

Online Voltage Prediction Using Gaussian Process Regression for Fault-Tolerant Photovoltaic Standalone Applications

Sanz-Alcaine, José Miguel; Sanz-Gorrachategui, Ivan; Pajovic, Milutin; Orlik, Philip V.

TR2022-038 May 19, 2022

Abstract

This paper presents a fault detection system for photovoltaic standalone applications based on Gaussian Process Regression (GPR). The installation is a communication repeater from the Confederación Hidrográfica del Ebro (CHE), public institution which manages the hydrographic system of Aragón, Spain. Therefore, fault-tolerance is a mandatory requirement, complex to fulfill since it depends on the meteorology, the state of the batteries and the power demand. To solve it, we propose an online voltage prediction solution where GPR is applied in a real and large dataset of two years to predict the behavior of the installation up to 48 hours. The dataset captures electrical and thermal measures of the lead-acid batteries which sustain the installation. In particular, the crucial aspect to avoid failures is to determine the voltage at the end of the night, so different GPR methods are studied. Firstly, the photovoltaic standalone installation is described, along with the dataset. Then, there is an overview of GPR, emphasizing in the key aspects to deal with real and large datasets. Besides, three online recursive multistep GPR model alternatives are tailored, justifying the selection of the hyperparameters: Regular GPR, Sparse GPR and Multiple Experts (ME) GPR. An exhaustive assessment is performed, validating the results with those obtained by Long ShortTerm Memory (LSTM) and Nonlinear Autoregressive Exogenous Model (NARX) networks. A maximum error of 127mV and 308mV at the end of the night with Sparse and ME respectively corroborates GPR as a promising tool.

Neural Computing and Applications 2022

Online Voltage Prediction Using Gaussian Process Regression for Fault-Tolerant Photovoltaic Standalone Applications

José Miguel Sanz-Alcaine¹ · Eduardo Sebastián² · Iván Sanz-Gorrachategui¹ · Carlos Bernal-Ruiz¹ · Antonio Bono-Nuez¹ · Milutin Pajovic³ · Philip V. Orlik⁴

Abstract

This paper presents a fault detection system for photovoltaic standalone applications based on Gaussian Process Regression (GPR). The installation is a communication repeater from the Confederación Hidrográfica del Ebro (CHE), public institution which manages the hydrographic system of Aragón, Spain. Therefore, fault-tolerance is a mandatory requirement, complex to fulfill since it depends on the meteorology, the state of the batteries and the power demand. To solve it, we propose an online voltage prediction solution where GPR is applied in a real and large dataset of two years to predict the behavior of the installation up to 48 hours. The dataset captures electrical and thermal measures of the lead-acid batteries which sustain the installation. In particular, the crucial aspect to avoid failures is to determine the voltage at the end of the night, so different GPR methods are studied. Firstly, the photovoltaic standalone installation is described, along with the dataset. Then, there is an overview of GPR, emphasizing in the key aspects to deal with real and large datasets. Besides, three online recursive multistep GPR model alternatives are tailored, justifying the selection of the hyperparameters: Regular GPR, Sparse GPR and Multiple Experts (ME) GPR. An exhaustive assessment is performed, validating the results with those obtained by Long Short-Term Memory (LSTM) and Nonlinear Autoregressive Exogenous Model (NARX) networks. A maximum error of 127mV and 308mV at the end of the night with Sparse and ME respectively corroborates GPR as a promising tool.

Keywords Battery management system, fault-tolerant system, gaussian process regression, photovoltaic standalone installation, sparse gaussian process regression, voltage prediction

1 Introduction

The progress in technological infrastructures and the awareness about environmental resource consumption are driving a revolution in how energy is supplied [1]. There is a need of not only providing stable energy but also accomplishing efficiency requirements and prediction capabilities to anticipate issues. This is especially important in critical services such as irrigation system [2],

telecommunication stations [3] and defense systems [4], where the system is not fed by the electrical distribution network but a Photo-Voltaic (PV) standalone installation. The energy in these systems is not consumed at the same rate at which is generated so the installation needs an Energy Storage System (ESS) to provide a stable power supply. This stability is subject to unpredictable meteorological phenomena, turning the efficient autonomy into a challenge. The solution involves designing an intelligent system capable of predicting future low power generation scenarios, preserving the infrastructures typically coordinated by low-cost embedded systems. In this sense, data-driven techniques are a must. Among others, GPR appears as a promising tool [5, 6] because of the absence of overfitting and the measurement of the uncertainty of the prediction. However, GPR requires a large computational effort, limiting its use in real-time applications. To cope with this, this paper analyses the performance of different GPR models using a real and large dataset, containing the waveforms of typically

✉ José Miguel Sanz-Alcaine, Eduardo Sebastián
jm_sanz@unizar.es; esebastian@unizar.es

¹ Department of Electronics Engineering and Communications, University of Zaragoza, Spain

² Department of Computer Science and Systems Engineering, University of Zaragoza, Spain

³ Analog Devices, Boston, USA. The author was with Mitsubishi Electric Research Laboratories, Cambridge, USA when this work was made.

⁴ Mitsubishi Electric Research Laboratories, Cambridge, USA.

measured variables in an ESS of a communication system (see Fig.1).

The problem of feature prediction in ESSs based on Battery Management Systems (BMSs) is a motivating topic. For instance, [7] works in predicting both State of Charge (SoC) and State of Health (SoH) by an Extended Kalman Filter (EKF) and an adaptive observer. A Probabilistic Neural Network is presented in [8] to predict the SoH. Following with the learning tools, [9] proposes a Support Vector Machine (SVM) model for regression. Nevertheless, due to the aforementioned advantages, GPR is a cornerstone in the state-of-the-art discussion. The authors in [10] apply Regular GPR to the prediction of the State-of-Charge (SoC) of a single battery. A similar approach is presented in [11], focused on the voltage prediction in a certain horizon applying a recursive GPR. Another interesting parameter to predict in BMSs is the SoH [12] and the Remaining Useful Live (RUL). In this regard, recent works like [13, 14] explore the use of GPR to determine the aging of Lithium-ion batteries and control their SoH under different life-cycle regimes. Furthermore, the authors of [15–17] have proved the convenience of GPR techniques to predict the health of Lithium-ion batteries under uncertain scenarios, modifying the structure of the underlining kernels to better consider empirical and electrochemical knowledge. Despite the great promise of the aforementioned instances, GPR is typically studied with artificial and/or informal datasets.

In contrast, the main contribution of this paper is the extension of GPR in BMS applications to a real, large, complex dataset, where the computational cost and memory use is crucial, and a significant level of unpredictability is involved. There are two critical parameters to predict: the evolution of the voltage during the day and the End Of Night (EON) voltage, the latter with especial impact to develop fault-tolerant systems. This study concerns a variety of GPR versions, kernels and hyperparameters, and compares the performance with LSTM and NARX techniques.

The outline of the paper is organized as follows. Section II introduces the dataset, detailing its main features. Then, Section III introduces GPR and describes the GPR models developed for this specific application. Section IV provides the results of an exhaustive evaluation of the models and compares with both an LSTM and NARX model. Finally, Section V includes the conclusions.

2 Dataset development

In this Section we present the real data set used, its main features and the context in which it has been obtained.



Fig. 1. PV installation in Sigena, region of Aragón, Spain.

The dataset comes from a PV standalone system which provides the energy supply to a communication installation. This installation is part of the hydrographic system of Aragón, Spain, which is managed by the CHE. The base station is located in one of the best Spanish irradiation environments in the Ebro basin (Sigena, Aragón). The standalone installation is powered by 24 lead-acid batteries (VRLA, Exide Classic Solar 2V 4600Ah C120) with nominal voltage of 2V in series and for a total nominal voltage of 48V and a capacity of 4600Ah, shown in Fig. 2. The installation, as it is seen in Fig. 1., has 24 solar panels (Scheuten multi 180P6), with 4680W of rated power. The equipment of the installation has a typical consumption of 350W, with occasional peaks up to 550W.

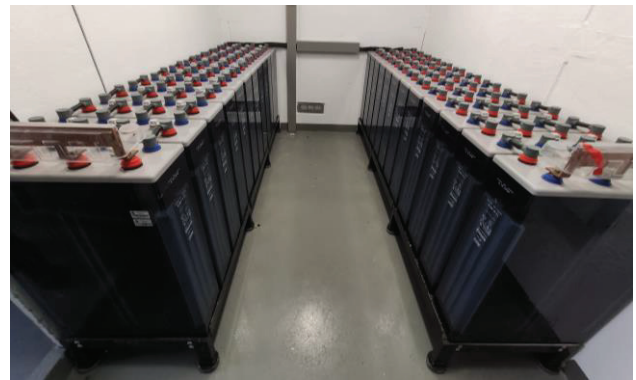


Fig. 2. Group of batteries of the PV standalone system.

The control system monitors the current, voltage and temperature waveforms of the batteries. PV Standalone system features irregular generation profiles that depend on weather and sun irradiation. They have three main generation-consumption dynamics: day-night cycles, cloudy periods (days or weeks), and the annual dynamic. Therefore, the reliability of the system is subject to

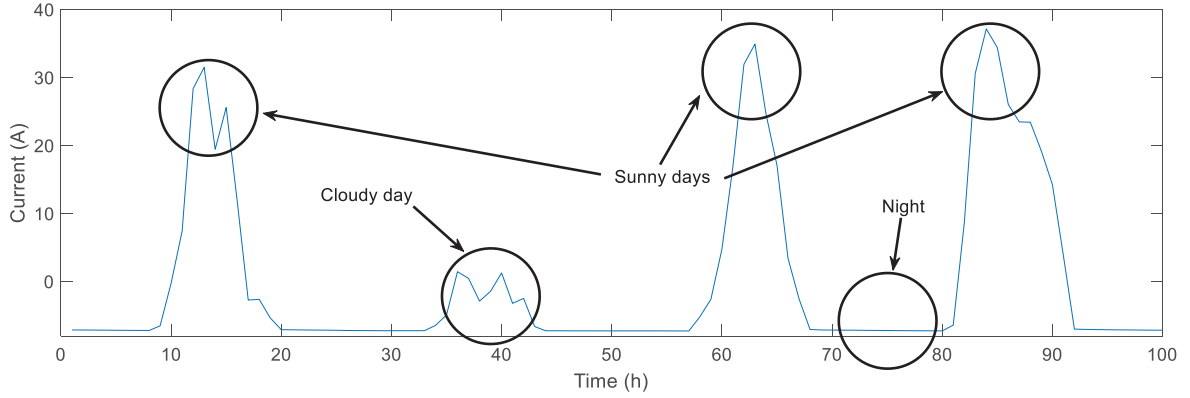


Fig. 3. Example of the current waveforms recorded in different weather scenarios.

significant uncertainties. In this application in particular, the prediction of a short-term system failure is especially interesting due to the expensive actions needed to restart the system. These critical system failures are mainly related to insufficient charge in the battery and could be detected by predicting an excessive low voltage level in the battery. A GPR based voltage predictor is a suitable option for this task, since it not only provides the prediction, but it also indicates the confidence interval of the prediction [18].

Data has been recorded during 2 years with a sampling rate of one sample every hour. Voltage, current and temperature waveforms are the information available and, as they are the typical variables in this type of installation, this work can be easily extended to similar systems. Below we include a sample instance of the structure and values of the dataset

Dataset	Voltage	→	{	Value	↗	e.g.	49.62 V
			Date	↘		01/01/2008 – 00:00	
	Current	→	{	Value	↗	e.g.	-7.2 A
			Date	↘		01/01/2008 – 00:00	
	Temperature	→	{	Value	↗	e.g.	6.3 °C
			Date	↘		e.g.	01/01/2008 – 00:00

Fig. 3 shows examples of current waveforms recorded in different weather scenarios, which manifest clear differences. According to the specifications of the client, i.e., the CHE, a failure prediction of 48 hours ahead is sufficient to avoid a critical failure, which is tackled by increasing the battery charge voltage level (floating voltage). As the consumption of the installation is nearly constant, the critical moment is the moment just before the sunrise, where the amount of charge is the lowest of the cycle.

Our data set is different from others due to the amount of real data available. In this sense, it is noteworthy that

there are two different phenomena in the database. The first is the periodical dynamics, which are predictable with an accurate model of a battery. The other is the random events associated with weather from which, due to its inherent quasi-chaotic nature, it is difficult to predict its impact in the batteries. The purely periodical dynamics are related to several phenomena: day/night, yearly and season-based dynamics. Thus, to account for all of them, the training set has been selected according to the greatest period, using a complete year of data. Thanks to the size of the database, tests are performed with data from the next year.

Fig. 4. illustrates the voltage waveforms recorded in the installation in 2008, from where it is clear that the floating voltage of the batteries is not constant. This is due to the temperature-regulated charge strategy of the solar regulator. Thus, the information encoded in that strategy provides important knowledge to our problem.

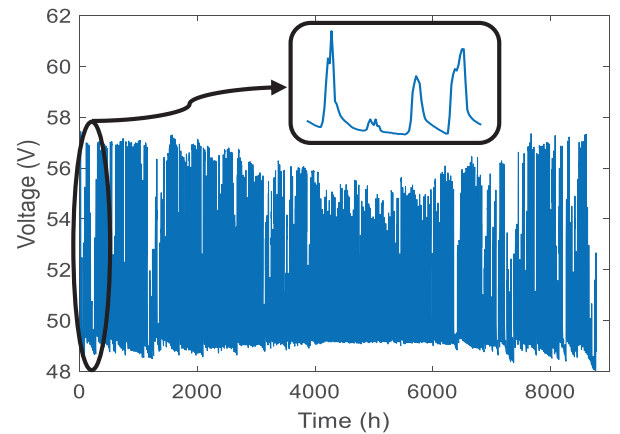


Fig. 4. Voltage waveforms from 2008.

To end up this Section, we introduce the metrics we will use along the paper to evaluate and study the different methods and hyperparameter selections. To evaluate the performance of the models, we choose the Root Mean

Square Error (RMSE) and the Maximum Absolute Error (MaxAE) as metrics, defining them as

$$RMSE(l) = \sqrt{\frac{1}{N_{test}} \sum_{t=1}^{N_{test}} (V_{t+l} - \hat{V}_{t+l})^2}, \quad (1)$$

$$MaxAE(l) = \max_{i=1, \dots, N_{test}} |V_{t+l} - \hat{V}_{t+l}|, \quad (2)$$

where N_{test} refers to the number of test samples and $l = 1, \dots, M$, where $M = 48$ hours for this work. In addition, we present two additional metrics which refer to RMSE and MaxAE at the end of the night

$$RMSE_{EON}(l) = \sqrt{\frac{1}{N_{test}} \sum_{t=1}^{N_{test}} (V_{t+l}^{EON} - \hat{V}_{t+l}^{EON})^2}, \quad (3)$$

$$MaxAE_{EON}(l) = \max_{i=1, \dots, N_{test}} |V_{t+l}^{EON} - \hat{V}_{t+l}^{EON}|. \quad (4)$$

This value of voltage is crucial in PV standalone systems as it gives the BMS the information of the voltage of the system if no more solar energy can be absorbed.

3 GPR models for large datasets

In this section we firstly present a brief review of the theory of GPR. After that, three versions of GPR (Regular GPR, Multiple Experts GPR and Sparse GPR) are detailed, connecting with the need of processing large quantities of data.

3.1 A brief introduction to theory

Most of the content of this subsection is adapted from [6] and it should be consulted by readers interested in further details about the mathematical background. Let $\mathcal{D} = (\mathbf{X}, \mathbf{y})$ denote a training dataset, comprising n D-dimensional input and scalar output pairs $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^n$, where $\mathbf{x}_i \in \mathbb{R}^D$, and the corresponding outputs $\mathbf{y} = \{y_i\}_{i=1}^n$, where $y_i \in \mathbb{R}$. It is assumed that there is an underlying nonlinear latent function $f(\cdot)$, which can be used to parameterize the probabilistic mapping between inputs and outputs

$$y_i = f(\mathbf{x}_i) + \varepsilon_i, \quad (5)$$

where ε_n denotes zero-mean additive Gaussian noise with variance σ_i^2 , i.e., $\varepsilon_i \sim \mathcal{N}(0, \sigma_i^2)$ and $\{\varepsilon_i\}_{i=1}^n$ form an independent and identically distributed (i.i.d) sequence. The GPR main hypothesis relies on assuming a priori that function values behave according to a multivariate Gaussian distribution

$$p(\mathbf{f}|\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n) = \mathcal{N}(\mathbf{0}, \mathbf{K}_{\mathbf{f}, \mathbf{f}}). \quad (6)$$

where, $\mathbf{f} = [f(\mathbf{x}_1), f(\mathbf{x}_2), \dots, f(\mathbf{x}_n)]^T$ is a vector of latent function values, and $\mathbf{0}$ is an $n \times 1$ vector whose elements are all 0. In addition, $\mathbf{K}_{\mathbf{f}, \mathbf{f}}$ is a covariance matrix, whose entries are given by the covariance function $\mathbf{K}_{i,j} = k(\mathbf{x}_i, \mathbf{x}_j)$, from now on named kernel function, evaluated at each pair of training inputs.

In GPR, the kernel function plays a major role, since it encodes the prior assumptions about the properties of the underlying latent function that we are trying to model. Indeed, one of the tasks of the designer is to select the kernel which best fits the phenomena modelled. An instance of kernel typically adopted is the Squared Exponential (SE) kernel, defined as

$$k_s(\mathbf{x}_i, \mathbf{x}_j) = \vartheta_0^2 \exp \left[-\frac{1}{2} \sum_{d=1}^D \left(\frac{x_{id} - x_{jd}}{l_d} \right)^2 \right], \quad (7)$$

where x_{id} and x_{jd} correspond to the d -th element of vectors \mathbf{x}_i and \mathbf{x}_j , respectively, and $\Theta = [\vartheta_0, l_1, \dots, l_D]^T$ denotes the hyperparameters. Distinctly, ϑ_0^2 denotes the signal variance and quantifies the variation of the underlying latent function from its mean, and l_D represents the characteristic length scale for each input dimension. Finally, l_D fixes the width of the kernel and thereby represents how smooth the functions in the model are.

In addition to the aforementioned kernel function, we incorporate the additive Gaussian white noise term into the selected kernel function

$$k(\mathbf{x}_i, \mathbf{x}_j) = k_s(\mathbf{x}_i, \mathbf{x}_j) + \sigma_i^2 \delta_{ij}, \quad (8)$$

where δ_{ij} denotes the Kronecker delta, which takes value of 1 if $i = j$ and 0 otherwise. Then, the distribution of \mathbf{y} , conditioned on the latent function values \mathbf{f} and the input \mathbf{X} , is given by

$$p(\mathbf{y}|\mathbf{f}, \mathbf{X}) = \mathcal{N}(\mathbf{f}, \sigma_i^2 \mathbf{I}), \quad (9)$$

where \mathbf{I} is an $n \times n$ identity matrix. Throughout integration over the latent function values \mathbf{f} and by using (6) and (9) the marginal distribution of \mathbf{y} can be obtained.

$$p(\mathbf{y}|\mathbf{X}, \Theta) = \int p(\mathbf{y}|\mathbf{f}, \mathbf{X}) p(\mathbf{f}|\mathbf{X}) d\mathbf{f} = \mathcal{N}(\mathbf{0}, \mathbf{K}_{\mathbf{f}, \mathbf{f}}) + \sigma_i^2 \mathbf{I}. \quad (10)$$

Then, the marginal log-likelihood, which refers to the marginalization over the function values \mathbf{y} can be written as

$$\begin{aligned} \log p(\mathbf{y}|\mathbf{X}, \theta) &= -\frac{1}{2}\mathbf{y}^T(\mathbf{K}_{f,f} + \sigma_i^2\mathbf{I})^{-1} \\ &\quad -\frac{1}{2}\log|\mathbf{K}_{f,f} + \sigma_i^2\mathbf{I}| \\ &\quad -\frac{n}{2}\log 2\pi, \end{aligned} \quad (11)$$

where $|\cdot|$ is the determinant of a matrix. The optimum hyperparameters $\hat{\theta}$ are found by maximizing the marginal log-likelihood. To do so, we use the partial derivatives of the marginal log-likelihood, from where we obtain that

$$\begin{aligned} \frac{\partial}{\partial \theta_i} \log p(\mathbf{y}|\mathbf{X}, \theta) &= \\ &-\frac{1}{2}\text{tr}\left(\left(\mathbf{K}_{f,f} + \sigma_i^2\mathbf{I}\right)^{-1} \frac{\partial(\mathbf{K}_{f,f} + \sigma_i^2\mathbf{I})}{\partial \theta_i}\right) \\ &+\frac{1}{2}\mathbf{y}^T(\mathbf{K}_{f,f} + \sigma_i^2\mathbf{I})^{-1} \frac{\partial(\mathbf{K}_{f,f} + \sigma_i^2\mathbf{I})}{\partial \theta_i} (\mathbf{K}_{f,f} + \sigma_i^2\mathbf{I})^{-1} \mathbf{y}. \end{aligned} \quad (12)$$

It is important to notice that the complexity of computing (11) is dominated by the inversion of $\mathbf{K}_{f,f} + \sigma_i^2\mathbf{I}$, which requires a computational time of $\mathcal{O}(n^3)$. Hence, a simple implementation of the GPR is advisable for datasets with up to a thousand training examples.

The characterization in (12) allows the use of any gradient based optimization method to optimize the marginal log-likelihood function (11). It is important to note that, generally, objective functions are nonconvex with respect to the hyperparameters which can lead to convergence to a local optimum. In order to tackle this problem, gradient-based optimization can be performed with different initial points, and the optimal hyperparameters that yield the largest marginal log-likelihood can be chosen. Once we obtain the optimal hyperparameters, it is possible to express the joint distribution of the training outputs \mathbf{y} and the test output y_* as

$$\begin{aligned} p(\mathbf{y}, y_*|\mathbf{X}, x_*, \theta) &= \mathcal{N}\left(\begin{bmatrix} \mathbf{0} \\ 0 \end{bmatrix}, \begin{bmatrix} \mathbf{K}_{f,f} + \sigma_i^2\mathbf{I} & \mathbf{K}_{*,f} \\ \mathbf{K}_{f,*} & K_{**} + \sigma_i^2 \end{bmatrix}\right), \end{aligned} \quad (13)$$

where the asterisk $*$ is used as a shorthand for f_* , which is the corresponding latent function value at the test input, $\mathbf{K}_{*,f} = [k(\mathbf{x}_1, \mathbf{x}_*), \dots, k(\mathbf{x}_n, \mathbf{x}_*)]^T$ and $K_{**} = k_s(\mathbf{x}_*, \mathbf{x}_*)$. The target of the GPR is to find the predictive distribution of the test output y_* which are conditioned on both the dataset \mathcal{D} and test input \mathbf{x}_* . Thus, it is possible to marginalize the joint distribution (13) over the training

dataset output \mathbf{y} and to obtain that the predictive distribution of the test output, y_* , is Gaussian distributed, i.e.,

$$p(y_*|\mathbf{X}, \mathbf{y}, \mathbf{x}_*, \theta) = \mathcal{N}(\mu_*, \Sigma_*), \quad (14)$$

where the mean and the covariance of the predictive distribution are given, respectively, in the following

$$\begin{aligned} \mu_* &= \mathbf{K}_{*,f}(\mathbf{K}_{f,f} + \sigma_i^2\mathbf{I})^{-1} \mathbf{y}, \\ \Sigma_* &= \sigma_i^2 + K_{**} - \mathbf{K}_{*,f}(\mathbf{K}_{f,f} + \sigma_i^2\mathbf{I})^{-1} \mathbf{K}_{f,*}. \end{aligned} \quad (15) \quad (16)$$

This mean, which is effectively the point estimate of the test output, is obtained as a linear combination of the noisy dataset outputs \mathbf{y} . Also, the variance of the predictive distribution, Σ_* in (16), serves as a measure of the uncertainty in the estimate of the test output. After performing the inversion of $\mathbf{K}_{f,f} + \sigma_i^2\mathbf{I}$, the computational complexity of the testing stage is $\mathcal{O}(n)$ and $\mathcal{O}(n^2)$ for the mean μ_* and the variance Σ_* respectively, which makes the proposed method highly appropriate for online operation. As a remark, notice that the computation of the inverse of $\mathbf{K}_{f,f} + \sigma_i^2\mathbf{I}$ can be speeded up to improve the computational burden of the online regression. Since $\mathbf{K}_{f,f}$ does not change after training, we can compute $\mathbf{K}_{f,f}^{-1}$ and then use the matrix inversion lemma (Chapter 8 of [6]) to calculate $(\mathbf{K}_{f,f} + \sigma_i^2\mathbf{I})^{-1}$ by simply computing online $(\sigma_i^2\mathbf{I})^{-1}$.

Fig. 5 shows a first approximation to our prediction problem. All the typical data available in the installation is fed (voltage, temperature and current). However, at first sight, it is not possible to determine the relevance and influence of this data in the prediction, so an extensive discussion and justification of which type of information is really needed is discussed in Section IV.



Fig. 5. Schematic block of Regular GPR.

The output of the GPR model is an estimation of the future sample of voltage data, V_{k+1} , predicted from $L+1$ past samples of voltage V_k, \dots, V_{k-L} , current I_k, \dots, I_{k-L} , and temperature T_k, \dots, T_{k-L} ; and future current I_{k+1} and temperature T_{k+1} samples. Thus, L is the memory length of the number of previous samples processed. Since the power demand of the installation is known, we can feed the GPR models with future current and temperature

Algorithm 1: Training stage of the one step predictor

- (1) **Input:** Dataset $\mathcal{D}\{(\mathbf{x}_{k_1}, y_{k_1}), \dots, (\mathbf{x}_{k_n}, y_{k_n})\}$
- (2) **Input:** Kernel function $k(\cdot, \cdot)$ parametrized by Θ
- (3) Run iterative optimization to obtain $\hat{\Theta}$
- (4) Evaluate inverse covariance matrix $\mathbf{K}_{\mathbf{f}, \mathbf{f}}^{-1}$
- (5) **Output:** learnt hyperparameters $\hat{\Theta}$
- (6) **Output:** inverse covariance matrix $\mathbf{K}_{\mathbf{f}, \mathbf{f}}^{-1}$

Algorithm 2: Direct estimation stage of the one-sample predictor at instant k

- (1) **Input:** Dataset $\mathcal{D}\{(\mathbf{x}_{k_1}, y_{k_1}), \dots, (\mathbf{x}_{k_n}, y_{k_n})\}$
- (2) **Input:** Kernel function $k(\cdot, \cdot)$ and hyperparameters $\hat{\Theta}$
- (3) **Input:** Inverse covariance matrix $\mathbf{K}_{\mathbf{f}, \mathbf{f}}^{-1}$
- (4) **Input:** Measurements
 $V_k I_k T_k V_{k-1} I_{k-1} T_{k-1} \dots V_{k-L} I_{k-L} T_{k-L}$
- (5) **Input:** Future current and temperature $I_{k+1} T_{k+1}$
- (6) $\mathbf{y} = [y_{k_1}, y_{k_2}, \dots, y_{k_n}]^T$
- (7) Format \mathbf{x}_{*k} as in (17)
- (8) $\mathbf{K}_{*, \mathbf{f}} = k(\mathbf{x}_{*k}, \mathbf{x}_{k_i}), i = 1, \dots, n$
- (9) $K_{**} = k_S(\mathbf{x}_{*k}, \mathbf{x}_{*k})$
- (10) μ_{*k} and Σ_{*k} from Eqs. (15,16)
- (11) **Output:** Predicted mean μ_{*k} and variance Σ_{*k}

waveforms. This data is assembled in a vector \mathbf{x}_{*k} , of length $3(L+1)+2$, formatted as:

$$\mathbf{x}_{*k} = [I_{k+1} T_{k+1} V_k I_k T_k V_{k-1} I_{k-1} \dots T_{k-L} \dots V_{k-L} I_{k-L}]^T \quad (17)$$

The corresponding mean and variance output from the GPR, given \mathbf{x}_{*k} , is denoted as μ_{*k} (from Eq. (15)) and Σ_{*k} (from Eq. (16)) respectively, using the sub-index k to emphasize that they are the estimations at instant k . The output y_k of the algorithm is the voltage at the sample $k+1$

$$y_k = \mu_{*k} = V_{k+1}. \quad (18)$$

GPR techniques consist in two processes. In the first part, GPR is trained offline to learn the relationship between the inputs \mathbf{x}_{k_i} and outputs \mathbf{y} . Then, optimal values of the hyperparameters of the chosen kernel are determined through a conjugate gradient method based on a training dataset, \mathcal{D} . In the second part, online voltage prediction of the battery is performed based on voltage, current, and temperature measurements of the battery. The mean μ_{*k} of the predicted distribution represents the estimated voltage V_{k+1} . Additionally, the standard deviation ($\Sigma_{*k}^{1/2}$) of the predicted distribution permits us to build a confidence interval to measure the accuracy of the prediction. In this paper, we consider a typical 95% confidence interval [6]

$$[V_{k+1} - 1.96\Sigma_{*k}^{1/2}, V_{k+1} + 1.96\Sigma_{*k}^{1/2}]. \quad (19)$$

The designer can choose the level of confidence interval depending on the particularities of applications. As standard deviation decreases, the confidence interval becomes smaller, and it indicates a more accurate prediction. GPR techniques can be used to directly provide the prediction of a variable one sample ahead. In this sense, Algorithms 1 and 2 show the training and estimations stages for the one step predictor. In this model, this means 1 hour in advance. However, as a 48 hours prediction is pursued, the GPR versions in this work are implemented in a recursive multi-step form, so recursion will be assumed henceforth.

3.2 Regular GPR

The scope of this method is to obtain the estimated voltage corresponding to $k+z$ time, until reaching the desired $M=48$ hours prediction for our purposes. This means that the predicted output of the method corresponds to the expected voltage at time instant $k+z+1$. Due to the recursive structure, in the first step, what it is estimated is the mean $\mu_{*k}^{(1)}$ calculated through (15). This value is then fed back along with voltage, current and temperature at time instants $k, k-1, \dots, k-L-1$, and current and temperature $I_{k+1}, T_{k+1}, I_{k+2}, T_{k+2}$ in a one-step prediction algorithm to obtain $\mu_{*k}^{(2)}$. This process is repeated until we obtain the 48 step-ahead voltage prediction. Regarding the implementation, the training process is the same treated in Algorithm 1. However, in the testing stage vectors are formatted as

$$\mathbf{x}_{*k}^{(z)} = [I_{k+m} T_{k+m} \mathbf{V}_k^{(z)} \mathbf{I}_k^{(z)} \mathbf{T}_k^{(z)}]^T, \quad (20)$$

with

$$\mathbf{V}_k^{(z)} = [\mu_{*k}^{(z-1)} \cdots \mu_{*k}^{(1)} V_k \cdots V_{k-L+z-1}]^T \quad (21)$$

$$\mathbf{I}_k^{(z)} = [I_{k+m-1} \cdots I_{k+1} I_k \cdots I_{k-L+z-1}]^T \quad (22)$$

$$\mathbf{T}_k^{(z)} = [T_{k+m-1} \cdots T_{k+1} T_k \cdots T_{k-L+z-1}]^T \quad (23)$$

A flow chart of the operational stage of this estimation method is shown in Algorithm 3.

Algorithm 3: Estimation stage of the Regular Recurrent GPR based predictor

- (1) **Input:** Dataset $\mathcal{D}\{(\mathbf{x}_{k_1}, y_{k_1}), \dots, (\mathbf{x}_{k_n}, y_{k_n})\}$
 - (2) **Input:** Kernel function $k(\cdot, \cdot)$ and hyper-parameters $\hat{\Theta}$
 - (3) **Input:** Inverse covariance matrix $\mathbf{K}_{f,f}^{-1}$
 - (4) **Input:** Measurements
 $V_k I_k T_k V_{k-1} I_{k-1} T_{k-1} \cdots V_{k-L} I_{k-L} T_{k-L}$
 - (5) **Input:** Future current and temperature $I_{k+1} T_{k+1}$
 - (6) $\mathbf{y} = [y_{k_1}, y_{k_2}, \dots, y_{k_n}]^T$
 - (7) **for** $z = 1: M$ **do**
 - (8) Format $\mathbf{x}_{*k}^{(z)}$ as in (20)
 - (9) $\mathbf{K}_{*,f} = k(\mathbf{x}_{*k}^{(z)}, \mathbf{x}_{k_i})$, $i = 1, \dots, n$
 - (10) $K_{**} = k(\mathbf{x}_{*k}^{(z)}, \mathbf{x}_{*k}^{(z)})$
 - (11) Eq (15,16)
 - (12) **end for**
 - (13) **Output:** Predicted voltages $\hat{V}_{k+z} = \mu_{*k}^{(z)}$ and variance $\Sigma_{*k}^{(z)}$, $z = 1, \dots, M$
-

3.2.1 Train/test dataset selection

The GPR model is trained with 720 samples of data, i.e., a complete month. To capture all kind of phenomena, the 720 samples come from an equally spaced selection of the original training dataset (the complete year of 2008). In each selection, a complete day is captured in to preserve the temporary properties and waveform features, so the training set can be seen as a cascade of days from different seasons and weather conditions. This set is the input of the training algorithm (Algorithm 1). For testing, the complete month of March 2009 has been used, selected due to the fact that it involves all kinds of uncertain phenomena (both

sunny and cloudy periods) since it belongs to a low irradiation season.

3.2.2 Memory length selection

Prior to applying Algorithms 1 and 3, Memory Length (L) must be established. The selection of L is relevant for the final deployment of the algorithm because of its impact in the computational cost. Indeed, we recall that the focus of this paper is to validate GPR techniques in real applications with large quantities of data and evaluate different GPR-based techniques. The final selection of L should be made considering the available computational resources.

A priori, a larger L could provide better results because of more data is involved in each prediction. The results of an exhaustive preliminary test with a Regular GPR and several L values are shown in Fig. 6., where RMSE over the predicted voltage is represented to track the performance of the model with respect to L . As there is not a clear optimized value of L we opted for a technical selection of $L = 15$ samples. This value is a compromise between the computational cost and the accuracy of the calculations.

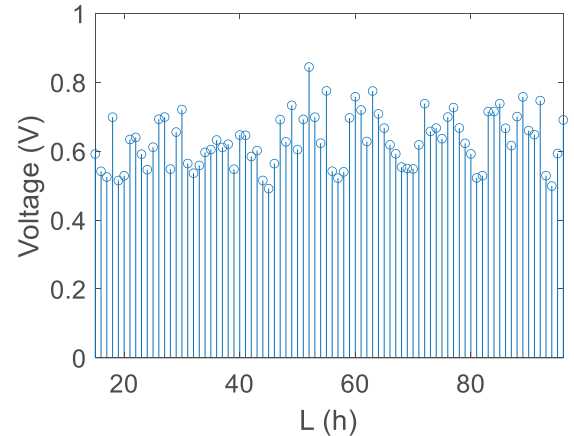


Fig. 6. Evolution of the error with the memory length.

3.2.3 Kernel function selection

The selection of the kernel used in the GPR model is also required. Conceptually, the kernel is the tool that captures the relationships within the training data. In this application, it is especially difficult to detect these hidden relations because of the 48 hours recurrent approach. In non-recurrent previous related works [10, 11] SE kernel exhibits good performances. However, in this work we make an extensive test of different kernels, as proposed in [19]. Table 1 describes the compared kernels. According to the results over the RMSE and MaxAE, the kernel that

gives a better performance for our database is the Rational Quadratic (RQ) [20]. Table 2 shows the best results of the tests, where it can be appreciated that more complex kernels do not improve the performance whilst computational cost is increased. In all cases, Automatic Relevance Determination is used.

3.3 GPR based Multiple Experts

In this Section a modification of the Regular GPR is proposed. The idea is to model different types of canonical current waveforms (related to specific daily dynamics

during a year), or classes. Several GPR-based “experts” are designed, each one related to a class. A non-supervised clustering algorithm is used for a more accurate classification. Self-Organizing Maps (SOM) [21–23] is the selected technique to classify training data in a visual map. To characterize a day, five features are chosen

- Voltage at the beginning of the discharge.
- Voltage at the end of the discharge (at the sunrise of the day).
- Duration in hours of the charging process.
- Hours of the discharge.
- Charge stored.

After training a 10x6 SOM by using the SOM Toolbox [24] MATLAB® library, every input pattern (each day) is projected to its closest neuron, in such a way that nearby neurons are tuned to similar patterns (similar days). The resulting SOM can be considered as a two-dimensional picture of the database, where clusters of similar days could be identified. After that, a K-means clustering algorithm [25, 26] has been applied to the outputs of the SOM to establish the different classes obtained. K-means needs the number of clusters to make the separation, so for establishing this number of clusters the algorithm has been applied several times changing this parameter.

The results show that using more than five classes does not provide relevant information. Fig. 7. is the final result of this process. Each sample is labelled regarding its characteristics in the following classes: low irradiation days, winter days with high SoC, winter days with low SoC, spring days, and summer days.

For a better understanding of the differences among the type of days, Fig. 8., shows the mean current waveform of each data related to each type of day. With this tool all the days of the original training database are classified and tagged. For further details, [27] is included as a pointer. Afterwards, five GPR experts are designed and trained, using for training days of their corresponding class.

The same parameters obtained with Regular GPR (train size, test set, memory length of 15 hours, RQ kernel) are used in the five experts. In the inference stage, after each expert estimates the future sample, one of the experts is chosen and its output is reinjected for the next iteration. As the confidence interval calculated by the GPR provides information about the accuracy of the estimation, the expert with the most accurate confidence interval is selected as the one to provide the final output.

3.4 Sparse GPR

Sparse GPR [5, 28] is a modification of the Regular GPR to be able to process large datasets. For larger datasets, sparsity solves the problem of the computational cost. The computational cost of a regular GPR is reduced by introducing inducing variables and modifying the joint prior distribution, $p(f_*, \mathbf{f})$. Let $\mathbf{u} = [u_0, \dots, u_m]^T$ denote

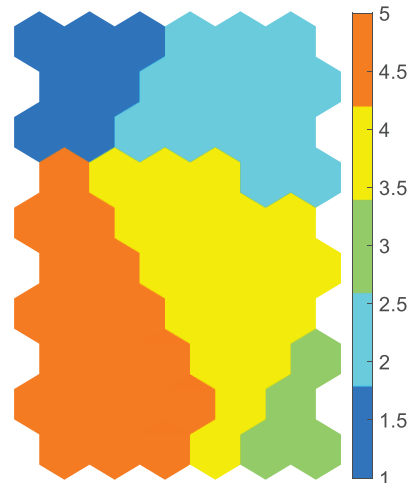


Fig. 7. Results of K-means applied to SOM.

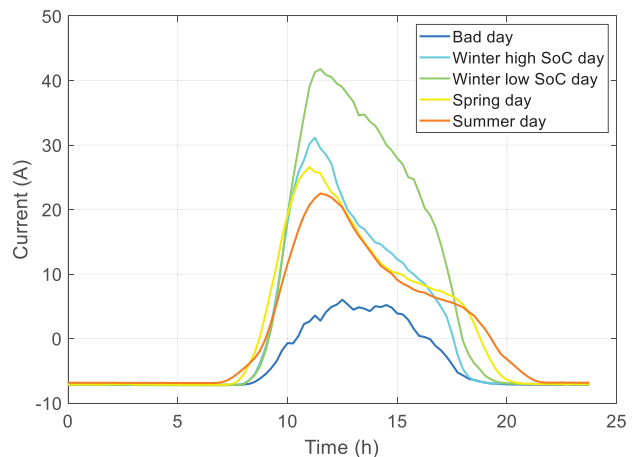


Fig. 8. Mean waveforms of the five classes of days from SOM.

Table 1 List of kernels used in the comparison

SEard	$k_s(\mathbf{x}_i, \mathbf{x}_j) = \vartheta_0^2 \exp \left[-\frac{1}{2} \sum_{d=1}^D \left(\frac{x_{id} - x_{jd}}{l_d} \right)^2 \right]$	ϑ_0^2 is the signal variance, l_d is the characteristic length scale
RQard	$k_s(\mathbf{x}_i, \mathbf{x}_j) = \vartheta_1^2 \left(1 + \frac{1}{2\beta} \sum_{d=1}^D \left(\frac{x_{id} - x_{jd}}{\mu_d} \right)^2 \right)^{-\beta}$	ϑ_1^2 is the signal variance, β is a smoothness parameter, μ_d is the characteristic length scale
Maternard	$k_s(\mathbf{x}_i, \mathbf{x}_j) = \vartheta_2^2 \frac{1}{\Gamma(\nu)2^{\nu-1}} \left[\sqrt{2\nu} \sum_{d=1}^D \left(\frac{x_{id} - x_{jd}}{\rho_d} \right) \right]^\nu K_\nu \left(\sqrt{2\nu} \sum_{d=1}^D \left(\frac{x_{id} - x_{jd}}{\rho_d} \right) \right)$	ϑ_2^2 is the signal variance, ν is a smoothness parameter, ρ_d is the characteristic length scale, $\Gamma(\cdot)$ is the Gamma function, $K_\nu(\cdot)$ is the modified Bessel function of the second kind
NN	$k_s(\mathbf{x}_i, \mathbf{x}_j) = \vartheta_3^2 \arcsin \frac{\sigma_p^2(1 + \mathbf{x}_{id}^T \mathbf{x}_{jd})}{\sqrt{(1 + \sigma_p^2 + \sigma_p^2 \mathbf{x}_{id}^T \mathbf{x}_{id})(1 + \sigma_p^2 + \sigma_p^2 \mathbf{x}_{jd}^T \mathbf{x}_{jd})}}$	ϑ_3^2 is the signal variance, σ_p^2 is the scaling factor

Table 2 Comparative of kernels using Regular GPR

	RQard	SEard	SEard+SEard+NN [11]	Maternard
RMSE (V)	0.592	3.315	1.699	3.071
MaxAE (V)	1.596	1.635	1.919	2.045
Train time (s)	13.2	13.4	31.6	10.7
Test time (s)	51.0	50.1	369.2	67.6

the inducing variables which correspond to a set of input locations $X_{\mathbf{u}}$ called inducing points, where $m \leq n$ is the number of inducing points. The inducing points are chosen as a subset of the data points. Given the inducing points, the joint prior distribution, $p(f_*, \mathbf{f})$ can be rewritten as

$$p(f_*, \mathbf{f}) = \int p(f_*, \mathbf{f}|\mathbf{u})p(\mathbf{u})d\mathbf{u}, \quad (25)$$

where $p(\mathbf{u}) = \mathcal{N}(\mathbf{0}, K_{\mathbf{u},\mathbf{u}})$. For the approximation of $p(f_*, \mathbf{f})$, it is assumed that f_* and \mathbf{f} are conditionally independent given \mathbf{u} in the following [5]

$$p(f_*, \mathbf{f}) \approx q(f_*, \mathbf{f}) = \int q(f_*|\mathbf{u})q(\mathbf{f}|\mathbf{u})p(\mathbf{u})d\mathbf{u}. \quad (26)$$

Besides, it is assumed that the training conditional $q(\mathbf{f}|\mathbf{u})$ is fully independent and the test conditional keeps exact as

$$q(\mathbf{f}|\mathbf{u}) = \prod_{i=1}^n p(f_n|\mathbf{u}) = \mathcal{N}(K_{\mathbf{f},\mathbf{u}}, K_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u}, \text{diag}[K_{\mathbf{f},\mathbf{f}} - Q_{\mathbf{f},\mathbf{f}}]), \quad (27)$$

$$q(f_*|\mathbf{u}) = p(f_*|\mathbf{u}), \quad (28)$$

where $\text{diag}[A]$ denotes the diagonal matrix in which the entries outside the main diagonal are all zero. The distributions in (27) and (28) can be substituted in (26), so that integrating over \mathbf{u} gives the joint prior

$$q(\mathbf{f}, f_*) = \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} Q_{\mathbf{f},\mathbf{f}} - \text{diag}[Q_{\mathbf{f},\mathbf{f}} - K_{\mathbf{f},\mathbf{f}}] & Q_{\mathbf{f},*} \\ Q_{*,\mathbf{f}} & K_{*,*} \end{bmatrix}\right), \quad (29)$$

where $Q_{\mathbf{a},\mathbf{b}} = K_{\mathbf{a},\mathbf{u}}K_{\mathbf{u},\mathbf{u}}^{-1}K_{\mathbf{u},\mathbf{b}}$ is a low-rank matrix (i.e., rank m). The predictive distribution can be obtained by using the above joint prior distribution,

$$q(y_*|\mathbf{X}, \mathbf{y}, \mathbf{x}_*, \Theta) = \mathcal{N}(\tilde{\mu}_*, \tilde{\Sigma}_*), \quad (30)$$

where

$$\tilde{\mu}_* = K_{*,\mathbf{u}}\Omega K_{\mathbf{u},\mathbf{f}}\Lambda^{-1}\mathbf{y}, \quad (31)$$

$$\tilde{\Sigma}_* = \sigma_i^2 + K_{*,*} - Q_{*,*} + K_{*,\mathbf{u}}\Omega K_{\mathbf{u},*}. \quad (32)$$

Here, $\Omega = (K_{\mathbf{u},\mathbf{u}} + K_{\mathbf{u},\mathbf{f}}\Lambda^{-1}K_{\mathbf{f},\mathbf{u}})^{-1}$ and $\Lambda = \text{diag}[K_{\mathbf{f},\mathbf{f}} - Q_{\mathbf{f},\mathbf{f}} + \sigma_i^2\mathbf{I}]$. Notice that the only matrix requiring inversion is the $n \times n$ diagonal matrix Λ , which translates into a significant reduction in computational complexity. The computational cost of training becomes $\mathcal{O}(m^2n)$, i.e., linear in n and quadratic in m . Larger values of m should yield to better accuracy at the expense of an increase in the

computational requirements. Additionally, the complexity of the testing stages is $\mathcal{O}(m)$ and $\mathcal{O}(m^2)$ for calculating the mean and the variance, respectively.

Among many different kinds of sparsity algorithms [5, 29], Fully Independent Training Conditional (FITC) seems to get the best performance [30] and, therefore, it is the one used in this work. Regarding the selection of m , several simulations have been made with an increasing number of inducing points. Fig. 9. shows that there is almost no need of increasing inducing points beyond 80 for our database (RMSE is beyond that point), so $m = 80$ is selected.

There are several methods to select the position of the inducing points (like the powerful, but time consuming, Farthest Point Clustering (FPC) [31]). However, the most typical approximation is to select equally spaced points because of the low computational cost involved. After that, this subset of data points is also optimized when included in the training stage, along with the other hyperparameters, with the gradient descent algorithm. The other parameters are the same as in the other models. The iterative process of Regular GPR is also carried on with the Sparse GPR.

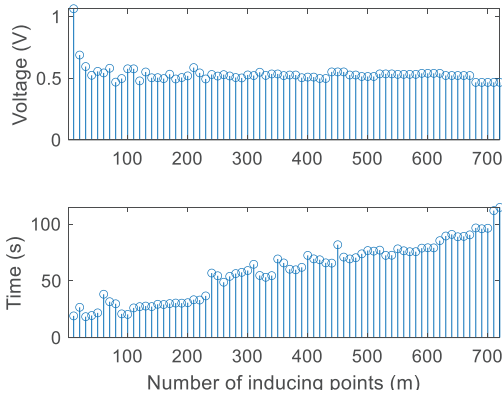


Fig. 9. Study of the influence of the number of inducing points in the voltage prediction error and the computational time.

The number of inducing points is tested up to 720 samples because this corresponds to one month of data, the same magnitude used to train Regular and ME models. Algorithm 4 details the flow chart of the Sparse method, similar but not equal to Algorithm 3 for Regular GPR.

As a last remark, one of the advantages of GPR for the application point of view is that it provides the confidence interval in Eq. (19). To verify the performance of the trained model with respect to the actual behavior of the installation we must wait M hours to compare the measured voltage with the predicted one. This gives a procedure to check the accuracy of the trained model, since with an accurate GPR we should see, at least, a 95% of the times of the measured voltage lying within the confidence interval.

Algorithm 4: Estimation with Sparse recursive GPR predictor

- (1) **Input:** Dataset $\mathcal{D}\{(\mathbf{x}_{k_1}, y_{k_1}), \dots, (\mathbf{x}_{k_n}, y_{k_n})\}$
 - (2) **Input:** Kernel function $k(\cdot, \cdot)$ parametrized by Θ
 - (3) Random selection of m inducing points, X_u
 - (4) Run optimization: $\hat{\Theta} = \arg \min_{\Theta} NLL(\mathcal{D}, \Theta, X_u)$
 - (5) Precompute Ω , Λ^{-1} , $\mathbf{K}_{f,f}^{-1}$, $K_{u,u} = k(X_u, X_u)$, $K_{f,u} = k(\mathbf{x}_{k_i}, X_u)$, $i = 1, \dots, n$
 - (6) $\mathbf{y} = [y_{k_1}, y_{k_2}, \dots, y_{k_n}]^T$
 - (7) **for** $z = 1: M$ **do**
 - (8) Format $\mathbf{x}_{*k}^{(z)}$ as in (20)
 - (9) $\mathbf{K}_{*,f} = k(\mathbf{x}_{*k}^{(z)}, \mathbf{x}_{k_i})$, $i = 1, \dots, n$
 - (10) $K_{**} = k(\mathbf{x}_{*k}^{(z)}, \mathbf{x}_{*k}^{(z)})$
 - (11) Eq (31,32)
 - (12) **end for**
 - (13) **Output:** Predicted voltages $\hat{V}_{k+z} = \hat{\mu}_{*k}^{(z)}$ and variance $\hat{\Sigma}_{*k}^{(z)}$, $z = 1, \dots, M$
-

4 Results and discussion

In this Section we validate the proposed models in the experimental database. The experiments are implemented in MATLAB©2020a in a 1.8-GHz Intel Core i7-10510U CPU.

In order to test the performance of the three proposed models, it is first necessary to see the relevance of the selection of the input variables. By this, we study which variables (voltage, current and temperature) are relevant to predict future voltage. There are four possibilities:

- Future current and temperature profiles known.
- Only future current profile known.
- Only future temperature profile known.
- Neither current nor temperature known (not possible in the context of this problem).

In all cases past profiles are known. To test the influence of the input parameters in the prediction we use the Regular

GPR, as long as the results are expandable to the other models. The resulting error metrics are listed in Table 3.

Table 3 Error metrics as a function of the inputs considered in the models

	I and T	I only	T only
$RMSE$ (V)	0.592	0.590	2.283
$MaxAE$ (V)	1.596	1.603	5.700
$RMSE_{EON}$ (V)	0.225	0.228	0.279
$MaxAE_{EON}$ (V)	0.739	0.728	1.539

The results show that temperature has almost no influence on voltage prediction and therefore it can be omitted in future analysis. This is related to the high thermal inertia of the battery pack, since they contain tons of liters of water. Sudden irradiation changes caused by a cloud are not enough to vary relevantly the temperature. In practice, this result helps to reduce the complexity of the system since the matrix dimensions get reduced. As the second method in Table 3 gets similar error metrics to the first method but with less measures, it is chosen to test the models.

Once the future input profile has been chosen, we discuss the performance of the three different algorithms. In contrast with Regular and ME GPR, as Sparse GPR can deal with larger datasets, it is trained with the entire 2008 year. In order to keep a fair comparison between the algorithms, they all are tested with March 2009.

To illustrate the performance of the three methods, in Fig. 10. we show the waveforms obtained with the Regular GPR implementation. The model fits the real waveform accurately even in low irradiation days. These results manifest that there exists a periodic error located at midday which matches with the highest value of the confidence interval, and therefore of standard deviation, during a single day. This is because, at this point of the day, any interfering cloud may reduce the irradiation on the panels and, thus, there is a higher uncertainty. The error at the end of the night is low in all cases. This is an intuitive result since clouds do not really have an impact on the irradiation at night (which is nonexistent). As the waveforms from the three GPR models are similar, Table 4 provides quantitative information.

Table 4 Error metrics with only future current profile as input, for a battery with a total nominal voltage of 48V

	Regular	ME	Sparse
$RMSE$ (V)	0.590	0.517	0.469
$MaxAE$ (V)	1.603	2.751	2.138
$RMSE_{EON}$ (V)	0.228	0.190	0.238
$MaxAE_{EON}$ (V)	0.728	0.308	0.127

All methods perform, to a greater or lesser extent, a proper voltage estimation in a month with heterogeneous events. ME GPR improves the RMSE of Regular GPR but Sparse GPR gets the better performance of all methods when it comes to this metric. Besides, Regular GPR has the lower MaxAE so it seems to be the most reliable option when the system needs accurate voltage estimation. In the PV standalone application field, where the EON voltage estimation has a strong interest, all methods reduce RMSE and MaxAE values with a noteworthy performance of ME and Sparse. We remark that using GPR for prediction (extrapolation) in PV standalone applications represents a more demanding task than in the typical regression problem (interpolation) because of the uncertainty of the results.

As means of comparison, we apply the same approach to two non GPR related algorithms in this application: Nonlinear Autoregressive Exogenous Model (NARX) networks [32, 33] and Long Short-Term Memory (LSTM) [34, 35], both well-known techniques to process and predict real time data series. The theoretical and implementation details can be found in [36]. The tests are carried with the same conditions of GPR: the same year for training, the same month for testing, same L and failure prediction of 48 hours ahead. The results are in Table 5.

Table 5 Error metrics of GPR techniques vs NARX and LSTM for a battery with a total nominal voltage of 48V

	ME	Sparse	NARX	LSTM
$RMSE$ (V)	0.517	0.468	0.469	0.454
$MaxAE$ (V)	2.751	2.138	3.741	3.861
$RMSE_{EON}$ (V)	0.190	0.238	0.095	0.220
$MaxAE_{EON}$ (V)	0.308	0.127	0.083	0.197
Train time (s)	60	24	55.7	5437
Test time (s)	416	47	0.041	2.7
Exe. time (ms)	610	69	0.06	4

Results are rather similar, which confirms the usefulness of GPR as a prediction tool. Only NARX shows slightly better results predicting EON situations (40mV better than GPR) but, in contrast, it has higher MaxAE figures (1.6V worse than GPR). Train and test times information are not especially important because they are only related to the design phase of the algorithm. Execution time (Exe. Time), which is the time needed to make one single prediction of a sample, is more important regarding to implementation. However, as a single prediction is carried out each hour, there is enough time for each execution. For example, the most time-consuming technique (ME) spends only 0.61 seconds. Therefore, the time-scale difference between the execution time and rate of data stream (in the order of 10^3) makes the system robust against failures due to overpassing the computational burden of the installation.

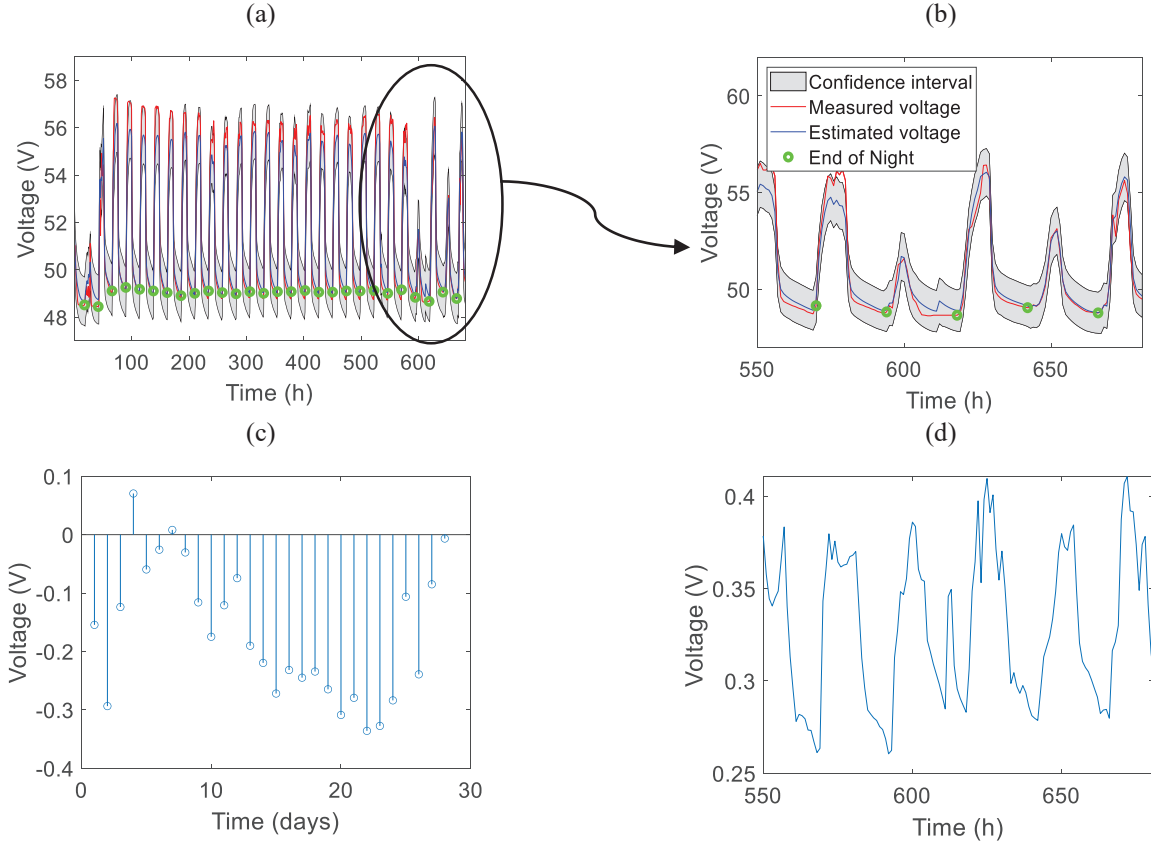


Fig. 10. Results of the prediction with Regular GPR and current as the only known future value: (a) waveforms over the month test, (b) zoom to the crucial days of the month, (c) $RMSE_{EON}$ obtained in the month test, (d) Σ obtained in the crucial days of the month.

Comparing the execution times in Table 5 it can be seen that the computational improvement of Sparse GPR is of an order of magnitude in both training and test stages. This is reasonable if we consider the theoretical computational complexities of both methods, shown in Table 6.

Table 6 Computational cost of Regular and Sparse GPR

	Storage	Training	Mean	Variance
Regular	$O(n^2)$	$O(n^3)$	$O(n)$	$O(n^2)$
Sparse	$O(mn)$	$O(m^2n)$	$O(m)$	$O(m^2)$

Besides, it is important to note that GPR algorithms have an advantage over the other techniques, that is, they provide the confidence interval. This measure of the algorithm accuracy enhances the robustness and allows to design more sophisticated fault-tolerant protocols.

5 Conclusion

In this paper, we have analyzed the problem of future voltage estimation in PV standalone installations with three different recursive multistep GPR models: Regular GPR, Multiple Expert GPR and Sparse GPR. In contrast with previous work on the field, a large dataset with voltage,

current and temperature from 2 years has been used. We have evaluated the impact of kernel selection, memory length and the influence of the temperature in the behavior of the system. Results have revealed how simple kernels as RQ provide a good trade-off between computational time and error metrics. It has been observed that a short memory length of $L=15$ hours is good enough to make accurate estimations. We have seen how, due to the high thermal inertia of the battery pack, temperature is not a representative input parameter for the model and can be omitted. Simulation results have shown an RMSE value of 590mV for Regular GPR, improved to 517mV and 469mV for Multiple Experts and Sparse GPR respectively. Furthermore, with respect to the fault-tolerant application, ME and Sparse models exhibit a robust performance thanks to a low MaxAE and the confidence interval, compared to the results with NARX and LSTM solutions. In future work, we will study the prediction capabilities of these algorithms for SoC and SoH parameters in PV standalone applications, and we will extend our proposal to an online version implemented in the real installation. Besides, we will prepare and process the dataset to open it to the general public.

Compliance with ethical standards

Acknowledgements The authors would like to acknowledge the support from Sociedad Ibérica de Construcciones Eléctricas S.A. and from Confederación Hidrográfica del Ebro.

Conflict of interest The authors declare that they have no conflict of interest.

Funding This work was supported in part by the Spanish MINECO under Grant RTC-2015-3358-5 and the LMP_16_18 Grant from the Aragon government RIS3 program.

References

- Vazquez S, Lukic S, Galvan E, Franquelo LG, Carrasco JM, Leon JI (2011) Recent advances on energy storage systems. *Proc Industrial Electron Conf* 4636–4640
- Taki M, Rohani A, Soheili-Fard F, Abdeshahi A (2016) Assessment of energy consumption and modeling of output energy for wheat production by neural network (MLP and RBF) and Gaussian process regression (GPR) models. *J Clean Prod* 172:3028–3041
- Sechilariu M, Wang B, Locment F (2013) Building integrated photovoltaic system with energy storage and smart grid communication. *IEEE Trans Ind Electron* 60:1607–1618
- Yu W, Xue Y, Luo J, Ni M, Tong H, Huang T (2016) An UHV grid security and stability defense system: Considering the risk of power system communication. *IEEE Trans Smart Grid* 7:491–500
- Quiñero-candela J, Rasmussen CE, Herbrich R (2005) A unifying view of sparse approximate Gaussian process regression. *J Mach Learn Res* 6:1935–1959
- Rasmussen CE, Williams CKI (2006) *Gaussian processes for machine learning*. MIT Press
- El Mejdoubi A, Oukaour A, Chaoui H, Gualous H, Sabor J, Slamani Y (2016) State-of-Charge and State-of-Health Lithium-Ion Batteries' Diagnosis According to Surface Temperature Variation. *IEEE Trans Ind Electron* 63:2391–2402
- Lin HT, Liang TJ, Chen SM (2013) Estimation of battery state of health using probabilistic neural network. *IEEE Trans Ind Informatics* 9:679–685
- Hu JN, Hu JJ, Lin HB, Li XP, Jiang CL, Qiu XH, Li WS (2014) State-of-charge estimation for battery management system using optimized support vector machine for regression. *J Power Sources* 269:682–693
- Sahinoglu GO, Pajovic M, Sahinoglu Z, Wang Y, Orlik P V., Wada T (2018) Battery State-of-Charge Estimation Based on Regular/Recurrent Gaussian Process Regression. *IEEE Trans Ind Electron* 65:4311–4321
- Pajovic M, Sahinoglu Z, Wang Y, Orlik P V., Wada T (2017) Online data-driven battery voltage prediction. *Proc Int Conf Ind Informatics* 827–834
- Liu D, Pang J, Zhou J, Peng Y (2012) Data-driven prognostics for lithium-ion battery based on Gaussian process regression. *Proc Progn Syst Heal Manag Conf*
- Cong X, Member S, Zhang C, Member S, Jiang J (2020) A Hybrid Method for the Prediction of the Remaining Useful Life of Lithium - ion Batteries with Accelerated Capacity Degradation. *IEEE Trans Vehic Tech* 69, 11: 12775 - 12785
- Liu J, Chen Z (2019) Remaining Useful Life Prediction of Lithium-Ion Batteries Based on Health Indicator and Gaussian Process Regression Model. *IEEE Access* 7:39474–39484
- Liu K, Hu X, Wei Z, Li Y, Jiang Y (2019) Modified Gaussian Process Regression Models for Cyclic Capacity Prediction of Lithium-Ion Batteries. *IEEE Trans Transp Electrification* 5:1225–1236
- Liu K, Shang Y, Ouyang Q, Widanage WD (2020) A Data-driven Approach with Uncertainty Quantification for Predicting Future Capacities and Remaining Useful Life of Lithium-ion Battery. *IEEE Trans Ind Electron* 68, 4: 3170-3180
- Liu K, Li Y, Hu X, Lucu M, Widanage WD (2020) Gaussian Process Regression with Automatic Relevance Determination Kernel for Calendar Aging Prediction of Lithium-Ion Batteries. *IEEE Trans Ind Informatics* 16:3767–3777
- Williams CKI, Rasmussen CE (1996) Gaussian processes for regression. *Adv Neural Inf Process Syst* 8 8:514–520
- Duvenaud DK (2014) *Automatic model construction with Gaussian processes* (Doctoral thesis).
- Rasmussen CE, Nickisch H (2010) Gaussian processes for machine learning (GPML) toolbox. *J Mach Learn Res* 11:3011–3015
- Kohonen T (2001) *Self-organizing maps*, 3rd ed.
- Kohonen T (1997) Exploration of very large databases by self-organizing maps. *IEEE Int Conf Neural Networks - Conf Proc*.
- Kangas J, Kohonen T, Laaksonen J, Simula O, Venta O (1989) Variants of self-organizing maps. *Int Joint Conf Neur Net*. 2:517-522.
- Vesanto Juha, Himberg Johan, Alhoniemi Esa PJ (2000) *Self-Organizing Map in Matlab: Som Toolbox*
- Wagstaff K, Cardie C, Rogers S, Schrödl S (2001) Constrained K-means Clustering with Background Knowledge. *Int Conf Mach Learn ICML pages:577–584*
- Likas A, Vlassis N, J. Verbeek J (2003) The global k-means clustering algorithm. *Pattern Recognit* 36:451–461
- Sanz-Gorrahategui I, Pastor-Flores P, Guillen-Asensio A, Artal-Sevil JS, Bono-Nuez A, Martin-Del-Brio B, Bernal-Ruiz C (2020) Unsupervised clustering of battery waveforms in off-grid PV installations. *Int Conf Ecol Veh Renew Energies - Conf Proc*.
- Bauer M, Van Der Wilk M, Rasmussen CE (2016) Understanding probabilistic sparse Gaussian Process approximations. *Adv Neural Inf Process Syst* 1533–1541
- Ozcan G, Pajovic M, Sahinoglu Z, Wang Y, Orlik P V., Wada T (2016) Online battery state-of-charge estimation based on sparse gaussian process regression. *IEEE Power Energy Soc Gen Meet - Conf Proc*

30. Chalupka K, Williams CKI, Murray I (2013) A framework for evaluating approximation methods for Gaussian process regression. *J Mach Learn Res* 14:333–350
31. Gonzalez TF (1985) Clustering to minimize the maximum intercluster distance. *Theor Comput Sci* 38:293–306
32. Chen S, Billings SA, Grant PM (1990) Non-linear system identification using neural networks. *Int J Control* 51:1191–1214
33. Siegelmann HT, Horne BG, Giles CL (1997) Computational capabilities of recurrent NARX neural networks. *IEEE Trans Syst Man, Cybern Part B Cybern* 27:208–215
34. Graves A, Schmidhuber J (2005) Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks* 18:602–610
35. Greff K, Srivastava RK, Koutnik J, Steunebrink BR, Schmidhuber J (2017) LSTM: A Search Space Odyssey. *IEEE Trans Neural Networks Learn Syst* 28:2222–2232
36. Guillen-Asensio A, Sanz-Gorrachategui I, Pastor-Flores P, Artal-Sevil JS, Bono-Nuez A, Martin-Del-Brio B, Bernal-Ruiz C (2020) Battery state prediction in photovoltaic standalone installations. *Int Conf Ecol Veh Renew Energies - Conf Proc*.