)

2.4 Silhouette Analysis

In this report, to evaluate the quality of clustering and determine the optimal number of clusters, the silhouette method is employed. The silhouette coefficient measures how well an individual observation fits within its assigned cluster compared to other clusters, offering a geometric and interpretable criterion for clustering validation.

Let $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\} \subset \mathbb{R}^M$ be a set of N observations assigned to K clusters denoted by $\mathcal{C}_1, \ldots, \mathcal{C}_K$. For a given point $\mathbf{x}_i$, the silhouette coefficient $s(i)$ is defined as:

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}},$$

where $a(i)$ denotes the average intra-cluster distance, i.e., the mean distance between $\mathbf{x}_i$ and all other points in its own cluster $\mathcal{C}_{c(i)}$:

$$a(i) = \frac{1}{|\mathcal{C}_{c(i)}| - 1} \sum_{\substack{\mathbf{x}_j \in \mathcal{C}_{c(i)} \\ j \neq i}} d(\mathbf{x}_i, \mathbf{x}_j).$$

In contrast, $b(i)$ represents the minimum average distance from $\mathbf{x}_i$ to points in any other cluster $\mathcal{C}_k$ with $k \neq c(i)$, defined as:

$$b(i) = \min_{k \neq c(i)} \left(\frac{1}{|\mathcal{C}_k|} \sum_{\mathbf{x}_j \in \mathcal{C}_k} d(\mathbf{x}_i, \mathbf{x}_j) \right).$$

Here, $d(\cdot, \cdot)$ denotes the Euclidean distance. The silhouette value $s(i)$ lies in the interval $[-1, 1]$. A value close to 1 indicates that the observation is well matched to its own cluster and poorly matched to others. A value near zero implies that the point lies between two clusters, and negative values indicate possible misclassification.

To assess the global clustering quality, the average silhouette score across all data points is computed:

$$\bar{s} = \frac{1}{N} \sum_{i=1}^{N} s(i).$$

The optimal number of clusters K^* is the one that maximizes $\bar{s}$, indicating a clustering structure with high intra-cluster similarity and low inter-cluster similarity.(Januzaj et al., 2023)

2.5 Agglomerative Hierarchical Clustering

Agglomerative Hierarchical Clustering (AHC) is a bottom-up clustering method that builds a nested hierarchy of clusters by iteratively merging the two most similar clusters. The process begins with each observation forming its own singleton cluster. At each step, the pair of clusters with the smallest dissimilarity is merged, reducing the number of clusters by one. This procedure continues until all observations are grouped into a single cluster, producing a dendrogram that illustrates the hierarchical relationships among data points.

Formally, let $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\} \subset \mathbb{R}^M$ be the dataset with N observations. Denote the set of clusters at step t by $\mathcal{G}^{(t)} = \{G_1^{(t)}, \ldots, G_{K_t}^{(t)}\}$, where K_t is the number of clusters at iteration t. At each step, the algorithm selects the pair of clusters (G_p, G_q) with minimum inter-cluster dissimilarity $D(G_p, G_q)$ and merges them:

$$G_r = G_p \cup G_q, \quad \mathcal{G}^{(t+1)} = \mathcal{G}^{(t)} \setminus \{G_p, G_q\} \cup \{G_r\}.$$

The definition of dissimilarity $D(G_p, G_q)$ depends on the linkage method used. In this study, we apply *Ward's method*, which defines dissimilarity in terms of the increase in the total within-cluster sum of squares (WCSS) when two clusters are merged.

Let $m(G)$ denote the centroid of cluster G, and define the within-cluster error as:

$$E(G) = \sum_{\mathbf{x}_k \in G} \|\mathbf{x}_k - m(G)\|^2.$$

The increase in within-cluster error due to merging G_p and G_q is:

$$\Delta E(G_p, G_q) = E(G_p \cup G_q) - E(G_p) - E(G_q).$$

Ward's dissimilarity is then defined as $D(G_p, G_q) = \Delta E(G_p, G_q)$. This method tends to produce compact, spherical clusters and is particularly effective when data are assumed to follow Gaussian distributions.

The clustering process was implemented using the `AgglomerativeClustering` class from the `scikit-learn` library, employing Ward linkage on Euclidean distance. The number of clusters K was selected based on the silhouette analysis described previously. (Miyamoto, 2022, pp. 19–20, 24)

2.6 State Space Models

Analyzing the distribution of time series data $(Y_1, \ldots, Y_t)$ for $t \geq 1$ often poses significant challenges due to temporal dependencies, making assumptions such as independence or exchangeability typically inappropriate. Markovian dependencies, representing the simplest form of temporal dependence, are often assumed. A Markov chain describes a time series $\{Y_t\}_{t\geq 1}$ satisfying the property:

$$\pi(y_t \mid y_{1:t-1}) = \pi(y_t \mid y_{t-1}),$$

implying conditional independence of Y_t from $Y_{1:t-2}$ given Y_{t-1}. Hence, the joint distribution simplifies to:

$$\pi(y_{1:t}) = \pi(y_1) \prod_{j=2}^{t} \pi(y_j \mid y_{j-1}).$$

(Petris et al., 2009, p. 39-40)

State space models extend Markov chains by introducing an unobserved latent Markovian state process $\{\theta_t\}$, such that observed series $\{Y_t\}$ represent noisy measurements of these underlying states.

$$\begin{array}{ccccccccccc} \theta_0 & \to & \theta_1 & \to & \cdots & \to & \theta_{t-1} & \to & \theta_t & \to & \theta_{t+1} \to \ldots \\ \downarrow & & \downarrow & & & & \downarrow & & \downarrow & & \\ Y_1 & & Y_{t-1} & & & & Y_t & & Y_{t+1} & & \end{array}$$

Formally, a state space model consists of an $\mathbb{R}^p$-valued latent time series $\{\theta_t\}_{t\geq 0}$ that constitutes a Markov chain, and an $\mathbb{R}^m$-valued observable time series $\{Y_t\}_{t\geq 1}$ which are conditionally independent given the latent states, i.e., $p(Y_t \mid \theta_t, Y_{t-1}, \theta_{t-1}, \ldots) = p(Y_t \mid \theta_t)$. Specifically, each observation Y_t depends solely on the current state θ_t.

Thus, the joint distribution is fully specified as:

$$\pi(\theta_{0:t}, y_{1:t}) = \pi(\theta_0) \prod_{j=1}^{t} \pi(\theta_j \mid \theta_{j-1}) \pi(y_j \mid \theta_j).$$

Marginalizing over the hidden states yields:

$$\pi(y_{1:t}) = \int \pi(\theta_0) \prod_{j=1}^{t} \pi(\theta_j \mid \theta_{j-1}) \pi(y_j \mid \theta_j) \, d\theta_0 \ldots d\theta_t.$$

(Petris et al., 2009, p. 40-41)

2.6.1 Dynamic Linear Models (DLM)

Dynamic linear models are Gaussian linear state space models, specified by:

$$\theta_0 \sim N_p(c_0, C_0),$$

with observation and state equations for $t \geq 1$:

$$\begin{aligned} Y_t &= F_t \theta_t + v_t, \quad v_t \sim N_m(0, V_t), \\ \theta_t &= G_t \theta_{t-1} + w_t, \quad w_t \sim N_p(0, W_t), \end{aligned}$$

where $F_t \in \mathbb{R}^{m \times p}$ and $G_t \in \mathbb{R}^{p \times p}$ are known matrices, and $\{v_t\}$, $\{w_t\}$ are independent Gaussian noise sequences. Consequently, the posterior and predictive distributions remain Gaussian, simplifying inference considerably. (Petris et al., 2009, p. 41-42)

2.6.2 Kalman Filter

The Kalman filter sequentially estimates θ_t given observations $y_{1:t}$. Starting from $\theta_0 \sim N_p(c_0, C_0)$, the recursive steps are:

Prediction step:

$$\begin{aligned} \theta_t | y_{1:t-1} &\sim N_p(a_t, R_t), \quad a_t = G_t c_{t-1}, \quad R_t = G_t C_{t-1} G_t^\top + W_t, \\ Y_t | y_{1:t-1} &\sim N_m(f_t, Q_t), \quad f_t = F_t a_t, \quad Q_t = F_t R_t F_t^\top + V_t. \end{aligned}$$

Update (filtering) step:

$$\theta_t|y_{1:t} \sim N_p(c_t, C_t), \quad c_t = a_t + R_t F_t^\top Q_t^{-1}(y_t - f_t), \quad C_t = R_t - R_t F_t^\top Q_t^{-1} F_t R_t.$$

Forecasting future states and observations for $k \geq 1$:

$$\theta_{t+k}|y_{1:t} \sim N_p(a_{t+k}, R_{t+k}), \quad a_{t+k} = G_{t+k} a_{t+k-1}, \quad R_{t+k} = G_{t+k} R_{t+k-1} G_{t+k}^\top + W_{t+k},$$

$$Y_{t+k}|y_{1:t} \sim N_m(f_{t+k}, Q_{t+k}), \quad f_{t+k} = F_{t+k} a_{t+k}, \quad Q_{t+k} = F_{t+k} R_{t+k} F_{t+k}^\top + V_{t+k}.$$

(Petris et al., 2009, p. 53)

2.6.3 Particle Filter

Particle filtering approximates the posterior distribution of latent states θ_t by a discrete set of particles. The posterior expectation is approximated as:

$$\hat{E}(\theta_t \mid y_{1:t}) = \sum_{i=1}^{N} w_t^{(i)} \theta_t^{(i)}, \quad \theta_t^{(i)} \in \mathbb{R}^p.$$

Under Gaussian assumptions analogous to the Kalman filter, the particle filter is implemented as follows:

For initialization, draw N particles $\theta_0^{(i)} \sim \mathcal{N}_p(c_0, C_0)$, and assign equal weights $w_0^{(i)} = \frac{1}{N}$, for $i = 1, \ldots, N$.

For each time step $t \geq 1$:

1. Define the latent process:

$$\tilde{\theta}_t^{(i)} = G_t \theta_{t-1}^{(i)} + w_t^{(i)}, \quad w_t^{(i)} \sim \mathcal{N}_p(0, W_t),$$

   and draw N i.i.d. samples with $\tilde{\theta}_t^{(i)} \sim \mathcal{N}_p(G_t \theta_{t-1}^{(i)}, W_t)$.

2. Compute weights and normalize:

$$\tilde{w}_t^{(i)} = f_{Y_t}(y_t \mid F_t \tilde{\theta}_t^{(i)}, V_t), \quad w_t^{(i)} = \frac{\tilde{w}_t^{(i)}}{\sum_{j=1}^{N} \tilde{w}_t^{(j)}}.$$

3. Resample N particles $\theta_t^{(i)}$ from the discrete distribution defined by:

$$\mathbb{P}(\theta_t = \tilde{\theta}_t^{(i)}) = w_t^{(i)}, \quad i = 1, \ldots, N,$$

using multinomial resampling.

4. Estimate the posterior moments:

$$\hat{E}(\theta_t \mid y_{1:t}) = \bar{\theta}_t, \quad \hat{\mathrm{Cov}}(\theta_t \mid y_{1:t}) = \mathrm{Cov}(\theta_t^{(1)}, \ldots, \theta_t^{(N)}).$$

(Petris et al., 2009, p. 208-211)

Particle filters thus offer flexibility in scenarios involving non-linearities or non-Gaussian dynamics.

2.7 Parameter Optimization

To tune the parameters of the time series forecasting model, we minimized prediction error using cross-validated root-mean-squared error (RMSE) as the objective criterion. Let $\{y_t\}_{t=1}^T$ denote the observed time series and $\{\hat{y}_t\}_{t=1}^T$ the corresponding model predictions. The RMSE is defined as:

$$\mathrm{RMSE} = \sqrt{\frac{1}{T}\sum_{t=1}^{T}(y_t - \hat{y}_t)^2}.$$

(Karunasingha, 2022)

To assess generalization performance, K-fold cross-validation was employed. For each fold $k = 1, \ldots, K$, the model was trained on $K-1$ folds and evaluated on the remaining fold. The average RMSE across all folds served as the performance metric. Specifically, this study involved forecasting electricity consumption for three households, each represented by a separate time series. Using 50% of the data for training, model performance was assessed via cross-validated root mean squared error (CV-RMSE), defined as

$$\text{CV-RMSE} = \frac{1}{K}\sum_{k=1}^{K}\left(\frac{1}{H}\sum_{h=1}^{H}\mathrm{RMSE}_h^{(k)}\right) \tag{2}$$

where $H = 3$ denotes the number of households and $\mathrm{RMSE}_h^{(k)}$ is the prediction error for household h in fold k. This aggregated metric served as the objective criterion for parameter optimization.

A random search was then conducted over the model's parameter space. For each randomly drawn parameter set $\boldsymbol{\theta}$, the cross-validated RMSE was computed, and the configuration yielding the lowest score was selected. This strategy balances computational feasibility with effective exploration of the parameter space.

3 Statistical analysis

This analysis was conducted using Python and several scientific computing libraries. Data handling and visualization were performed with `pandas`, `numpy`, and `matplotlib`. Statistical modeling, including MANOVA and regression, was carried out using `statsmodels` (Seabold and Perktold, 2010), while machine learning tasks such as clustering and classification were performed using `scikit-learn` (Pedregosa et al., 2011). Additional tools included `patsy` for formula-based modeling and `scipy.optimize` for parameter tuning (Virtanen et al., 2020). Methodological guidance was based on concepts from *An Introduction to Statistical Learning* (James et al., 2013).

3.1 Temporal Aggregation and Smoothing

To facilitate statistical analysis, the original electricity consumption data—recorded at 15-minute intervals (96 measurements per day)—was aggregated to an hourly resolution, yielding 24 observations per day. This temporal aggregation serves to reduce data dimensionality while retaining the key structural features of daily energy usage.

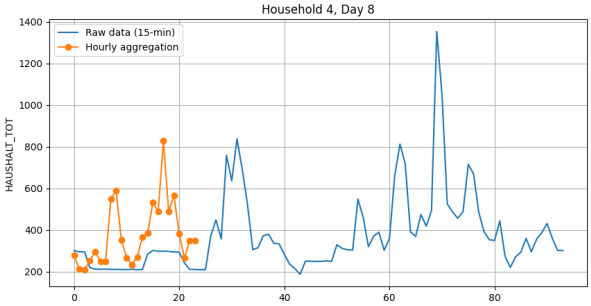


Figure 2: Comparison of raw 15-minute data and hourly aggregated consumption for Household 4 on Day 8

As depicted in 2 for Household 4 on Day 9, the aggregated hourly consumption closely follows the underlying shape of the original high-frequency data. This demonstrates

that the essential temporal consumption dynamics are preserved, allowing meaningful patterns and daily usage behaviors to be effectively captured. Hence, the 24-hour time points are adopted consistently throughout the report for all subsequent analyses.

Following temporal aggregation, two types of smoothing—daily smoothing and weekday-level smoothing—are employed to reduce short-term variability and enhance the interpretability of hourly consumption patterns. Daily smoothing is applied to the 24 hourly values for each household and each day using a centered moving average, as defined in Equation 1, with a window size of 5. This produced a smoothed version, denoted as v_1, which reduces random fluctuations while retaining the overall daily trend.

Subsequently, weekday-level smoothing is conducted across recurring weekday contexts. Specifically, the data were grouped by household, weekday, and hour, and the same centered moving average (window size 5) was applied within each group. This procedure yields a generalized load profile, denoted as v_2, that captures onsis- tent patterns in electricity usage across similar days while minimizing irregular changes.

Figure 3 compares the raw electricity consumption with the daily (v_1) and weekday-level (v_2) smoothed profiles over the 24-hour aggregated time format. The comparison illustrates how the smoothing procedures reduce short-term variability while preserving overall usage trends.

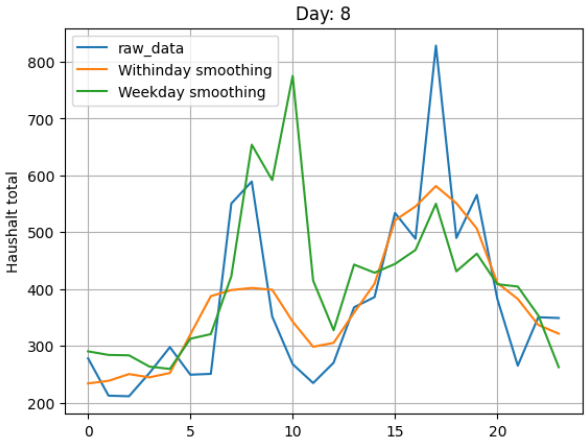


Figure 3: Comparison of raw, daily smoothed (v_1), and weekday-level smoothed (v_2) electricity consumption for Household 4 on Day 8.

3.2 Model Selection

To identify the most appropriate modeling framework, we evaluated four multivariate linear models across 30 randomly selected households under two smoothing regimes: daily smoothing (v_1) and weekday-profile smoothing (v_2). The models vary in how they encode temporal patterns—either using the full 7-level weekday factor or a binary weekday/weekend indicator—and whether they include interaction terms. The four model variants are as follows:

- **Model A**: Main effects only, with 7 weekday levels.
- **Model B**: Main effects and interactions, with 7 weekday levels.
- **Model C**: Main effects only, using a 2-level weekday/weekend factor.
- **Model D**: Main effects and interactions, using the 2-level factor.

Model Fit across all households Multivariate Analysis of Variance (MANOVA) is applied to both Model B and Model D, representing the 7-level weekday encoding and the 2-level weekday (weekday vs. weekend) encoding, respectively, to evaluate the significance of explanatory factors under the v_1 and v_2 smoothing scheme.

As shown in Table 8 and 9, for the v_1-smoothed data, both models demonstrated strong explanatory power. Notably, in Model B, the factors involving *season* and the *weekday* $\times$ *season* interaction were statistically insignificant, suggesting limited group separation along these dimensions. Conversely, Model D revealed statistically significant contributions from all major factors except for *weekday* $\times$ *season*, suggesting a better fit to the temporal structure of the data.

To complement the hypothesis testing, the Akaike Information Criterion (AIC) is calculated for each model to assess overall model fit. As shown in Table 1, Model D—which uses a simplified 2-level weekday encoding with interaction terms—achieved the lowest AIC (2,526,499.69). This suggests that it strikes the most effective balance between model complexity and explanatory performance.

For the v_2-smoothed data, Table 10, 11 indicatesall explanatory terms are statistically significant in both Model B and Model D. However, based on the Akaike Information Criterion (AIC) in 1, Model B—featuring full weekday encoding with interaction terms—achieved the lowest value, indicating optimal fit under the v_2 smoothing regime.

Table 1: Comparison of AIC Values for Competing Model Specifications Across Two Dataset Versions

Model Specification	**AIC (Version 1)**	**AIC (Version 2)**
Model A: Main Effects (7-day)	2,540,286.83	2,960,485.97
Model B: Interaction (7-day)	2,531,224.43	**2,881,492.91**
Model C: Main Effects (2-day)	2,540,883.30	2,962,439.78
Model D: Interaction (2-day)	**2,526,499.69**	2,910,585.79

Hence, for the v_2-smoothed data, the model structure cannot be further simplified without sacrificing explanatory power.

Model Fit for separate households In addition to evaluating overall fit, we applied the MANOVA separately to each household, focusing on the effects of season and weekday—encoded via a binary is_weekday indicator. As shown in Table 2, the *season* effect was statistically significant in all 30 households, and the `is_weekday` effect in 29 households; the *season* × `is_weekday` interaction was significant in 22 households.

Table 2: Household-level significance of Wilks' λ tests (daily smoothing, v_1).

Effect	Significant Households	Insignificant HH IDs
Season	30	—
is_weekday	29	37
is_weekday × season	22	$4, 6, 12, 14, 23, 25, 27, 37$

Moreover, the per-household AIC results in Appendix Table 12 show that Model D (main effects plus interaction with binary weekday encoding) achieved the lowest AIC for 28 household. For consistency, we therefore adopt Model D for all subsequent analyses.

Subsequently, under the v_2 smoothing regime, Model B (main effects with interaction using full weekday encoding) yielded the lowest AIC, as shown in Table 13, indicating the best model fit. However, for consistency and simplicity, Model D (main effects with interaction using binary weekday encoding) is employed in all subsequent analyses for both smoothing approaches.

3.3 Hierarchical Clustering of Household Parameters

In this step, hierarchical clustering is applied to 30 households using the parameter estimates derived from Model B (main effects with interaction), based on both the **v1** and **v2** smoothing approaches.

firstly, for the **v1** smoothing, a silhouette score was computed to evaluate cluster quality. The resulting PCA-based scatterplot (Figure 6) revealed two well-separated clusters, achieving a silhouette score of **0.69**. This high score indicates strong intra-cluster similarity and substantial inter-cluster separation.

To assess whether simpler temporal segmentations can yield comparable structure, clustering was also performed using parameter estimates derived from *weekday-only* and *weekend-only* subsets of the data. For this, Model B was reduced to include only the seasonal effect, and fitted separately to the weekday and weekend data. Both variants produced two clusters, with silhouette scores of **0.69** (weekday) and **0.68** (weekend)—comparable to the full model. As illustrated in Figure 4, the resulting cluster structure was consistent across both segmentations, with **23 households** in Cluster 0 and **7 households** in Cluster 1.

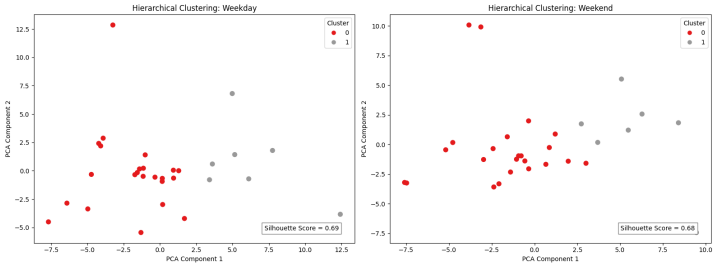


Figure 4: Clustering based on weekday/weekend-only subset for v1

Similarly, Figure 7 and Figure 8 presents the results for the **v2** smoothing, where weekday- and weekend-based clustering produced slightly higher silhouette scores of **0.63** and **0.62**, respectively, compared to **0.62** for the full model.

Given the comparable or improved cluster quality, reduced model complexity, and increased interpretability, the weekday/weekend-only configuration is adopted for all subsequent analyses.

3.4 Cluster Assignment via K-Nearest Neighbors

Following the clustering of households based on weekday- and weekend-only data using the simplified model $y_i \sim \text{season} + \varepsilon$, a classification model was developed to assign new households to the identified clusters.

To this end, a K-Nearest Neighbors (KNN) classifier was trained using the parameter vectors of the 30 households in the training set, each labeled with its corresponding cluster from the previous step. The input features to the classifier are the intercept along with season coefficients derived from the fitted models, and the target labels are the cluster assignments.

For each of the three held-out test households, the same seasonal model was fitted separately for weekday and weekend data to extract parameter estimates. These parameter vectors were then fed into the trained KNN classifier to predict the most likely cluster assignment, thereby enabling consistent model-based segmentation of unseen households.

Table 3 presents the assigned cluster for each test household based on both v1 and v2 smoothing configurations.

Table 3: Cluster assignments for test households based on weekday/weekend segmentation under v1 and v2 smoothing

House ID	**v1**		**v2**	
	Weekday	**Weekend**	**Weekday**	**Weekend**
8	cluster 0	cluster 0	cluster 0	cluster 0
11	cluster 0	cluster 0	cluster 0	cluster 1
34	cluster 0	cluster 0	cluster 0	cluster 0

3.5 Load Profile Estimation

The daily load profiles for each cluster is estimated based on household-level predictions from a simplified multivariate model. For each household i, and separately for weekday and weekend data, the following model was fitted:

$$\mathbf{y}_i \sim \text{season} + \varepsilon$$

where $\mathbf{y}_i \in \mathbb{R}^{n \times 24}$ denotes hourly consumption over n days, and *season* is a continuous covariate.

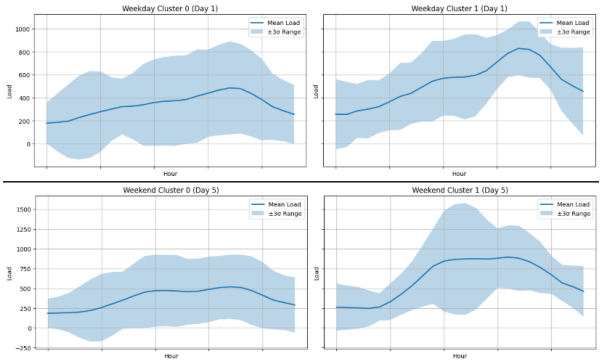


Figure 5: cluster wise load profile for v1

Predicted hourly profiles for each day d were computed as:

$$\hat{\mathbf{y}}_{i,d} = \mathbf{x}_{i,d}^{\top} \boldsymbol{\beta}_i, \quad \text{with } \mathbf{x}_{i,d} = [1, \text{season}_d]^{\top}$$

Cluster-level mean load profiles were then obtained by averaging across all households in each cluster $\mathcal{C}_c$:

$$\bar{y}_{c,d,h} = \frac{1}{|\mathcal{C}_c|} \sum_{i \in \mathcal{C}_c} \hat{y}_{i,d,h}$$

This yielded smoothed estimates of typical daily demand patterns per cluster.

To visualize variability across time, we also computed the standard deviation of predicted values across members of each cluster and plotted the mean profile with a $\pm 3\sigma$ band:

$$\sigma_{c,d,h} = \sqrt{\frac{1}{|\mathcal{C}_c|} \sum_{i \in \mathcal{C}_c} (\hat{y}_{i,d,h} - \bar{y}_{c,d,h})^2}$$

The band $\bar{y}_{c,d,h} \pm 3\sigma_{c,d,h}$ represents an empirical envelope capturing most of the within-cluster variation in predicted consumption patterns.

Figure 5 shows the cluster-wise average daily load profiles for a representative weekday and weekend under the v_1 smoothing. Cluster 1 consistently exhibits higher average consumption than Cluster 0, with more pronounced evening peaks—especially on weekends—indicating systematically higher energy use among Cluster 1 households.

3.6 kalman Filter

To jointly forecast electricity consumption across three households (#8, #11, and #34), we employ a multivariate Gaussian linear state-space model that incorporates household-specific observation noise and a shared latent dynamic structure. The model is formulated as:

$$\begin{aligned}
&\mathbf{x}_0 \sim \mathcal{N}_p(\mathbf{c}_0, \mathbf{C}_0), \\
&\mathbf{x}_t = \theta_x \cdot \mathbf{x}_{t-1} + \theta_v \cdot \mathbf{v}_t + \mathbf{w}_t, \quad \mathbf{w}_t \sim \mathcal{N}_p(\mathbf{0}, \mathbf{W}), \\
&\mathbf{z}_t = \theta_z \cdot \mathbf{x}_t + \mathbf{v}_t, \quad \mathbf{v}_t \sim \mathcal{N}_m(\mathbf{0}, \mathbf{V}),
\end{aligned}$$

where $\mathbf{x}_t \in \mathbb{R}^3$ denotes the latent state vector, $\mathbf{z}_t \in \mathbb{R}^3$ is the observed consumption vector, and $\mathbf{v}_t \in \mathbb{R}^3$ represents known exogenous inputs derived from cluster-level load profiles.

This model corresponds to the classical dynamic linear model (DLM) structure:

$$\mathbf{G} = \theta_x \cdot \mathbf{I}_3, \quad \mathbf{F} = \theta_z \cdot \mathbf{I}_3, \quad \mathbf{B} = \theta_v \cdot \mathbf{I}_3, \quad \mathbf{C}_0 = \mathbf{I}_3,$$

with household-specific noise covariances modeled as diagonal matrices:

$$\mathbf{W} = \mathrm{diag}(q_8, q_{11}, q_{34}), \quad \mathbf{V} = \mathrm{diag}(r_8, r_{11}, r_{34}).$$

This specification enables recursive estimation via the Kalman filter and supports personalized forecasting based on cluster-informed inputs.

The Kalman filter was applied to the first 50% of the available time-series data for each of the three households. Model performance was evaluated using a objective function defind in 2.The model parameters –including the state transition coefficient θ_x, the input effect θ_v, the observation scaling θ_z, and the noise variances specific to the home q_h, r_h –were optimized using a random search of more than 200 trials. The optimal parameter set corresponded to the lowest CV-RMSE in all folds and households.

Under the cluster-based load profile framework, the multivariate Kalman filter was independently applied to the *v1* and *v2* smoothed datasets. In the *v1* setting, all three test households (Households 8, 11, and 34) were consistently assigned to Cluster 0 for

both weekdays and weekends (see Table 3), resulting in a shared cluster-level load profile used as input, stratified by day type. In contrast, for *v2*, a uniform weekday profile was applied across all households. However, on weekends, Households 8 and 34 shared the same profile, while Household 8 uniquely adopted the Cluster 1 weekend profile, as reflected in the predictive assignment shown in Table 3.

In a complementary set of experiments, the Kalman filter was also calibrated and applied without exogenous load profile inputs. Across all configurations—both with and without load profiles— Table4 illustrates the optimal parameter for v1.Subsequent analysis was also performed for the `v2` smoothing configuration using the same procedure.

Table 4: Optimal Kalman Filter Parameters by Setting

Setting	Θ_X	Θ_v	θ_z	Q_{diag}	R_{diag}
Weekday	0.4970	0.4308	0.6257	[9.4448, 7.3696, 9.9191]	[25.9738, 35.0268, 21.8945]
Weekend	0.3550	0.5949	0.1580	[0.3831, 8.2278, 3.6083]	[6.4403, 26.1599, 38.5227]
No Load Profile	0.9890	0.8846	0.8312	[2.5687, 4.2573, 1.1747]	[32.4325, 6.2845, 13.1311]

Figure 9 demonstrates the forecasted power consumption curve based on weekday, weekend load profile, and without load profiles.

3.7 particle fiter

To complement the Kalman filtering framework, a particle filter was implemented to estimate latent consumption states under normality assumptions. The latent states evolved as

$$x_t = \theta_x x_{t-1} + \theta_v v_t + \varepsilon_t, \quad \varepsilon_t \sim \mathcal{N}(0, \sigma_q^2),$$

and observations followed

$$z_t = \theta_z x_t + \eta_t, \quad \eta_t \sim \mathcal{N}(0, \sigma_r^2).$$

In this study, the filter was initialized with $M = 1000$ particles sampled from $\mathcal{N}(0, \Sigma)$, where Σ was a diagonal matrix with entries equal to the empirical variance of each observed dimension. All particles were assigned uniform weights initially. The hyperparameters (θ_x, θ_v, θ_z, σ_q, σ_r) were optimized via random search over 100 trials by minimizing the RMSE between the filtered and observed series, following the same procedure as used for the Kalman filter. Subsequent analysis is also performed for the

`v2` smoothing configuration using the same procedure. table 5 illustrates the optimal parameters based for v1.

Table 5: Optimal Particle Filter Parameters by Setting

Setting	θ_x	θ_v	θ_z	σ_q	σ_r	Particles (M)
Weekday	0.9025	0.09248	0.95010	5.49896	37.8820	1000
Weekend	0.9025	0.09248	0.95012	5.49890	42.3258	1000
No Load Profile	0.9841	0.00000	1.36090	8.21000	30.1722	1000

Figure 10 demonstrates the forecasted power consumption curve based on weekday, weekend load profile, and without load profiles.

4 Comparative Forecasting Performance

Comparision between table 6 and table 7 demonstrates that in general `v2` smoothing consistently outperforms `v1` across both Kalman and particle filters. Incorporating cluster-based load profiles significantly improves forecast accuracy compared to the no-load-input scenario. Kalman filtering generally yields lower RMSEs than particle filtering under optimal conditions. The best overall performance is observed with `v2` smoothing and exogenous input, confirming the benefit of structured temporal information in short-term load forecasting.

Household	**Smoothing**	**Non-Weekend**	**Weekend**	**Without Load**
8	v1	241.30	309.68	473.00
	v2	172.03	207.37	447.26
11	v1	210.39	213.15	408.16
	v2	175.08	148.39	475.33
34	v1	161.77	879.43	916.22
	v2	161.29	120.10	914.31

Table 6: Kalman filter RMSEs for three households under different smoothing levels and input conditions.

Household	Smoothing	Weekend	Weekday	Without Load
8	v1	262.51	316.51	447.62
	v2	211.30	191.40	450.29
11	v1	229.33	215.33	409.40
	v2	233.34	205.23	410.21
34	v1	171.22	847.22	916.38
	v2	155.13	162.13	699.14

Table 7: Particle filter RMSEs for three households under different smoothing levels and input conditions.

5 Summary

This project examined short-term forecasting of household electricity consumption using Kalman and particle filters, applied to multivariate time series data from 33 households. Two smoothing strategies (v1 and v2) were employed to construct structured daily load profiles, and MANOVA with AIC-based model selection was used to capture key temporal patterns. Households were clustered based on estimated model parameters, enabling the use of representative load profiles as exogenous inputs in forecasting. Forecasts were generated for three test households under different configurations, and performance was assessed using RMSE. The results show that incorporating cluster-level inputs significantly improves forecast accuracy, and that v2 smoothing consistently outperforms v1. Kalman filtering generally produced more accurate results than the particle filter, highlighting the strength of linear state-space modeling when supported by meaningful exogenous information.

For future work, selecting the optimal model separately for v1 and v2 before performing forecasting may further enhance accuracy by aligning model complexity with the smoothing structure.

Bibliography

French, A., Macedo, M., Poulsen, J., Waterson, T., and Yu, A. (2008). Multivariate analysis of variance (manova). *San Francisco State University*.

Helwig, N. E. (2017). Multivariate linear regression. *University of Minnesota, http://users. stat. umn. edu/ helwig/notes/mvlr-Notes. pdf*.

Hyndman, R. J. (2011). Moving averages.

James, G., Witten, D., Hastie, T., and Tibshirani, R. (2013). *An Introduction to Statistical Learning*. Springer.

Januzaj, Y., Beqiri, E., and Luma, A. (2023). Determining the optimal number of clusters using silhouette score as a data mining technique. *International Journal of Online & Biomedical Engineering*, 19(4).

Karunasingha, D. S. K. (2022). Root mean square error or mean absolute error? use their ratio as well. *Information Sciences*, 585:609–629.

Miyamoto, S. (2022). *Theory of Agglomerative Hierarchical Clustering*. Behaviormetrics: Quantitative Approaches to Human Behavior. Springer Singapore, Singapore, 1 edition. 33 b/w illustrations, 2 illustrations in colour.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12:2825–2830.

Petris, G., Petrone, S., and Campagnoli, P. (2009). *Dynamic Linear Models with R*. Springer New York, New York, NY. Available from: ProQuest Ebook Central. [Accessed 16 June 2025].

Seabold, S. and Perktold, J. (2010). Statsmodels: Econometric and statistical modeling with python. In *Proceedings of the 9th Python in Science Conference*.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C., Polat, I., Feng, Y., Moore, E. W., VanderPlas, J., Laxalde, D.,

Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A., Ribeiro, A., Pedregosa, F., and van Mulbregt, P. (2020). Scipy 1.0: Fundamental algorithms for scientific computing in python. *Nature Methods*, 17:261–272.

Appendix

A Additional figures

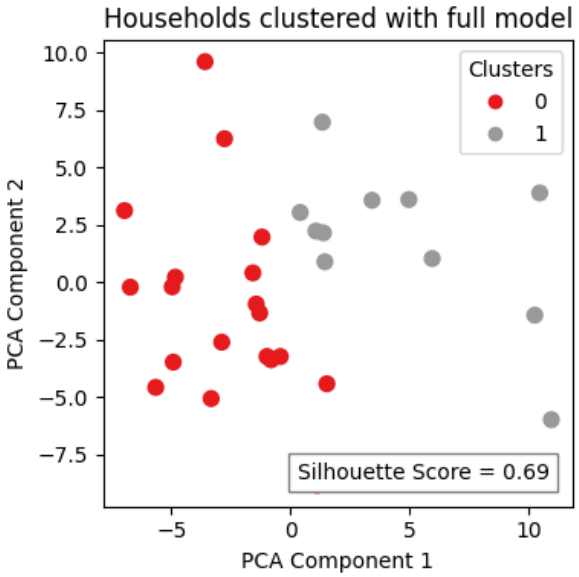


Figure 6: Households clustered with full model

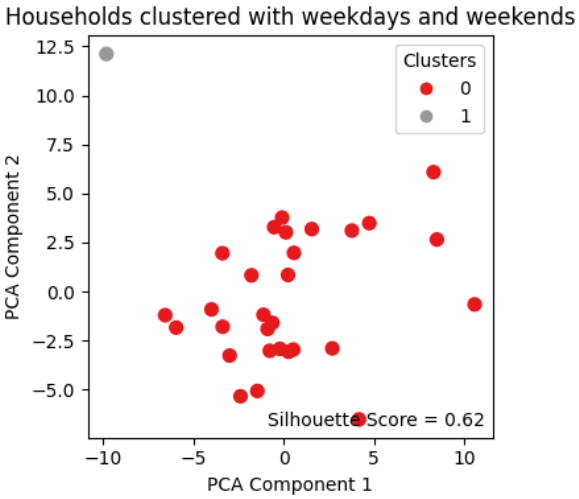


Figure 7: Households clustered with full model(v2)

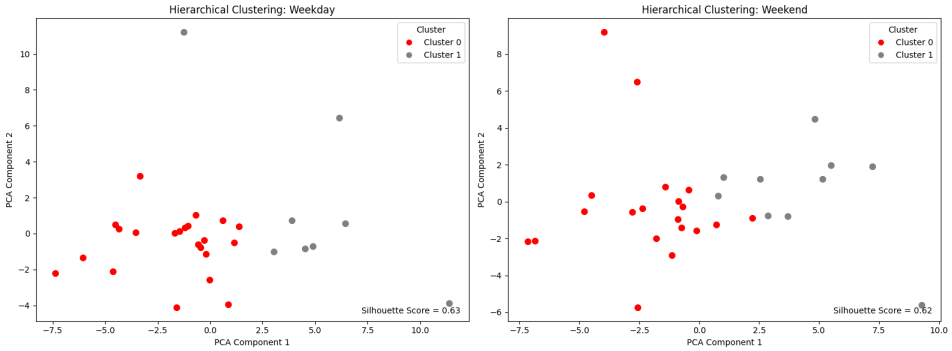
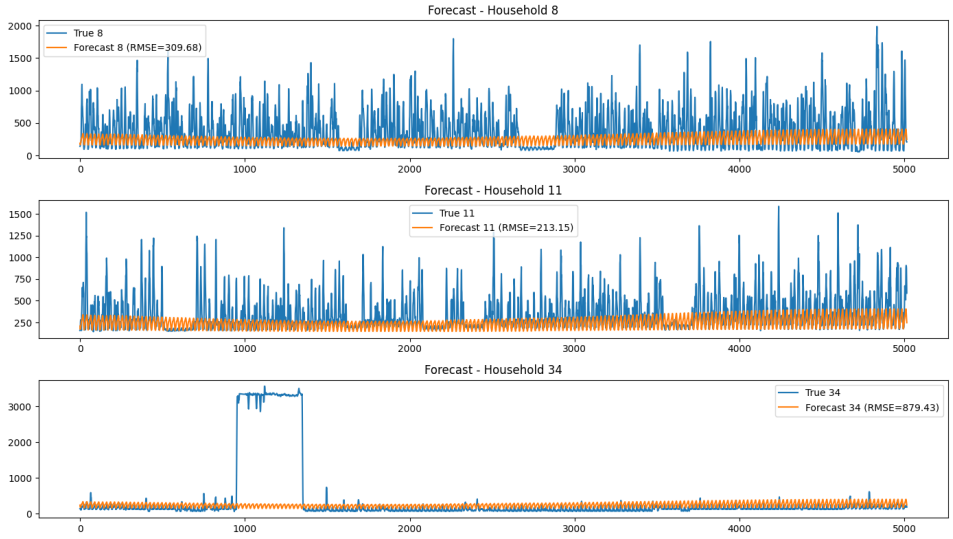
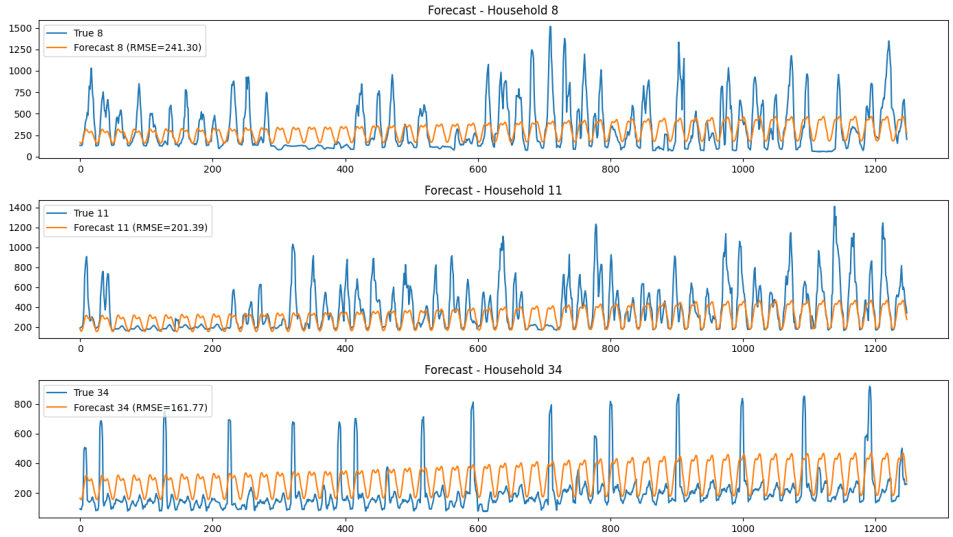


Figure 8: Clustering based on weekday/weekend-only subset for v2

(a) Forecast results for Households using the Kalman filter with weekday load profiles.



(b) Forecast results for Households using the Kalman filter with weekend load profiles.



(c) Forecast results for Households using the Kalman filter without load profile input.

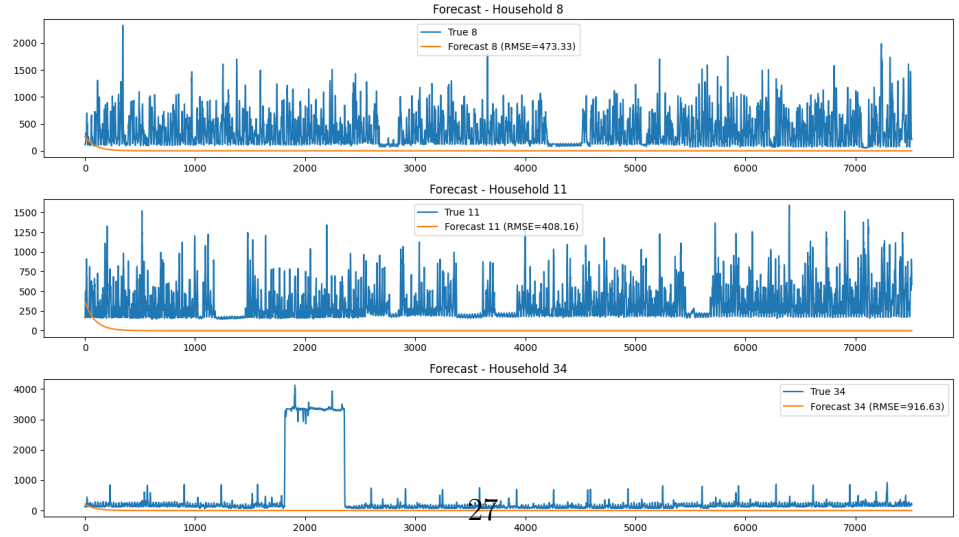
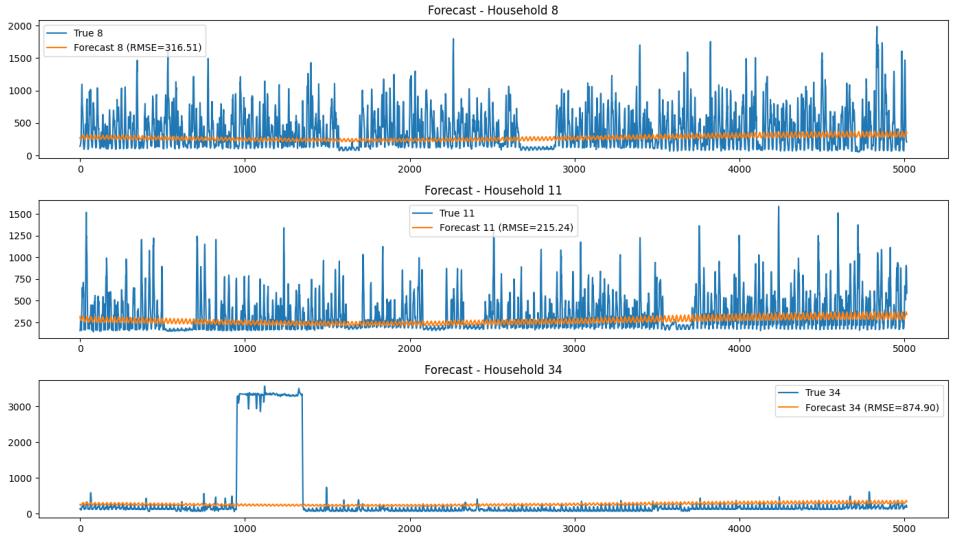
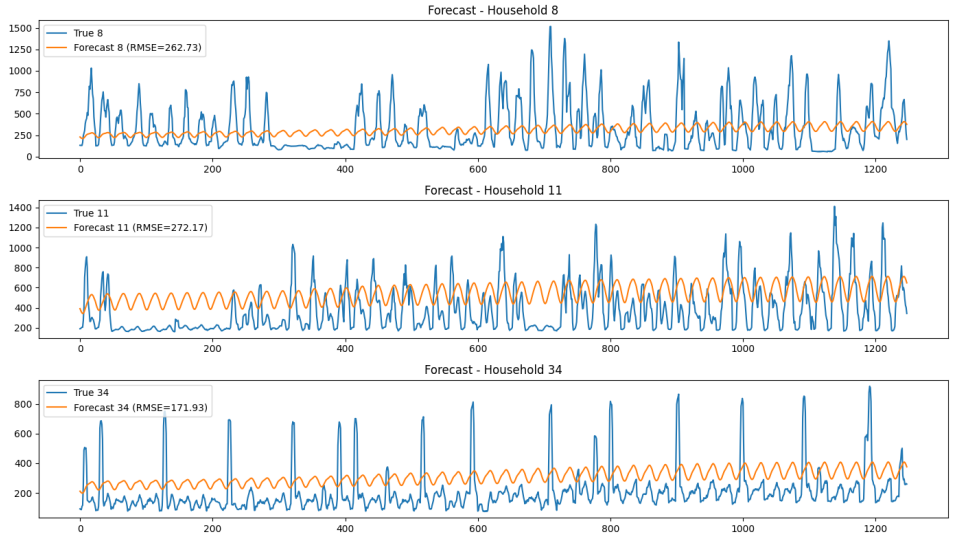


Figure 9: Kalman filter forecasts for v1-smoothed data under different exogenous input configurations. Each subplot shows the true and predicted load across three households.

(a) Forecast results for Households using the Particle filter with weekday load profiles.



(b) Forecast results for Households using the Particle filter with weekend load profiles.



(c) Forecast results for Households using the Particle filter without load profile input.

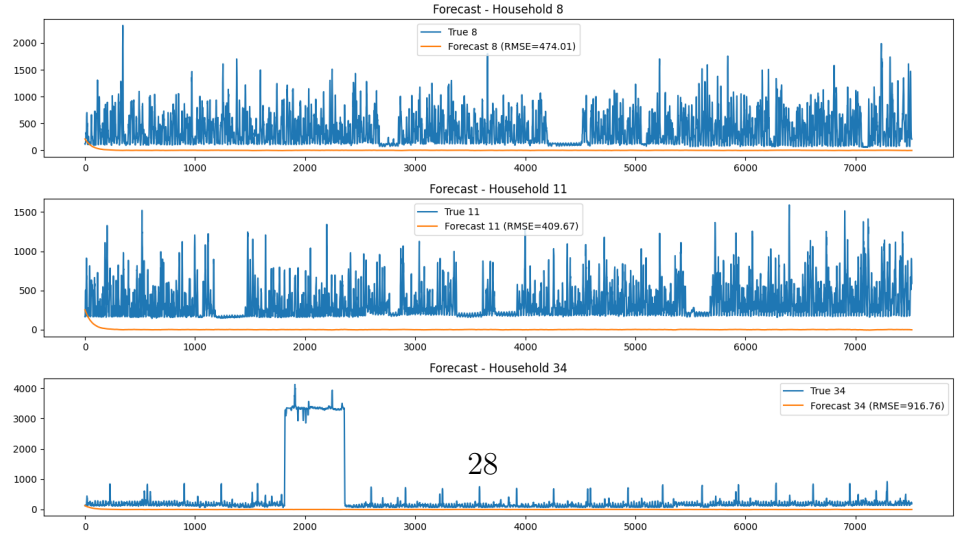


Figure 10: Particle filter forecasts for v1-smoothed data under different exogenous input configurations. Each subplot shows the true and predicted load across three households.

B Additional Tables

Table 8: MANOVA Results for v1 (Model with 7 weekday levels)

Effect	Wilks Lambda	Num DF	F Value	Pr > F
house_hold	0.014611	720	63.754813	0.0000
weekday	0.907454	144	6.582881	0.0000
house_hold:weekday	0.016001	4176	10.478571	0.0000
season	0.994254	24	2.330659	0.0002
house_hold:season	0.366172	696	14.366447	0.0000
weekday:season	0.949836	144	3.475082	0.0000
house_hold:weekday:season	0.066544	4176	6.659500	0.0000

Table 9: MANOVA Results for v1 (Model with weekday/weekend)

Effect	Wilks Lambda	Num DF	F Value	Pr > F
house_hold	0.000133	720	158.439719	0.0000
is_weekday	0.958094	24	18.186248	0.0000
house_hold:is_weekday	0.205745	696	23.680068	0.0000
season	0.970407	24	12.679906	0.0000
house_hold:season	0.070162	696	40.987774	0.0000
is_weekday:season	0.982526	24	7.394699	0.0000
house_hold:is_weekday:season	0.424629	696	12.577042	0.0000

Table 10: MANOVA Results for v2 (Model with 7 weekday levels)

Effect	Wilks' Lambda	Num DF	F Value	Pr > F
house_hold	0.014611	720	63.754813	0.0000
weekday	0.907454	144	6.582881	0.0000
house_hold:weekday	0.016001	4176	10.478571	0.0000
season	0.994254	24	2.330659	0.0002
house_hold:season	0.366172	696	14.366447	0.0000
weekday:season	0.949836	144	3.475082	0.0000
house_hold:weekday:season	0.066544	4176	6.659500	0.0000

Table 11: MANOVA Results for v2 (Model with weekday/weekend encoding)

Effect	**Wilks' Lambda**	**Num DF**	**F Value**	**Pr > F**
house_hold	0.000133	720	158.439719	0.0000
is_weekday	0.958094	24	18.186248	0.0000
house_hold:is_weekday	0.205745	696	23.680068	0.0000
season	0.970407	24	12.679906	0.0000
house_hold:season	0.070162	696	40.987774	0.0000
is_weekday:season	0.982526	24	7.394699	0.0000
house_hold:is_weekday:season	0.424629	696	12.577042	0.0000

Table 12: AIC values for four model variants by household (daily smoothing, v1)

Household	AIC_main7	AIC_interaction7	AIC_main2	AIC_interaction2	Use 2-level interact
3	83220.50	82472.44	83693.81	82275.26	Yes
4	85690.57	85012.73	86118.85	84767.74	Yes
5	89700.12	89236.71	90120.63	89089.25	Yes
6	19554.94	19539.27	19536.05	19389.59	Yes
7	89932.58	89373.65	90403.30	89182.25	Yes
9	100868.01	100344.43	101341.95	100209.40	Yes
10	95100.64	94275.08	95534.05	94075.99	Yes
12	91985.53	91302.59	92425.88	91079.62	Yes
14	84530.66	84493.85	84829.11	84286.20	Yes
16	88205.32	87667.85	88664.89	87509.71	Yes
17	59427.33	59090.19	59704.22	58941.68	Yes
18	89506.79	88871.90	89937.65	88874.46	No
19	87527.93	86572.61	88070.33	86440.43	Yes
20	90022.28	89633.87	90540.76	89523.54	Yes
21	82347.93	81320.21	82839.03	81158.82	Yes
22	90649.77	89698.76	91148.51	89550.51	Yes
23	86559.05	85389.80	87034.90	85159.73	No
25	70677.96	70329.26	71084.27	70268.60	Yes
27	64339.40	63422.84	64820.54	63227.41	Yes
28	87358.38	86477.64	87907.19	86413.51	Yes
29	78708.77	77947.73	79250.88	77814.58	Yes
30	83578.16	82916.61	84129.81	82783.96	Yes
31	79497.68	79014.14	79934.62	78908.02	Yes
32	84820.25	84083.79	85663.98	84449.87	Yes
35	97278.16	96613.64	97754.65	96479.82	Yes
36	80362.37	79788.39	80863.21	79669.60	Yes
37	35741.28	35714.35	35796.98	35438.36	Yes
38	88341.07	87413.22	88885.47	87293.40	Yes
39	82163.07	81619.90	82616.38	81396.69	Yes
40	66420.44	66000.72	66708.95	65830.30	Yes

Table 13: AIC values for four model variants by household (daily smoothing, v2)

Household	AIC_main7	AIC_interaction7	AIC_main2	AIC_interaction2	Use Interaction 7
3	94965.17	93031.97	95979.36	93955.52	True
4	97511.81	95776.28	98554.62	96604.87	True
5	102416.62	100858.89	103466.01	101815.79	True
6	21493.63	20510.84	22083.72	21645.53	True
7	102050.51	100555.40	103061.91	101424.36	True
9	113804.39	112044.75	114926.37	113178.06	True
10	108450.23	106347.17	109383.99	107247.86	True
12	101535.93	99904.90	102685.82	100860.46	True
14	96980.71	96019.89	97978.08	96900.05	True
16	100552.10	99125.81	101569.77	100075.38	True
17	66676.99	65141.46	67561.77	66246.32	True
18	102722.75	100753.08	103746.14	101980.15	True
19	100789.78	98391.42	102180.45	99852.18	True
20	101960.85	100508.36	103200.74	101484.79	True
21	94227.56	91937.78	95265.35	92966.72	True
22	102636.98	100371.44	103811.25	101522.03	True
23	99306.00	97282.95	100326.90	98161.64	True
25	80002.66	78430.26	81309.00	79880.84	True
27	73333.42	71333.92	74304.61	72220.30	True
28	101731.30	99412.49	102977.31	100705.35	True
29	91989.93	90302.35	93257.41	91510.77	True
30	95448.34	93705.30	96613.36	94658.48	True
31	91620.27	89976.56	92783.01	91183.64	True
32	98847.23	96627.35	100795.40	98960.68	True
35	109239.06	107136.12	110356.36	108326.01	True
36	92804.30	90998.74	93983.64	92213.18	True
37	40223.78	39376.33	40766.62	40068.79	True
38	100871.50	98760.46	102101.19	99927.95	True
39	94914.99	93344.84	96006.43	94173.48	True
40	75211.82	73815.80	76040.90	74617.88	True

Appendix: Code Structure and Workflow Overview

1. **Library Setup**
   Installed and imported necessary Python packages, including `pandas`, `numpy`, `matplotlib`, `statsmodels`, and `scikit-learn`.

2. **Data Loading and Merging**
   Loaded 33 individual household CSV files, extracted household identifiers, and concatenated them into a unified DataFrame.

3. **Datetime Parsing and Feature Engineering**
   Converted timestamps to datetime format and derived additional features such as day of year, hour, weekday, `is_weekday`, and a sinusoidal `season` variable.

4. **Data Aggregation**
   Aggregated 15-minute consumption data into hourly averages, grouped by household, day, and hour.

5. **Smoothing**
   Applied two smoothing strategies to generate structured load profiles:

   - `v1`: Rolling mean within each day.
   - `v2`: Rolling mean across same weekday-hour combinations.

6. **Wide-format Pivoting**
   Reshaped the smoothed data into wide-format daily profiles with 24 hourly columns for both `v1` and `v2`.

7. **Missing Value Handling**
   Evaluated missing values after smoothing and filled remaining gaps using linear interpolation.

8. **Multivariate Analysis (MANOVA)**
   Applied MANOVA to assess the effects of weekday, season, and household. Model selection was based on Akaike Information Criterion (AIC).

9. **Clustering and Classification**
   Performed KMeans clustering on estimated model parameters and used KNN for classifying test households.

10. **Kalman and Particle Filter Forecasting**
    Calibrated and applied both Kalman and particle filters to forecast household electricity consumption under multiple configurations.

11. **Model Evaluation**
    Compared forecasting accuracy across filters, smoothing levels, and input types using root mean squared error (RMSE).