Linköping University | Department of Computer and Information Science
Master's thesis, 30 ECTS | Statistics and Machine Learning
2021 | LIU-IDA/LITH-EX-A--2021/001--SE

Reduction of Power Supply Units Hardware Alarms in Radio Base Stations using Machine Learning

Agustín Valencia González

Supervisor : Sanjiv Dwivedi
Examiner : Oleg Sysoev

External supervisor : Oleg Gorbatov, Lackis Eleftheriadis

Linköpings universitet
SE–581 83 Linköping
+46 13 28 10 00 , www.liu.se

To Ana María.

Abstract

Radio Base Stations are one of the most important elements of mobile communication networks, and Power Supply Units are the components that feed them with energy. Thus, by transitivity, Power Supply Units are vital for radio networks. Any malfunctioning in them could be critical and need to be adequately addressed, which might imply high operational costs.

The current work aims to set the ground for improving the mechanisms of how the Radio Base Stations report their Power Supply Units hardware faults to the Network Operations Centre, so they are notified only when the operational continuity is at risk.

A state-of-the-art study has been made regarding power consumptions modelling and forecasting in Radio Base Stations and other domains, finding that *Prophet*, a Generalized Additive Model developed by Facebook, shows promising forecasting capabilities in power demand research in other domains.

At the first stage, time series analysis techniques are applied to preprocess the data to construct a database with uncorrupted information by estimating the process that produced them and then imputing them if there is missing data in any of the features by running a Kalman smoother. Later, these models are also used to forecast the future samples to set a baseline performance.

Secondly, it is estimated the future power consumptions using the *Prophet* model. It is analysed and explained how *Prophet* had been conceived to work, how it learns, and how to use it in an actual application through an example from data of an Radio Base Stations. Its performance is compared against the baselines.

Then, the power consumption is translated into Power Headroom terms, which is the feature of interest for the communications engineers. Then an initial alarm criterion is derived from the critical operational conditions definition.

Finally, the methods are thoroughly tested by running experiments for a set of Radio Base Stations to obtain a more substantial and general overview of the resulting performance of the proposed solution. Showing that *Prophet* achieves an R^2 score of 0.86 in the testing set for an hourly-long-range prediction of three days.

Acknowledgements

I want to thank my supervisors Oleg Gorbatov and Lackis Eleftheriadis from Ericsson Research, for their constant support, feedback and encouragement throughout this research work. I would like to also thanks Markus Andersson for trusting me and giving me the opportunity to work in such an amazing team.

I want to thank my examiner Oleg Sysoev and my opponent Namita Sharma, from Linköpings Universitetet, for their advice, constructive criticism and for showing me other perspectives from which to observe this research.

I am thankful to those who supported me and gave me the energy and strength needed throughout these two years: Sofia, Marcos, Martín, Josefina, Bayu, Ismail, José, Alba, Ying, Zuxiang and many more. You are, without a doubt, part of the journey that has taken me to this point.

Last but not least, I am very grateful to those who are no longer with us, who gave me life and raised me to become the man I am.

List of Abbreviations

3GPP	3rd Generation Partnership Project
AIC	Akaike Information Criterion
API	Application Programable Interface
CRITIC	Criteria Importance Through Intercriteria Correlation
EENNP	Evolutionary Ensemble Neural Network Pool
GAM	Generalized Additive Model
GRU	Gated Recurrent Unit
HMC	Hamiltonian Monte Carlo
LSTM	Long Short-Term Memory
MAE	Mean Absolut Error
MCAR	Missing Completely at Random
MCMC	Markov Chain Monte Carlo
MLP	Multi-Layer Perceptron
MNO	Mobile Network Operator
MOBE	Mean Out-of-Bounds Error
NOC	Network Operations Centre
NUTS	no-U-turn sampler
OBE	Out-of-Bounds Error
OSS	Operations Support System
PDU	Power Distribution Unit
PSU	Power Supply Unit
RBS	Radio Base Station
RF	Random Forest
RMSE	Root Mean Square Error
RU	Radio Unit
SVR	Support Vector Regression

Contents

List of Figures

List of Tables

1 Introduction

1.1 Motivation

1.1.1 Context

Mobile communications have shaped how people interact with each other in modern societies. Therefore, it is valid to state that mobile networks are fundamental building blocks for modern lifestyles. Usually, its infrastructure underlies silent and unnoticed. Nonetheless, when they stop working as expected, unpleasant situations, and even chaotic ones, can happen.

Usually, the most visual element of mobile networks for the common eye is the antenna towers. Although, there are several more hardware infrastructure and software that enable communications as we know them.

Figure 1.1: RBS infrastructure. All rights belong to Ericsson

A Radio Base Station (RBS) is the element of telecommunication networks that receives and broadcasts electromagnetic waves to the environment in a determined area and, therefore, one of the essential pieces to enable mobile communications.

Mobile networks are geographically distributed in units called *antenna site* considering multiple cells which comprise more than one cell –usually three–. Depending on its technology, it will vary its components and the architecture in which they are connected, as shown in Figure 1.2. Nonetheless, the main principles stand.

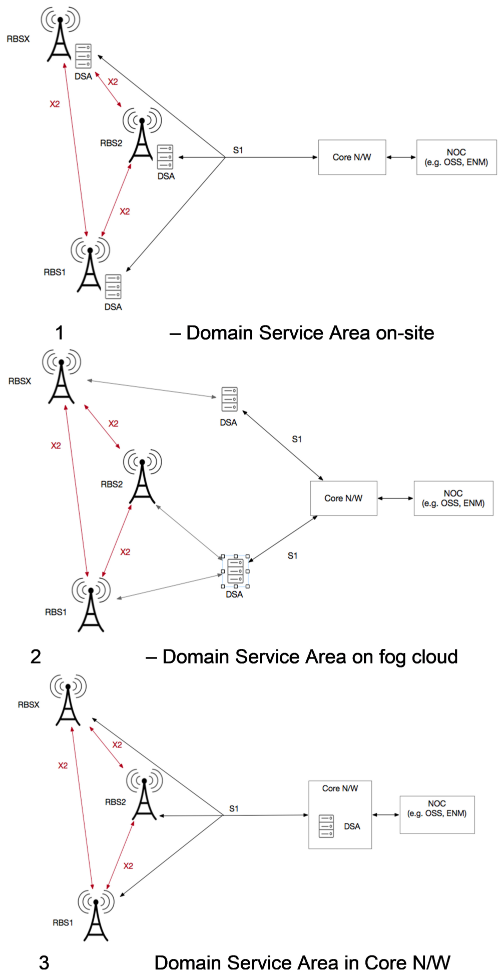


Figure 1.2: Different Radio Access Networks architectures

In present RBSs, in case of any hardware fault, an alarm is triggered and sent to the fault management system (reporting system) and informed to the Operations Support System (OSS) in order to be processed and handled by the Network Operations Centre (NOC). Engineers in the NOC should raise a request to send the field workforce to replace the faulty unit based on the alarm data.

The reality is that not all of the alarms affect the radio performance or degradation of the radio traffic, which implies that there is no urgency in sending service personnel to the RBS site until its operational continuity is at risk.

Setting the right time for sending field workers to the RBS to replace hardware, which sometimes might be located in very isolated or hard to reach locations, would positively affect the maintenance and operational costs for the Mobile Network Operators (MNOs).

As a Statistics and Machine Learning research, the current work does not aim to provide deep knowledge on telecommunications networks. Thus, some communications concepts will not be thoroughly developed, which does not imply any lack of statistical research rigorousness.

1.1.2 PSU alarms and power headroom

The Power Supply Unit (PSU) feeds with power the entire RBS. Nonetheless, usually, one station may contain several PSUs for operational robustness in case of faults, as shown in Figure 1.3. This redundancy implies that when an alarm is received from a PSU, sometimes, there is no need for an immediate replacement of the faulty PSU hardware since the RBS has still enough power available to continue its operation without blackout risk.

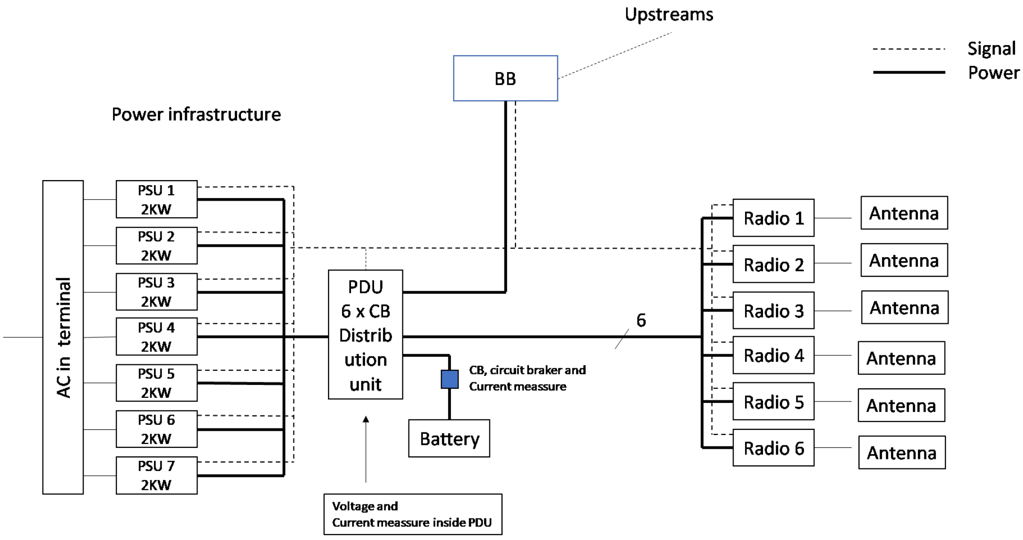


Figure 1.3: Example of a power infrastructure in an RBS[1]

Other components to note to understand better the features are the Power Distribution Unit (PDU) which is the unit that manages the power of the system and distributes it according to different operational scenarios. The Radio Units (RUs) are, as its name explains, the units in which the power is converted into electromagnetic waves modulated to carry information and then uses the antennas to broadcast them into the ambient. Last but not least are the power lines, which are not a unit themselves, but interconnects the units so the RBS itself could work as a system. The power lines are the veins that carry the energy to all the elements within the system. Nonetheless, they are not ideal, and power gets dissipated along them. The longer the lines are, the higher the losses related to them.

Added to the power surplus given by redundant PSUs, there is also the fact that they do not use all the available power all the time because the radio traffic has a clear seasonal component as shown in Figure 1.4. Thus, the RUs are not constantly transmitting at their maximum capacity.

[1]This is a general overview of the power infrastructure in an RBS, i.e., each station may differ from others depending on how the ad-hoc solution has been designed

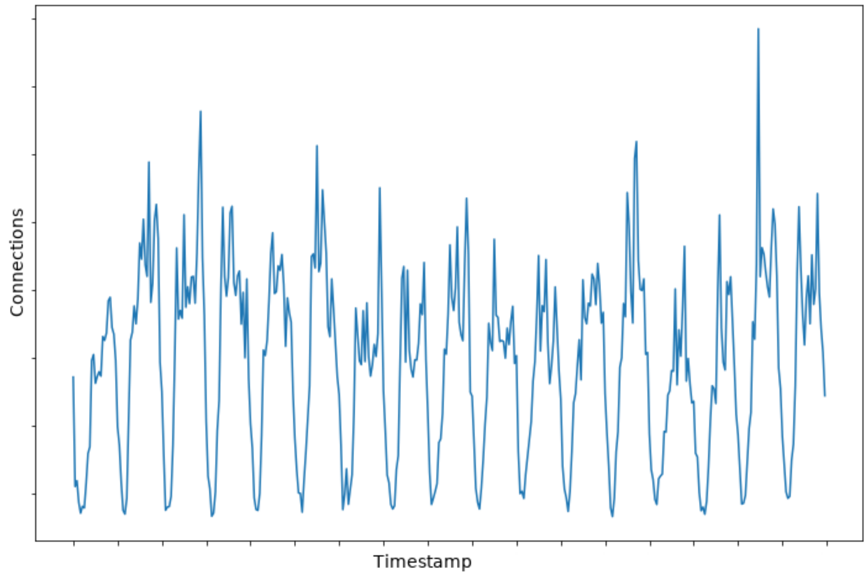


Figure 1.4: Example of variation of a number of connections in a RBS

The *power headroom* P_h, in a new approach for the PSU, then is defined as the difference between the power consumption P_{cons} and the maximum available power P_{max}.

$$P_h = P_{max} - P_{cons} \tag{1.1}$$

Where P_{cons} can be understood as *the consumed power seen from the PSU* [1, 2], which will comprise the RUs power consumption, static power consumption, cooling system consumption, the losses in the lines, etc. For simplicity, as this is not a communications thesis, this definition will not be developed in-depth.

In Figure 1.5 it is shown how the amount of PSUs is related to the power headroom and how a decrease in the amount of working PSUs would diminish the power headroom value but still give a reasonable operational margin.

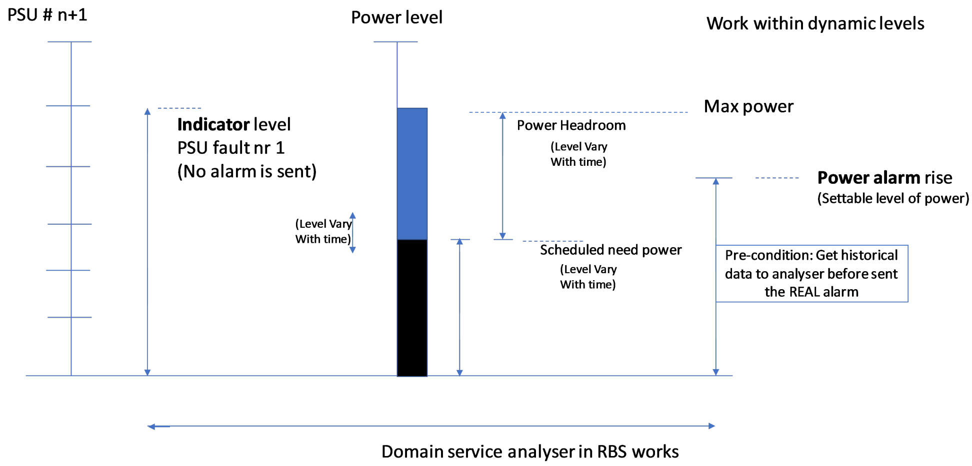


Figure 1.5: Power headroom example

1.2 State-of-the-art

The research on power consumptions in RBSs is not a new topic in the communications community, and they can be seen from two different research perspectives. The first stands from a modelling point of view. It can have a top-to-down approach [3], i.e., focusing on the consumptions first, and based on them, describe the system. Alternatively, there is a bottom-up perspective, in which the overall power consumption of the system is described by the consumptions of each independent building block depending on their different operational modes [4][2]. Both have in common that they are descriptive works and have constrained their data to pure hardware technologies and their power characteristics. Their primary purpose has been modelling and simulation of communication networks.

The second perspective uses more data than just hardware description. As a consequence, traffic metrics have been shown to be a fundamental covariate to explain power consumption fluctuations in RBSs [5]. Furthermore, STL decomposition has been applied to model the traffic load in RBSs. Then, Holt-Winter's technique [6] has been used to forecast its values, so the power management can adopt power-saving strategies when it is not needed to broadcast at maximum power capacity [7].

When it comes to power consumptions forecast in domains other than communications, the field has also been fruitful. Sensor networks power consumptions using Markov chains [8], power consumptions in data centres using LASSO, elastic-net, Random Forest (RF), KNN, and Multi-Layer Perceptron (MLP) [9], coal-mining power consumption forecast using pure statistics [10] or laser-melting processes using linear regression [11] to mention some.

Furthermore, households power demand forecast is a very fertile domain on its own. Linear regression, decision trees, RF, Support Vector Regression (SVR) and MLP have been applied to predict the demand in northern Morocco [12] finding that MLP performs better than the others, Long Short-Term Memory (LSTM) networks have been applied to learn household consumptions and compared against Gradient Boosting Tree (GBT) and SVR approaches finding that the LSTM model, which they have called *PowerLSTM* outperforms the others[13]. A novel Evolutionary Ensemble Neural Network Pool (EENNP) method is proposed in [14] and applied to power consumptions in western Norway.

The well-used Holt-Winters model has been compared against Facebook's Prophet model in long-range power loads forecast in Kuwait, reaching a MAPE ~ 2% for a prediction horizon of 720 days. Their predictions robustness has been tested by injecting Gaussian noise in different intensities at a fixed (unspecified) forecasting horizon, concluding that Prophet can perform robust power demand predictions while obtaining $R^2 = 0.96$ at a noise intensity of 80% [15]. In the same direction, it has been shown that Prophet performs better than ARIMA models for a long-range of 30 days on power demand prediction using external environmental regressors in an airport in Belgium [16]. In both studies, Prophet's weekly seasonal component is used to recommend the most suitable day of the week to perform maintenance related to power consumptions.

Furthermore, in [17], it has been shown the potential of Prophet for long-range predictions by extending it to work together with a Gated Recurrent Unit (GRU) as an attention layer plus a Criteria Importance Through Intercriteria Correlation (CRITIC) node to weigh both predictions optimally. For a prediction horizon of ~ 6 months, the proposed architecture shows a MAE = 8.5 against to 10.1 and 20.1 from Prophet on its own and ARIMA, respectively.

Thus, the robustness of the predictions made by Prophet models for long-range time horizons makes it an interesting alternative to be applied in communications research. Additionally, it has shown to perform better than standard methods in other fields, which are also highly related to human behaviours. No publication has been found doing so during the research.

[2] This study considers: power amplifier, analogue frontend, digital baseband, digital control and power system

1.3 Aim

The following work aims to develop a statistical or machine learning method that reports an alarm if and only if the power headroom in a RBS will reach unsafe operational levels based on installed power capacity and power consumption forecasts.

Based on state-of-the-art findings, exploring Prophet's capabilities to also endow of long-range robustness the proposed solution.

1.4 Research questions

1. What is the best way to handle real-world telecommunications and power time series to provide useful structures to mine and learn from them?
2. What are suitable forecast techniques to predict power utilisation in an RBS?
3. Given the power forecast, what are suitable criteria to raise alarms if and only if the RBS operational continuity is at risk?

1.5 Delimitations

Even though one of the overall goals of the present work has been reducing operational costs for MNOs, this has been done without considering the logistics under the maintenance or operational constraints of hardware replacement stock or even RBS geographical location. Therefore, the only factor of operational costs current work tries to optimise is *when* a hardware replacement alarm is raised.

2 Data analysis

2.1 Data acknowledgements

Due to data usage legal restrictions and Ericsson's customers' privacy, internal identifiers used by the company's operations, dates in all timestamps have been anonymised in the thesis to be available for the public domain. Moreover, the data structures and sensitive data values have not been presented in the thesis.

2.2 Data description

The available data is in the form of time series indexed by an RBS unique identifier and the time stamp corresponding to sample time. The data is sparse in different files depending on the domains from where the features have been measured, i.e., the power supply, power distribution, radio traffic, cabinet climate domains, etc. A brief explanation of the information provided by the time series and some random samples from the database are shown in the following subsections. It should be noted that configuration data for base stations is included. This data shows hardware equipment and their settings, for example, the number of PSUs.

2.2.1 Power supply

Power supply interruptions from the Power Grid distribution

This measurement shows how stable the electric power supply from the AC distribution is, i.e., it is a time counter in which the energy supply has been interrupted from the public power distribution. As a consequence, RBS utilise power from other sources such as batteries.

PSU utilisation statistics

PSU utilisation refers to the percentage of PSU's power capacity being consumed by the loads over a determined time. In the available data for this project, this is described by the average, minimum, maximum and standard deviation of these measures over different time durations.

These statistics are, in fact, the most crucial feature in the current work. It can be understood as the complement of the power headroom in per cent. In order to obtain the power values in terms of Watts units, it is needed to know beforehand the installed power capacity.

As there are no direct power headroom measurements, this has been chosen as the target variable, and then the power headroom will be derived from this quantity. In the Figure 2.1, the average of a PSU utilisation is shown as an example.

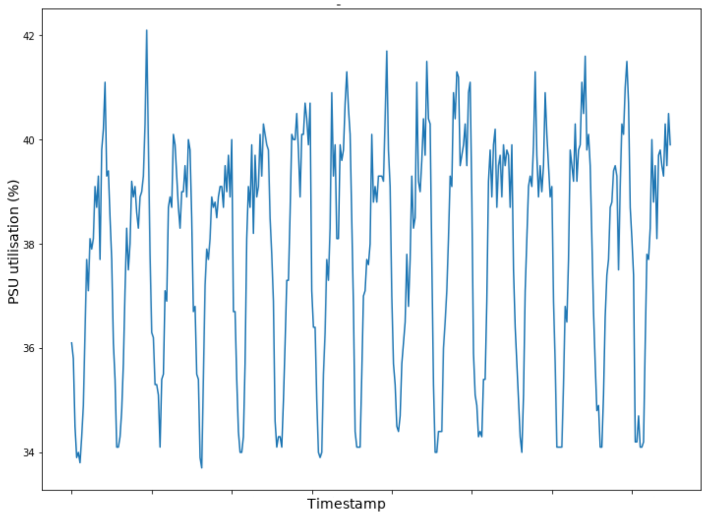


Figure 2.1: Average PSU utilisation example

2.2.2 Radio energy

Radio units voltage statistics

The RUs are fed with energy by the PDU. Nonetheless, this is not a constant and might change over time. The minimum, maximum, average and standard deviation statistics show how these fluctuations have been over a determined time. Figure 2.2 shows the standard deviation of the voltage perceived by the RUs.

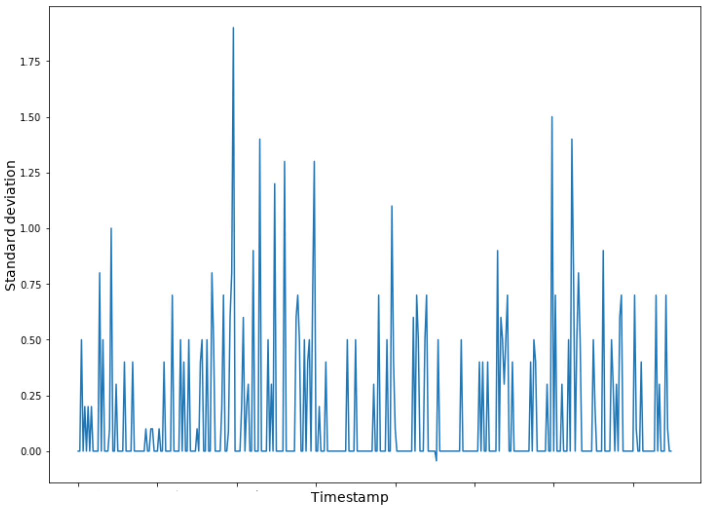


Figure 2.2: Standard deviation of RU voltage example

Radio units power consumption statistics

Supplied energy and power consumption are not the same. Whereas the former refers to the available energy to be used, the consumed power refers to the actual power used for broadcasting purposes. The minimum, maximum, average and standard deviation statistics have been measured and reported over a determined time. Figure 2.3 presents the average of the consumed power by the RU as an example.

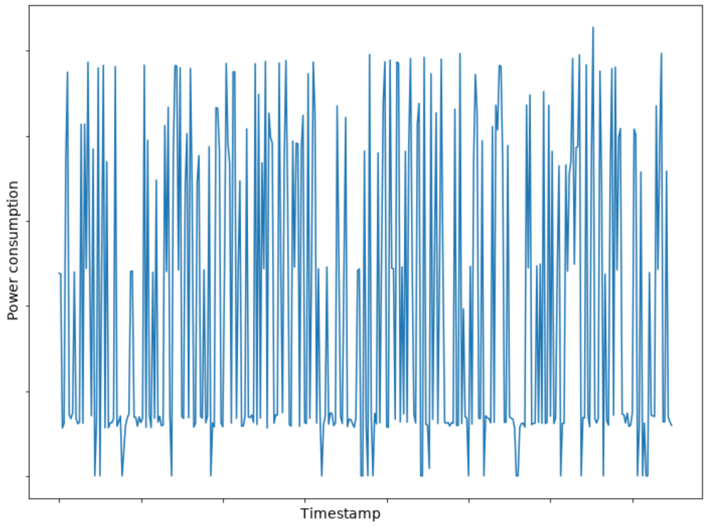


Figure 2.3: Average RU power consumption example

2.2.3 Power distribution

As shown in Figure 1.3, the RUs are not fed directly by the PSUs but by the PDU which manages the different power sources of the RBS. These measurements do not need to be the same as the voltage and power measurements in the RUs because the power lines that connect them could be up to 60 metres long, which might introduce some losses to the system.

Like other features, these are sampled and presented as their minimum, maximum, average and standard deviation over a determined time. Figure 2.4 shows the average of the PDU's output voltage as example.

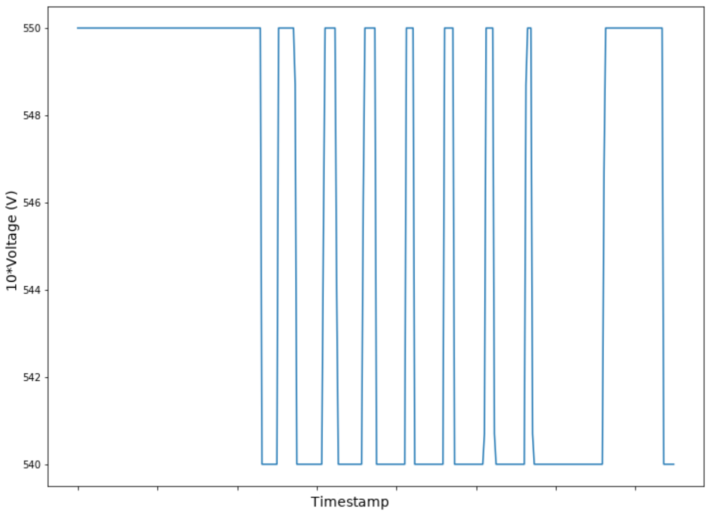


Figure 2.4: Average PDU voltage example

2.2.4 Radio traffic

In the data measured from radio traffic, it is possible to observe the highly seasonal characteristics of users behaviour influenced by the day-night cycle. Examples of data are shown in Figures 2.5. 2.6, and 2.7.

Number of connections

This measurement corresponds to the number of established connections to the radio traffic.

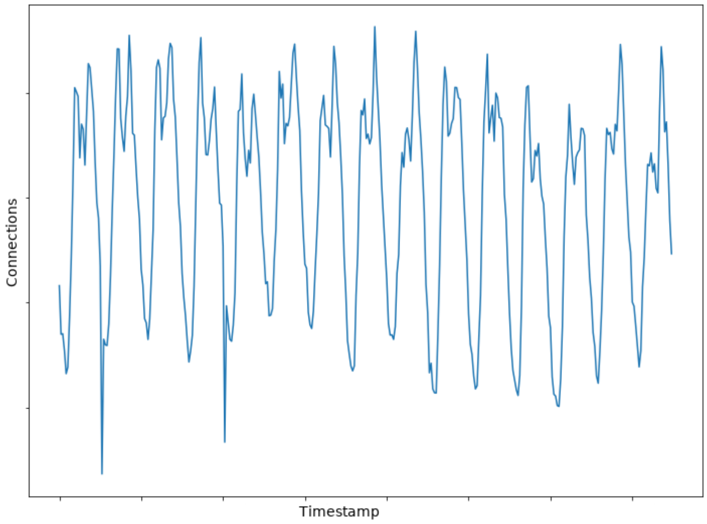


Figure 2.5: Connections requests signal example

Data blocks

This signal corresponds to the number of resource blocks connected to the traffic load in the uplink and downlink.

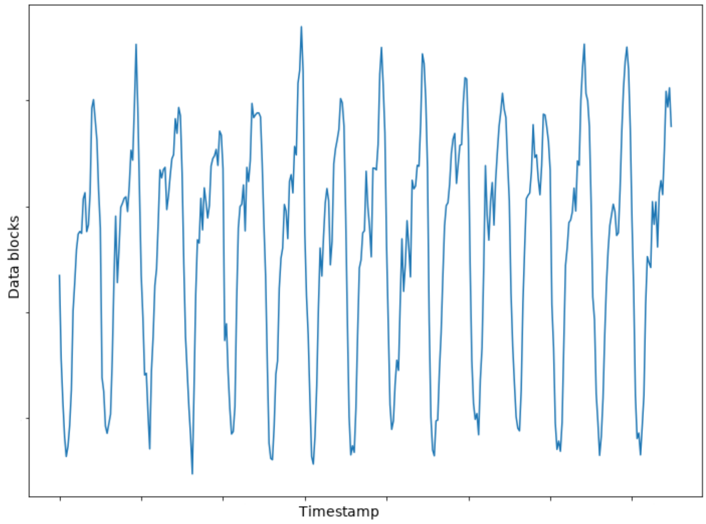


Figure 2.6: Data blocks signal example

Active Users

This feature shows the number of active users connected to the radio traffic load in the uplink and downlink.

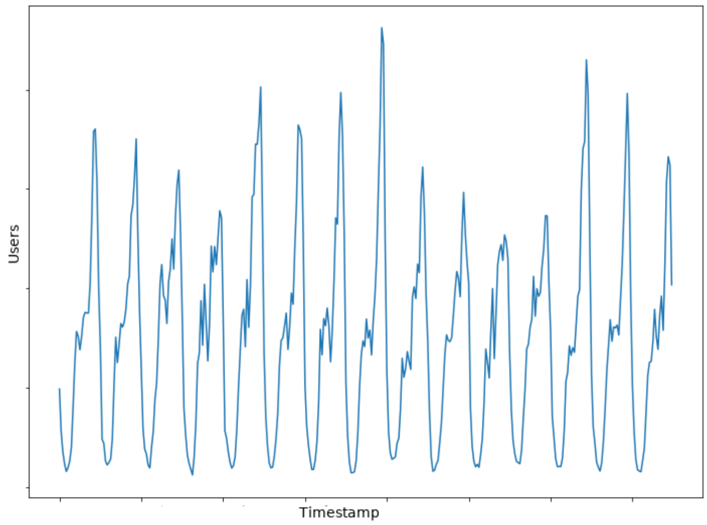


Figure 2.7: Active users signal example

2.2.5 Climate

Average cabinet temperature

In electronic components, high temperature is highly correlated with power dissipation; therefore, its increments in the hardware units will also increase the temperature within the cabinet. An example of average cabinet temperature time series is shown in Figure 2.8.

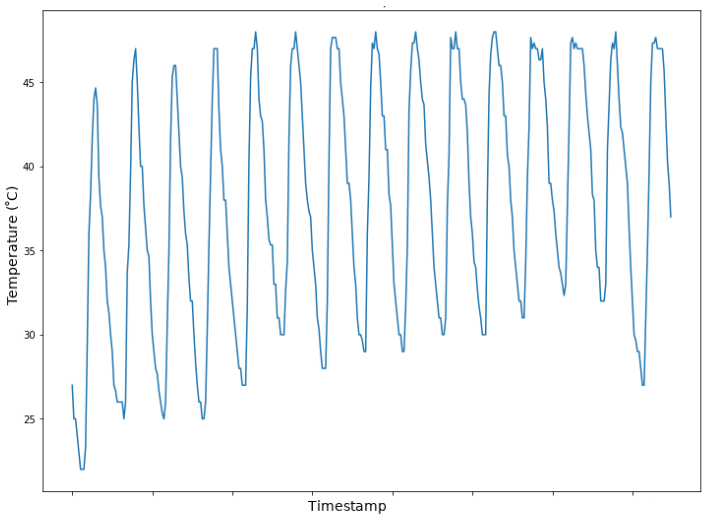
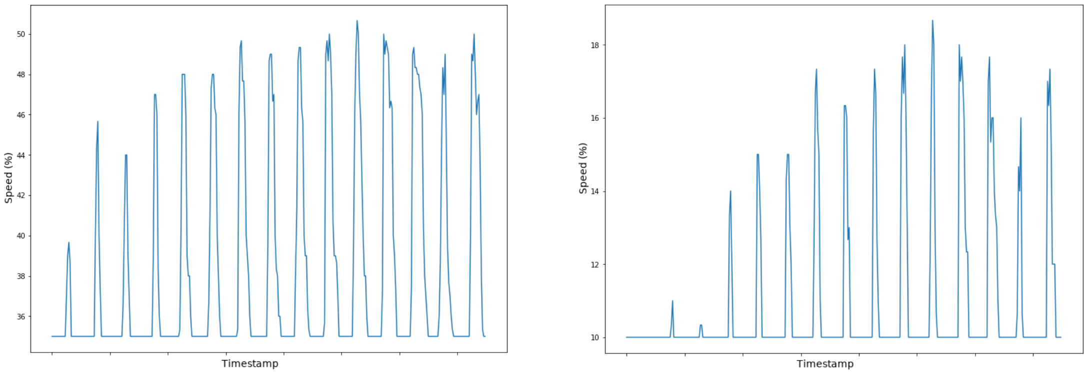


Figure 2.8: Cabinet temperature signal example

Internal and external fan speed

The cabinet is designed to keep the hardware safe. When the cabinet temperature is considered *high*, the fans, one internal and the other external, will be managed to cool down the hardware units. Therefore, the higher the temperature, the faster the fans will run. The reported values correspond to percentages of possible speed values, i.e., a zero value represents a steady fan, whereas a 100 means maximum velocity. Internal and external fan speed signals are shown in Figure 2.9a and 2.9b.



(a) Internal fan speed (b) External fan speed

Figure 2.9: Internal and external fan speed signals examples

3 Theory

3.1 Data imputation and database construction

In general, when measuring any variable, there is always a chance of collecting the data and missing some samples. Examples can be people not answering some questions in surveys or incomplete medical records, or even public entities failing to report data or choosing not to disclose it. Sampling data from sensors and reporting it via wireless communication channels is not invulnerable to this type of issues. There can be unforeseen technical issues such as a car crash damaging some units, power blackouts, or even a misconfiguration done by a negligent engineer.

Therefore, handling missing data is a problem that has been under research for a long time. Being Multiple Imputation [18], Expectation-Maximization [19], Nearest Neighbours [20] and hot-deck [21] the most popular techniques to deal with them.

In the following sections, it is going to be exposed how it is possible to obtain approximations of the processes that produce a given time series, and these approximations can be used to *fill the gaps* of missing data through imputation processes.

3.1.1 Notation

In the following sections, mathematical notations will be used from time series analysis. In order to stand on common ground, notations are defined and explained their meaning.

Backshift operator

The *backshift* (B) or *lag* (L) operators refer to the same operation and both can be found in the literature with no different meaning. In the current work it will , let us define 3.1.1

Definition 3.1.1 (Backshift operator)**.** *Let B denote a backshift or a lag operation in a given series $\{Y_t\} = \{Y_1, Y_2, \ldots, Y_n\}$ such that $t, n \in \mathbb{N}, t \leq n$ as*

$$BY_t = Y_{t-1}, \quad t > 1 \tag{3.1}$$

Property 1 (Inverse Backshift)*.* The inverse operation of a lag it is a forward step, thus

$$B^{-1}Y_t = Y_{t+1} \tag{3.2}$$

Property 2. The backshift exponentiation introduced in Property 1 can be generalised to denote back/forward shifts in multiple steps as

$$B^k Y_t = Y_{t-k} \tag{3.3}$$

Property 3 (Backshift algebra). B is a linear operator. Therefore the following is always true.

$$B(aX_t + bY_t + c) = aBX_t + bBY_t + c \tag{3.4}$$
$$B^j B^k Y_t = B^{j+k} Y_t \tag{3.5}$$

Property 4 (Backshift polynomials). The backshift operations can be treated as polynomials to explain more complex expressions.

Let $\{Y_t\} = \{Y_1, Y_2, \dots, Y_n\}$ be a random process and $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \dots, \alpha_n)$ such that.

$$\begin{aligned} \alpha_{t-1}Y_{t-1} + \alpha_{t-2}Y_{t-2} + \ldots + \alpha_{t-k}Y_{t-k} &= \alpha_{t-1}BY_t + \alpha_{t-2}B^2Y_t + \ldots + \alpha_{t-k}B^kY_t \\ &= (\alpha_{t-1}B + \alpha_{t-2}B^2 + \ldots + \alpha_{t-k}B^k)\{Y_t\} \\ &= \alpha(B)\{Y_t\} \end{aligned} \tag{3.6}$$

The reduction in (3.6) shows the power of the backshift operator, allowing to reduce such large expression into a short and elegant one. Nonetheless, a common flaw in this notation that could create confusion is that it is not needed to specify the polynomial order (k in the example). It can be assumed that whenever the order is not specified is then a general expression for any k-th order polynomial. Nonetheless, whenever its specification is needed, such as to specify an AR or MA model order, a subscript to the notation can be added: $\alpha_k(B)$ in this example.

Difference operator

It is also used the (backward) difference operator used in finite difference calculus to simplify long differences expressions.

Definition 3.1.2 (Difference operator)**.** *Let* ∇ *denote the difference operator in a given series* $\{Y_t\} = \{Y_1, Y_2, \dots, Y_n\}$ *such that* $t, n \in \mathbb{N}, t \leq n$ *as*

$$\nabla Y_t = Y_t - Y_{t-1}, \quad t > 1 \tag{3.7}$$

Property 5 (Difference and backshift operators relation). As an operation between lagged values, it is possible to define ∇ in terms of B

$$\begin{aligned} \nabla Y_t &= Y_t - Y_{t-1} \\ \nabla Y_t &= Y_t - BY_t \\ \nabla Y_t &= (1 - B)Y_t \end{aligned} \tag{3.8}$$

Property 6 (High order differences). Let us obtain the second-order difference as

$$\begin{aligned} \nabla(\nabla Y_t) &= \nabla Y_t - \nabla Y_{t-1} \\ \nabla^2 Y_t &= \nabla Y_t - \nabla BY_t \\ \nabla^2 Y_t &= (1 - B)\nabla Y_t \\ \nabla^2 Y_t &= (1 - B)(1 - B)Y_t \\ \nabla^2 Y_t &= (1 - B)^2 Y_t \end{aligned} \tag{3.9}$$

It can be shown that the development in (3.9) can be extended to the d-th order difference operator having that.

$$\nabla^d Y_t = (1 - B)^d Y_t \tag{3.10}$$

3.1.2 ARIMA models

It is defined that a time series $\{X_t\}$ can be modelled by an *integrated autoregressive moving average* (ARIMA) model if there exist a d-th difference $\{Y_t\} = \nabla^d\{X_t\}$ that can be modelled as an ARMA model [22].

Expanding the definition

$$\begin{aligned} \{Y_t\} &= \nabla^d\{X_t\}, \quad d > 0 \\ &= (1 - B)\nabla^{d-1}\{X_t\} \\ &\vdots \\ &= (1 - B)^{d-1}\nabla\{X_t\} \\ &= (1 - B)^d\{X_t\} \end{aligned} \tag{3.11}$$

Therefore, $(1-B)^d\{X_t\}$ will be explained by an ARMA(p, q) model if the following recursive equation has a causal and stationary solution:

$$Y_t - \phi_1 Y_{t-1} - \cdots - \phi_p Y_{t-p} = Z_t + \theta_1 Z_{t-1} + \cdots + \theta_q Z_{t-q}$$

Where $\{Z_t\}$ is an uncorrelated noise sequence and ϕ_i and θ_j are the autoregressive and moving-average weights, respectively. Written using backshift operator:

$$\phi(B)\{Y_t\} = \theta(B)Z_t \quad , \quad Z_t \sim \mathcal{N}(0, \sigma^2) \tag{3.12}$$

Replacing 3.11 in 3.12 the ARIMA(p, d, q) process that explains $\{X_t\}$ can be defined as:

$$\phi(B)(1 - B)^d\{X_t\} = \theta(B)Z_t \tag{3.13}$$

Where p is the p-order of the AR(p) process, q the q-order of the MA(q) process and d the successive differencing steps until obtaining a stationary ARMA(p, q) process.

3.1.3 SARIMA

It is said that a time series has a seasonality of period s if $Y_t = Y_{t-s}$. Written using the backshift operator this is $Y_t = B^s Y_t$.

Using the same logical development than for ARIMA derivation, it can be said that $\{Y_t\}$ follows a Seasonal ARIMA (SARIMA) process with period s [22] :

$$\{Y_t\} \sim \text{ARIMA}(p, d, q)(P, D, Q)_s$$

If and only if (3.14) Is a causal ARMA process defined by (3.15).

$$\{X_t\} = (1 - B)^d(1 - B^s)^D\{Y_t\} \tag{3.14}$$

$$\phi(B)\Phi(B^s)X_t = \theta(B)\Theta(B^s)Z_t, \quad \{Z_t\} \sim \mathcal{N}(0, \sigma^2) \tag{3.15}$$

Therefore, replacing (3.14) in (3.15), the SARIMA definition can be rewritten as

$$\phi(B)\Phi(B^s)(1-B)^d(1-B^s)^D\{Y_t\} = \theta(B)\Theta(B^s)Z_t, \quad \{Z_t\} \sim \mathcal{N}(0,\sigma^2)$$

Most of the literature obviate a constant c as part of the model since it does not add more insight about the concepts around these derivations. Nonetheless, as it will be seen in the following section it is considered as part of the auto-ARIMA estimation, therefore, the final ARIMA definition in current work is given by:

$$\phi(B)\Phi(B^s)(1-B)^d(1-B^s)^D\{Y_t\} = c + \theta(B)\Theta(B^s)Z_t, \quad \{Z_t\} \sim \mathcal{N}(0,\sigma^2) \tag{3.16}$$

Where p, d and q refer to the AR, integral and MA parameters of the ARIMA model, respectively. P, D and Q are the AR, integral and MA parameters of the seasonal construction for the period s.

3.1.4 State space representation

A state-space representation of a time series is suitable for cases in which the underlying nature of a process is hidden and cannot be determined. Nonetheless, indirect observations can be measured.

Thus, state-space representations are given by two equations. The *observation equation* (3.17) relates w-dimensional observable measures as a linear function of the v-dimensional noisy hidden state, and the *state equation* (3.18) determines how the hidden state will transition from time t to $t+1$.

$$\boldsymbol{Y}_t = G\boldsymbol{X}_t + \boldsymbol{W}_t \tag{3.17}$$

$$\boldsymbol{X}_{t+1} = F\boldsymbol{X}_t + \boldsymbol{V}_t \tag{3.18}$$

Where F is a $v \times v$ matrix, G a $w \times v$ matrix, $\{\boldsymbol{W}_t\} \sim \mathcal{N}(0,R)$, $\{\boldsymbol{V}_t\} \sim \mathcal{N}(0,Q)$ and $E(\boldsymbol{V}_s\boldsymbol{W}_t^T) = 0$ for all t, s.

3.1.5 Structural models

The classical structural models are defined in terms of trend (m), seasonal (s) and noise (ε) components (3.19). Although useful for some applications, they can be too deterministic for some others.

$$X_t = m_t + s_t + \varepsilon_t \tag{3.19}$$

State-space representations allow bringing more flexibility to these components. Therefore, it is natural to extend these concepts to a state-space domain.

3.1.5.1 Local level model

To show how the model is built up, it will be derived by adding component by component from the most straightforward model: the random walk. Let $\{Y_t\}$ be the observable variable from a state-space (3.20) in which the hidden state corresponds is a random variable (3.21), which in fact will determine the *local level model* [22].

$$Y_t = M_t + W_t, \quad W_t \sim \mathcal{N}(0,\sigma_w^2) \tag{3.20}$$

$$M_{t+1} = M_t + Vt, \quad V_t \sim \mathcal{N}(0,\sigma_v^2) \tag{3.21}$$

3.1.5.2 Local linear trend model

It is not difficult to extend this local level model to a *local linear trend model* by adding a slope state B_t.

$$M_t = M_{t-1} + B_{t-1} + V_{t-1} \tag{3.22}$$

Introducing randomness into the slope also:

$$B_t = B_{t-1} + U_t, \quad U_t \sim \mathcal{N}(0, \sigma_u^2) \tag{3.23}$$

As now this model contains multiple state, in order to write the model in state-space form, it can be defined the state vector:

$$\boldsymbol{X}_t = (M_t, B_t)^T \tag{3.24}$$

Thus, using (3.24), (3.22) and (3.23) can be rewritten as

$$\boldsymbol{X}_{t+1} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \boldsymbol{X}_{t+1} + \boldsymbol{V}_t, \quad t = 1, 2, \dots \tag{3.25}$$

Such that

$$\boldsymbol{V}_t = (V_t, U_t)^T \tag{3.26}$$

The observation equation for the process $\{Y_t\}$ is then by

$$Y_t = \begin{bmatrix} 1 & 0 \end{bmatrix} \boldsymbol{X}_t + W_t \tag{3.27}$$

Thus, from (3.25) and (3.27) the missing pieces to define the state-space equations are:

$$F = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \tag{3.28}$$

$$G = \begin{bmatrix} 1 & 0 \end{bmatrix} \tag{3.29}$$

$$Q = \begin{bmatrix} \sigma_v^2 & 0 \\ 0 & \sigma_u^2 \end{bmatrix} \tag{3.30}$$

$$R = \sigma_w^2 \tag{3.31}$$

3.1.5.3 Noisy seasonal model

Same as the classical structural models, the seasonal component s_t with period d has the properties [23]

$$s_{t+d} = s_t$$
$$\sum_{t=1}^{d} s_t = 0$$

Expanding it as in [22] it is obtained that:

$$s_{t+1} = -s_t - \dots - s_{t-d+2}, \quad t = 1, 2, \dots \tag{3.32}$$

From (3.32) it can be constructed a generalised sequence $\{Y_t\}$ by adding a random variable $S_t \sim \mathcal{N}(0, \sigma_s^2)$

$$Y_{t+1} = -Y_t - \dots - Y_{t-d+2} + S_t, \quad t = 1, 2, \dots \tag{3.33}$$

Now, to put it into a state-space form, it is defined the state vector:

$$\boldsymbol{X} = (Y_t, Y_{t-1}, \dots, Y_{t-d+2})^T \tag{3.34}$$

Therefore the observation equation for $\{Y_t\}$

$$Y_t = \begin{bmatrix}1 & 0 & 0 & \cdots & 0\end{bmatrix}\boldsymbol{X}, \quad t = 1, 2, \dots \tag{3.35}$$

$\{\boldsymbol{X}\}$ in the state equation

$$\boldsymbol{X}_{t+1} = F\boldsymbol{X}_t + \boldsymbol{V}_t, \quad t = 1, 2, \dots \tag{3.36}$$

$$\boldsymbol{V}_t = (S_t, 0, \dots, 0)^T \tag{3.37}$$

$$F = \begin{bmatrix} -1 & -1 & \cdots & -1 & -1 \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{bmatrix} \tag{3.38}$$

3.1.5.4 Structural time series general model

As in (3.19), to construct a general (additive) structural time series model, it is needed to sum each of the components, i.e., the general model is built as a result of merging the linear trend and the noisy seasonal models.

The state vector is then:

$$\boldsymbol{X}_t = \begin{bmatrix}\boldsymbol{X}_t^1 \\ \boldsymbol{X}_t^2\end{bmatrix} = \begin{bmatrix}(M_t, B_t)^T \\ (Y_t, Y_{t-1}, \dots, Y_{t-d+2})^T\end{bmatrix} = \begin{bmatrix}\begin{bmatrix}M_t \\ B_t\end{bmatrix} \\ \begin{bmatrix}Y_t \\ Y_{t-1} \\ \vdots \\ Y_{t-d+2}\end{bmatrix}\end{bmatrix} \tag{3.39}$$

The state equation

$$\boldsymbol{X}_{t+1} = \begin{bmatrix}F_1 & 0 \\ 0 & F2\end{bmatrix}\boldsymbol{X}_t + \begin{bmatrix}\boldsymbol{V}_t^1 \\ \boldsymbol{V}_t^2\end{bmatrix} \tag{3.40}$$

Where F_1 and F_2 are the matrices defined in (3.28) and (3.38) respectively. $\boldsymbol{V}_t^1$ and $\boldsymbol{V}_t^2$ are (3.26) and (3.37).

And the observations equation:

$$Y_t = \begin{bmatrix}1 & 0 & 1 & 0 & \cdots & 0\end{bmatrix}\boldsymbol{X}_t + W_t \tag{3.41}$$

3.1.6 Kalman prediction

Let $\boldsymbol{X} = (X_1, \dots, X_v)^T$ be a random vector. Then it can be defined

$$P_t(\boldsymbol{X}) \coloneqq (P_t(X_1), \dots, P_t(X_v))^T \tag{3.42}$$

Such that

$$P_t(X_i) \coloneqq P(X_i | Y_0, \dots, Y_t) \tag{3.43}$$

Is the best linear predictor of X_i in terms of all components of $Y_0, Y_1, \dots, Y_t$ [22]

Then the Kalman one-step predictor for a state-space model given by (3.17) and (3.18) is defined as

$$\hat{\boldsymbol{X}} \coloneqq P_{t-1}(\boldsymbol{X}_t) \tag{3.44}$$

And their error covariance matrices:

$$\Omega_t = E[(\boldsymbol{X}_t - \hat{\boldsymbol{X}}_t)(\boldsymbol{X}_t - \hat{\boldsymbol{X}}_t)^T] \tag{3.45}$$

The Kalman predictive recursions are then defined by
Initial conditions:

$$\begin{aligned} \hat{\boldsymbol{X}}_1 &= P(\boldsymbol{X}_1|\boldsymbol{Y}_0) \\ \Omega_1 &= E[(\boldsymbol{X}_1 - \hat{\boldsymbol{X}}_1)(\boldsymbol{X}_1 - \hat{\boldsymbol{X}}_1)^T] \end{aligned} \tag{3.46}$$

Recursions:

$$\begin{aligned} \hat{\boldsymbol{X}}_{t+1} &= F_t\hat{\boldsymbol{X}}_t + \Theta_t\Delta_t^{-1}(\hat{\boldsymbol{Y}}_t - G_t\hat{\boldsymbol{X}}_t) \\ \Omega_{t+1} &= F_t\Omega_t F_t^T + Q_t - \Theta_t\Delta_t^{-1}\Theta_t^T \end{aligned} \tag{3.47}$$

Where,

$$\begin{aligned} \Delta_t &= G_t\Omega_t G_t^T + R_t \\ \Theta_t &= F_t\Omega_t G_t^T \end{aligned} \tag{3.48}$$

3.1.7 Structural models estimation

Consider a vector $\boldsymbol{\theta}$ whose components can fully parametrise the state space given by (3.17) and (3.18).

Therefore, it is possible to find $\hat{\boldsymbol{\theta}}_{MLE}$ by maximising the observations in $\{Y_t\}$ with respect to the parameters in $\boldsymbol{\theta}$.

If the conditional probability density of $\boldsymbol{Y}_t|\boldsymbol{Y}_{t-1},\ldots,\boldsymbol{Y}_0$ is $f(\cdot|\boldsymbol{Y}_{t-1},\ldots,\boldsymbol{Y}_0)$, then the likelihood can be expressed as

$$\mathcal{L}(\boldsymbol{\theta};\boldsymbol{Y}_1,\ldots,\boldsymbol{Y}_n) = \prod_{t=1}^{n} f(\boldsymbol{Y}_t|\boldsymbol{Y}_{t-1},\ldots,\boldsymbol{Y}_0) \tag{3.49}$$

In general (3.49) is hard to solve. But, if it is assumed that $\boldsymbol{Y}_0$, $\boldsymbol{X}_1$ and $\boldsymbol{W}_t$, $\boldsymbol{V}_t, t = 1, 2, \ldots$ are *jointly Gaussian*, then the resulting conditional densities will have the form [22]

$$f(\cdot|\boldsymbol{Y}_{t-1},\ldots,\boldsymbol{Y}_0) = (2\pi)^{-w/2}\,(\det\Delta_t)^{-1/2}\exp\left\{-\frac{1}{2}\boldsymbol{I}_t^T\Delta_t^{-1}\boldsymbol{I}_t\right\} \tag{3.50}$$

Where

$$\boldsymbol{I}_t = \boldsymbol{Y}_t - P_{t-1}\boldsymbol{Y}_t = \boldsymbol{Y}_t - G\hat{\boldsymbol{X}}_t \tag{3.51}$$

And $P_{t-1}\boldsymbol{Y}_t$ and $\Delta_t, t \geq 1$ are obtained from the Kalman prediction recursions.

Finally, under the Gaussian assumptions, the likelihood can be rewritten as [22]

$$\mathcal{L}(\boldsymbol{\theta};\boldsymbol{Y}_1,\ldots,\boldsymbol{Y}_n) = (2\pi)^{-\frac{nw}{2}}\left(\prod_{j=1}^{n}\det\Delta_j\right)^{-\frac{1}{2}}\exp\left\{-\frac{1}{2}\sum_{j=1}^{n}\boldsymbol{I}_t^T\Delta_t^{-1}\boldsymbol{I}_t\right\} \tag{3.52}$$

Now, for any value of $\boldsymbol{\theta}$, the likelihood values $\mathcal{L}(\boldsymbol{\theta};\boldsymbol{Y}_1,\ldots,\boldsymbol{Y}_n)$ can be calculated using the help of Kalman recursions. Thus, in order to find $\hat{\boldsymbol{\theta}}_{MLE}$ it is needed to run some non-linear optimisation algorithm to find the best $\boldsymbol{\theta}$ by maximising $\mathcal{L}$[1].

[1] Constrained to the optimisation algorithm behaviour

3.1.8 Kalman smoothing

A time series smoothing consists in the estimation of the hidden states $\boldsymbol{X}_t$ given the complete time series $\boldsymbol{Y}_1, \ldots, \boldsymbol{Y}_n$ such that $t > n$ [24]. Which is especially suitable when there is any missing data point at a time $t < n$, this is:

$$\boldsymbol{X}_{t|n} = P_n \boldsymbol{X}_t \tag{3.53}$$

An the error covariance matrices [22]

$$\Omega_{t|n} = E\left[(\boldsymbol{X}_t - \boldsymbol{X}_{t|n})(\boldsymbol{X}_t - \boldsymbol{X}_{t|n})^T\right] \tag{3.54}$$

Then, the Kalman iterations for the smoothing problem are [22]

$$P_n \boldsymbol{X}_t = P_{n-1} \boldsymbol{X}_t + \Omega_{t,n} G_n^T \Delta_n^{-1} (\boldsymbol{Y}_n - G_n \hat{\boldsymbol{X}}_n) \tag{3.55}$$

$$\Omega_{t,n+1} = \Omega_{t,n} \left[F_n`\Theta_n \Delta_n^{-1} G_n\right]^T \tag{3.56}$$

$$\Omega_{t|n} = \Omega_{t|n-1} - \Omega_{t,n} G_n^T \Delta_n^{-1} G_n \Omega_{t,n}^T \tag{3.57}$$

Given the initial conditions in (3.58), which are obtained from the Kalman prediction.

$$\hat{\boldsymbol{X}}_t = P_{t-1} \boldsymbol{X}_t \tag{3.58}$$

$$\Omega_{t,t} = \Omega_{t|t-1} = \Omega_t \tag{3.59}$$

3.2 Prophet forecasting model

Prophet is a statistical model developed by engineers at Facebook, which mainly fits its models in Stan and has been open-sourced with public Application Programable Interfaces (APIs) for python and R languages [25]. It takes a regression-fitting approach to learn the time series and then forecasts by extrapolating such models and adding uncertainty in a Bayesian framework.

It uses a decomposable time series model [26] with three main components: trend, seasonality and holidays:

$$y(t) = g(t) + s(t) + h(t) + \sum_{i=1}^{N} x_i(t) + \varepsilon_t \tag{3.60}$$

Where $g(t)$ is a trend function that models non-periodic fluctuations, $s(t)$ capture different kinds of seasonalities, and $h(t)$ represents special situations that could add irregularities to the other components and a random error term ε.

The model is similar to a Generalized Additive Model (GAM) [27], which allows adding more and more components as needed. This is expressed in (3.60) by the sum of the N external regressors $x(t)$. The seasonalities are modelled by an exponential smoothing approach [28].

3.2.1 Trend model

Prophet has two trend models: a saturating growth one, which is similar to model population growths in ecosystems [29] and a piecewise model to give the flexibility of trend changes in determined learnt changepoints [25].

The basic saturated growth model is given by

$$g(t) = \frac{C}{1 + \exp\{-k(t - m)\}} \tag{3.61}$$

Where $\quad C$: Carrying capacity, $\quad k$: Growth rate, $\quad m$: Offset parameter

Nonetheless, the saturated growth model in (3.61) does not meet all the usual requirements for some of the applications in which the saturating ceiling varies over time or in which the growth rate is not constant. Therefore, the authors have extended the model as follows.

Let $s_j = 1, \ldots, S$ be the number of changepoints where the growth rate is allowed to change. Then, let δ_j be the change rate that happens at s_j, and $\boldsymbol{\delta} \in \mathbb{R}^s$ the vector defined by all δ_j. The growth rate at any time t is then the base rate k plus all the adjustments up to t.

$$k + \sum_{j=1}^{t<s_j} \delta_j \tag{3.62}$$

Expression (3.62) can be rewritten as:

$$\begin{gathered} k + \boldsymbol{a}(t)^T \boldsymbol{\delta} \\ \text{Where} \quad a_j = \begin{cases} 1 & , \quad \text{if} \quad t > s_j \\ 0 & , \quad \text{otherwise} \end{cases} \end{gathered} \tag{3.63}$$

Also, when the growth rate is adjusted, the m needs to be adjusted to ensure continuity.

$$\gamma_j = \left(s_j - m - \sum_{l<j} \gamma_l \right) \left(1 - \frac{k + \sum_{l<j} \delta_l}{k + \sum_{l\leq j} \delta_l} \right) \tag{3.64}$$

Then the linear trend with changepoints is defined by (3.65) piecewise logistic growth model by (3.66) [25].

$$g(t) = \left(k + \boldsymbol{a}(t)^T \boldsymbol{\delta}\right) t + \left(m + \boldsymbol{a}(t)^T \boldsymbol{\gamma}\right) \tag{3.65}$$

$$g(t) = \frac{C(t)}{1 + \exp\left\{-(k + \boldsymbol{a}(t)^T \boldsymbol{\delta})(t - (m + \boldsymbol{a}(t)^T \boldsymbol{\gamma}))\right\}} \tag{3.66}$$

3.2.2 Trend forecast

After the model has learnt from a history of T time points with S changepoints from which each point has a rate change $\delta_j \sim \text{Laplace}(0, \tau)$.

The trend will keep its last growth rate constant. The forecasts will be made by extrapolating the GAM and simulating samples from $\text{Laplace}(0, \lambda)$ where λ is a variance inferred from the data from the maximum likelihood estimate of the rate scale parameter (3.67).

$$\lambda = \frac{1}{2} \sum_{j=1}^{S} |\delta_j| \tag{3.67}$$

Once λ has been inferred, the trend forecast and its uncertainty are obtained from

$$\forall j > T \quad , \quad \begin{cases} \delta_j = 0 \quad \text{w.p.} \frac{T-S}{S} \\ \delta_j \sim \text{Laplace}(0, \lambda) \quad \text{w.p.} \frac{S}{T} \end{cases} \tag{3.68}$$

3.2.3 Seasonalities

By using the GAM flexibility, it is possible to add different seasonality periods. 365.25 or 7, for yearly and weekly seasonalities, respectively, –in days measurements– for example. These seasonalities are modelled by the use of Fourier series [30].

Therefore, for every given period P, it can be defined that.

$$s(t) = \sum_{n=1}^{N} \left(a_n \cos\left(\frac{2\pi nt}{P}\right) + b_n \sin\left(\frac{2\pi nt}{P}\right) \right) \tag{3.69}$$

Which has $2N$ params to learn:

$$\boldsymbol{\beta} = [a_1, b_1, a_2, b_2, \ldots, a_N, b_N]^T$$

In order to make the structures more manageable, it is defined a matrix of seasonalities comprised of the seasonal vectors for each time step.

$$\boldsymbol{X}(t) = \left[\cos\left(\frac{2\pi(1)t}{P}\right), \sin\left(\frac{2\pi(1)t}{P}\right), \ldots, \cos\left(\frac{2\pi(N)t}{P}\right), \sin\left(\frac{2\pi(N)t}{P}\right) \right] \tag{3.70}$$

Thus, the seasonal component is expressed then as.

$$s(t) = \boldsymbol{X}(t)\boldsymbol{\beta} \tag{3.71}$$

$$\text{Where} \quad \beta \sim \mathcal{N}(0, \sigma^2) \tag{3.72}$$

3.2.4 Holidays and festivities

Prophet also allows considering non-seasonal events that can significantly affect the forecasts due to changes in peoples behaviour. Namely, Muslim Ramadan, Chinese New Year or the US's Super Bowl.

It is assumed that each holiday is independent. For each holiday i, D_i is the set of past and future dates of it.

Like seasonalities, a regressors matrix is generated indicating whether the time t happens during holiday i and a parameter κ_i to model its influence in the forecast.

$$Z(t) = [\mathbf{1}(t \in D_1), \ldots, \mathbf{1}(t \in D_L)] \tag{3.73}$$

$$\therefore \quad h(t) = Z(t)\boldsymbol{\kappa} \tag{3.74}$$

$$\boldsymbol{\kappa} \sim \mathcal{N}(0, \nu^2) \tag{3.75}$$

For practical cases, importing the holidays' dates can be done using the python-holidays open-source package [31]. It is possible also to include other parameters to extend the holiday component to the neighbouring days. For more details, the reader can further investigate in [25].

3.2.5 Hamiltonian Monte Carlo and Prophet fitting

As mentioned, Prophet's core has been implemented in Stan [32]. As a consequence, the R and python APIs work only as intermediaries between the application code and Stan's model definition.

Stan, as a Bayesian framework, uses Markov Chain Monte Carlo (MCMC) algorithms to sample from the defined distributions: Hamiltonian Monte Carlo (HMC) [33] and its adaptive variation no-U-turn sampler (NUTS) [34]. These approaches resemble the Hamiltonian mechanics, which describes the evolution of a system over time, its position q and momentum p, as the partial derivatives of the Hamiltonian function $\mathcal{H}(q,p)$ with respect to the other parameter, i.e., the derivative of the Hamiltonian with respect to the position results in the derivative of the momentum with respect to the time and the same in the opposite sense as defined in (3.76).

$$-\frac{\partial \mathcal{H}}{\partial q} = \frac{dp_i}{dt} \tag{3.76}$$

$$\frac{\partial \mathcal{H}}{\partial p} = \frac{dq_i}{dt} \tag{3.77}$$

Where

$$\mathcal{H}(q, p) = U(q) + K(p) \tag{3.78}$$

Being $U(q)$ the potential energy which is defined to be the negative log-probability density of the distribution for q desired to be sampled (sometimes denoted as θ in Bayesian literature), and $K(p)$ the negative log-probability density of multi-normal $\mathcal{N}(0, \Sigma)$ [35].

The HMC iterations consist of two steps. In the first step, samples for the momentum variables are drawn from their multi-normal distribution. In the second step, it is used Leapfrog numerical integration [36] to solve the Hamiltonian equations to run a Metropolis-Hastings accept/reject step later to propose a new state of the system.

It should be noted that for maximum a posteriori estimation, it is used Limited-memory-Broyden–Fletcher–Goldfarb–Shanno algorithm (L-BFGS) optimisation method [37]. L-BFGS is an iterative method for solving unconstrained nonlinear optimisation problems using a limited amount of computer memory.

4 Methods

4.1 Overall pipeline architecture

By joining all the parts exposed in the previous chapters, the resulting pipeline proposed as a solution is shown in Figure 4.1. Based on the flow presented, the obtained results are summarised in the following sections.

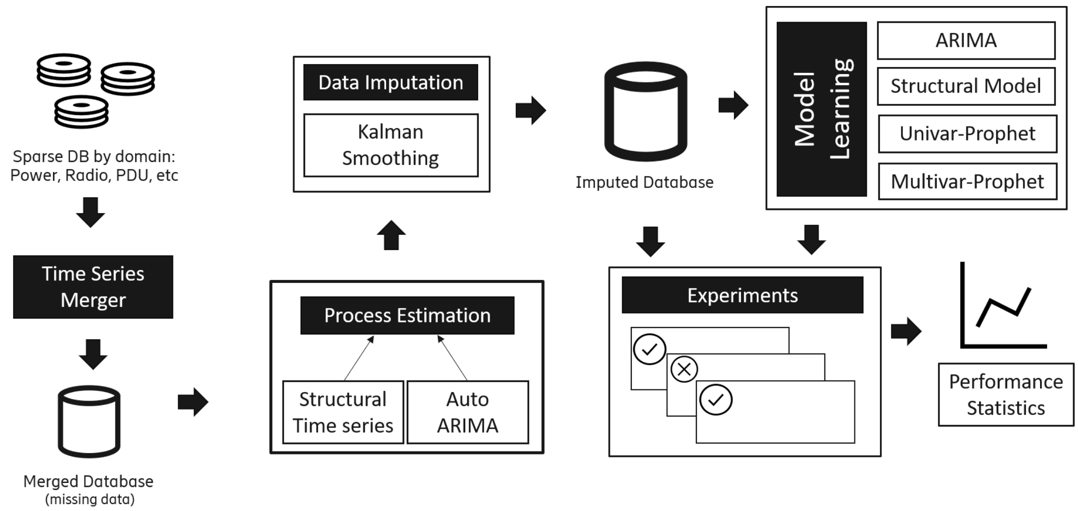


Figure 4.1: Overall pipeline architecture

4.2 Database building and time series merging

As mentioned, the data is sparse across several files depending on its domain. Therefore, all the data has been merged into one consolidated database to make it more manageable to handle. Then, there is the problem of determining the best approach to join all these data without corrupting its integrity as a valid time series; this means having an invariant sampling time with fully defined observations vectors.

Among the data tables, it can be seen that the two fields that are always present, and therefore are suitable to use as keys to index all the different tables and join them, are the timestamp of the sample and the RBS that has produced it. Therefore, the tuples $(\mathrm{date}_i, \mathrm{rbs}_i)$ will be used as keys. This means, having two relations $R(x_1, \ldots, x_n)$ and $S(y_1, \ldots, y_m)$ a third relation could be constructed $Q(z_1, \ldots, z_k) = R \times S$ such that $R(\vec{x}_i) = S(\vec{y}_i)$ be the keys [38].

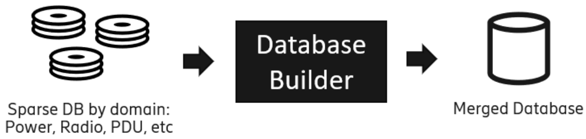


Figure 4.2: Database consolidation

In an ideal situation, all the time series for a given RBS would have no missing data and identical timestamps, so just intersecting them would result in a complete-time series with equal sampling time, and no information would be lost. Nonetheless, that is not the current case. As real-world data, technical issues happen now and then. There could be missing samples, or the timestamps between different files are not equidistant and, therefore, a join using them will not perform as expected.

The present work has been implemented mainly in Python with a few exceptions in R. The joining part is not an exception; thus, pandas `merge` options are available to use: *left, right, outer, inner* or *cross*. Merge operations in pandas work on two sets [39]. Thus, for joining multiple series, successive two-sided `merge` operations have to be performed.

4.2.1 Strategy analysis: Inner join

An inner join constitutes an intersection between both sets. The implementation in pandas allows indicating the keys to be intersected, and as a result, a tuple is returned containing the matched keys with the remaining members. This can be expressed as the following

$$Q(date_i, rbs_i, x_{i_1}, \ldots, x_{i_n}, y_{i_1}, \ldots, y_{i_m}) =$$
$$R(date_i, rbs_i, x_{i_1}, \ldots, x_{i_n}) \bigcap S(date_i, rbs_i, y_{i_1}, \ldots, y_{i_m})$$

Furthermore, in a more programming-friendly manner, it can also be expressed as a SQL query. The expression above would be as the following and graphically as a 3D Venn diagram as in Figure 4.3.

```sql
SELECT *
FROM R
INNER JOIN S
ON R.date = S.date AND R.rbs = S.rbs;
```

The most significant disadvantage of this strategy relies on losing the observations that are not indexed in both datasets. Such nuisance will imply losing the series' head and tails if both series do not have the same start and finish times. If the missing samples are within the series, it will corrupt the constant sampling constraint. Given the amount of data available, these are situations to be avoided.

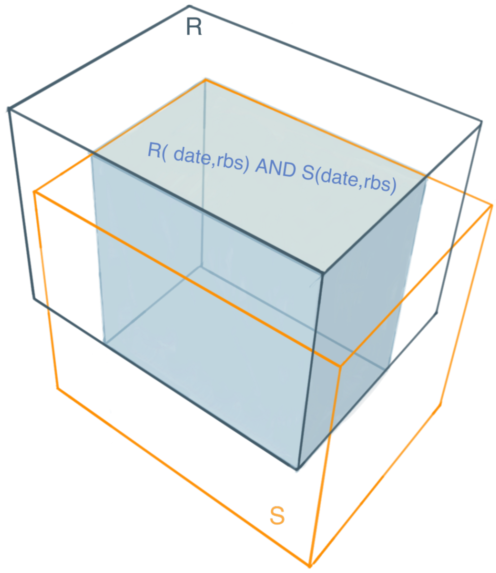


Figure 4.3: Inner join

4.2.2 Strategy analysis: Outer join approach

Whereas an inner join implies an intersection, an outer join can be understood as a union. In pandas, the union implementation is performed in the keys domain, and the rest of the vector is preserved. If there is some missing value in terms of samples or the full covariates, `NaN` will be used to fill in.

Given the relations R and S from Section 4.2:

$$R(date_i, rbs_i, x_{i_1}, \ldots, x_{i_n}), S(date_i, rbs_i, y_{i_1}, \ldots, y_{i_m})$$

The resulting *outer-joined* relation Q on keys $(\mathrm{date}_i, \mathrm{rbs}_i)$

$$Q(date_i, rbs_i) = R(date_i, rbs_i) \bigcup S(date_i, rbs_i)$$

Such that the missing values on each of the relations are declared as NaN

$$Q(x_i) = \mathrm{NaN}, \forall R(y_i) : (date_i, rbs_i) \in S \land (date_i, rbs_i) \notin S$$
$$Q(y_i) = \mathrm{NaN}, \forall S(x_i) : (date_i, rbs_i) \in R \land (date_i, rbs_i) \notin R$$

Again, in a more programming-friendly fashion, the SQL query would look like the following and a more explanatory Venn diagram as in 4.4.

```
SELECT *
FROM R
FULL OUTER JOIN S
ON R.date = S.date AND R.rbs = S.rbs;
```

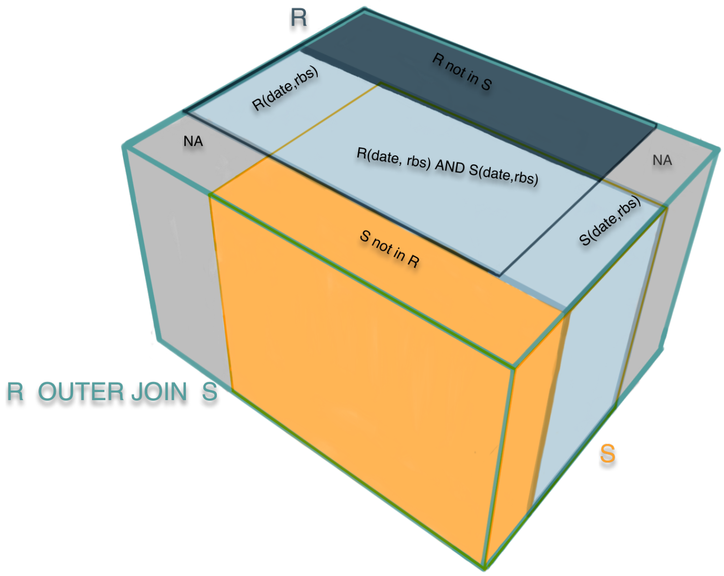


Figure 4.4: Outer join

The drawback of this approach is related to its insertions of NaNs, which will imply either postprocessing after merging all the time series or preprocessing before the learning stage.

4.2.3 Chosen strategy

Given the analysis, it has been decided that not losing information is more critical than having to dedicate efforts to postprocessing it later to handle the non-computable values. Therefore, the outer join approach will be used for the following steps.

4.3 Data Imputation

As in the previous step, and based on time constraints, it has been decided not to implement algorithms from scratch but rely on existing known packages to avoid investing in developing time.

Moritz et al. have analysed several univariate time series imputation implementations in R [40], which results later have been compiled in the `imputeTS` R package [41]. Therefore, it is reasonable to rely on their work and use their results to decide which imputing strategy should be taken. Thus, for the current work, the `imputeTS` package will estimate a structural time series model from the data and then perform the Kalman smoothing to fill the gaps.

It should be noted that before performing the Kalman smoothing, it is needed to have a state representation of the model, for which `imputeTS` supports Auto-ARIMA state-space estimation and structural time series model fitted by maximum likelihood. In addition, it should be mentioned that the development has been mainly done in python. Nonetheless, for this phase, an R package will be called from Python using the `rpy2` module as an interface to bounce between both languages.

4.3.1 Auto-ARIMA algorithm

Despite the fact that in the library the procedure is called *auto-ARIMA*, it does also support SARIMA models.

The main goal of auto-tuning these models is to choose the appropriates p, q, d, P, Q and D values so the model can make a good approximation of the process. If d and D are known, p, q, P, Q can be selected by using an information criterion such as the Akaike Information Criterion (AIC) [42]:

$$\mathrm{AIC} = -2\log(L) + 2(p + q + P + Q + k) \tag{4.1}$$

Where $k = 1$ if $c \neq 0$ in equation (3.16) and 0 otherwise, and L is the maximised likelihood of the model fitted to $(1 - B)^d(1 - B^s)^D\{Y_t\}$ [43].

`ImputeTS` library does not implement the (S)ARIMA estimation on its code. Instead, it uses the `forecast` package to do for it [41].

The `forecast` library implements the Hyndman-Khandakar algorithm for 3.16, which uses an heuristic that combines unit root tests, AIC minimisation and MLE maximisation as shown in Figure 4.5 has been transcribed in form of a DFD diagram the algorithm the authors have described in [43].

4.4 Database construction pipeline

It has been implemented a pipeline to build a unified and non-corrupted database from the sparse-raw files so that the forecasting section could learn from reliable data. In Figure 4.6, it is shown how the implemented blocks interact with each other to accomplish this task.

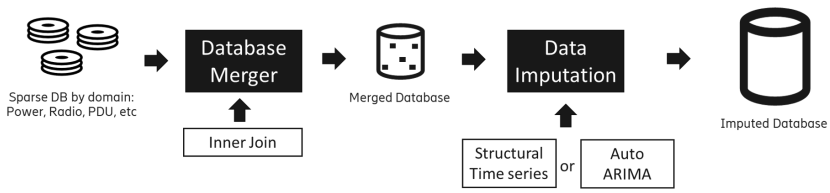


Figure 4.6: Database construction algorithm

4.5 Prophet fitting

Prophet's Stan core [32] can be found in their github repository[1] for further detailed research. The API exposes several more parameters to tune the models beyond the ones explained in Chapter 3. Some of them refer to the amount of MCMC samples used to fit the trend, the limit of trend changepoints, the confidence interval size or some regularization variables used in the Stan model, among others.

Another handy feature that allows using GAMs is the ability to plot each component on its own, which allows the analyst to spot abnormalities when debugging a model. These plots will be shown and explained in Chapter 5.

[1] https://github.com/facebook/prophet

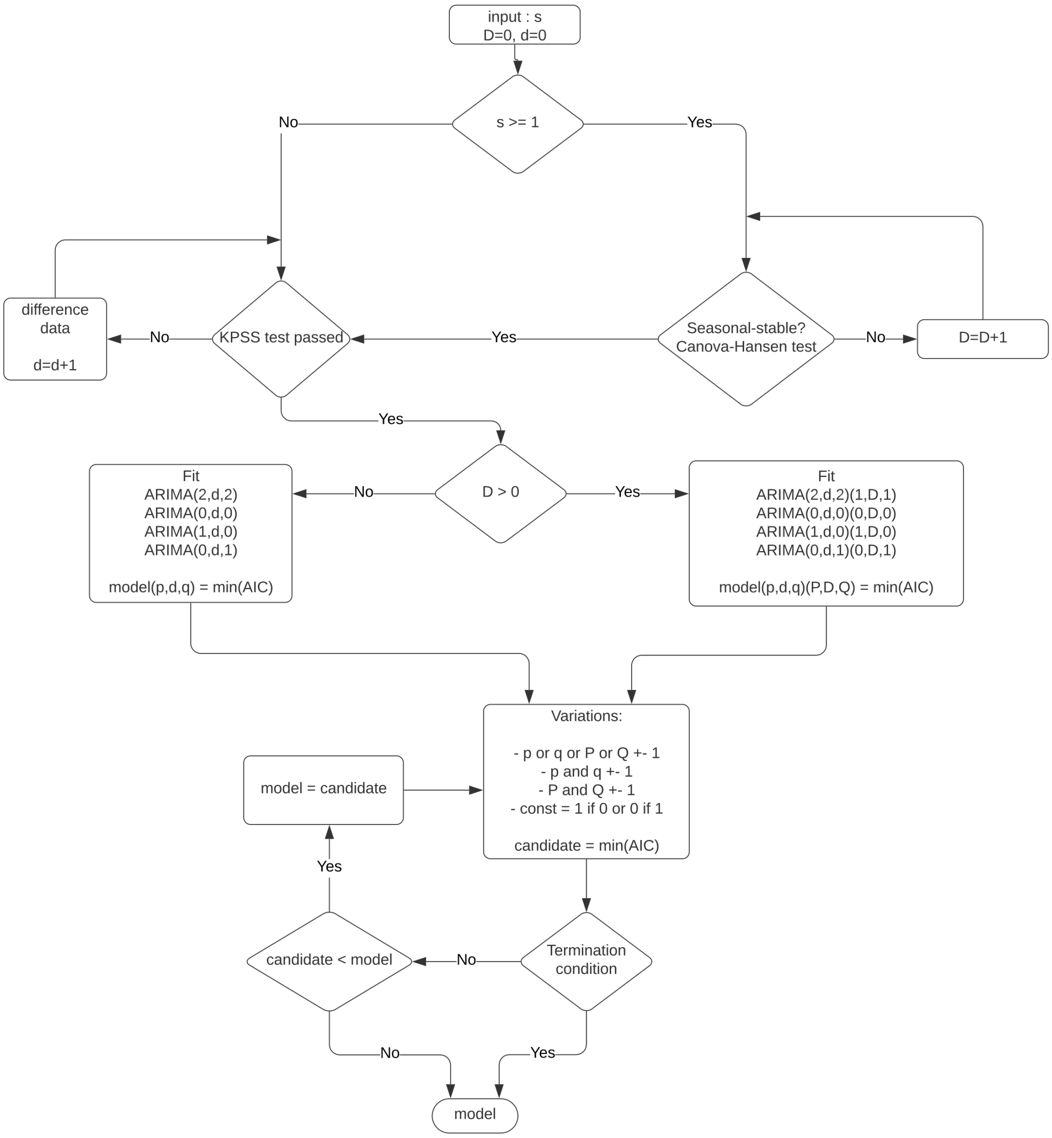


Figure 4.5: Hyndman-Khandakar algorithm for Auto-ARIMA estimation

5 Results

5.1 Comparative imputing experiments

A set of thorough experiments has been designed to compare both model estimation techniques and empirically determine which one performs better for the current dataset.

The starting point is a database containing approximately two months of measurements from several RBSs. Then, the database is mined to find $n = 50$ signals per feature with no missing data. After that, it has been artificially removed a given ratio of data using a uniform distribution so that it can be simulated a Missing Completely at Random (MCAR) scenario [44].

The two algorithms run to estimate the process models and use them to run the Kalman smoothing. It is used the coefficient of determination R^2 to compare the performance of the models.

$$R^2 = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2} \tag{5.1}$$

Where y_i refers to the i-th data sample, $\hat{y}$ its estimated value and $\bar{y}$ the series mean.

Thus, having perfect predictions would cancel out the right-hand term's numerator and produce a score equal to 1. Therefore, the closer the score is to 1, the better quality has been the data imputation.

The process of removing data, imputing and computing the coefficient of determination value is performed iteratively while increasing the amount of simulated missing data. The mean of the R^2 values from the selected RBS are reported.

In Figure 5.1, it is shown the experiment's algorithm pseudocode in order to give more context of the meaning of the results plots.

```
input : database, ratio_step, ratio_min, ratio_max
output: mean_scores

let n ← rbs batch size
let grid ← [ratio_min : ratio_step : ratio_max]
let mean_scores ← []

foreach feature in database.get_features() do:
    sites ← database.get_random_sites(n)
    i ← 0
    foreach ratio in grid do:
        let arima_scores ← []
        let structural_scores ← []
        j ← 0
        foreach site in sites do:
            data ← database.get_site_data(site, feature)
            sim_data ← remove_uniform_random_data(data, ratio)

            imputed_arima ← impute_with_arima(sim_data)
            imputed_structural ← impute_with_structural_model(sim_data)

            arima_scores[j] ← R^2(data, imputed_arima)
            structural_scores[j] ← R^2(data, imputed_structural)
            j ← j + 1
        mean_scores[i] ← (ratio, mean(arima_scores), mean(structural_scores))
        i ← i + 1
return mean_scores
```

Figure 5.1: Comparative imputing experiment pseudocode.

Figure 5.2 shows the results of the imputation comparison for the radio traffic as example. It can be seen that the R^2 value decreases –as expected– when the missing data ratio increases. It also can be seen that, in this example, and most of them, structural models performs better than ARIMA models for imputing.

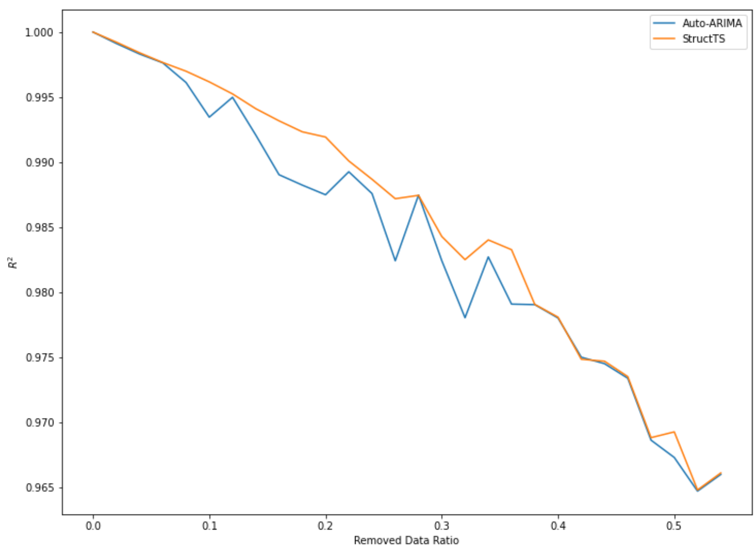


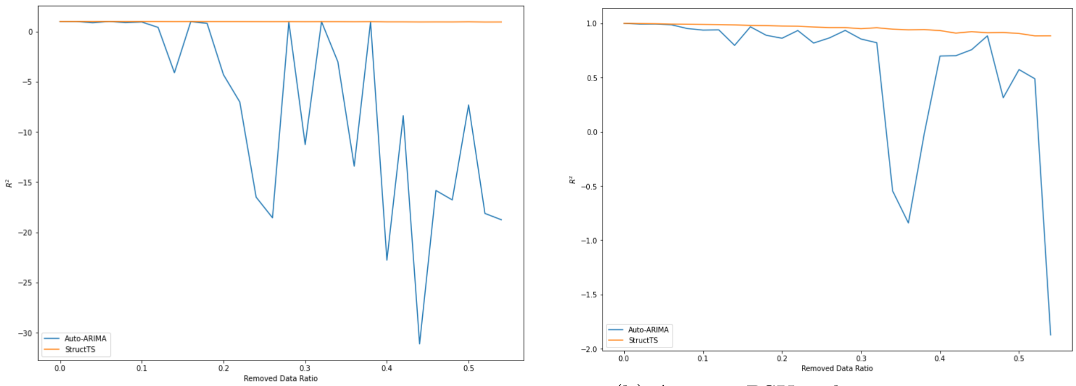
Figure 5.2: Radio traffic load imputation

Although it was obtained promising results for some of the imputed signals, there were other cases where the experiment did not perform as expected. In the following subsections, they will be discussed.

5.1.1 Unintuitive R^2 values

There are cases in which Auto-ARIMA estimation shows poor and even negative R^2 values as shown in Figure 5.3. These are unintuitive results, as R^2 values are usually expected to be limited to the $[0, 1]$ interval.

Nonetheless, this is meant only for linear models, where the worst fitted model is assumed to be the observations mean [45]. Thus, having negative R^2 values implies that the observations mean explains more variance than the fitted model.



(a) PDU system voltage imputation. (b) Average PSU utilisation imputation.

Figure 5.3: Imputation experiments with bad results.

5.1.2 Optim convergence failures

The model estimation threw runtime exceptions for some features due to the R code calls to the `optim` library not converging. These exceptions were found to be an actual bug in the library that occurs when the function being optimised tends to a constant value as shown in Figure 5.4. A dedicated routine was written to catch whenever this exception was thrown and run a simple interpolation instead of estimating the model.

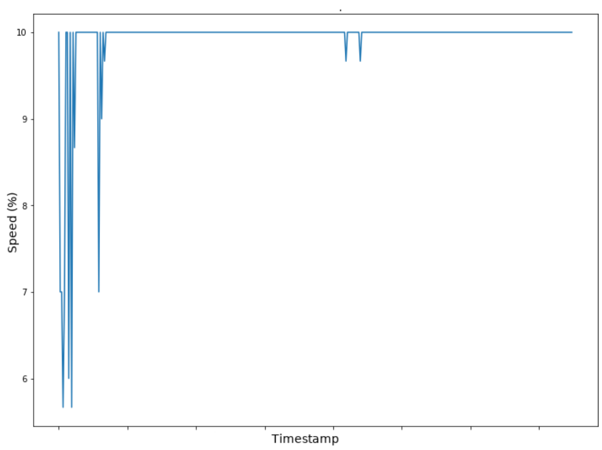


Figure 5.4: Example of problematic signals for Auto-ARIMA estimation.

5.1.3 Conclusion of the imputing experiments

The imputations using ARIMA models have shown to be unstable for some signals. Even though when, for the same signals, the structural model does not overperform either, it stills better than the very negative R^2 scores produced by the ARIMA approximations. In conclusion, based on this experiment's results, the structural models' approach has been chosen to impute the missing data and construct the database.

5.2 Forecasting initial experiments

The average PSU load signal shown in Figure 5.5 has been chosen to run the following experiments. It is not a particularly easy signal since it contains some severe outliers in the first days that turned the power to almost zero, and in the end, the average power consumption seems to decrease.

It is necessary to notice that in every forecasting model, the further the prediction horizon is, the more uncertainty and, thus, the lower performance. For this experiment, it has been decided to leave the last 10% of data for testing purposes.

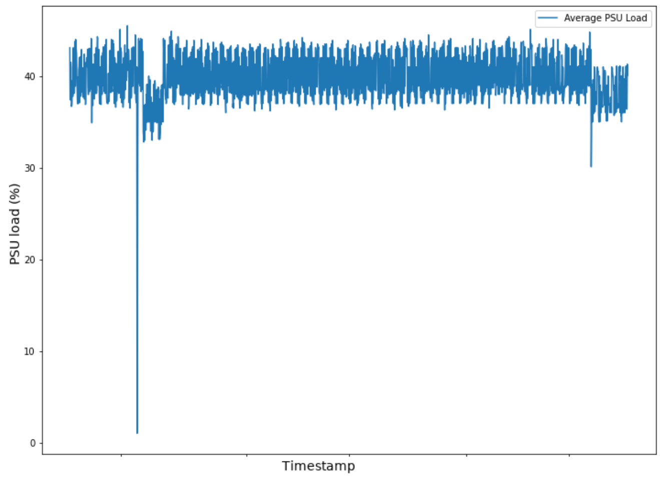


Figure 5.5: Average PSU load signal used for the forecasting experiments

In the following sections, the baselines models will be implemented first. Then, results of the univariate Prophet model will be shown. The models will be evaluated by using the scores described in Section 5.2.1.

5.2.1 Evaluation criteria

Coefficient of determination

As previously mentioned, the coefficient of determination R^2 is a measure of how many variance of $\hat{y}$ is explained by the variance in y in a linear regression context. It is compared against the considered *worst possible linear approximation* which corresponds to the samples mean.

Although the current application is not linear, R^2 is still considered a valuable score to compare predictor performances. Nonetheless, as the linearity assumption is not met, its values would reside in $(-\infty, 1]$, where 0 stills the mean value, but it is no longer considered the worst possible fit.

Mean absolute error

It measures, in absolute terms, the deviations from the true values. Mean Absolut Error (MAE) is preferred, given its interpretability, over Root Mean Square Error (RMSE) [46].

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N} |y_i - \hat{y}_i| \tag{5.2}$$

Mean out-of-bounds error

As a result of the fitting or forecasting process, Prophet's output is comprised by the mean prediction $\hat{y}_t$ and also its lower and upper boundaries $\hat{y}_t - \Delta\hat{y}_t$ and $\hat{y}_t + \Delta\hat{y}_t$, respectively. Where $\Delta\hat{y}_t$ is the computed confidence interval half-magnitude for the time t. The width of this interval is defined by the user and defaults to 80%

Let the Out-of-Bounds Error (OBE) be the out-of-confidence-bands error defined as zero if the true value y_t lies inside the confidence interval, and if it lies outside, defines the distance to the closest confidence boundary as defined in (5.3). Then the Mean Out-of-Bounds Error (MOBE) of N samples can be obtained by taking the mean of these values to obtain an overall performance score as in (5.4).

$$\text{OBE}_t = \begin{cases} 0 & \text{, if } \hat{y}_t - \Delta\hat{y}_t \le y_t \le \hat{y}_t + \Delta\hat{y}_t \\ \mathbf{min}\left\{ \left|y_t - (\hat{y}_t - \Delta\hat{y}_t)\right|, \left|y_t - (\hat{y}_t + \Delta\hat{y}_t)\right| \right\} & \text{, otherwise} \end{cases} \tag{5.3}$$

$$\text{MOBE} = \frac{1}{N}\sum_{t=1}^{N} \text{OBE}_t \tag{5.4}$$

5.2.2 Baseline predictions models

For the imputation step, the seasonal ARIMA and structural models have been learnt to fill the missing values by applying the Kalman smoothing algorithm. Furthermore, these models can also be used to forecast future observations. They are not expected to overperform. Nonetheless, predicting with them can be used as a benchmark to compare other models' improvements over their baseline.

Structural model predictions

In Figure 5.6, it is shown the overall performance of the structural time series predictor. It can be seen that its predictive mean tends to follow the test dataset mean, whereas the confidence interval is wider the further the prediction horizon is. In Appendix A.1.1, more detailed plots are available for the reader without overpopulating the main text of the report.

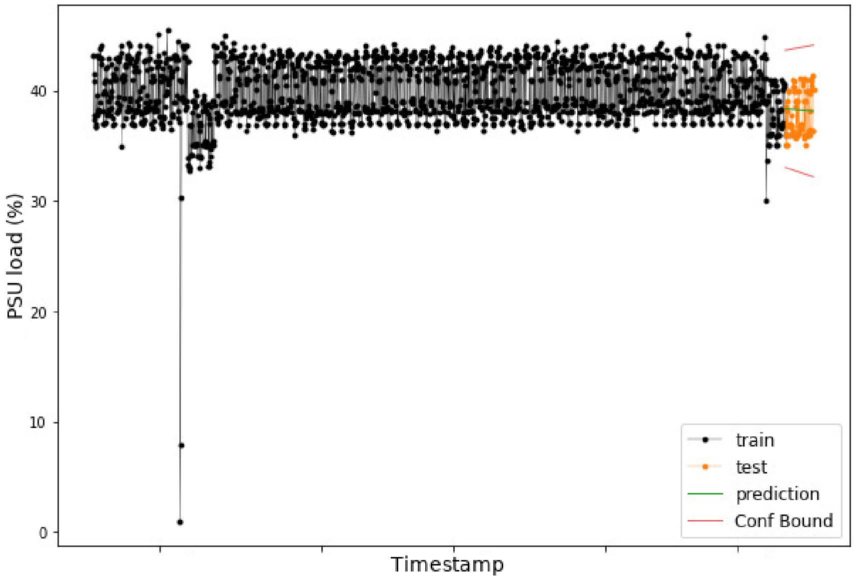


Figure 5.6: Structural model 3-days baseline overall performance

When plotting how the predictions $\hat{y}$ and the real values y are distributed, the ideal prediction would be denoted by the positive unitary line. Therefore, the results in Figure 5.7 can be considered very poor.

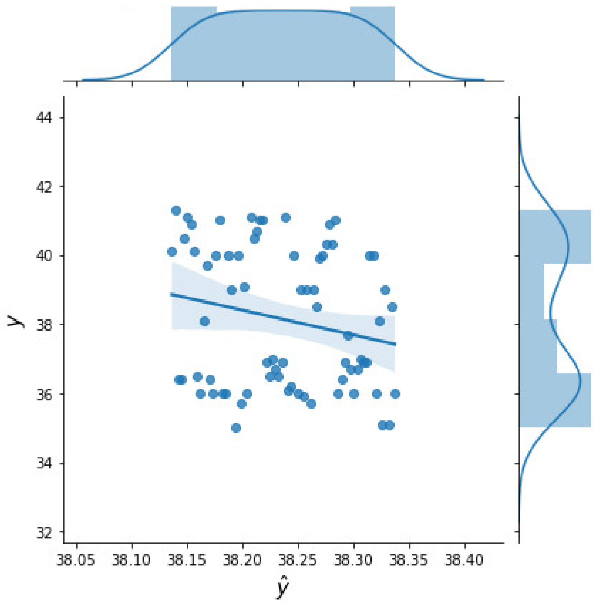


Figure 5.7: Structural model 3-days baseline joint distribution

Auto-ARIMA implementation

Same as in Section 5.2.2, an Auto-ARIMA model is trained and used to make predictions. In this case, it can be seen that, in the short-range, the predictive mean tries –poorly– to follow the actual data fluctuations. However, later, in the long-range converges to the data mean. More detailed plots of the results can be found in Appendix A.1.2.

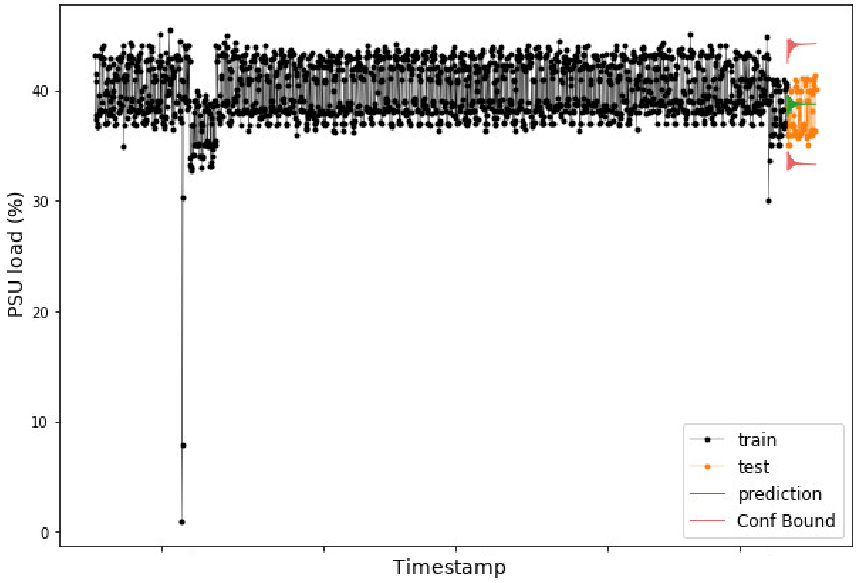


Figure 5.8: Auto-ARIMA 3-days baseline overall performance

Although the joint distribution of predictions and true values has been slightly improved, stills very poor.

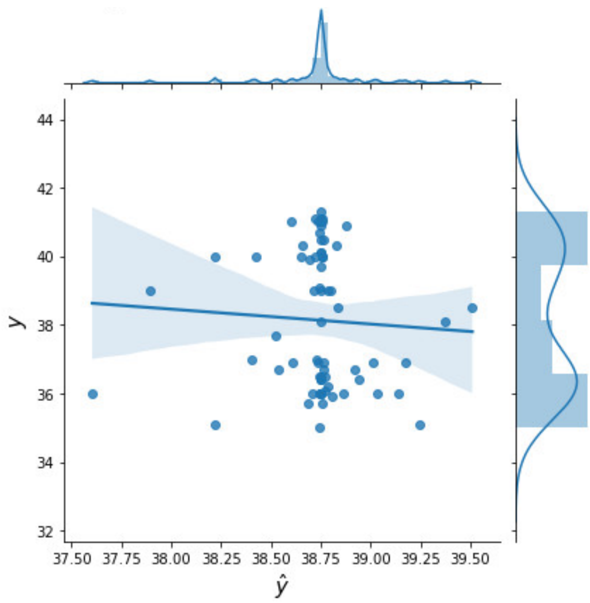


Figure 5.9: Auto-ARIMA 3-days baseline joint distribution

Overall baselines scores

In Table 5.1, it is summarised the long-range prediction of both baselines. They perform very poorly; nonetheless, as naive baselines, it is not expected any other outcome.

Score	Structural Model	Auto-ARIMA
MAE	1.87	1.95
R^2	-0.02	-0.12
MOBE	0	0
RMSE	2.02	2.12

Table 5.1: Baselines long-range performance

5.2.3 Univariate Prophet model implementation

Figure 5.10 shows the complete results of training and testing for a univariate Prophet model. In comparison to the baselines, now it can be observed that the predictive mean tends to follow a more clear seasonal pattern. Nonetheless, in the testing set, it appears to miss the trend change.

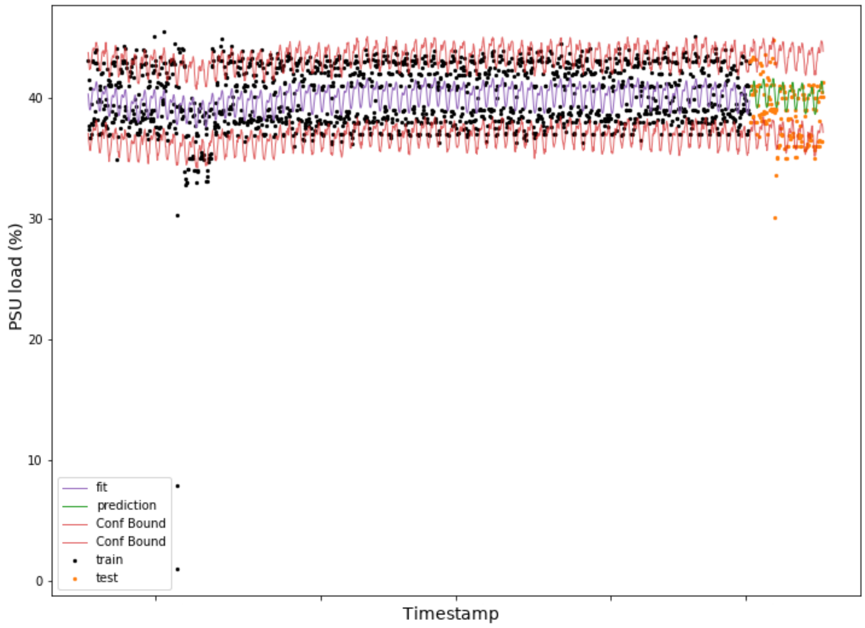


Figure 5.10: Training, test and predictions from the univariate Prophet model

More detailed plots can be found in Appendix A.2.

Model learnt components

Figure 5.11 presents the learnt approximation of every component in the GAM. Although the trend did not overfit the outliers, it stills learnt that there is a breakpoint and changed the piecewise trend.

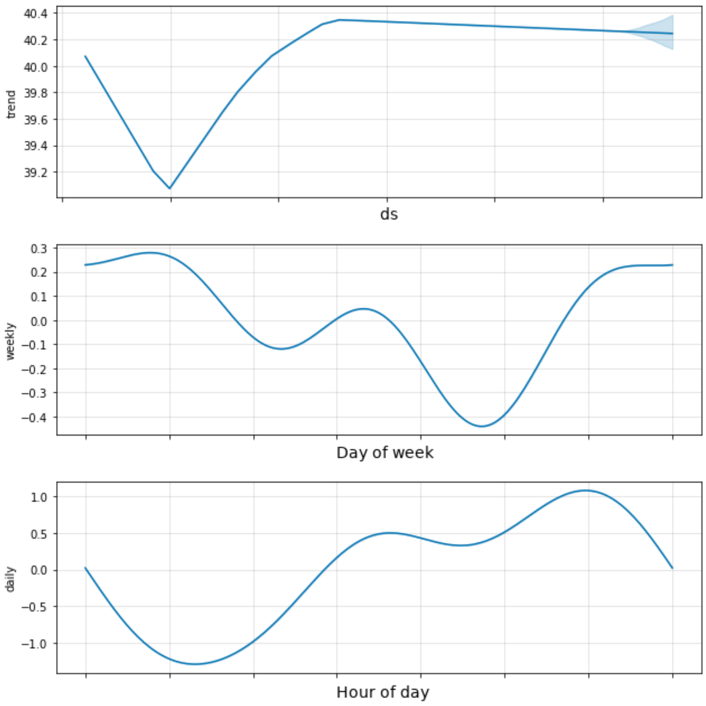
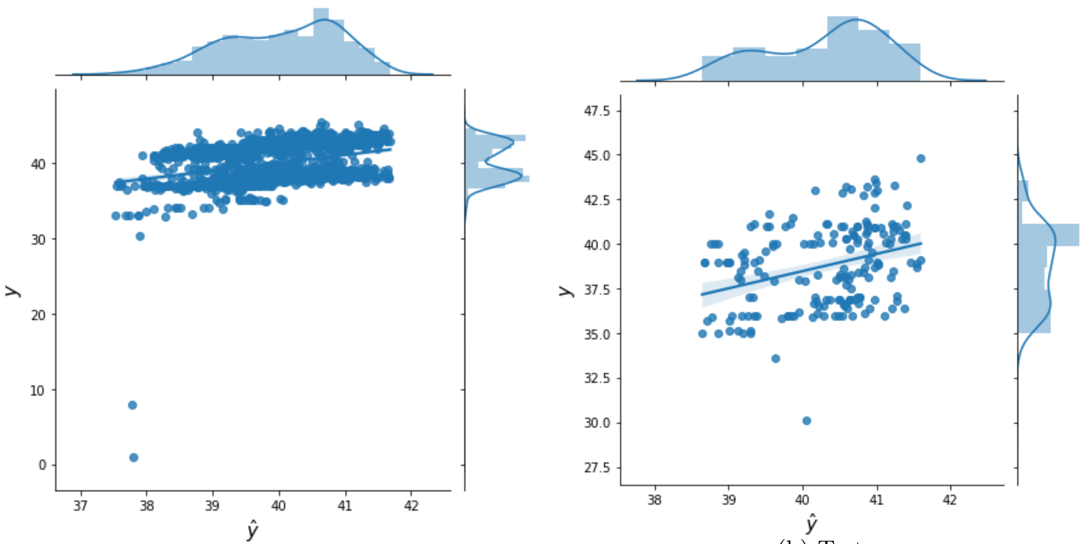


Figure 5.11: Learnt components from the univariate Prophet model

Data and predictions joint distribution

If the joint distributions for the true values y and the predicted values $\hat{y}$ are plotted, it can be seen how the outlier in the training set was not learnt and how the approximation becomes erratic in the test set. Nonetheless, they already show a considerable improvement from the baselines distributions.



(a) Train

(b) Test

Figure 5.12: Univariate Prophet joint distributions of y and $\hat{y}$

5.2.4 Prophet implementation using exogenous regressors

To fully exploit the flexibility of GAMs, Prophet's API allows defining custom regressors, which in the current work will be called exogenous variables as a resemblance of SARIMAX models. These variables are the ones explained in Chapter 2.

The overall performance can be seen in Figure 5.13. Compared to the univariate model, it can be seen that the confidence bounds now are narrower since there is less uncertainty in the predictions. It also can be seen that the model now can predict the trend change in the test set.

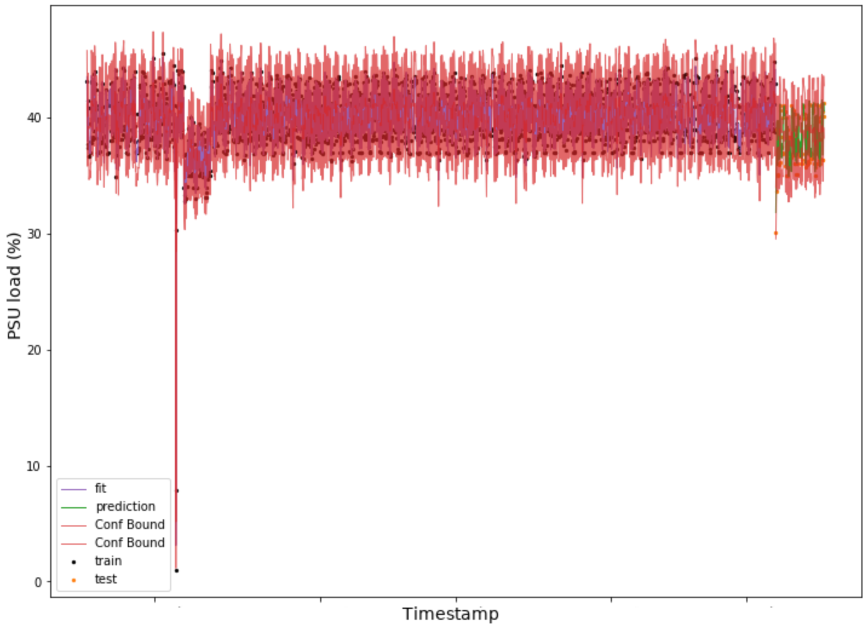


Figure 5.13: Training, test and predictions from the multivariate Prophet model

Model learnt components

The model components now show the additive factor of all the exogenous regressors, which seems to contain the information not learnt by the univariate training residuals.

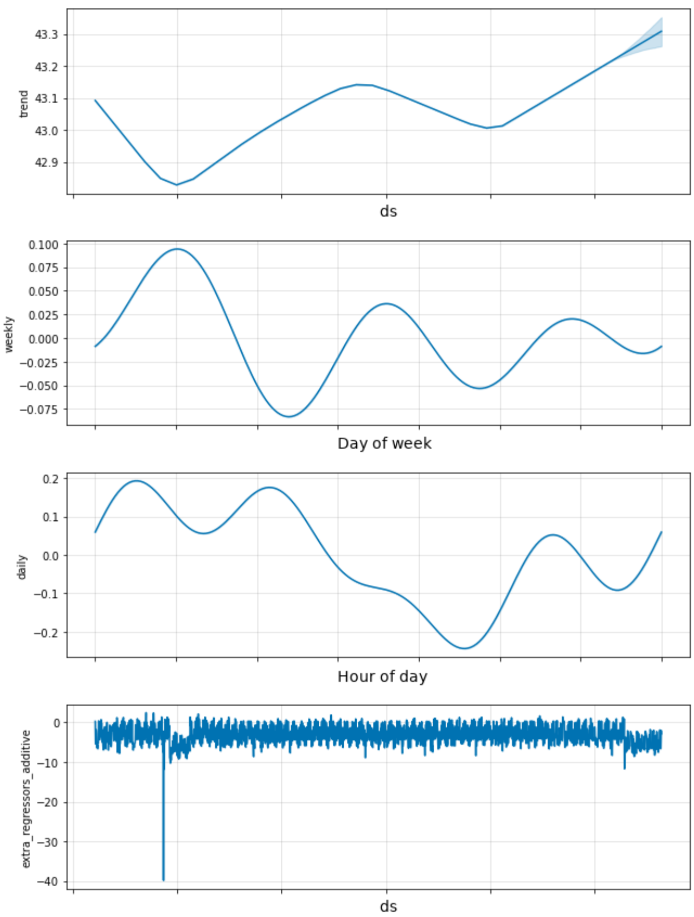


Figure 5.14: Learnt components from multivariate Prophet

Data and predictions joint distribution

The joint distributions for y and $\hat{y}$ in Figure 5.15b now shows that the model has a significant improvement in its predictions accuracy compared against the baselines.

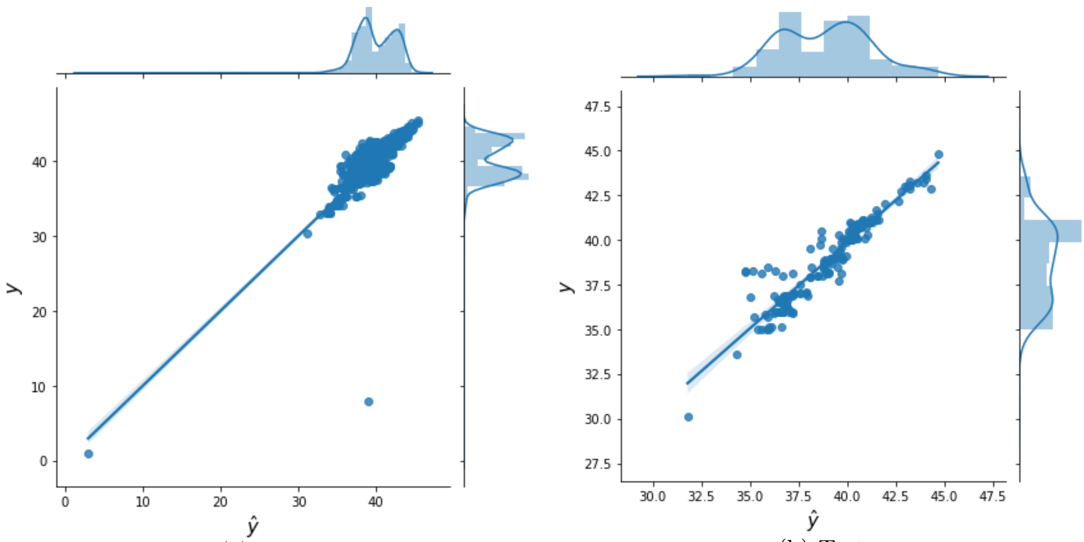
In the plot for the training set, it can be seen that the trend model approximates the abrupt drop. Nonetheless, the positive trend in the correlation between y and $\hat{y}$ in the test set shows that the model is not overfitting the training data. Moreover, the trend drop in the testing set is also approximated without issues.

5.2.5 Example forecast performance comparison

In order to quantify the improvement of the long-range predictions made by the usage of the Prophet model, in Table 5.2 are summarised their performance scores.

	MAE	R^2	MOBE	RMSE
Univar - Train	2.16	0.14	0.04	2.53
Univar - Test	2.11	-0.57	0.04	2.70
Multivar - Train	0.55	0.83	0.05	1.13
Multivar - Test	0.57	**0.85**	0.03	**0.85**

Table 5.2: Example of long-range prediction performance by the Prophet model



(a) Train (b) Test

Figure 5.15: Multivariate Prophet joint distributions of y and $\hat{y}$

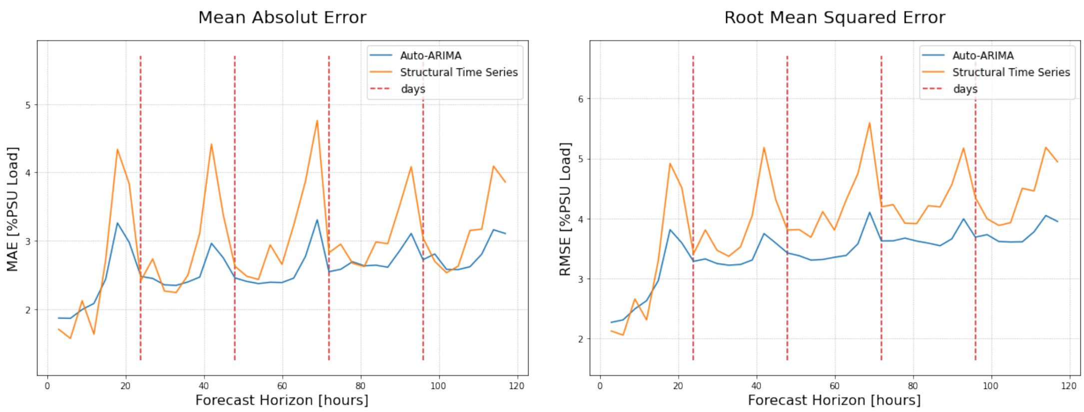
In this example, the significant improvement done by using the exogenous predictors in the multivariate model can be observed, obtaining a $R^2 = 0.85$, which for a non-linear case can be considered a good approximation.

5.3 Exhaustive forecasting experiments

After visualising how the models comparatively perform in an example case, it is needed to obtain a general view of how they perform. Therefore, similar to the experiments in Section 5.1, it is performed an exhaustive iterative evaluation of the predictions while moving the predicting horizon further and further.

In the following subsections, the exhaustive baseline results are presented, and then Prophet model results are obtained to visualise its general improvement.

5.3.1 Baselines predictions moving horizon



(a) Baseline MAE (b) Baseline RMSE

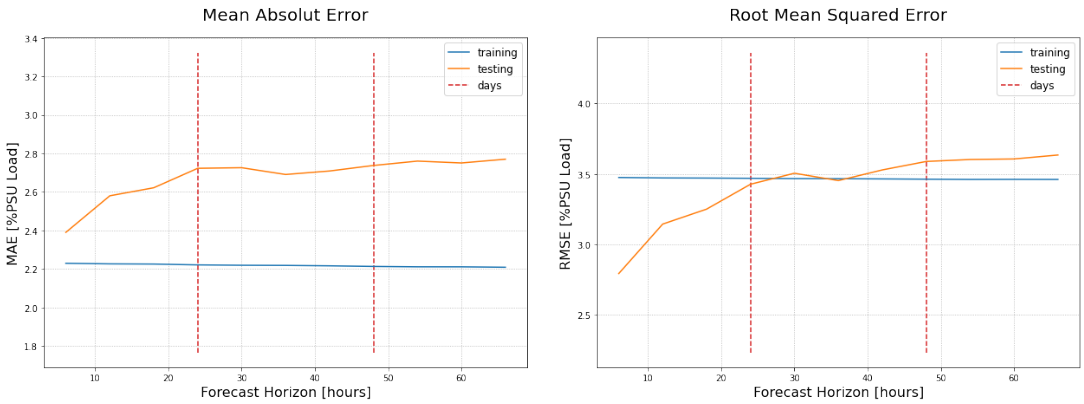
Figure 5.16: Baseline predictions performance vs forecast horizon time

Figure 5.16 shows together the Auto-ARIMA and structural model performances. The curves are interesting due to their peak-valley seasonal-alike shapes. Their possible explanation is related to the original signal peaks and valleys variability, i.e., the minima are more show less variability than the maxima. Therefore, the predictions for the minima are less error-prone than for the maxima.

On the other hand, it can be seen that, opposed to the results for the imputation phase, the structural model provides poorer predictions than the ARIMA. Thus, if any of the baselines would have to be chosen, the Auto-ARIMA would be the best –less-worse– candidate.

5.3.2 Univariate Prophet prediction moving horizon

This set of experiments have been found to require hefty computing power and time. Since the following overall experiment took ~ 17 hours to finish, it has been limited to sampling 50 models performances for a three days long-range horizon maximum.



(a) MAE (b) RMSE

Figure 5.17: Univariate Prophet predictions performance vs forecast horizon time

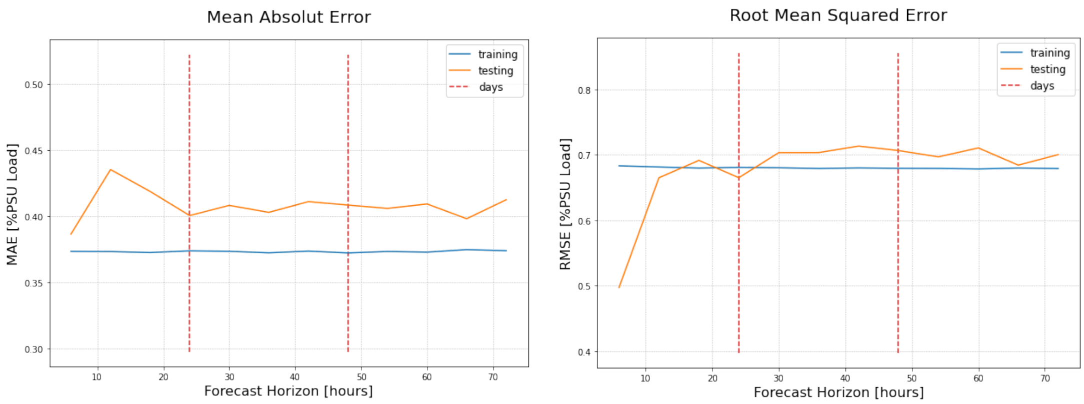
In the results in Figure 5.17, it can be observed that as long the time horizon increases, as expected, the prediction errors tend to increase also. While MAE show that the prediction errors are consistently higher than the training error, the RMSE shows lower scores for the testing set than for the training when the time horizon is close to the present time. This shift can be explained by the fact that RMSE penalises higher the large errors compared to MAE.

It is important to notice that, as the PSU utilisation signal is already in per-cents, these results are easily interpretable as per-cent error also. Therefore, in the long-range, the univariate Prophet model, shows that in average it can achieve MAE ~ 2.5 and RMSE ~ 3.6 PSU per-cent utilisation.

5.3.3 Multivariate Prophet prediction moving horizon

Now it is the turn of the Prophet model with external regressors that showed the best performance in the first example. Therefore, it is expected to maintain that tendency but now with more confidence that the performance in the example was not some fortunate random event.

These experiments have consumed even more computing time than the previous in Section 5.3.2 having spent more than 18.5 hours to finish.



(a) MAE (b) RMSE

Figure 5.18: Multivariate Prophet predictions performance vs forecast horizon time

The results in Figure 5.18 show that the long-range prediction results for the Prophet model using exogenous regressors are consistently good having both, MAE and RMSE below 1%.

Other noticeable results are the ones shown in Figure 5.19, in which the R^2 scores in long-range are ~ 0.88. Although it might seem unintuitive to have better R^2 in the long-range rather than in the short term, it can be explained by the amount of data used to compute the score: the fewer data, the likely to have *a general bad approximation*.

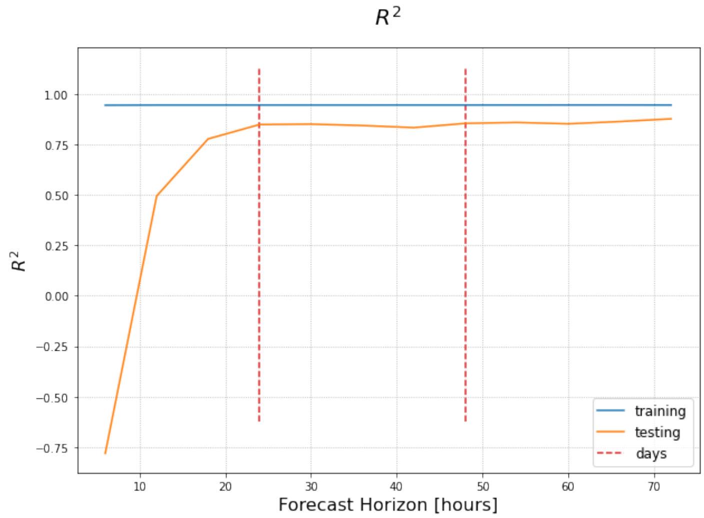
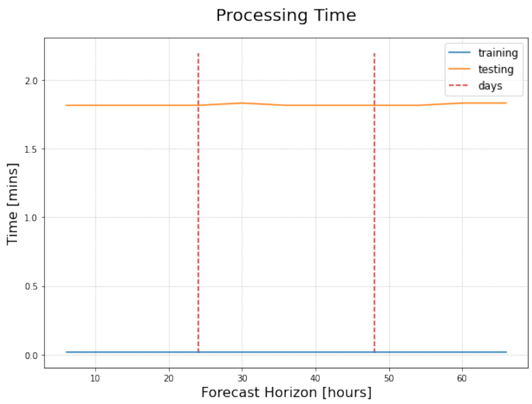


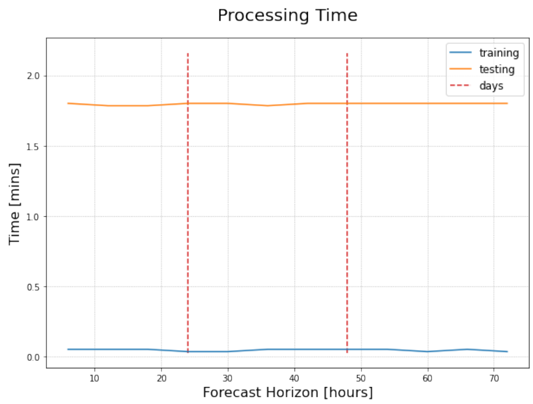
Figure 5.19: Multivariate Prophet model R^2 vs forecast horizon time

5.3.4 Time performances

Figure 5.20 shows the average consumed time for running the exhaustive experiments in Ericsson's computing farm. It can be seen that the testing time is very similar for univariate and multivariate case. On the other hand, the training time for the multivariate model is slightly higher than the univariate one. Nonetheless, both are under half of a second on average. It is also interesting that the longer the prediction range, the times remain almost constant.



(a) Univariate experiment computing time



(b) Multivariate experiment computing time

Figure 5.20: Experiments computing times

5.4 Performance summary

Finally, this section summarises the average performances per day for all the models under analysis for ease of comparison. Table 5.3 shows that the Prophet model with exogenous regressors vastly outperforms the other methods in terms of MAE.

Model	**1 day**	**2 day**	**3 day**
Structural	2.55	3.17	3.08
ARIMA	2.59	2.812	2.70
UniVar-Prophet	2.88	2.96	3.01
MultiVar-Prophet	**0.41**	**0.40**	**0.40**

Table 5.3: MAE Summary of models performance by day

Likewise, in Table 5.4, it can be observed the long-range stability of the multivariate Prophet model having, again, the best RMSE scores.

Model	**1 day**	**2 day**	**3 day**
Structural	3.27	4.31	4.35
ARIMA	3.22	3.77	3.81
UniVar-Prophet	3.54	3.88	4.02
MultiVar-Prophet	**0.62**	**0.71**	**0.68**

Table 5.4: RMSE Summary of models performance by day

Lastly, Table 5.5 presents how the multivariate Prophet model is even capable of having acceptable R^2 scores, which in this type of data is not an easy task.

Model	**1 day**	**2 day**	**3 day**
Structural	-7.23	-8.90	-2.90
ARIMA	-95.76	-2.25	-0.23
UniVar-Prophet	-203.76	-1.04	-0.98
MultiVar-Prophet	**0.33**	**0.85**	**0.86**

Table 5.5: R^2 Summary of models performance by day

5.5 Power headroom estimation

Until this point of the work, the research has been mainly focused on PSU loads forecasting. Nonetheless, as mentioned in the introductory context, the goal is to predict the power headroom defined in (1.1). However, as there are no direct measurements of power headroom that can be learnt, it is needed to derive it from the power consumption.

Therefore, to provide a power headroom forecast, after estimating the loads' consumptions' future values, it is needed a further step to translate that information into power headroom terms. The following sections will propose a way to manage to achieve it and discuss other aspects of its technical applicability.

5.5.1 Power headroom derivation as PSU utilisation complement

As it has been exposed in Section 2.2.1, the power load is a measure of *how many percentual power capacity it is being used at that time.* Thus, it is straightforward to claim that the percentual power headroom is its complement.

If the interest is to obtain a measurement in Watts units, the installed power capacity in the RBS, P_{max}, is needed to be known so that its proportion can be computed as follows.

$$\begin{aligned} P_{h[\%]} &= 100 - P_{L[\%]} \\ \text{Where } P_{h[\%]} &: \text{Power headroom in percents} \\ \text{and } P_{L[\%]} &: \text{Power loads consumptions in percents} \end{aligned} \tag{5.5}$$

If the interest is to obtain a measurement in Watts units, it is needed to know the installed power capacity, P_{max}, in the RBS so that its proportion can be computed as showed in (5.6).

$$\begin{aligned} P_h &= P_{max} \cdot \frac{P_{h[\%]}}{100} \\ \text{Where } P_{max} &: \text{Installed power capacity in Watts} \end{aligned} \tag{5.6}$$

In Figure 5.21, it is shown the PSU utilisation signal used as example test in Chapter 3.2 and its derived power headroom.

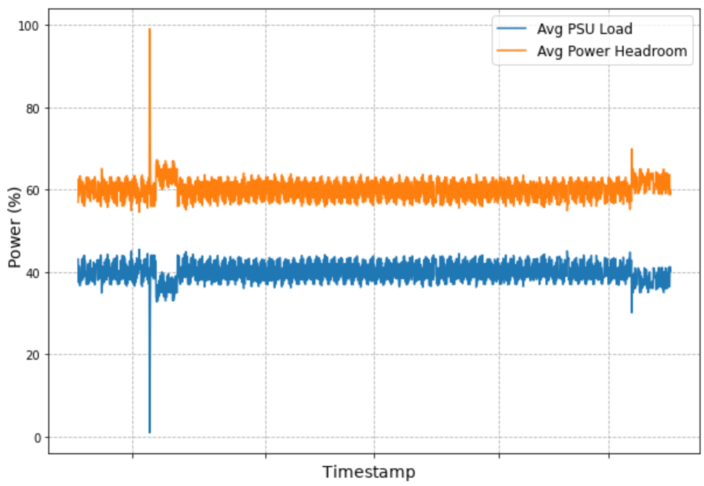


Figure 5.21: Power headroom derivation from PSU Load

5.5.2 Criterion considerations

In general terms, an alarm is meant to make someone notice that something abnormal is happening in a given process or, in a forecasting context, may (or will) occur in the future and support the operator response [47].

In the current works' context, the alarm meaning can be conceived as the RBS not having enough power to manage to keep its by-design behaving.

Although it should be possible to define how likely a power overload is given the system's current state in probability terms, triggering an alarm is a binary task. The system is operating either in a *"safe"* or in an *"abnormal"* region. Then crossing a boundary that separates these two regions is the trigger for the alarm.

There are different techniques to tune an optimal trigger level. It could be done based on a system variable, or latent variable, or even a joint distribution of the two kinds [48].

Choosing the best possible boundary is considered to be out of the current work scope. Nonetheless, as an initial approach, it can be proposed to be settable by the user according to their needs and expertise.

From this assumption and the derivations in (5.5) and (5.6), the following guidelines should be taken into consideration.

Values interpretation

Percentages values are not always able to fully represent the system conditions. For example, it is very different having 10% of power headroom where the total capacity is 10 kW and 1 kW since the relative oscillations could be drastically different. Therefore, the power magnitude should be taken into consideration.

PSUs do not partially fail

The power capacity is not continuous. It will have discrete values as shown in Figure 5.22, and also, the PSUs can have different power contributions. This quantisation means that in order to set the threshold is needed to take into account the worst-case scenario, which can be depicted by the sentence: *"can the RBS work if the next failure is the most power contributing PSU?"*

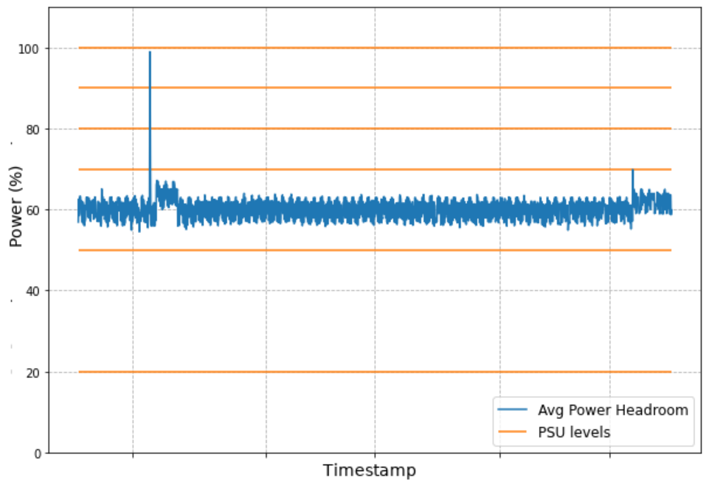


Figure 5.22: Power headroom and discrete availability in percents

5.5.3 Alarm triggering and the n-level safety criteria

Let P_{max} be the maximum power capacity installed and working in an RBS, $\bar{P}_{Load}$ the average PSU utilisation and Δp the confidence interval used when training the Prophet model. Let N be the amount of working PSUs, let $\{P_{PSU_1}, \ldots, P_{PSU_N}\}$ such that $P_{PSU_1} > \ldots > P_{PSU_N}$ are the descending sorted power contributions from the working PSUs.

Then, let the n-level safety criteria, be the n amount of worst-case PSU failures to keep as operational margin:

$$\begin{aligned} P_{max} - \sum_{i=1}^{n} P_{PSU_i} &> \bar{P}_{Load} + \Delta p \\ P_{max} - \left\{ \bar{P}_{Load} + \Delta p \right\} &> \sum_{i=1}^{n} P_{PSU_i} \end{aligned} \tag{5.7}$$

However, as the power headroom is by definition the difference between the installed power capacity and the power being consumed, it can be said that:

$$\begin{aligned} P_h &= P_{max} - \left\{ \bar{P}_{Load} + \Delta p \right\} \\ P_h &> \sum_{i=1}^{n} P_{PSU_i} \\ \therefore P_{crit_n} &= \sum_{i=1}^{n} P_{PSU_i} \end{aligned} \tag{5.8}$$

Where P_{crit_n} is the n-level criteria threshold. Thus, whenever the power headroom forecast $\hat{P}_h$ at time $t + k$, being t the current time, meets the condition in (5.9) a fault alarm must be triggered

$$\hat{P}_{h_{t+k}} \geq P_{crit_n} \tag{5.9}$$

6 Discussion

6.1 Results

After analysing the results from the imputation experiments and the prediction generalisation, the structural time series and multivariate Prophet model have been chosen for each stage, respectively. In Figure 6.1 it is shown how the pipeline has been defined.

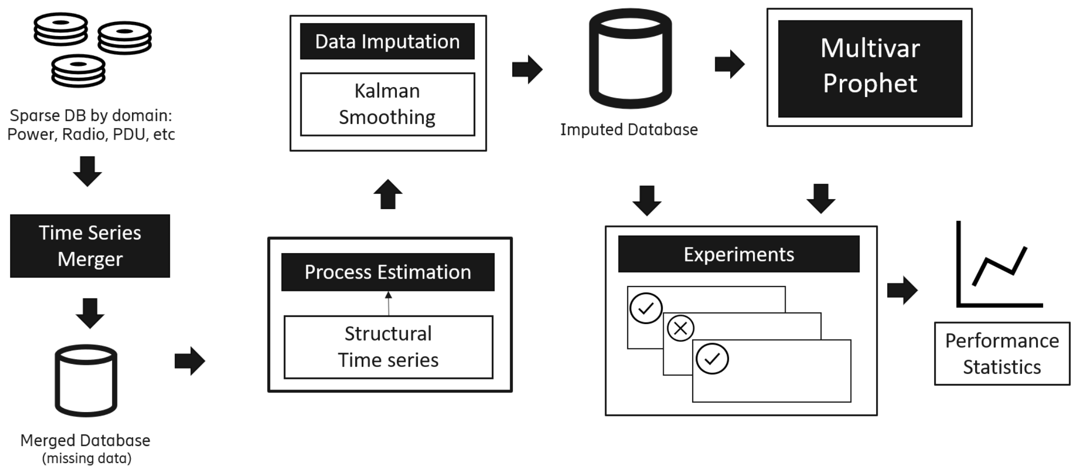


Figure 6.1: Final pipeline architecture

In the imputation stage, the Auto-ARIMA model has shown to be unstable for some features in the dataset, whereas the structural models have not. This inconsistency should not be understood as a failure of the SARIMA approach itself but as an instability in the auto-ARIMA algorithm not reaching a proper model through its heuristics. Thus, even though the auto-ARIMA algorithm has been published and implemented in a popular library, it still has some arbitrary steps.

When it comes to comparing the baselines performances against the univariate Prophet model, it can be seen that the latter sometimes performs even worse than the baselines, which might induce some critical thoughts about the Prophet model as an insufficient framework.

Nonetheless, when adding exogenous regressors, the model shows its potential by lowering the error rates significantly—even more, reaching $R^2 > 0.8$ values, which can be considered a reasonable approximation due to the nature of the target signal.

Although a power headroom derivation from the PSU utilisation has been presented, the n-level criteria have not been tested in a case that meets them and triggers an alarm.

6.2 Methods

6.2.1 Time series merging

In the current case, to build the database, it has been decided to merge the time series with an outer join and then impute the missing data rather than using inner joins and reduced data. Nevertheless, this does not mean that this approach is always the recommended approach. A further study that could have been done is to extract the *longest* uncorrupted piece of time series $\{\boldsymbol{U}_t\} \subseteq \{\boldsymbol{Y}_t\}$ and run statistical tests to measure their similarities and determine if subseries $\{\boldsymbol{U}_t\}$ is *representative* of the complete series $\{\boldsymbol{Y}_t\}$.

This scenario could happen when having a very long time series of highly seasonal data. Unfortunately, it has not been the case for the available data since extrapolating two months to an entire year can be considered already only a proof of concept. Trimming the time series, even more is undesired.

6.2.2 Time series imputation

It has been only used univariate approaches to estimate the model and smoothen the signals, which implicitly –and naively– implies that the features are independent and no correlations can be found between them. This decision was intentionally made to set a baseline and, as the first attempt on this domain within the company, prioritising simplicity over sophistication.

It should be noted that multivariate imputing techniques could improve the processing time and the imputation accuracy.

6.2.3 Forecast

The Prophet model was chosen as a predictive model for its novelty, flexibility, well-proven usage in the social networks industry, and good performance in the long-range forecasts. Besides its theoretical attractiveness, it is also well suited for time-limited work due to its well-designed APIs and available documentation. Nonetheless, there are no strong arguments to state that it is the *best-suited* model since it has been only compared against the baselines.

The baselines have been implemented using only univariate approaches. Therefore comparing the multivariate Prophet model against them might no the fairest. Other multivariate methods might be used as baselines in such case.

Other than the Prophet model, a boosted decision trees regressor was explored using XGBoost, which showed promising results. Nonetheless, its current implementation status can be considered half-done and, therefore, its results cannot be conclusive.

6.3 Ethical considerations

The project aims to open a research line towards the decreasing of telecom operator companies operational costs. Nonetheless, this is not limited to a purely economic benefit to the companies. Maintenance duties can imply some risks to the field workforce when, for instance, the RBS is located in a remote location. Therefore, this project effects will be also beneficial for the field personnel.

Environmentally wise, reducing the number of trips and interventions in remote locations, which could be close to nature wildlife, will also reduce the pollution levels related to them and invade the wildlife space.

Although the project's domain is in mobile communications, it is directly related to human behaviours. Therefore, analysing how this may be affected by social biases and whether its effects could be more beneficial towards one group of people or detrimental to others is, in fact, a moral obligation from the researcher's and developer's perspective.

In that line, it is crucial to state that different groups of people behave differently. Therefore, the seasonalities learnt from data from one country may not reflect the customs in others, especially when it is related to holidays or temperature. As a consequence, it can be stated that when a system of this nature is deployed to service, it needs to learn from data coming from the region where it is being deployed or some other statistically similar.

7 Conclusion

It has been explored and evaluated different techniques to manage univariate time series from sparse file sources and synchronise them in just one multivariate time series data structure. For this case, if they would be merged using an inner join approach, it would affect the length of the series since only the intersections would be included. Even more, if there was any missing sample, it could violate the constant sampling time assumption. Hence, using an inner join approach was more reasonable.

Nonetheless, by using an outer join to merge the time series, NaN values were introduced, making Prophet fail. Consequently, it was decided to impute them by using Kalman smoothing from model approximations obtained from structural time series estimation by maximum likelihood or a SARIMA auto fitting using the algorithm in [43]. It was observed that when simulating MCAR from complete data and then comparing the estimates, structural time series modelling tends to approximate better and behave more stable than auto-ARIMA models.

Since the only forecasting model, other than the baselines, with conclusive results, has been the Prophet model, there are no rooted proofs for concluding that it is, in fact, the most suitable model for this application. Nonetheless, it still shows the ability to estimate future power consumptions in a long-rage horizon accurately.

It has been analytically derived a formula to compute, in real-time, the power headroom. Considering that the power reductions are not continuous but quantised, it has been concluded that the power headroom cannot be less than the most significant PSU power contribution. Furthermore, this conclusion has been extended for n consecutive worst-case scenarios, and the n-level safety criterion has been proposed.

8 Future work

Imputing the data was not initially considered as part of the project. However, after running into issues with the Prophet model due to NaN values, it had to be decided between two options. First, the prediction model could have been changed for another that could handle missing data, or on the other hand, the merged time series could have been processed to fill the gaps so Prophet would not fail. The latter was chosen, leaving the former unexplored due to time constraints. Future work could research pointing towards that direction and evaluate how much improvement is obtained from imputing the data and conclude whether this non-informed decision was correct or not.

It will also be interesting to research multivariate time series imputation techniques and replace the several independent univariate imputations done in this work, overtaking the naive –and known wrong– assumption of independence between the features. In the same stage, the multivariate processes estimations should not be discarded since they can be reused to forecast power consumptions.

Further research must explore other forecasting alternatives, such as the mentioned XG-Boost regressor with promising inconclusive results or RNN or Transformers, and the new architectures that mix the Prophet model with other layers, to mention some.

The proposed alarm criteria have been analytically derived. Although it works to set a reasonable threshold, it still relies on an expert to tune its value. In order to fully automate the new predictive alarm system, it is also needed to apply an optimisation technique to find the best alarm criterion.

When it comes to evaluating an eventual deployment of the service, the current approach needs to collect some time of data from the actual RBSs since it has been conceived as *ad hoc* solutions, which might not be the ideal solution at all.

The initial conceptualisation of this work was to train a faults predictor, so the installed power capacity was the most critical variable to predict. Unfortunately, after mining more than a thousand RBSs data, only six faults suspicions were found, forcing the change in the approach from events prediction to a power demand forecasting assuming the power capacity as a constant. The final solution must withdraw this assumption and change the P_{max} value to another forecast $P_{max}(t)$ being evaluated by faults predictors.

During the research, it has been noted that, in appearance, there is no unique pattern preceding a PSU fault. Thus, if the final goal is to pre-train some models to be deployed without collecting data on-site, these suspected failure modes must be studied. It has been proposed to cluster the fault modes and investigate the possibilities to treat them as if they were the same RBS allowing to deploy pre-trained models depending on the RBS class.

A Further results

A.1 Baseline predictions

A.1.1 Structural time series

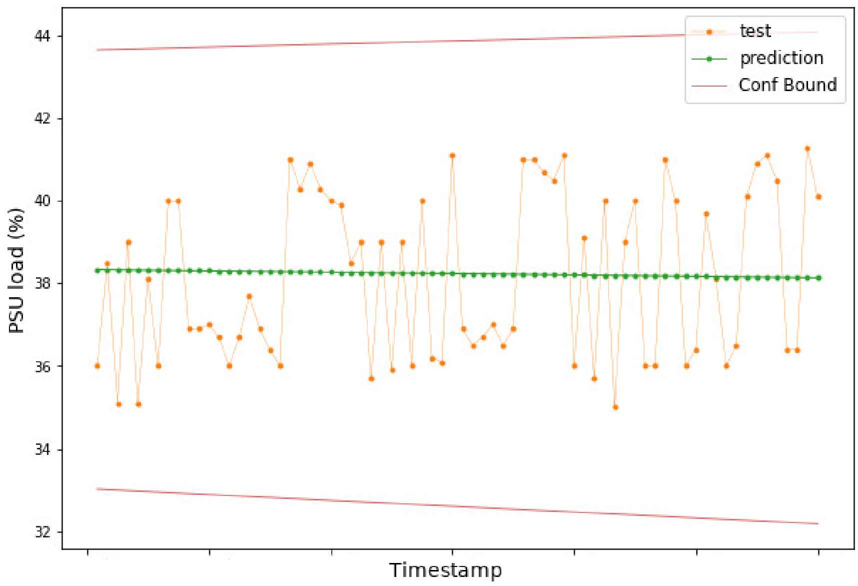


Figure A.1: Structural model baseline predictions

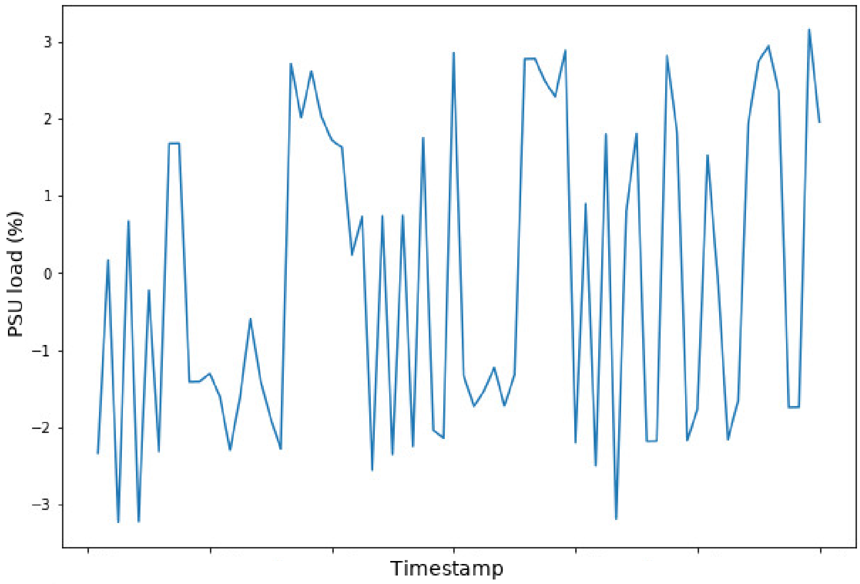


Figure A.2: Structural model baseline residuals

A.1.2 Auto-ARIMA

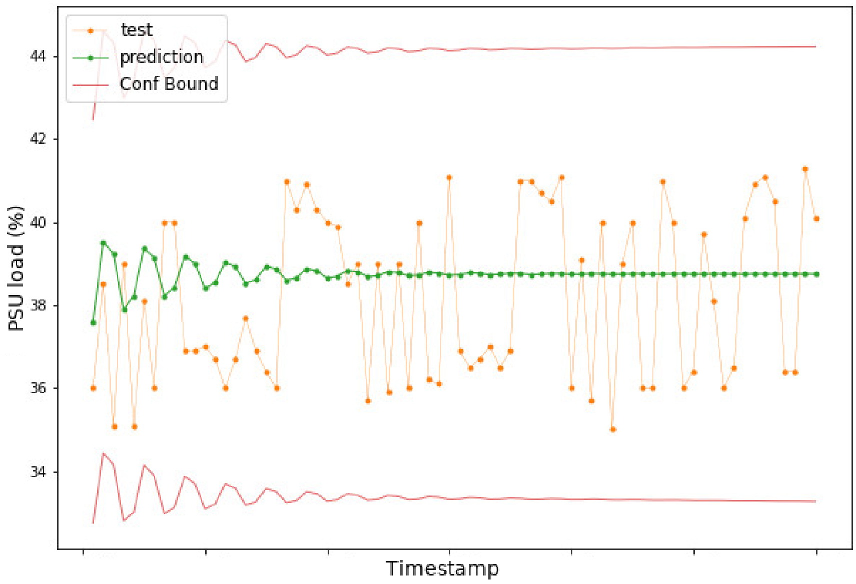


Figure A.3: Auto-ARIMA baseline predictions

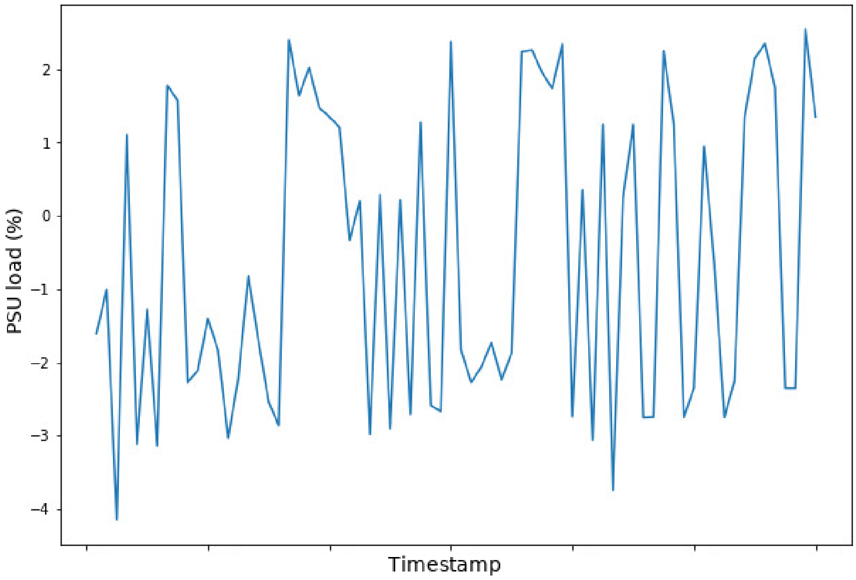


Figure A.4: Auto-ARIMA baseline residuals

A.2 Univariate Prophet

A.2.1 Training fitting

In training is obtained a very poor R^2 value, which from the plot in figure A.5 can be expected, since $\hat{y}$ looks like mostly an average of the seasonal components. Nonetheless, if it is considered that predicting an interval is acceptable, then the MOBE shows a very good result.

It can also be observed that the trend component did not overfit to the outliers.

MAE	2.20
R^2	0.11
MOBE	0.08

Table A.1: Univariate prophet training scores

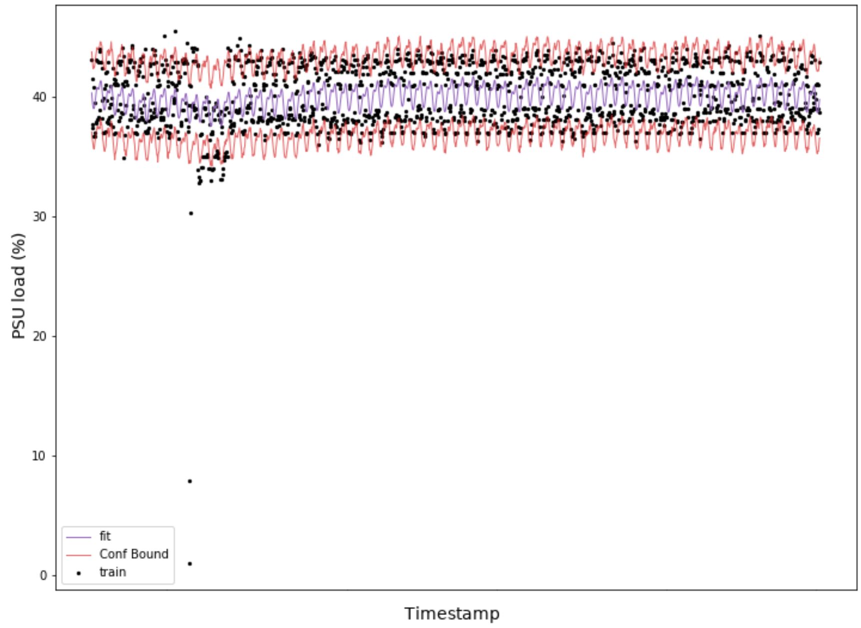


Figure A.5: Training fitting from univariate prophet

A.2.2 Test predictions

When it comes to the predictions performance, the R^2 shows even worse results. The MOBE is also high since the trend component was not able to foresee the decreasing in the middle of the test set as shown in Figure A.6.

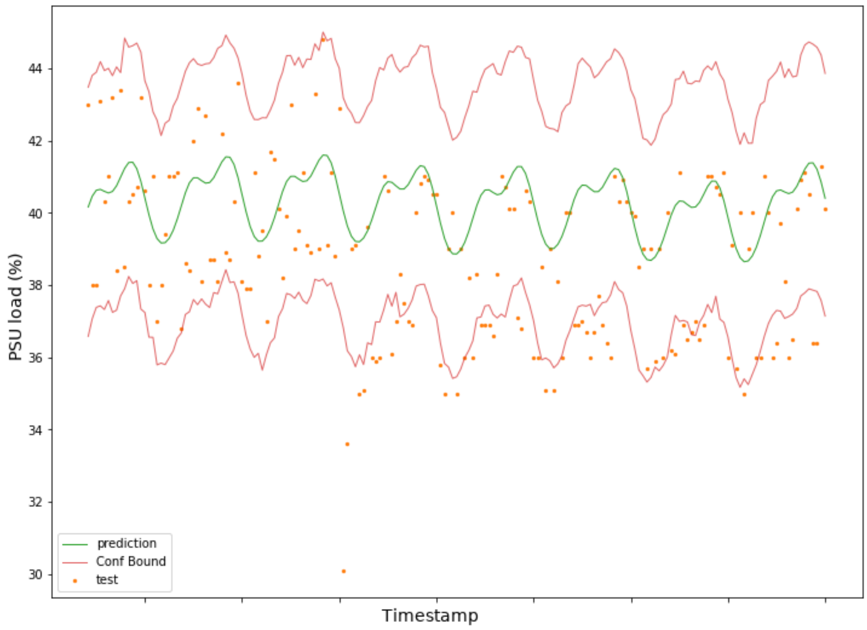


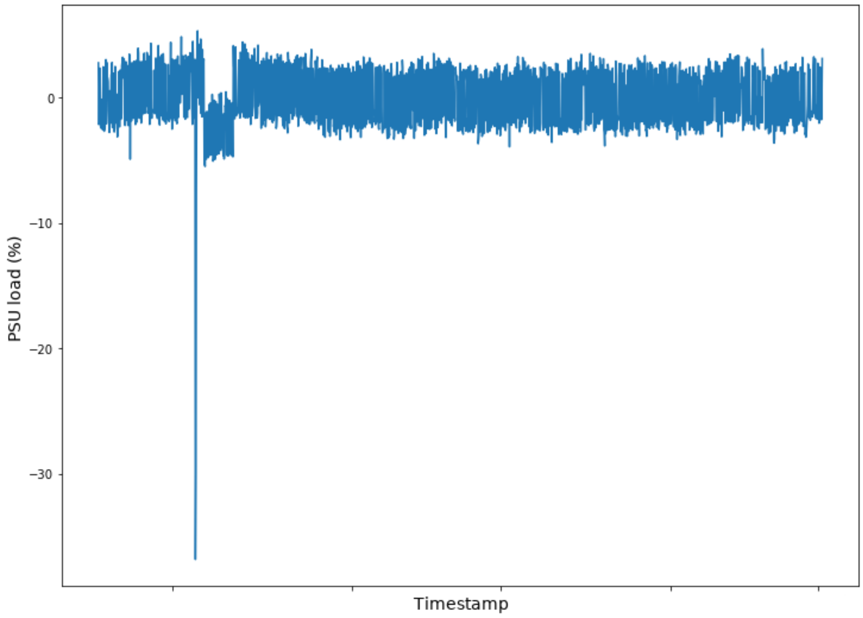
Figure A.6: Test predictions from the univariate Prophet model

MAE	2.12
R^2	-0.33
MOBE	0.25

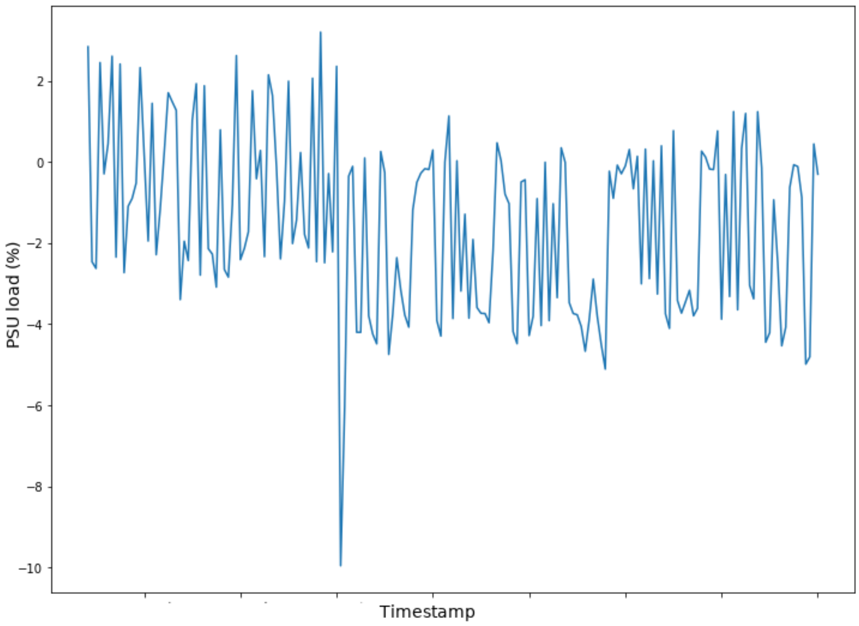
Table A.2: Univariate prophet prediction performance

A.2.3 Residual analysis

The residuals in the training set still show somehow resembles the original data pattern. This could be interpreted that there is still patterns to learn from data, i.e., this model is not complete.



(a) Residuals from train fit



(b) Residuals from test predictions

Figure A.7: Univariate prophet residuals

A.3 Multivariate Prophet

A.3.1 Test predictions

The results of the predictions have notoriously improved in this case. The R^2 value is close to 0.9, which for such a complex signal shows how powerful is the model.

The MOBE has even decreased in the test set, this could be due to training outliers that still being difficult to approximate without overfitting the trend component.

MAE	0.52
R^2	0.88
MOBE	0.05

Table A.3: Multivariate prophet prediction performance

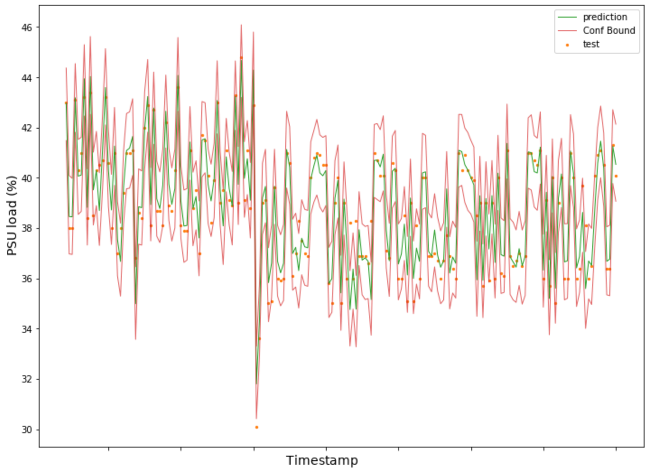
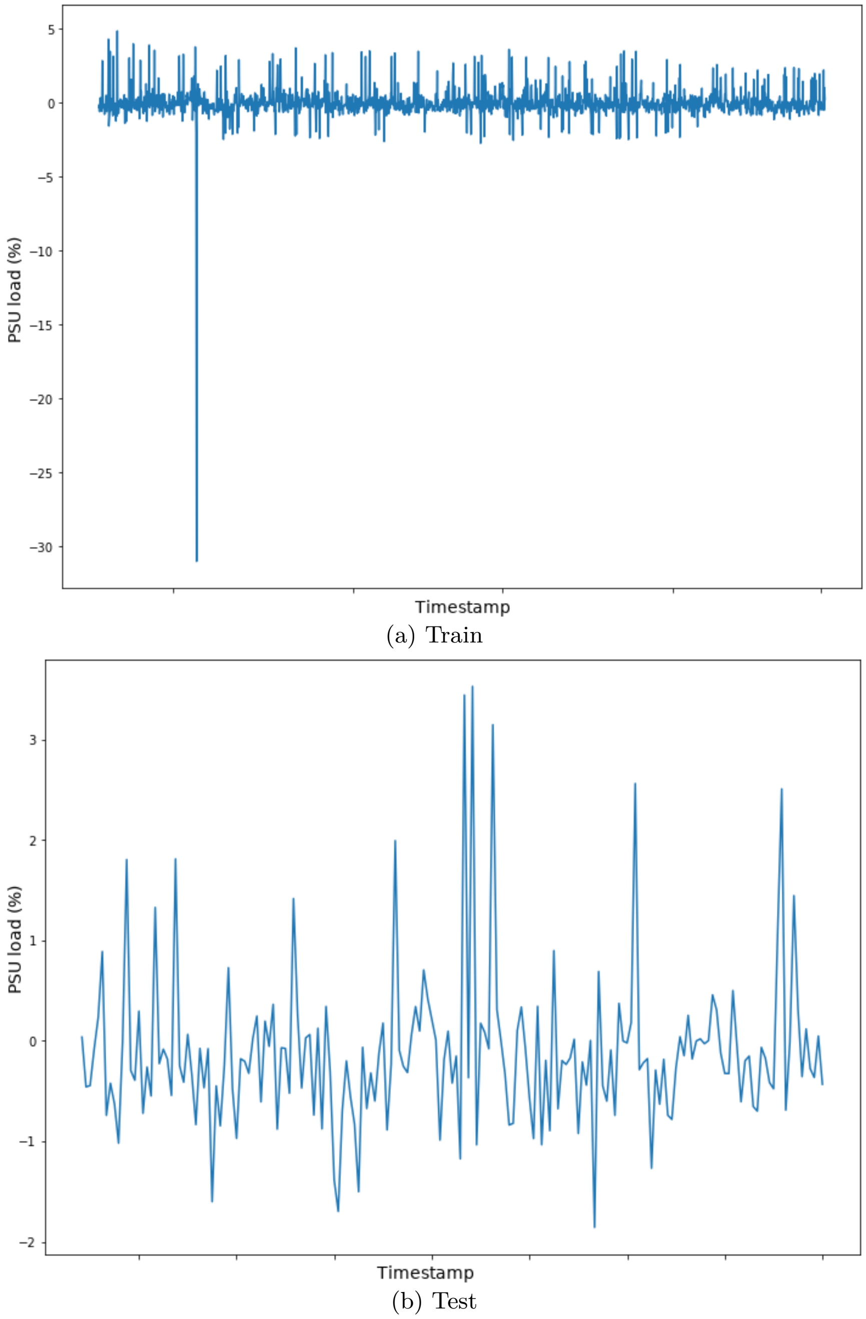


Figure A.8: Test predictions from multivariate prophet

A.3.2 Residual analysis

The residuals show that the patterns have been better learnt than in the univariate case.



(a) Train

(b) Test

Figure A.9: Multivariate prophet residuals

A.4 Exhaustive experiments

A.4.1 Baselines predictions moving horizon

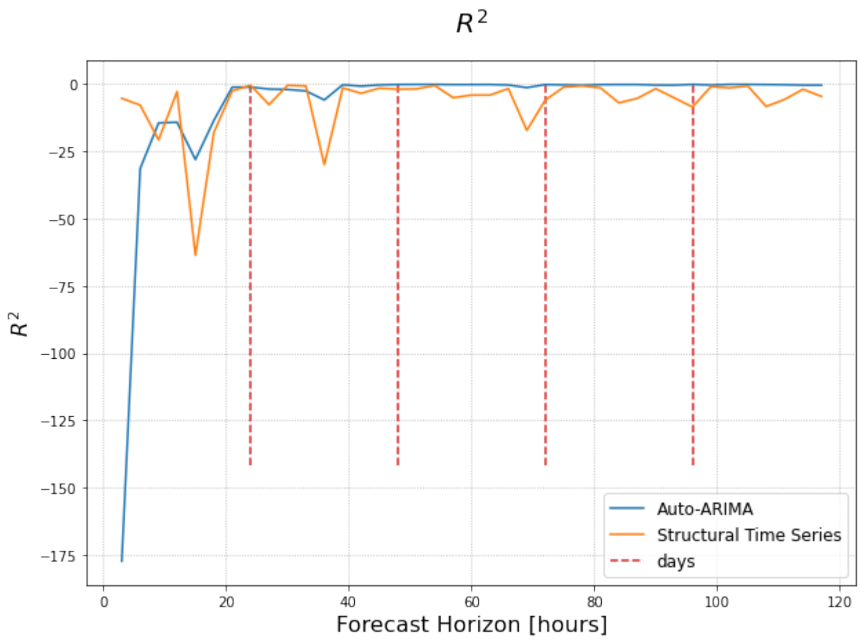


Figure A.10: Baseline R^2

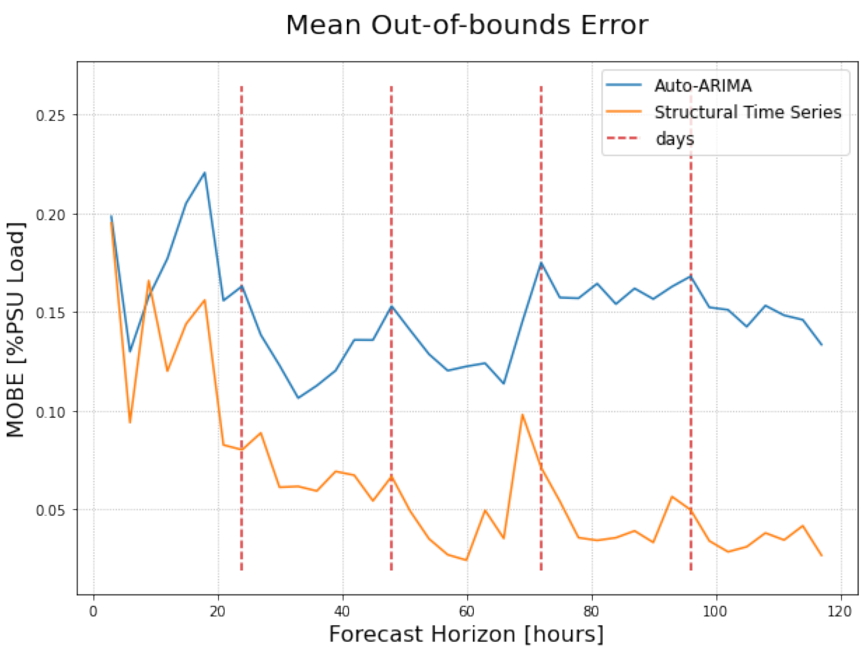


Figure A.11: Baseline MOBE

A.4.2 Univariate Prophet prediction moving horizon

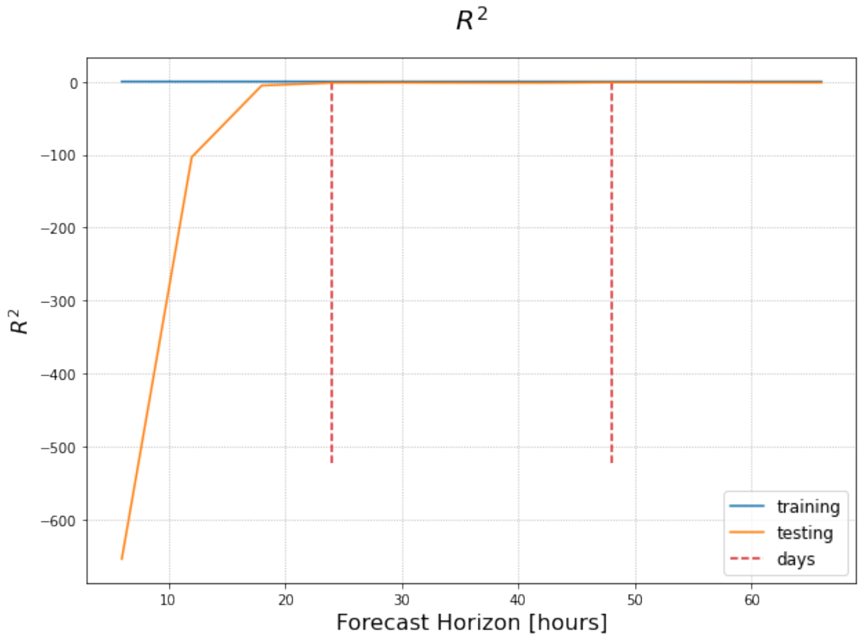


Figure A.12: Univariate moving prediction horizon: R^2

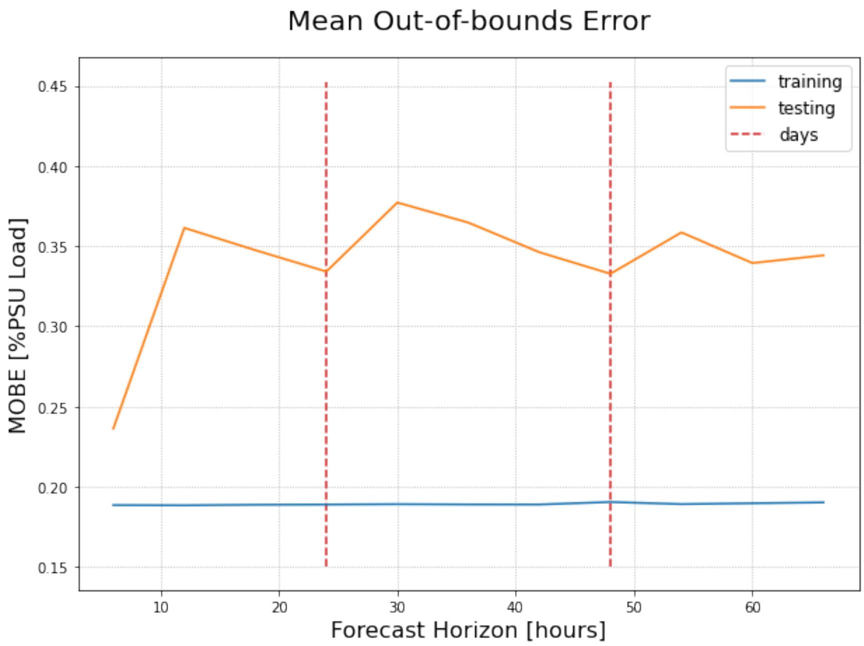


Figure A.13: Univariate moving prediction horizon: MOBE

A.4.3 Multivariate Prophet prediction moving horizon

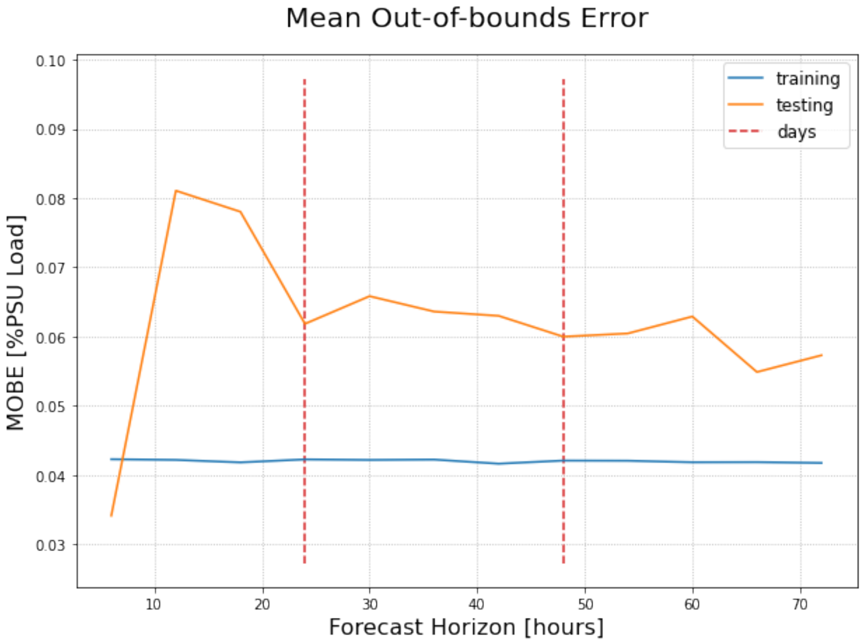


Figure A.14: Multivariate moving prediction horizon: MOBE

Bibliography

[1] Bilal Muhammad and Abbas Mohammed. "Uplink closed loop power control for LTE system." In: *2010 6th international conference on emerging technologies (ICET)*. IEEE. 2010, pp. 88–93.

[2] E-UTRA Physical Layer Procedures. "3GPP TS 36.213." In: *V11. 0.0, Sep* (2012).

[3] Oliver Arnold, Fred Richter, Gerhard Fettweis, and Oliver Blume. "Power consumption modeling of different base station types in heterogeneous cellular networks." In: *2010 Future Network & Mobile Summit*. IEEE. 2010, pp. 1–8.

[4] Bjorn Debaillie, Claude Desset, and Filip Louagie. "A flexible and future-proof power model for cellular base stations." In: *2015 IEEE 81st Vehicular Technology Conference (VTC Spring)*. IEEE. 2015, pp. 1–7.

[5] Alexandra Mourato, David Duarte, Iola Pinto, and Pedro Vieira. "A novel and realistic power consumption model for multi-technology radio networks." In: *URSI Radio Science Bulletin* 2018.364 (2018), pp. 20–29.

[6] Peter R Winters. "Forecasting sales by exponentially weighted moving averages." In: *Management science* 6.3 (1960), pp. 324–342.

[7] Pierpaolo Piunti, Simone Morosi, and Enrico Del Re. "Traffic Forecast and Power Consumption Management in Cellular Networks." In: *Traffic* 40.60 (), p. 80.

[8] Mounir Achir and Laurent Ouvry. "Power consumption prediction in wireless sensor networks." In: *Proceedings of the 16th ITC Specialist Seminar on Performance Evaluation of Wireless and Mobile Systems*. Citeseer. 2004.

[9] T Deepika and P Prakash. "Power consumption prediction in cloud data center using machine learning." In: *Int J Electr Comput Eng (IJECE)* 10.2 (2020), pp. 1524–1532.

[10] DV Antonenkov and DB Solovev. "Mathematic simulation of mining company's power demand forecast (by example of "Neryungri" coal strip mine)." In: *IOP Conference Series: Earth and Environmental Science*. Vol. 87. 3. IOP Publishing. 2017, p. 032003.

[11] Jingxiang Lv, Tao Peng, Yingfeng Zhang, and Yuchang Wang. "A novel method to forecast energy consumption of selective laser melting processes." In: *International Journal of Production Research* (2020), pp. 1–17.

[12] Abdulwahed Salam and Abdelaaziz El Hibaoui. "Comparison of Machine Learning Algorithms for the Power Consumption Prediction : - Case Study of Tetouan city –." In: *2018 6th International Renewable and Sustainable Energy Conference (IRSEC)*. 2018, pp. 1–5. DOI: 10.1109/IRSEC.2018.8703007.

[13] Yao Cheng, Chang Xu, Daisuke Mashima, Vrizlynn LL Thing, and Yongdong Wu. "PowerLSTM: power demand forecasting using long short-term memory neural network." In: *International Conference on Advanced Data Mining and Applications*. Springer. 2017, pp. 727–740.

[14] Songpu Ai, Antorweep Chakravorty, and Chunming Rong. "Household power demand prediction using evolutionary ensemble neural network pool with multiple network structures." In: *Sensors* 19.3 (2019), p. 721.

[15] Abdulla I Almazrouee, Abdullah M Almeshal, Abdulrahman S Almutairi, Mohammad R Alenezi, and Saleh N Alhajeri. "Long-Term Forecasting of Electrical Loads in Kuwait Using Prophet and Holt–Winters Models." In: *Applied Sciences* 10.16 (2020), p. 5627.

[16] RJ Chadalavada, S Raghavendra, and V Rekha. "Electricity requirement prediction using time series and Facebook's PROPHET." In: *Indian Journal of Science and Technology* 13.47 (2020), pp. 4631–4645.

[17] Yuan jiang Li, Yi Yang, Kai Zhu, and Jinglin Zhang. "Clothing Sale Forecasting by a Composite GRU-Prophet Model With an Attention Mechanism." In: *IEEE Transactions on Industrial Informatics* (2021).

[18] Donald B. Rubin. "Multiple Imputation after 18+ Years." In: *Journal of the American Statistical Association* 91.434 (1996), pp. 473–489. DOI: 10.1080/01621459.1996.10476908.

[19] Arthur P Dempster, Nan M Laird, and Donald B Rubin. "Maximum likelihood from incomplete data via the EM algorithm." In: *Journal of the Royal Statistical Society: Series B (Methodological)* 39.1 (1977), pp. 1–22.

[20] PM Vacek and T Ashikaga. "An examination of the nearest neighbor rule for imputing missing values." In: *Proc. Statist. Computing Sect., Amer. Statist. Ass* (1980), pp. 326–331.

[21] Barry L Ford. *An Overview of Hot-Deck Procedures: Incomplete Data in Sample Surveys, Vol. 2*. 1983.

[22] Peter J Brockwell, Peter J Brockwell, Richard A Davis, and Richard A Davis. *Introduction to time series and forecasting*. Springer, 2016.

[23] Agustin Maravall. "On structural time series models and the characterization of components." In: *Journal of Business & Economic Statistics* 3.4 (1985), pp. 350–355.

[24] James Durbin and Siem Jan Koopman. *Time Series Analysis by State Space Methods*. Aug. 2001.

[25] Sean J Taylor and Benjamin Letham. "Forecasting at scale." In: *PeerJ Preprints* 5 (Sept. 2017), e3190v2. ISSN: 2167-9843. DOI: 10.7287/peerj.preprints.3190v2. URL: https://doi.org/10.7287/peerj.preprints.3190v2.

[26] Andrew C Harvey and Simon Peters. "Estimation procedures for structural time series models." In: *Journal of forecasting* 9.2 (1990), pp. 89–108.

[27] Trevor Hastie and Robert Tibshirani. "Generalized additive models: some applications." In: *Journal of the American Statistical Association* 82.398 (1987), pp. 371–386.

[28] Everette S Gardner Jr. "Exponential smoothing: The state of the art." In: *Journal of forecasting* 4.1 (1985), pp. 1–28.

[29] George Evelyn Hutchinson. *An introduction to population ecology*. 504: 51 HUT. 1978.

[30] Andrew C Harvey and Neil Shephard. "10 Structural time series models." In: (1993).

[31] Maurizio Montel. *python-holidays*. https://github.com/dr-prodigy/python-holidays. 2021.

[32] Bob Carpenter, Andrew Gelman, Matthew D Hoffman, Daniel Lee, Ben Goodrich, Michael Betancourt, Marcus A Brubaker, Jiqiang Guo, Peter Li, and Allen Riddell. "Stan: a probabilistic programming language." In: *Grantee Submission* 76.1 (2017), pp. 1–32.

[33] Michael Betancourt. *A Conceptual Introduction to Hamiltonian Monte Carlo*. 2018. arXiv: 1701.02434 [stat.ME].

[34] Matthew D Hoffman and Andrew Gelman. "The No-U-Turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo." In: *J. Mach. Learn. Res.* 15.1 (2014), pp. 1593–1623.

[35] Radford M Neal et al. "MCMC using Hamiltonian dynamics." In: *Handbook of markov chain monte carlo* 2.11 (2011), p. 2.

[36] Michel A Cuendet and Wilfred F van Gunsteren. "On the calculation of velocity-dependent properties in molecular dynamics simulations using the leapfrog integration algorithm." In: *The Journal of chemical physics* 127.18 (2007), p. 184102.

[37] R. Fletcher. *Practical methods of optimization*. Chichester; New York: Wiley, 1987.

[38] Priti Mishra and Margaret H Eich. "Join processing in relational databases." In: *ACM Computing Surveys (CSUR)* 24.1 (1992), pp. 63–113.

[39] The pandas development team. *pandas-dev/pandas: Pandas*. Version latest. Feb. 2020. DOI: 10.5281/zenodo.3509134. URL: https://doi.org/10.5281/zenodo.3509134.

[40] Steffen Moritz, Alexis Sardá, Thomas Bartz-Beielstein, Martin Zaefferer, and Jörg Stork. "Comparison of different Methods for Univariate Time Series Imputation in R." In: *arXiv e-prints*, arXiv:1510.03924 (Oct. 2015), arXiv:1510.03924. arXiv: 1510.03924.

[41] Steffen Moritz and Thomas Bartz-Beielstein. "imputeTS: Time Series Missing Value Imputation in R." In: *The R Journal* 9.1 (2017), pp. 207–218. DOI: 10.32614/RJ-2017-009.

[42] Hirotogu Akaike. "Information Theory and an Extension of the Maximum Likelihood Principle." In: *Selected Papers of Hirotugu Akaike*. Ed. by Emanuel Parzen, Kunio Tanabe, and Genshiro Kitagawa. New York, NY: Springer New York, 1998, pp. 199–213. ISBN: 978-1-4612-1694-0. DOI: 10.1007/978-1-4612-1694-0_15. URL: https://doi.org/10.1007/978-1-4612-1694-0_15.

[43] Rob J. Hyndman and Yeasmin Khandakar. "Automatic Time Series Forecasting: The forecast Package for R." In: *Journal of Statistical Software, Articles* 27.3 (2008), pp. 1–22. ISSN: 1548-7660. DOI: 10.18637/jss.v027.i03. URL: https://www.jstatsoft.org/v027/i03.

[44] K Rantou. "Missing Data in Time Series and Imputation Methods." In: *University of the Aegean, Samos* (2017).

[45] Dennis D. Wackerly, William Mendenhall III, and Richard L. Scheaffer. *Mathematical Statistics with Applications*. sixth edition. Duxbury Advanced Series, 2002.

[46] Cort J Willmott and Kenji Matsuura. "Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance." In: *Climate research* 30.1 (2005), pp. 79–82.

[47] International Electrotechnical Comission. *Management of alarm systems for the process industries*. 2014.

[48] Iman Izadi, Sirish L Shah, David S Shook, and Tongwen Chen. "An introduction to alarm analysis and design." In: *IFAC Proceedings Volumes* 42.8 (2009), pp. 645–650.