Universidad Zaragoza
1542

Trabajo Fin de Grado

Procesos de Regresión Gaussiana: Estudio de métodos Sparse para la predicción de tensión futura en equipos de comunicaciones

Gaussian Process Regression: Study of Sparse Methods to estimate future voltage in communication systems

Autor

José Miguel Sanz Alcaine

Directores

Dr. Carlos Bernal Ruiz

Dr. Antonio Bono Nuez

ESCUELA DE INGENIERÍA Y ARQUITECTURA
2019

Procesos de Regresión Gaussiana: Estudio de métodos Sparse para la predicción de tensión futura en equipos de comunicaciones

RESUMEN

Este Trabajo Fin de Grado forma parte de la línea Smart del proyecto de investigación "Battery life Extensor"(BATT-Ex), cuyos principales objetivos son proporcionar información relevante sobre la batería en sistemas de telecocomunicaciones y tomar decisiones inteligentes en base al conocimiento de su estado actual. En los últimos años las baterías han recibido mucha atención debido a las redes de comunicaciones y a la aparición de los Vehículos Híbridos (HEVs) y los Vehículos Eléctricos (EVs). Esto ha llevado a que centros de investigación como el Mitsubishi Electric Research Laboratories (MERL) en Boston o empresas como la Sociedad Ibérica de Construcciones Eléctricas (SICE) hayan querido profundizar en su investigación y colaborar en el siguiente trabajo de investigación. Con el presente trabajo se pretende mejorar las técnicas de estimación futura de tensión en baterías por procesado de la señal de tensión-corriente, en instalaciones aisladas de comunicaciones. Para ello se han utilizado técnicas de Inteligencia Artificial. En particular, se ha estudiado el uso de la regresión con procesos gaussianos (GPR) como herramienta de predicción y sus diferentes variantes Sparse para reducir la complejidad computacional del algoritmo. Para llevar a cabo la estimación se necesita información sobre las variables de interés. Para ello, se ha utilizado la base de datos de un sistema fotovoltaico de gran potencia ubicado en el monte del Monasterio de Sigena la cual contiene 10 años de datos recabados cada 15 minutos de parámetros como tensión, corriente o temperatura. De forma más concreta, se ha evaluado el GPR en su enunciado clásico, el método FITC Sparse y una variante con múltiples expertos GPR comparando los resultados de estimación de tensión aportando el perfil de corriente o temperatura futura.

Gaussian Process Regression: Study of Sparse Methods to estimate future voltage in communication systems

ABSTRACT

This Bachelor's Degree Final Project is part of the Smart line of the research project "Battery life Extensor"(BATT-Ex), that aims to provide relevant information about the battery for the user and to make intelligent decisions based on knowledge of its current state. In recent years batteries have received a lot of attention due to telecommunications networks and the emergence of Hybrid Vehicles (HEVs) and Electric Vehicles (EVs). This has led research centres such as Mitsubishi Electric Research Laboratories (MERL) in Boston or companies such as Sociedad Ibérica de Construcciones Eléctricas (SICE) to want to deepen their research and collaborate in the following research work. The aim of this work is to improve the techniques for future estimation of battery voltage by processing the voltage-current signal in isolated communications installations. Artificial Intelligence techniques have been used for this purpose. In particular, the use of Gaussian Process Regression (GPR) as a prediction tool and its different Sparse variants have been studied to reduce the computational complexity of the algorithm. In order to carry out the estimation, information on the variables of interest is needed. For this purpose, the database of a high power photovoltaic system located in the mountain of the Sigena Monastery has been used, which contains 10 years of data collected every 15 minutes from parameters such as voltage, current or temperature. More specifically, GPR has been evaluated in its classical statement, FITC Sparse approximation method and a variant with multiple GPR experts comparing the results of voltage estimation providing the current profile or future temperature.

Índice

Capítulo 1

Introducción

El presente trabajo forma parte del proyecto de investigación "Battery life Extensor"(BATT-Ex, RTC-2015-3358-5) el cual se desarrolla de forma conjunta entre el Grupo de Electrónica de Potencia y Microelectrónica de la Universidad de Zaragoza (GEPM), el Grupo de almacenamiento de Energía Eléctrica de la Universidad de Mondragón y la empresa Sociedad Ibérica de Construcciones Eléctricas (SICE). Además, para este proyecto concreto se ha contado con la colaboración del Mitsubishi Electric Research Laboratories (MERL). En este centro de investigación, miembros del equipo de trabajo han realizado diversas estancias para trabajar de forma conjunta en profundizar el conocimiento de las baterías. Como consecuencia, se han publicado, de forma conjunta, diversos artículos y se está colaborando actualmente en otro abordando los resultados obtenidos en este trabajo.

Dado el interés que este tipo de investigación tiene en la industria, SICE ha apoyado este trabajo a través de una beca de investigación.

1.1. Antecedentes

El proyecto BATT-Ex se centra en solucionar los problemas que tienen los sistemas de telecomunicaciones aislados. En este tipo de instalaciones, se hace necesario el uso de baterías para asegurar un correcto funcionamiento del sistema durante el día. Los principales problemas que presentan este tipo de sistemas conocidos como sistemas de almacenamiento de energía (ESS, *Energy Storage Systems*) son el dimensionamiento, el mantenimiento y la absorción de energía.

Figura 1.1: Sistema de comunicaciones alimentado por placas fotovoltaicas

Debido a la ubicación de este tipo de sistemas se hace necesario alimentarlos con paneles fotovoltaicos (Figura 1.1). Sin embargo, este tipo de fuente de energía presenta incertidumbre debido a los cambios meteorológicos. Se considera un sistema estable ya que existe una distinción clara de patrones nocturnos y diurnos pero son dependientes de pasos por nube y distintos fenómenos meteorológicos que alteran la producción de la energía durante el día. Estos patrones pueden verse en la Figura 1.2

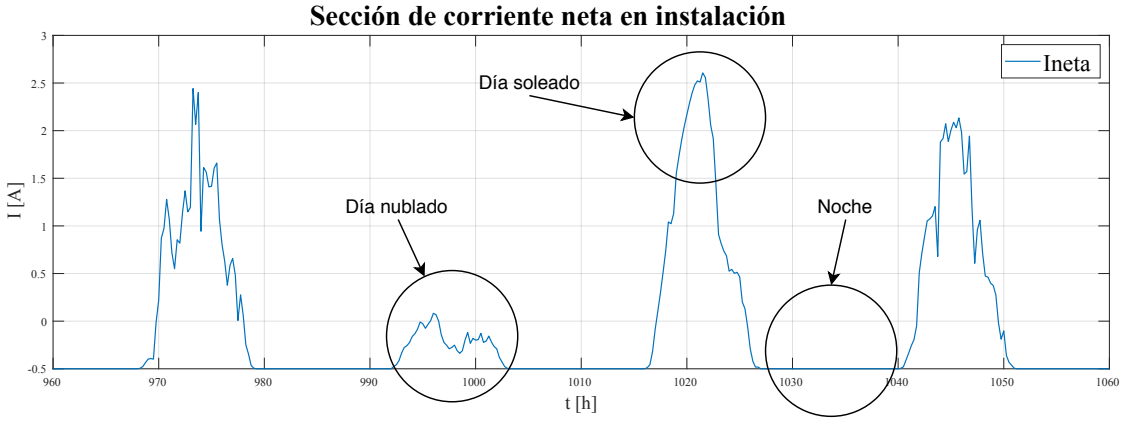

Figura 1.2: Patrones de corriente en función de la climatología

Es por ello que con el fin de aprovechar la energía generada y ser capaces de alimentar el sistema por la noche o en caso de fallo, se hace imprescindible el uso de baterías con una gestión inteligente.

Un ejemplo de aplicación de este tipo de sistemas se encuentra en las estaciones de

comunicaciones de 5G. Las nuevas bandas de frecuencia superior con menor eficiencia, los esquemas de modulación más eficientes, en términos de eficiencia espectral, pero peores en eficiencia de amplificación, la carga de computacional y el aumento de estaciones base lleva a problemas de suministro de energía. En las ciudades y, por lo general, en sistemas con conexión a la red no resulta un problema. Sin embargo, sí que lo es en las estaciones aisladas por lo que han de ser capaces de aprovechar al máximo la energía almacenada con el fin de reducir costes y mejorar la calidad de servicio.

1.2. Motivación y Objetivos

Los sistemas de gestión de baterías (BMS, *Battery Management Systems*) son los encargados del control de la batería y de asegurarse de su seguridad y correcto funcionamiento. Para ello, los BMS incluyen sensores de diferentes magnitudes físicas como tensión, corriente, temperatura, etc. Estas magnitudes son las que procesa el BMS para tratar de estimar el estado de la batería. Tradicionalmente, el estado de carga (SoC, *State of Charge*) y el estado de salud (SoH, *State of Health*) han sido los principales parámetros a tener en cuenta a la hora de averiguar el estado de una batería. El SoC representa la carga interna de la batería en función de su carga máxima actual mientras que el SoH se refiere al estado en que se encuentra la batería respecto al estado inicial. Ambos parámetros se encuentran al 100 % cuando la batería se fabrica sin defectos. Existen multitud de técnicas para estimar estos parámetros. Mayoritariamente se ha trabajado con modelos equivalentes de batería y filtros de Kalman [1]-[2]. Sin embargo, recientemente se están comenzando a usar técnicas basadas en análisis de datos (*Data-driven methods*) como las Redes Neuronales o las Máquinas de vectores de soporte [3]-[4]. A través de este tipo de estimaciones se obtiene un resultado más certero. Pese a su buen funcionamiento, el envejecimiento de las baterías provoca que este tipo de métodos requieran de un reentrenamiento lo cual es un problema para la estimación tanto del SoC como del SoH ya que no hay un acceso fácil a los valores reales de SoC o SoH. Es por ello, y porque en los sistemas fotovoltaicos el parámetro que limita la operación del sistema es la tensión mínima que admite el convertidor de potencia, que en este trabajo se trata de estimar un parámetro diferente, la tensión que el sistema va a ser capaz de suministrar en un determinado momento. Este parámetro resulta de gran utilidad ya que puede ser medido con precisión haciendo así un reentrenamiento sencillo.

En el presente trabajo, se ha estudiado la predicción de tensión futura dados tres perfiles de variables físicas conocidas por el BMS:

- **Perfil de corriente y temperatura futura conocido**
- **Perfil de corriente futura conocido pero no de temperatura**
- **Perfil de temperatura futura conocido pero no de corriente**

Frente a otras técnicas basadas en análisis de datos, la estimación de tensión permite un simple reentrenamiento por lo que puede aportar estimaciones certeras pese al envejecimiento de la batería. El objetivo es estimar la tensión correspondiente al tiempo $t + T$ asumiendo que los perfiles de alguna de las tres situaciones es conocido desde t hasta $t+T$. Esta situación de conocer patrones futuros es realista ya que, en la práctica, un operario o el BMS puede ingresar al algoritmo distintos patrones correspondientes a distintas situaciones, desde el funcionamiento normal al más crítico. De esta forma se sabe qué estación de almacenamiento tiene probabilidad de fallar en caso de que suceda dicho patrón y así actuar en consecuencia. En este trabajo contamos con una base de datos realista y de gran tamaño lo que ofrece un amplio espectro de posibilidades para las técnicas basadas en análisis de datos. Se ha analizado las capacidades que tiene el GPR en el ámbito de las baterías. Además, se ha realizado una aproximación GPR Sparse para reducir la carga computacional que tiene el GPR con grandes volúmenes de datos. Por otro lado, debido a la variabilidad de tipos de días que se pueden encontrar en la base de datos, se ha hecho una clasificación de tipos de días a través de mapas autoorganizados (SOM, *Self-organizing map*). Con los días ya clasificados según su tipología, se ha entrenado un modelo GPR con cada tipo de día y se ha hecho un test con un periodo distinto de la base de datos. El experto GPR que en cada muestra obtiene menor varianza es el que se toma como válido.

1.3. Estado del Arte

La estimación de parámetros de baterías a través de regresión con Procesos Gaussianos de forma exacta ya ha sido probada con éxito, por parte de nuestros compañeros del MERL, en la estimación del SoC y del SoH [5]-[6]-[7]. Tanto en [8] como en [9] se hace uso de aproximaciones Sparse en el ámbito de las baterías para estimar el SoC ya que en ambos artículos hacen uso de grandes volúmenes de datos que son lo suficientemente complejos como para que un GPR exacto no pueda tratarlos.

En [10] hacen uso, al igual que en nuestro caso, de este algoritmo como estimador de tensión a través de un GPR exacto. Sin embargo, en todos estos trabajos, se han utilizado bases de datos creadas de, forma sintética, a partir de modelos de baterías. Las peculiaridades de una base de datos real con casuísticas de días tan diversas no existe en bases de datos sintéticas y por por ello no se ha abordado previamente la posibilidad de crear expertos GPR en este ámbito. Alejándonos del sector de las baterías si que existen casos de uso de expertos GPR vistos, por ejemplo, en síntesis de voz [11]. En este caso, existen dos expertos, uno para la duración de las sílabas y otro para la duración de una llamada telefónica.

1.4. Herramientas

El proyecto ha sido realizado sobre Matlab 2018a. Se ha escogido esta herramienta porque, además de ofrecer un entorno de trabajo muy versátil, cuenta con la *toolbox* GPML y la *toolbox* de SOM. La *toolbox* GPML desarrollada por Carl Edward Rasmussen está en continua actualización abordando cada vez más tipos de aproximaciones Sparse y cubre el abanico completo del GPR incluyendo funciones de minimización, covarianzas, densidades de probabilidad y capacidad de testing. El amplio abanico de posibilidades que aporta el GPML provoca que ciertos métodos se engloben en la misma función y resulte ineficientes. Se ha trabajado en su mejora, realizando los cálculos matriciales necesarios. Por otro lado, la *toolbox* de SOM, desarrollada por la Universidad Tecnológica de Helsinki [12], permite el entrenamiento de los mapas, visualizaciones complejas, y herramientas de clustering como la K-*means*.

Capítulo 2

Regresión con Procesos Gaussianos

Los Procesos Gaussianos son una herramienta matemática cuya teoría básica fue introducida por Kolmogorov y Wiener [13]-[14] en los años cuarenta. Sin embargo, no fue hasta los años setenta cuando se comenzaron a utilizar como herramienta de regresión en el campo de la Geoestadística [15] llevándose, años mas tarde, a otros contextos. Pese a ello, fueron C. E. Rasmussen junto a Christopher K. I. Williams, a finales de siglo, quienes introdujeron esta técnica en el campo de la inteligencia artificial [16] haciendo, años mas tarde, un estudio más profundo [17] que ha sido tomado como objeto de estudio obligatorio para todo aquel que investigue en GPR.

Existen múltiples ventajas que nos llevan a hacer uso de los Procesos Gaussianos como herramienta de regresión:

- **Expresión analítica cerrada.** Dado una función de covarianza y observaciones, podemos computar la distribución a posteriori de forma cerrada. Esta propiedad no es muy usual en modelos no paramétricos.

- **No existe overfitting**. Al contrario que otras técnicas como las redes neuronales, esta técnica de regresión no hace un sobre entrenamiento de los datos.

- **Aporta un intervalo de confianza**. A través de la varianza de la distribución, se puede establecer un intervalo en el que el dato estimado se debe encontrar. Esto aporta una información extra al sistema sobre cómo de buena es esta estimación.

Su principal limitación reside en la carga computacional y la memoria ya que crece a razón de $\mathcal{O}(n^3)$ siendo n el número de datos de entrenamiento.

2.1. Introducción a los Procesos Gaussianos

Tal y como define C. E. Rasmussen [17] un Proceso Gaussiano (GP) es una colección de variables aleatorias, que cumplen que cualquier subconjunto finito de la colección, tiene una distribución Gaussiana

Un GP viene completamente especificado por su función de media y de covarianza. Se puede definir la función media $m(\mathbf{x})$ y la función de covarianza $k(\mathbf{x}, \mathbf{x}')$ de un proceso real $f(\mathbf{x})$ como

$$m(\mathbf{x}) = \mathbb{E}[f(\mathbf{x})] \tag{2.1}$$

$$k(\mathbf{x}, \mathbf{x}') = [(f(\mathbf{x}) - m(\mathbf{x}))(f(\mathbf{x}') - m(\mathbf{x}')] \tag{2.2}$$

y será denotado como

$$f(\mathbf{x}) \sim \mathcal{GP}(m(\mathbf{x})), k(\mathbf{x}, \mathbf{x}')) \tag{2.3}$$

Normalmente se asume que la media de la función es cero. Sin embargo, si es necesario, se puede incorporar el conocimiento de su valor al modelo.

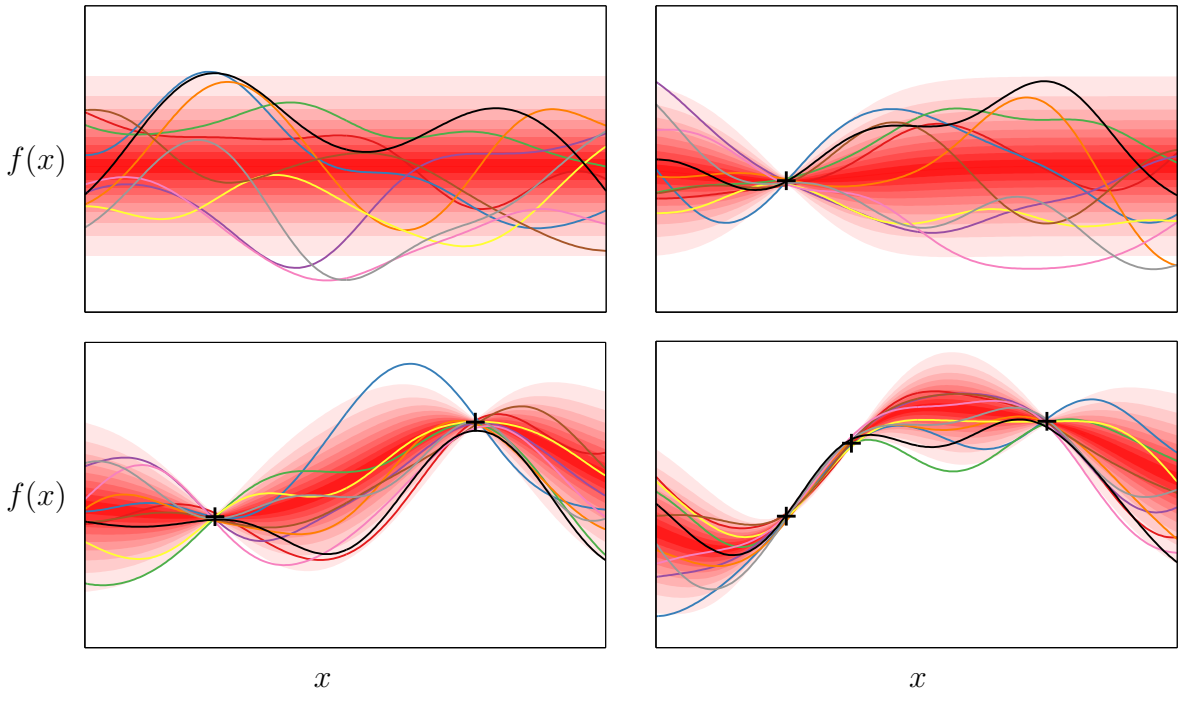

Figura 2.1: Una representación visual de un proceso gaussiano que modela una función unidimensional. Los diferentes tonos de rojo representan el intervalo de confianza de cada en cada punto. Las líneas de colores muestran diferentes realizaciones del proceso. *Esquina superior izquierda:* Un GP no condicionado por ningún punto. *Las gráficas restantes* La distribución a posteriori tras haber sido condicionada con diferente número de datos. [18]

Definición 1 (Distribución normal multivariante)**.** *Se dice que un vector aleatorio* $\mathbf{x}$ *sigue una distribución normal multivariante si tiene la siguiente función de densidad de probabilidad conjunta*

$$f_{\mathbf{x}}(x_1, \ldots, x_n) = \frac{1}{(2\pi)^{n/2}|K|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}^T)K^{-1}(\mathbf{x} - \boldsymbol{\mu}) \right)$$

donde $\boldsymbol{\mu} = [\mu(x_1), \ldots, \mu(x_n)]^T$ *representa la esperanza de* $\mathbf{x}$ *y* $K = AA^T$ *es la matriz de covarianza de las componentes de* $\mathbf{x}$*.*

Los procesos gaussianos se dice que son consistentes y por tanto cumplen la propiedad de marginalización.

Definición 2 (Propiedad de marginalización)**.** *Dados dos vectores* $\mathbf{x}$ *y* $\mathbf{y}$ *aleatorios y gaussianos:*

$$\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu_x}, K_{1,1})$$

$$\mathbf{y} \sim \mathcal{N}(\boldsymbol{\mu_y}, K_{2,2})$$

se cumple

$$\begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} \sim \mathcal{N}(\boldsymbol{\mu}, K) = \mathcal{N}\Big(\begin{bmatrix} \boldsymbol{\mu_x} \\ \boldsymbol{\mu_y} \end{bmatrix}, \begin{bmatrix} K_{1,1} & K_{1,2} \\ K_{2,1} & K_{2,2} \end{bmatrix} \Big)$$

Esta propiedad expresa como, al tratar de forma conjunta ambos vectores, la distribución resultante será una normal que depende de los parámetros de las distribuciones de los vectores y de la correlación entre ambos (K_{12} y K_{21}).

Con dicha propiedad y con el siguiente teorema se presenta la base de funcionamiento del GPR.

Teorema 1. *Sean* $\mathbf{x}$ *e* $\mathbf{y}$ *dos vectores aleatorios conjuntamente gaussianos*

$$\begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} \sim \mathcal{N}\Big(\begin{bmatrix} \boldsymbol{\mu_x} \\ \boldsymbol{\mu_y} \end{bmatrix}, \begin{bmatrix} A & C \\ C^T & B \end{bmatrix} \Big) \tag{2.4}$$

la distribución marginal de $\mathbf{x}$*y la condicional de* $\mathbf{x}$ *dado* $\mathbf{y}$ *son*

$$\begin{aligned} \mathbf{x} &\sim \mathcal{N}(\boldsymbol{\mu_x}, A) \\ \mathbf{x}|\mathbf{y} &\sim \mathcal{N}(\boldsymbol{\mu_x} + CB^{-1}(\mathbf{y} - \mathbf{y}_y), A - CB^{-1}C^T) \end{aligned} \tag{2.5}$$

La utilidad de este teorema en la regresión reside en que si conocemos la distribución de un conjunto de variables aleatorias gaussianas podemos dividirlo en dos subconjuntos $\mathbf{x}$ e $\mathbf{y}$ donde, si conocemos los valores que toma $\mathbf{y}$, podemos condicionar el valor del subconjunto $\mathbf{x}$. Como veremos más adelante, esta división en subconjuntos serán el conjunto de datos de entrenamiento y el de test.

2.2. Regresión con Procesos Gaussianos

Dado un conjunto de datos de entrenamiento $\mathcal{D} = (\mathbf{x}_i, y_i), i = 1, \ldots, n$ donde $\mathbf{x}_i$ es el vector de entrada e y_i la salida real, escalar y ruidosa, buscamos calcular la distribución de los valores de salida de test f_* (y_* si incluimos ruido) para una entrada de test $\mathbf{x}_*$. En el caso más simple (el cual tratamos aquí) asumimos que el ruido es aditivo, independiente y Gaussiano, de tal forma que la relación entre la función $f(\mathbf{x})$ y las observaciones ruidosas vienen dadas por una normal multivariante de la forma

$$y_i = f(\mathbf{x}_i) + \epsilon_i, \quad \text{donde} \quad \epsilon_i \sim \mathcal{N}(0, \sigma^2_{ruido}), \tag{2.6}$$

donde σ^2_{ruido} es la varianza del ruido.

La regresión con GPs es un método Bayesiano que asume un GP a priori sobre funciones. Se asume que la distribución de los valores de las funciones se comporta de la forma

$$p(\mathbf{f}|\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n) = \mathcal{N}(\mathbf{0}, K) \tag{2.7}$$

donde $\mathbf{f} = [f_1, f_2, \ldots, f_n]^T$ es un vector de valores observables de funciones, $f_i = f(\mathbf{x}_i)$. Remarcar que el GP trata los valores observables de funciones f_i como variables aleatorias que se relacionan con sus correspondientes entradas. Para que la expresión anterior tenga sentido K tiene que ser una matriz de covarianza y por tanto ser simétrica y semidefinida positiva. Esta función de covarianza, que compone la distribución a priori, no debe depender de los datos (puede depender de parámetros adicionales como se verá en la sección 2.4). De aquí en adelante, por simplicidad, se omitirá la relación condicional con las entradas ya que el modelo de GP y todas las expresiones siguientes vienen condicionadas por esta relación.

Para estimar se ha de hacer uso del Teorema de Bayes a través del cual se parte de una distribución a priori conjunta entre los valores de entrenamiento y los de test, $\mathbf{f}$ y $\mathbf{f}_*$ y se combina con la función de verosimilitud $p(\mathbf{y}|\mathbf{f})$ para obtener la distribución posterior conjunta

$$p(\mathbf{f}, \mathbf{f}_*|\mathbf{y}) = \frac{p(\mathbf{f}, \mathbf{f}_*)p(\mathbf{y}, \mathbf{f})}{p(\mathbf{y})} \tag{2.8}$$

El último paso para producir la distribución de los valores a estimar será marginalizar los valores de entrenamiento no deseados

$$p(\mathbf{f}_*|\mathbf{y}) = \int p(\mathbf{f}, \mathbf{f}_*|\mathbf{y})d\mathbf{f} = \frac{1}{p(\mathbf{y})} \int p(\mathbf{f}, \mathbf{f}_*)p(\mathbf{y}, \mathbf{f})d\mathbf{f} \tag{2.9}$$

Tanto la distribución de verosimilitud a priori como la distribución independiente son

distribuciones gaussianas por lo que se puede aplicar la Propiedad [2]

$$p(\mathbf{f}, \mathbf{f}_*) = \mathcal{N}\left(0, \quad \begin{bmatrix} K_{\mathbf{f},\mathbf{f}} & K_{*,\mathbf{f}} \\ K_{\mathbf{f},*} & K_{*,*} \end{bmatrix}\right), \quad \text{y} \quad p(\mathbf{y}, \mathbf{f}) = \mathcal{N}(\mathbf{f}, \sigma^2_{ruido} I) \tag{2.10}$$

donde $K_{\mathbf{f},\mathbf{f}} = k(\mathbf{x}, \mathbf{x})$, $K_{\mathbf{f},*} = k(\mathbf{x}, \mathbf{x}_*)$, $K_{*,\mathbf{f}} = k(\mathbf{x}_*, \mathbf{x})$ y $K_{*,*} = k(\mathbf{x}_*, \mathbf{x}_*)$

Una vez hallada la forma de esta distribución conjunta, se debe marginalizar el valor de las salidas $\mathbf{f}_*$ con respecto del valor de las entradas $\mathbf{f}$ obteniendo la distribución de predicción. Para ello, se hace uso del Teorema [1]

$$p(\mathbf{f}_*, \mathbf{y}) = \mathcal{N}(\mu_*, \sigma^2_*) \tag{2.11}$$

$$\mu_* = K_{*,\mathbf{f}}(K_{\mathbf{f},\mathbf{f}} + \sigma^2_{ruido} I)^{-1}\mathbf{y} \tag{2.12}$$

$$\sigma^2_* = K_{*,*} - K_{*,\mathbf{f}}(K_{\mathbf{f},\mathbf{f}} + \sigma^2_{ruido} I)^{-1} K_{\mathbf{f},*} \tag{2.13}$$

Este resultado aporta la fórmula buscada para poder hacer estimaciones con GPR. El valor de media es el mejor estimador de la variable $\mathbf{f}_*$ y por tanto, el valor tomado como verdadero en el proceso de predicción . Además, se tiene en cuenta el valor de incertidumbre que la varianza aporta para fijar un intervalo de confianza. Este intervalo, se suele tomar como dos veces la desviación típica de cada valor x.

2.3. Métodos Sparse en GPR

Tras haber obtenido una expresión cerrada del estimador GPR (Ec. 2.23) vemos como la inversión de la matriz de tamaño $n \times n$ tiene una complejidad de $\mathcal{O}(n^3)$, siendo n el numero de datos de entrenamiento. Una vez calculada dicha matriz, la estimación de la media y de la varianza tiene un coste de $\mathcal{O}(n)$ y $\mathcal{O}(n^2)$ respectivamente. Esto lleva a problemas de implementación cuando se manejan grandes volúmenes de datos. El objetivo de los métodos de aproximación Sparse es reducir la complejidad computacional del algoritmo GPR.

La mayoría de las aproximaciones Sparse están basadas en el uso de un subconjunto de m variables $\mathbf{u} = [u_1, \ldots, u_m]^T$. A este subconjunto de variables de entrenamiento se le denomina variables inducidas (*Inducing variables*) y representan una generalización del total de la base de datos. Son variables del Proceso Gaussiano (como $\mathbf{f}$ y $\mathbf{f}^*$) y se corresponden con los índices $\mathbf{X_u}$ los cuales se denominan entradas inducidas (*Inducing points*). Mientas que las variables $\mathbf{u}$ siempre se marginalizan en la distribución de predicción, la elección de los *inducing points* resulta de gran relevancia en el resultado final de la estimación.

Existen multitud de métodos para la elección de los *Inducing points* [19]. Sin embargo, para esta explicación se parte de que este subconjunto de la base de datos ya ha sido escogido.

Al igual que en la expresión 2.9, se puede obtener $p(\mathbf{f}_*, \mathbf{f})$ integrando sobre $\mathbf{u}$ la distribución a priori conjunta $p(\mathbf{f}_*, \mathbf{f}, \mathbf{u})$

$$p(\mathbf{f}_*, \mathbf{f}) = \int p(\mathbf{f}_*, \mathbf{f}, \mathbf{u}) d\mathbf{u} = \int p(\mathbf{f}_*, \mathbf{f}|\mathbf{u}) p(\mathbf{u}) d\mathbf{u} \quad \text{donde} \quad p(\mathbf{u}) = \mathcal{N}(\mathbf{0}, K_{\mathbf{u},\mathbf{u}}) \tag{2.14}$$

La expresión anterior es exacta con $K_{\mathbf{u},\mathbf{u}}$=$k(\mathbf{X}_\mathbf{u}, \mathbf{X}_\mathbf{u})$. Es en este punto donde se basan la mayoría de las aproximaciones Sparse. Es posible aproximar la distribución a priori conjunta asumiendo que $\mathbf{f}$ y $\mathbf{f}_*$ son condicionalmente independientes dado $\mathbf{u}$.

$$p(\mathbf{f}_*, \mathbf{f}) \simeq q(\mathbf{f}_*, \mathbf{f}) = \int q(\mathbf{f}_*|\mathbf{u}) q(\mathbf{f}|\mathbf{u}) p(\mathbf{u}) d\mathbf{u} \tag{2.15}$$

El origen de que a este tipo de variables se les denomine inducidas viene por el hecho de que $\mathbf{f}$ y $\mathbf{f}_*$ solo se pueden comunicar a través de $\mathbf{u}$ por lo que $\mathbf{u}$ induce una dependencia entre los datos de entrenamiento y los de test tal y como se puede ver en la Figura 2.2

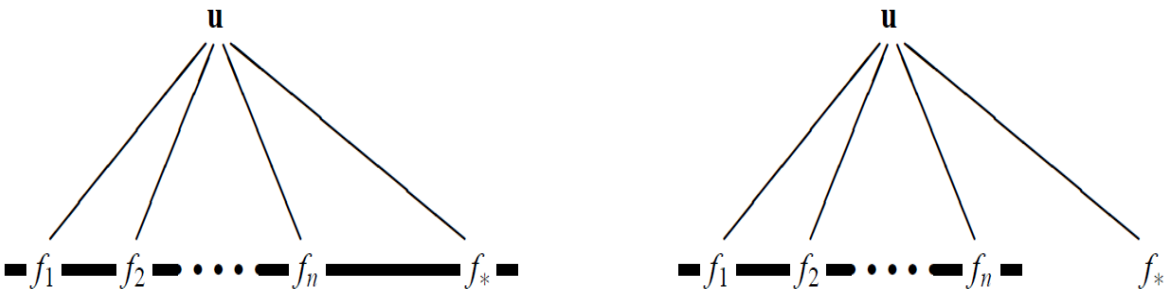

Figura 2.2: Relación entre variables con aproximaciones Sparse. [20]

Es a partir de este punto del algoritmo donde se ramifican las distintas aproximaciones Sparse. En ellas se proponen distintas hipótesis respecto las distribuciones $q(\mathbf{f}|\mathbf{u})$ y $q(\mathbf{f}_*|\mathbf{u})$. En la práctica, la aproximación que mejor ha funcionado ha sido la aproximación FITC (*Fully Independent Training Conditional*) [20]-[21] y por este motivo es por lo que se ha utilizado en este trabajo.

FITC hace una aproximación de la función de verosimilitud

$$p(\mathbf{y}|\mathbf{f}) \simeq q(\mathbf{y}|\mathbf{u}) = \mathcal{N}(K_{\mathbf{f},\mathbf{u}} K_{\mathbf{u},\mathbf{u}}^{-1} \mathbf{u}, \text{diag}[K_{\mathbf{f},\mathbf{f}} - K_{\mathbf{f},\mathbf{u}} K_{\mathbf{u},\mathbf{u}}^{-1} K_{\mathbf{u},\mathbf{f}}] + \sigma^2_{ruido} I) \tag{2.16}$$

$K_{\mathbf{f},\mathbf{u}} = k(\mathbf{x}, \mathbf{X}_\mathbf{u})$. Además impone una independencia de los valores de salida $\mathbf{f}$ con respecto de $\mathbf{u}$ (Figura 2.3)

$$q(\mathbf{f}|\mathbf{u}) = \prod_{i=1}^{n} p(f_i|\mathbf{u}) = \mathcal{N}(K_{\mathbf{f},\mathbf{u}} K_{\mathbf{u},\mathbf{u}}^{-1} \mathbf{u}, \text{diag}[K_{\mathbf{f},\mathbf{f}} - K_{\mathbf{f},\mathbf{u}} K_{\mathbf{u},\mathbf{u}}^{-1} K_{\mathbf{u},\mathbf{f}}]) \tag{2.17}$$

$$q(\mathbf{f}_*|\mathbf{u}) = p(\mathbf{f}_*|\mathbf{u}) = \mathcal{N}(K_{*,\mathbf{u}}K_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u}, K_{*,*} - K_{*,\mathbf{u}}K_{\mathbf{u},\mathbf{u}}^{-1}K_{\mathbf{u},*}) \tag{2.18}$$

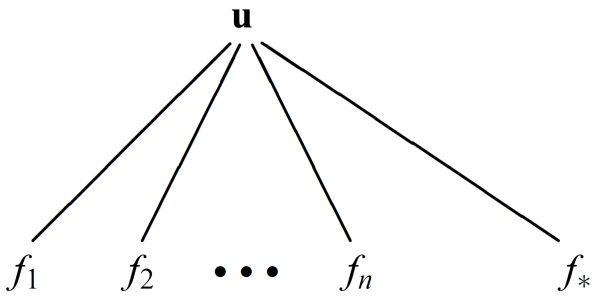

Figura 2.3: FITC Sparse con independencia entre los valores de salida $\mathbf{f}$ y $\mathbf{u}$. [20]

Al igual que se hizo en 3.7, se puede aplicar la Propiedad [2] para extraer la relación entre los valores de salida de entrenamiento y un solo caso de test.

$$p(\mathbf{f}, f_*) = \left(0, \quad \begin{bmatrix} K_{\mathbf{f},\mathbf{u}}K_{\mathbf{u},\mathbf{u}}^{-1}K_{\mathbf{u},\mathbf{f}} - \text{diag}[K_{\mathbf{f},\mathbf{u}}K_{\mathbf{u},\mathbf{u}}^{-1}K_{\mathbf{u},\mathbf{f}} - K_{\mathbf{f},\mathbf{f}}] & K_{\mathbf{f},\mathbf{u}}K_{\mathbf{u},\mathbf{u}}^{-1}K_{\mathbf{u},*} \\ K_{*,\mathbf{u}}K_{\mathbf{u},\mathbf{u}}^{-1}K_{\mathbf{u},\mathbf{f}} & K_{*,*} \end{bmatrix}\right) \tag{2.19}$$

Por último, para obtener la distribución de predicción con la aproximación FITC Sparse se debe marginalizar el valor de las salidas f_* con respecto de el valor de las entradas f a través del *Teorema* 1

$$p(\mathbf{f}_*, \mathbf{y}) = \mathcal{N}(\mu_*, \sigma_*^2) \tag{2.20}$$

$$\mu_* = K_{*,\mathbf{u}}\mathbf{\Sigma}K_{\mathbf{u},\mathbf{y}}\mathbf{\Lambda^{-1}}\mathbf{y} \tag{2.21}$$

$$\sigma_*^2 = K_{*,*} - K_{*,\mathbf{u}}K_{\mathbf{u},\mathbf{u}}^{-1}K_{\mathbf{u},*} + K_{*,\mathbf{u}}\mathbf{\Sigma}K_{\mathbf{u},*} \tag{2.22}$$

Donde $\mathbf{\Lambda} = \text{diag}[K_{\mathbf{y},\mathbf{y}} - K_{\mathbf{y},\mathbf{u}}K_{\mathbf{u},\mathbf{u}}^{-1}K_{\mathbf{u},\mathbf{y}} + \sigma^2 I_m]$ y $\mathbf{\Sigma} = (K_{\mathbf{u},\mathbf{u}} + K_{\mathbf{u},\mathbf{y}}\mathbf{\Lambda}^{-1}K_{\mathbf{y},\mathbf{u}})^{-1}$ El valor de media de la distribución de predicción anterior será el que utilicemos a la hora de estimar y la varianza formará el intervalo de confianza que tenemos alrededor de ese valor de media.

La siguiente tabla representa una comparativa de la carga computacional del algoritmo GPR y la aproximación FITC Sparse.

Método	Almacenamiento	Entrenamiento	Media	Varianza
GPR	$\mathcal{O}(n^2)$	$\mathcal{O}(n^3)$	$\mathcal{O}(n)$	$\mathcal{O}(n^2)$
FITC Sparse GPR	$\mathcal{O}(mn)$	$\mathcal{O}(m^2n)$	$\mathcal{O}(m)$	$\mathcal{O}(m^2)$

Tabla 2.1: Cargas Computacionales de las técnicas GPR

2.4. Kernels

Los GPs hacen uso de los Kernel para definir la covarianza entre dos variables aleatorias conjuntamente distribuidas

$$Cov[f(\mathbf{x}), f(\mathbf{x}')] = k(\mathbf{x}, \mathbf{x}') \tag{2.23}$$

Se trata de una función qué indica como de relacionadas están dos variables aleatorias. En el GPR esta función tiene un papel crucial ya que pese a que es un método matemáticamente no dependiente de los datos se debe escoger una función de kernel que se ajuste al comportamiento de estos. Cada kernel se corresponde con un conjunto

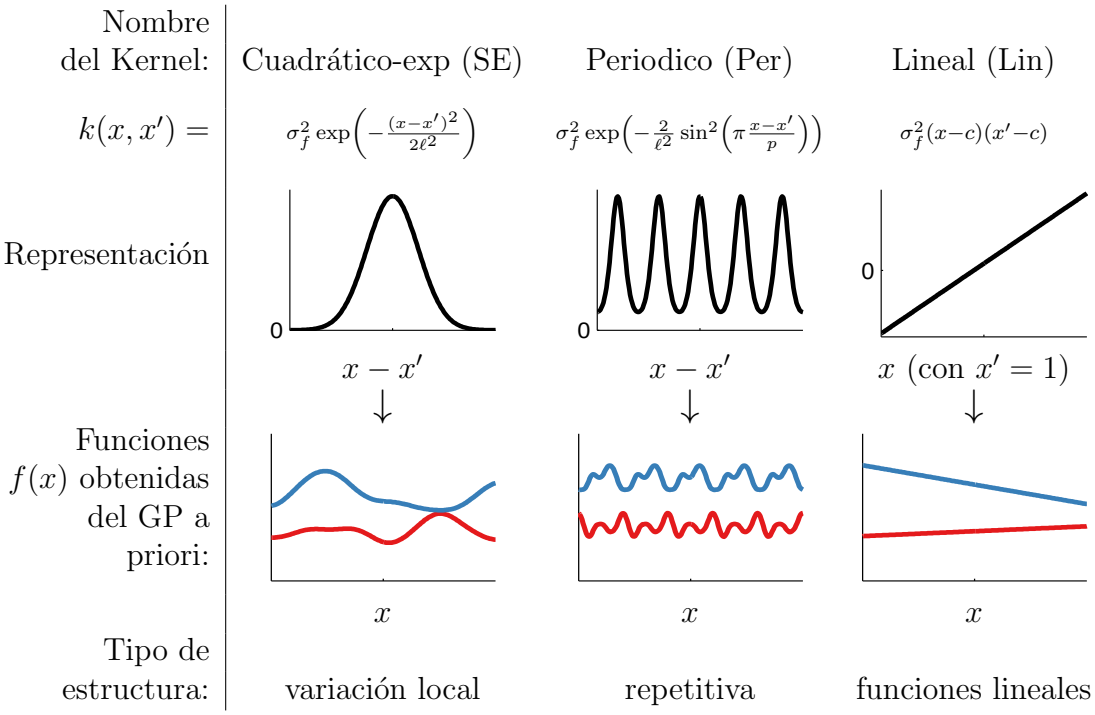

Figura 2.4: Ejemplo de distintas estructuras representadas por diferentes kernels. [18]

de suposiciones que se hacen sobre la función que se quiere modelar. Resulta notorio destacar que la mayoría de los kernel hacen uso de la distancia euclídea dentro de sus expresiones como herramienta de correlación. Algunos de los kernel más utilizados en GPR son:

- **Función cuadrática exponencial, SE:** Probablemente se trate del kernel más utilizado en el ámbito del GPR.

$$k_{SE}(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \exp\left(-\frac{1}{2}\sum_{k=1}^{d}\frac{(x_{i,k} - x_{j,k})^2}{l_k^2}\right) \tag{2.24}$$

El parámetro σ_f^2 es la varianza del Proceso Gaussiano mientras que el parámetro l_k^2, el cual se denomina longitud característica, expresa la correlación en la k-ésima dimensión del espacio de características. Se trata de una función estacionaria y anisotrópica (excepto si $l_k = l, \forall k$ caso en el que es isotrópica).

– **Función cuadrática racional, RQ:**

$$k_{RQ}(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \left(1 + \frac{1}{2\alpha} \sum_{k=1}^{d} \frac{(x_{i,k} - x_{j,k})^2}{l_k^2}\right)^{-\alpha} \tag{2.25}$$

Este kernel se puede ver como una suma infinita de kernels SE con diferentes longitudes características. Los GPs con este kernel esperan encontrarse con funciones que varían suavemente alrededor de estas longitudes. El parámetro α es un factor de peso para grandes y pequeñas variaciones de escala. Cuando $\alpha \to \infty$ el RQ es equivalente a SE.

En caso de que la función no pueda verse caracterizada por un solo modelo, existe la posibilidad de combinar kernels para modelar funciones más complejas a través de la suma o producto entre ellos. El kernel resultante será capaz de modelar estructuras que con kernels básicos no se podrían tratar. Sin embargo, se ha de tener en cuenta que el incremento de complejidad del kernel se traduce en un mayor coste computacional. La figura 2.5 muestra un ejemplo de selección de kernel de tal forma que conforme se avanza en los niveles del árbol, la complejidad del kernel aumenta.

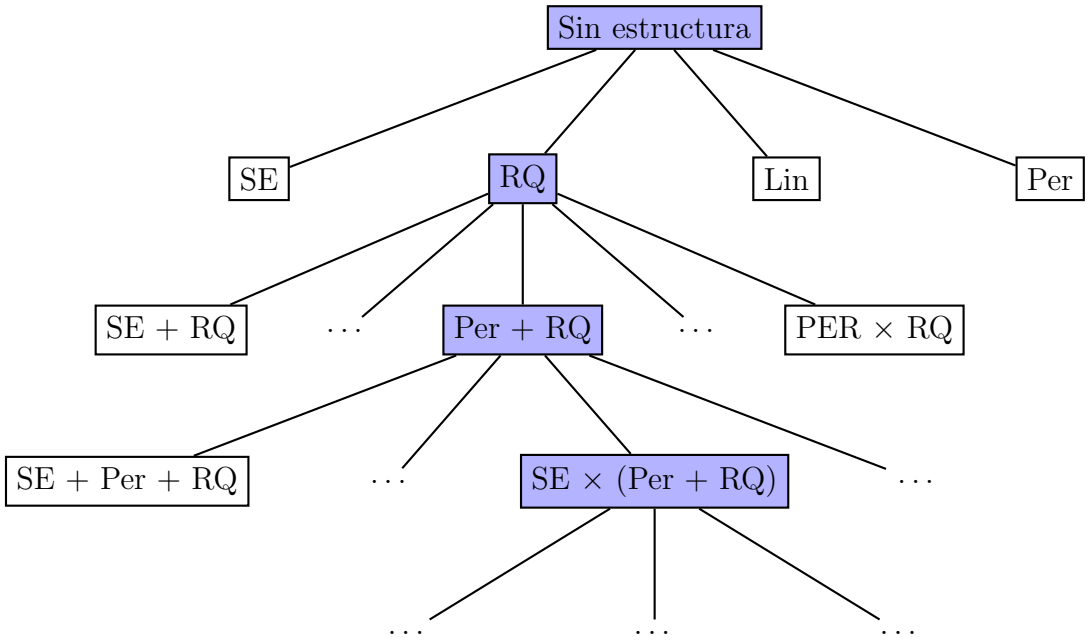

Figura 2.5: Un ejemplo de patrón de búsqueda de kernel con un incremento de complejidad del modelo conforme crece la expresión del kernel. [18]

2.5. Ajuste de los hiperparámetros

A la hora de escoger un kernel para el GP hay ciertas propiedades que son sencillas de determinar a través del contexto como puede ser si la señal es o no estacionaria. Sin embargo, hay otras propiedades de las cuales tenemos una vaga información a priori. Este es el caso de parámetros como la longitud característica o el factor de peso vistos en la sección anterior. A este tipo de parámetros del kernel se les denomina hiperparámetros y para que el proceso pueda ser modelado de forma adecuada han de ajustarse a los datos. Este proceso de ajuste de hiperparámetros constituye la fase de entrenamiento del GP. Como ejemplo de hiperparámetros, para el caso de la función cuadrática exponencial,

$$\mathcal{H} = \left\{l, \sigma_f^2, \sigma^2\right\} \tag{2.26}$$

Existen diversos métodos para optimizar estos hiperparámetros como puede ser la búsqueda en rejilla. Sin embargo, en el caso de los GP predomina el uso de dos técnicas: validación cruzada y selección Bayesiana de modelos [17]. En este trabajo se ha hecho uso de esta última.

2.5.1. Selección Bayesiana de modelos

El objetivo en la Selección Bayesiana de modelos es conocer la distribución a posteriori. Sin embargo, obtener de forma analítica esta expresión resulta en ocasiones imposible pues implica calcular una integral que puede resultar intratable. Es por ello que se requieren de métodos alternativos como la aproximación analítica de la distribución o la búsqueda de parámetros, en nuestro caso hiperparámetros, que maximicen la verosimilitud.

A través del Teorema de Bayes, se puede expresar la distribución a posteriori de los hiperparámetros como

$$p(\mathcal{H}|\mathbf{X}, \mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{X}, \mathcal{H})p(\mathcal{H})}{p(\mathbf{y}|\mathbf{X})} \tag{2.27}$$

Donde $p(\mathbf{y}|\mathbf{X}, \mathcal{H})$ es la verosimilitud marginal que se debe maximizar y $p(\mathcal{H})$ es la distribución a priori de los hiperparámetros que describe el conocimiento previo que se tiene de estos. Como normalmente ese conocimiento es escaso o nulo, se suele dotar de una distribución uniforme para todo el rango de valores.

Esta verosimilitud marginal puede calcularse a través de la expresión:

$$p(\boldsymbol{y}|\mathbf{X}, \mathcal{H}) = \int p(\mathbf{y}|f, \mathbf{X}, \mathcal{H})p(f|\mathbf{X}, \mathcal{H})df \tag{2.28}$$

Aplicando la hipótesis de que sigue una distribución normal llegamos a la conclusión de que $\mathbf{y} \sim \mathcal{N}(0, K_{\mathbf{yy}})$, y con ello se puede obtener analíticamente el resultado de la integral cuyo logaritmo es:

$$\log p(\mathbf{y}|\mathbf{X}, \mathcal{H}) = -\frac{1}{2}\mathbf{y}^T K_{\mathbf{yy}}^{-1}\mathbf{y} - \frac{1}{2}\log|K_{\mathbf{yy}}| - \frac{n}{2}\log 2\pi \tag{2.29}$$

donde $K_{\mathbf{yy}} = K_{ff} + \sigma^2_{ruido} I$ es la matriz de covarianza de los valores de salida $\mathbf{y}$ (y K_{ff} es la matriz de covarianza de la salida sin ruido). A esta función se le suele denominar logaritmo negativo de la función de probabilidad conjunta (NLL, Negative Log-Likehood). Los tres términos de la expresión pueden tener una interpretación: el primer término representa como el modelo se ajusta a los datos, el segundo representa la complejidad del método dependiendo únicamente del kernel y las entradas al sistema y el tercero es una constante de normalización.

Por tanto, al no poder calcular la distribución a posteriori analíticamente, se opta por maximizar la verosimilitud con respecto a los hiperparámetros. Para ello, se hace necesario calcular su gradiente con respecto al i-ésimo elemento de $\mathcal{H}$ de la forma:

$$\frac{\partial \log p(\mathbf{y}|\mathbf{X}, \mathcal{H})}{\partial \mathcal{H}_i} = -\frac{1}{2} tr\left(K_{\mathbf{yy}}^{-1}\frac{\partial K_{\mathbf{yy}}}{\partial \mathcal{H}_i}\right) + \frac{1}{2}\mathbf{y}^T K_{\mathbf{yy}}^{-1}\frac{\partial K_{\mathbf{yy}}}{\partial \mathcal{H}_i}K_{\mathbf{yy}}^{-1}\mathbf{y} \tag{2.30}$$

Esta forma permite el uso de cualquier optimización basada en método de gradiente. La función objetivo no es necesariamente convexa por lo que el método puede converger en máximos locales. Una posible solución a este problema sería iniciar múltiples búsquedas basadas en gradiente y luego escoger el punto óptimo que más maximice la distribución marginal. Es en este punto del algoritmo donde se debe tener en cuenta el coste computacional del algoritmo ya que este método requiere de la inversión de la matriz $K_{\mathbf{yy}}$ de tamaño $n \times n$, y, por tanto, un coste de $\mathcal{O}(n^3)$. Con el objetivo de disminuir dicho coste, es común que dado que la matriz de covarianza es simétrica y semidefinida positiva, hacer uso de la Factorización de Cholesky [22] en el proceso de inversión de la matriz.

Capítulo 3

Estimación de tensión futura con GPR

La presente sección trata de abordar la explicación de cómo, a través del GPR, se puede estimar la tensión futura de una batería dada las siguientes casuísticas:

- **Perfil de corriente y temperatura futura conocido**
- **Perfil de corriente futura conocido pero no de temperatura**
- **Perfil de temperatura futura conocido pero no de corriente**

Como explicación más genérica se trata el caso de perfil de corriente y temperatura futuro conocido siendo los otros casos completamente análogos a este. Se asume que se cuenta con los valores de tensión, corriente y temperatura del pasado de la batería ya que será necesario apoyarse en ellos para hacer tanto el entrenamiento como la predicción. Los datos recogidos en la base de datos están tomados cada 15 minutos. Sin embargo, se ha comprobado que submuestrear la base de datos en un factor 4, obteniéndose así muestras de 1 hora reduce la carga del algoritmo sin deteriorar, de forma apreciable, los resultados.

En el proceso de entrenamiento es donde se han de ajustar los valores de los hiperparámetros del kernel. Para ello, si los datos de entrada están muestreados cada 1 hora, se entrena al modelo GPR para ser capaz de predecir la tensión de la batería en una muestra futura.

Se denotan de la forma V_t, I_t y T_t a los valores discretos de tensión, corriente y temperatura en el instante de referencia t. El objetivo es estimar la tensión futura de

la batería en los instantes $t+1, \ldots, t+L$. El valor de L representa el horizonte de predicción y, en el caso concreto de un sistema fotovoltaico, se fija a 48 horas ya que, como se ha estudiado en el marco del proyecto, que este es el tiempo que precisaría el sistema para tomar las medidas necesarias (como subir la tensión de flotación) en caso de una posible caída de la instalación. Para poder realizar la estimación, se debe contar con los valores futuros de corriente y temperatura en dichos instantes. Por tanto, con el fin de estimar la tensión futura $V_{t+1}, \ldots, V_{t+L}$, se hace uso de los valores corriente $I_{t+1}, \ldots, I_{t+L}$ y de temperatura $T_{t+1}, \ldots, T_{t+L}$.

3.0.1. Predicción de tensión a una muestra

Antes de abordar el caso más complejo de estimar a L muestras futuras, se plantea el caso más simple de predicción a una única muestra. Esta técnica, denominada *One-step voltage prediction* [10], busca estimar la muestra de tensión correspondiente al tiempo $t+1$ a través de las muestras pasadas de $t, \ldots, t-M$. Nos referiremos a M como la longitud de memoria (*Memory length*) y representa el número de muestras pasadas utilizadas para la estimación de tensión.

Se agrupan las medidas pasadas y los valores futuros de corriente y temperatura en un vector $\mathbf{x}_t$ de longitud $3M+2$ de la forma

$$\mathbf{x}_t = [I_{t+1} \quad T_{t+1} \quad V_t \quad I_t \quad T_t \quad \ldots \quad V_{t-M} \quad I_{t-M} \quad T_{t-M}]^T \tag{3.1}$$

La salida y_t es la tensión obtenida en el instante $t+1$ donde no se considera la existencia de ruido

$$y_t = V_{t+1} \tag{3.2}$$

La esencia del algoritmo reside en modelar la función de probabilidad conjunta de n salidas $y_{t1}, y_{t2}, \ldots, y_{tn}$ como una gaussiana de la forma

$$p(y_{t1}, y_{t2}, \ldots, y_{tn}) \sim \mathcal{N}(0, K_{\mathbf{f},\mathbf{f}}) \tag{3.3}$$

donde se toma media nula ya que, al inicio del proceso, se puede sustraer su valor de todos los valores de entrenamiento y se puede volver a incluir en la solución final. La matriz de covarianza $K_{\mathbf{f},\mathbf{f}} \in \mathbb{R}^{n \times n}$ viene dada por

$$\left[K_{\mathbf{f},\mathbf{f}}\right]_{ij} = k(\mathbf{x}_{t_i}, \mathbf{x}_{t_j}) \tag{3.4}$$

donde $K_{\mathbf{f},\mathbf{f}}$ es el kernel que modela la similitud entre las variables $\mathbf{x}_{t_i}$ y $\mathbf{x}_{t_j}$. En este caso, entradas de tensión, corriente y temperatura similares provocarán una estimación de tensión similar.

Los hiperparámetros $\mathcal{H}$ del kernel se ajustan directamente a través de los valores de entrenamiento. Los datos de entrada de la optimización de descenso por gradiente serán n pares de entradas y salidas $\mathcal{D} = (\mathbf{x}_{t_i}, y_{t_i})$ donde $i = 1, \ldots, n$. El logaritmo negativo de la función de probabilidad conjunta de los datos de entrenamiento viene dado a partir de la distribución 3.3. De esta optimización resultan unos nuevos valores de hiperparámetros $\widehat{\mathcal{H}}$.

$$NLL(\mathcal{D}, \mathcal{H}) = -\frac{1}{2}\mathbf{y}^T K_{\mathbf{f},\mathbf{f}}^{-1}\mathbf{y} - \frac{1}{2}\log|K_{\mathbf{f},\mathbf{f}}| - \frac{n}{2}\log 2\pi \tag{3.5}$$

donde $\mathbf{y} = [y_{t1}, y_{t2}, \ldots, y_{tn}]^T$ y los valores de la matriz $K_{\mathbf{f},\mathbf{f}}$ vienen rellenados a través de 3.4 para los hiperparámetros $\mathcal{H}$. Tras el proceso de gradiente descendiente, se obtiene un argumento que minimiza el NLL

$$\widehat{\mathcal{H}} = \text{argmin NLL}(\mathcal{D}, \mathcal{H}) \tag{3.6}$$

Algoritmo 1 Fase de entrenamiento en GPR predictor a una muestra

1: **Entrada**: Datos de entrenamiento $\mathcal{D} = (x_{t_1}, y_{t_1}), \ldots, (x_{t_N}, y_{t_n})$
2: **Entrada**: Kernel $k(.,.)$ e hiperparámetros $\widehat{\mathcal{H}}$
3: Ejecución de la optimización: $\widehat{\mathcal{H}} = \text{argmin NLL}(\mathcal{D}, \mathcal{H})$
4: Cálculo de la inversa de la matriz de covarianza $K_{\mathbf{f},\mathbf{f}}^{-1}$ para $\widehat{\mathcal{H}}$ usando 3.4
5: **Salida**: Hiperparámetros óptimos $\widehat{\mathcal{H}}$
6: **Salida**: Matriz de covarianza inversa $K_{\mathbf{f},\mathbf{f}}^{-1}$

Con el objetivo de estimar en la fase de test la salida de tensión en el instante y_t utilizaremos los valores de hiperparámetros optimizados $\widehat{\mathcal{H}}$ y los datos de entrenamiento $\mathcal{D}$. Como se ve en la ecuación 3.7, a través de la propiedad [2] se puede expresar la distribución conjunta de las salidas $y_{t1}, y_{t2}, \ldots, y_{tN}$ y de la salida objetivo y_t como una nueva distribución gaussiana.

$$p(y_{t1}, y_{t2}, \ldots, y_{tn}, y_t) = \mathcal{N}\left(0, \quad \begin{bmatrix} K_{\mathbf{f},\mathbf{f}} & K_{*,\mathbf{f}} \\ K_{\mathbf{f},*} & K_{*,*} \end{bmatrix}\right) \tag{3.7}$$

donde $K_{\mathbf{f},\mathbf{f}}$ fue obtenida en la expresión 3.4, $K_{*,\mathbf{f}} \in \mathbb{R}^{n\times 1}$ es el vector de covarianza evaluado en cada entrada x_{t_i}de la base de datos de entrenamiento $\mathcal{D}$ y la entrada de test $\mathbf{x}_t$

$$K_{*,\mathbf{f}} = k(\mathbf{x}_{t_i}, \mathbf{x}_t), i = 1, \ldots, n \tag{3.8}$$

mientras que $K_{*,*} = k(\mathbf{x}_t, \mathbf{x}_t)$ Es importante remarcar que tanto $K_{\mathbf{f},*}$ como $K_{*,\mathbf{f}}$ y $K_{*,*}$ han hecho uso de los nuevos hiperparámetros $\widehat{\mathcal{H}}$. Como vimos en 2.23, como la distribución conjunta es gaussiana, la distribución condicional también lo es,

$$p(y_t|y_{t1}, y_{t2}, \ldots, y_{tn}) = \mathcal{N}(\mu_t, \sigma_t^2) \tag{3.9}$$

donde la media μ_t y la varianza σ_t^2 vienen dados por

$$\mu_t = K_{*,\mathbf{f}} K_{\mathbf{f},\mathbf{f}}^{-1} \mathbf{y} \tag{3.10}$$

$$\sigma_t^2 = K_{*,*} - K_{*,\mathbf{f}} K_{\mathbf{f},\mathbf{f}}^{-1} K_{\mathbf{f},*} \tag{3.11}$$

El valor de media de la nueva distribución es el mejor estimador de la tensión a una muestra,

$$\widehat{V}_{t+1} = \mu_t \tag{3.12}$$

mientras que la varianza es la incertidumbre de la predicción y puede ser utilizada para especificar un intervalo de confianza del 95 %

$$[\mu_t - 1{,}96\sigma_t, \mu_t + 1{,}96\sigma_t] \tag{3.13}$$

Algoritmo 2 Fase de test en GPR Predictor a una muestra

1: **Entrada**: Datos de entrenamiento $\mathcal{D} = (\mathbf{x}_{t_1}, y_{t_1}), \ldots, (\mathbf{x}_{t_N}, y_{t_n})$
2: **Entrada**: Kernel $k(.,.)$ e hiperparámetros $\widehat{\mathcal{H}}$
3: **Entrada**: Matriz de covarianza inversa $K_{\mathbf{f},\mathbf{f}}^{-1}$
4: **Entrada**: Mediciones $V_t, I_t, T_t \ldots V_{t-M}, I_{t-M}, T_{t-M}$
5: **Entrada**: Corriente y temperatura futura I_{t+1}, T_{t+1}
6: $\mathbf{y} = [y_{t1}, y_{t2}, \ldots, y_{tn}]^T$
7: $\mathbf{x}_t = [I_{t+1} \quad T_{t+1} \quad V_t \quad I_t \quad T_t \quad \ldots \quad V_{t-M} \quad I_{t-M} \quad T_{t-M}]^T$
8: $K_{*,\mathbf{f}} = k(\mathbf{x_{t_i}}, \mathbf{x_t}), i = 1, \ldots, N$
9: $K_{*,*} = k(\mathbf{x_t}, \mathbf{x_t})$
10: $\mu_t = K_{*,\mathbf{f}} K_{\mathbf{f},\mathbf{f}}^{-1} \mathbf{y}$
11: $\sigma_t^2 = K_{*,*} - K_{*,\mathbf{f}} K_{\mathbf{f},\mathbf{f}}^{-1} K_{\mathbf{f},*}$
12: **Salida**: Media y varianza estimada μ_t y σ_t^2

3.0.2. Predicción recursiva multipaso de tensión (R-MSVP)

El objetivo del R-MSVP es ser capaz de estimar el valor de tensión de la muestra $t + L$. Para ello, la idea es tratar la tensión estimada $y_t^{(m)} = V_{t+m} \quad | \quad m = 1, \ldots, L$ como si fuera una tensión real a la hora de estimar $y_t^{(m+1)} = V_{t+m+1}$. El estimador de tensión a una muestra en un instante t estima el valor de media de tensión $\mu_t^{(1)}$ a través de la ecuación 3.10. El R-MSVP, con el fin de estimar la tensión correspondiente al instante $t + 2$, se apoya en la tensión predicha en $t + 1$, $\mu_t^{(1)}$ tomándola como un valor realmente medido. Este valor, junto a los demás parámetros de tensión corriente y temperatura, en los instantes $t, t-1, \ldots, t-M$ se reinyecta a los valores de corriente y temperatura $I_{t+1}, I_{t+2}, T_{t+1}$ y T_{t+2} en un algoritmo de estimador de tensión a una

muestra para así obtener $\mu_t^{(2)}$. Siguiendo este proceso, es posible llegar a predecir L muestras de tensión futuras para poder así llegar a las 48 horas de margen de reacción de BMS frente a posibles carencias de energía . Esta técnica ha sido utilizada con éxito en [10]-[7].

El proceso de entrenamiento del R-MSVP es el mismo que en el algoritmo de estimación a una muestra. Por otro lado, en el proceso de test, el R-MSVP en un instante t estima, de forma ordenada, las tensiones correspondientes a las muestras $t+1, \ldots, t+L$. La predicción de tensión correspondiente a los instantes $t+m$ se realiza a través del algoritmo de estimación a una muestra cambiando, en cada iteración, la entrada al sistema de la forma

$$\mathbf{x}_t = [I_{t+m} \quad T_{t+m} \quad \mathbf{V}_t^{(m)} \quad \mathbf{I}_t^{(m)} \quad \mathbf{T}_t^{(m)}]^T \tag{3.14}$$

donde

$$\mathbf{V}_t^{(m)} = [\mu_t^{(m-1)} \quad \ldots \quad \mu_t^{(1)} \quad V_t \quad \ldots \quad V_{t-M+m-1}]^T \tag{3.15}$$

$$\mathbf{I}_t^{(m)} = [I_{t+m-1} \quad \ldots \quad I_{t+1} \quad I_t \quad \ldots \quad I_{t-M+m-1}]^T \tag{3.16}$$

$$\mathbf{T}_t^{(m)} = [T_{t+m-1} \quad \ldots \quad T_{t+1} \quad T_t \quad \ldots \quad T_{t-M+m-1}]^T \tag{3.17}$$

El proceso de test, se ilustra en el algoritmo 3

Algoritmo 3 Fase de test en R-MSVP

1: **Entrada**: Datos de entrenamiento $\mathcal{D} = (\mathbf{x}_{t_1}, y_{t_1}), \ldots, (\mathbf{x}_{t_N}, y_{t_n})$
2: **Entrada**: Kernel $k(.,.)$ e hiperparámetros $\widehat{\mathcal{H}}$
3: **Entrada**: Matriz de covarianza inversa $K_{\mathbf{f},\mathbf{f}}^{-1}$
4: **Entrada**: Mediciones $V_t, I_t, T_t \ldots V_{t-M,} I_{t-M}, T_{t-M}$
5: **Entrada**: Corriente y temperatura futura $I_{t+1}, T_{t+1}, \ldots, I_{t+L}, T_{t+L}$
6: $\mathbf{y} = [y_{t1}, y_{t2}, \ldots, y_{tn}]^T$
7: **for** $m = 0$ to L **do**
8: Reajustar $\mathbf{x}_t^{(m)}$ como en 3.14
9: $K_{*,\mathbf{f}} = k(\mathbf{x}_t^{(m)}, \mathbf{x}_{t_i}), i = 1, \ldots, N$
10: $K_{*,*} = k(\mathbf{x}_t^{(m)}, \mathbf{x}_t^{(m)})$
11: $\mu_t^{(m)} = K_{*,\mathbf{f}} K_{\mathbf{f},\mathbf{f}}^{-1} \mathbf{y}$
12: $\sigma_t^{2^{(m)}} = K_{*,*} - K_{*,\mathbf{f}} K_{\mathbf{f},\mathbf{f}}^{-1} K_{\mathbf{f},*}$
13: **end for**
14: **Salida**: Media y varianza de tensión estimada $\widehat{V}_{t+m} = \mu_t^{(m)}$, $m = 1, \ldots, L$

Este algoritmo recursivo es el que ha sido utilizado en la totalidad de las pruebas de este trabajo ya que es el método que mejores resultados ha obtenido en la literatura frente a otros métodos como el Predictor paralelo multipaso de tensión (P-MSVP)

[10]. Habiendo fijado el parámetro de horizonte de predicción a $L = 48$ muestras con el objetivo de que el BMS sea capaz de adaptarse en función de las estimaciones de predicción, quedarán por fijar los parámetros de longitud de memoria (M), el kernel utilizado y los días escogidos tanto para el entrenamiento como para el test del algoritmo. Para ello, es importante tener en cuenta que el objetivo final de nuestro sistema es que sea implementable y por ello ha de existir un compromiso entre la carga computacional y las métricas de resultados.

Teniendo en cuenta que nuestra base de datos cuenta con multitud de años completos se ha considerado apropiado escoger como base de datos de entrenamiento un año completo. En concreto, se ha escogido 2008 ya que es el año que más estabilidad presenta en la base de datos puesto que la batería se acababa de renovar y partíamos de su estado óptimo.

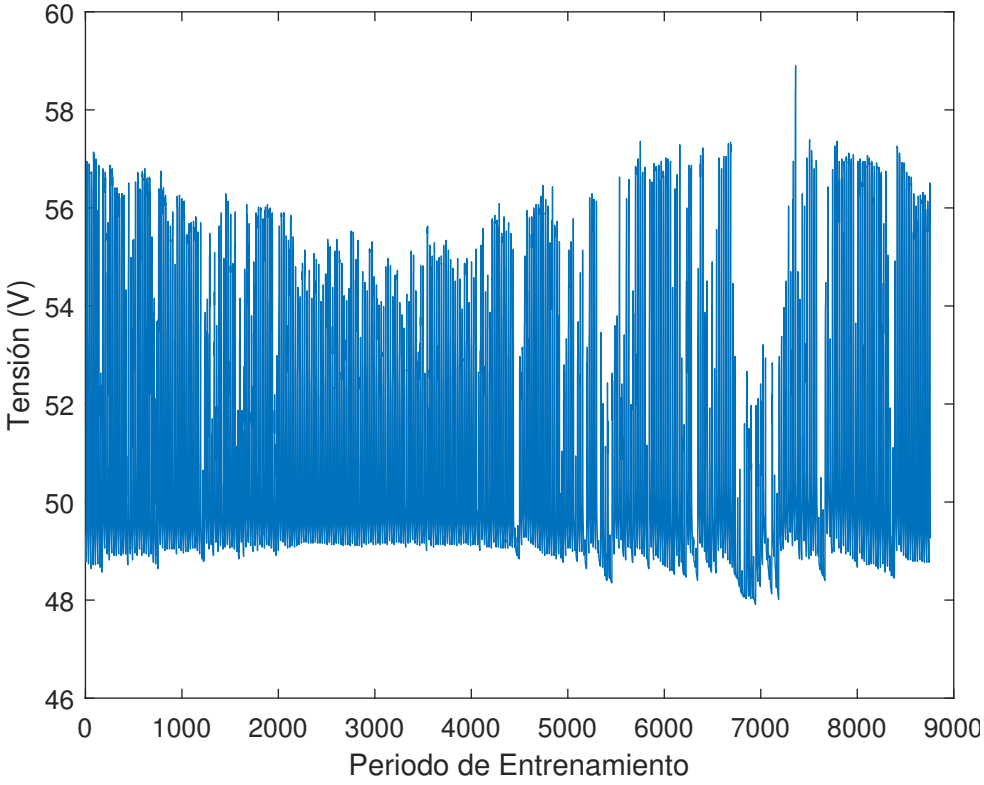

Figura 3.1: Tensión en la batería en el año 2008

Debido a que el GPR tiene una complejidad de $\mathcal{O}(n^3)$, tratar más de un millar de muestras en la fase de entrenamiento se vuelve inviable sin requerir de aproximaciones. Es por ello que, dentro de ese año de entrenamiento se han escogido muestras aleatorias de tal forma que en total conformen el tamaño equivalente a un mes (720 muestras de tensión corriente y temperatura). Escogiéndose de forma uniforme a lo largo de

un año, y no solamente un mes seguido, le damos la oportunidad al algoritmo de visualizar el comportamiento de la tensión en múltiples casuísticas de forma que sea capaz de tratarlas en la fase de test. Se ha probado a elegir un mes completo como enero obteniéndose buenos resultados en la estación de invierno. Sin embargo, en el momento en el que tratamos de abordar un día de verano sin reentrenar la estimación empeora de forma considerable.

En la Figura 3.2 se puede ver de forma visual el conjunto matricial que se utiliza como entrada al algoritmo. Se ha probado a normalizar los valores de entrada sin apreciar una mejora apreciable, por lo que se ha preferido que mantengan sus escalas originales.

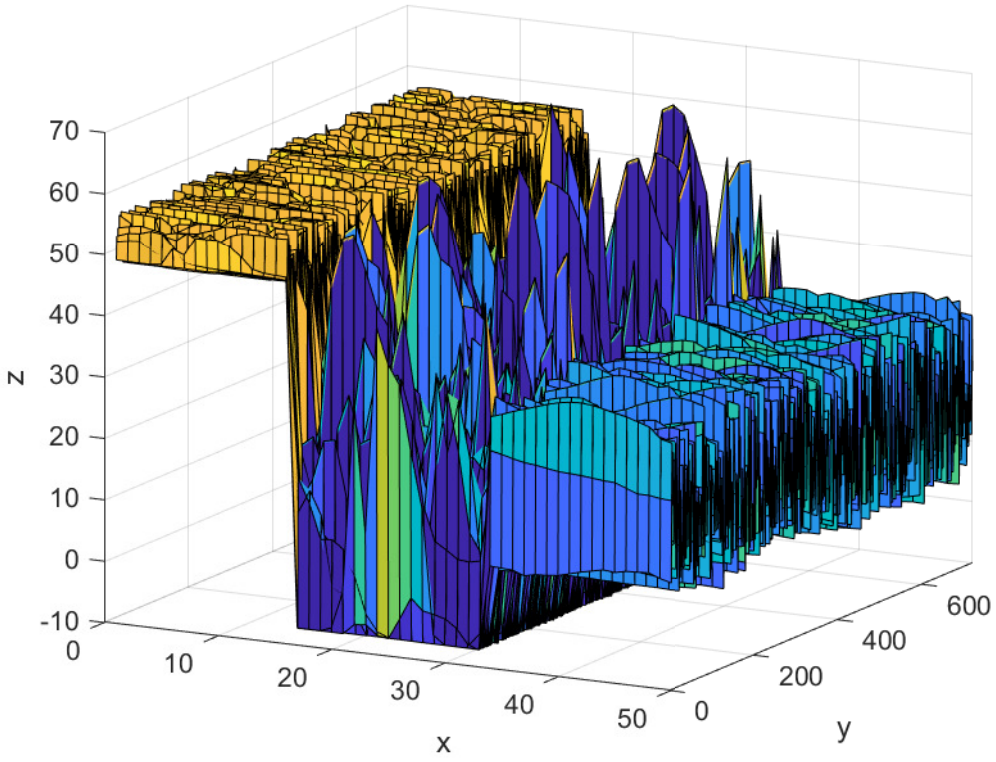

Figura 3.2: Representación visual de la base de datos de entrenamiento del algoritmo. Incluye como parámetros de entrada a la tensión, la corriente y la temperatura. El eje y representa las 720 muestras de entrenamiento. El eje x representa las M muestras pasadas en cada instante, incluyendo las muestras $t+1$ para el caso de la temperatura y la corriente. El eje z representa el valor de las amplitudes de las ondas de entrada: voltios, amperios y grados centígrados

En la fase de test, pese a que la estimación se realiza a 48 horas futuras se suele, al finalizar la estimación, repetir el proceso desplazando la posición actual de t a $t+1$

llegándose a desplazar muestra a muestra hasta un mes de tal forma que se simule un sistema real donde el algoritmo estima a 48 horas, una hora más tarde repite el proceso y así sucesivamente. A la hora de elegir dicho mes, al haber escogido una base de entrenamiento que toma muestras de todo el año, no existe una diferencia significativa entre un mes u otro del año. Por eso, y por escoger datos pertenecientes a un año diferente al de entrenamiento (validación cruzada) de la misma manera que se haría en un sistema realista, se ha escogido el mes de mayo de 2009 como objetivo de estimación.

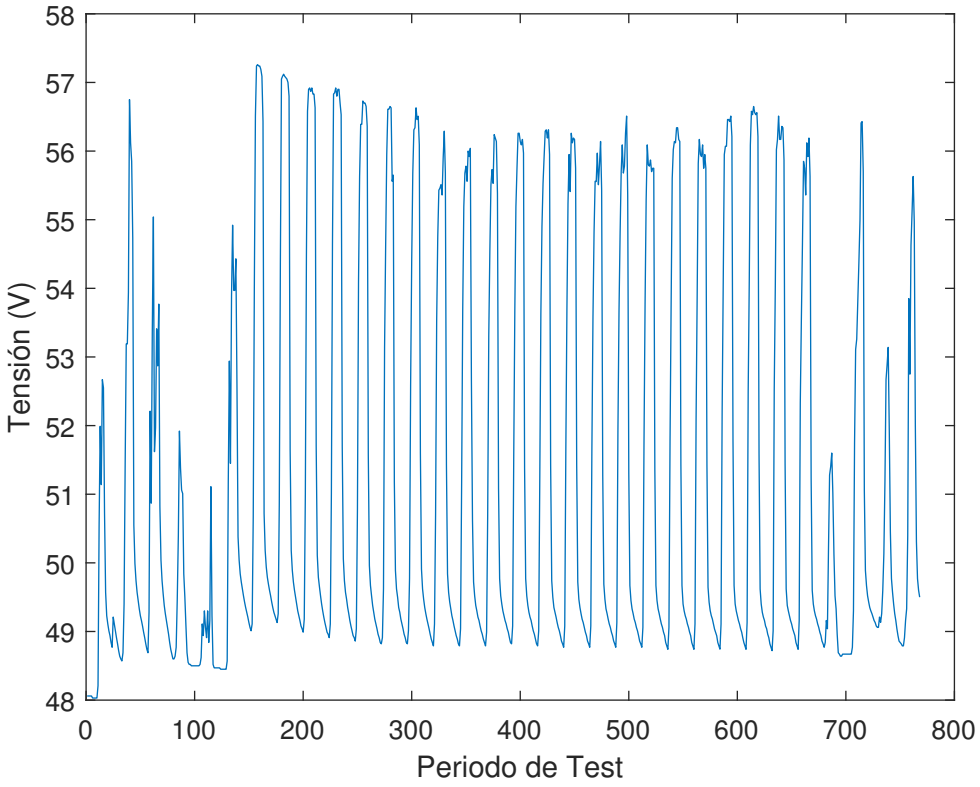

Figura 3.3: Forma de onda de tensión en la batería en el mes de test

3.0.3. Elección Kernel

La elección del kernel es compleja ya que se trata del único parámetro que realmente se escoge en función del tipo de datos. En [10] proponen una kernel compuesto por la suma de dos SE y un kernel de red neuronal. Siendo que se trata de una función formada a través de la suma de tres kernels elementales el objetivo era abordar un amplio espectro de posibles no linealidades. Sin embargo, este tipo de kernels complejos aumentan la coste computacional conforme mayor es su estructura (Figura 2.5). Se ha probado el uso de este kernel pero su complejidad computacional y los resultados poco certeros aportados, han llevado a su descarte.

En este trabajo se ha observado como, para la base de datos concreta que se maneja, un kernel con estructura simple como el RQ es capaz de modelar el comportamiento de la señal de tensión siguiendo, incluso, las variaciones que un paso por nube puede producir. Matlab aporta una herramienta llamada "Profiler"que ayuda a ver, en un scripts concreto, dónde se gasta la mayor parte del tiempo de ejecución. Como vemos, en la Figura 3.4, la llamada a la función del kernel ("covRQard") es la que más tiempo ha consumido en la totalidad de la ejecución resultando así un factor crítico en el proceso de implementación real.

Profile Summary

Function Name	Calls	**Total Time**	Self Time*	Total Time Plot (dark band = self time)
GPR_WITH_T_AND_C	1	79.968 s	11.604 s	
covRQard	65577	56.186 s	3.072 s	
covScale	65577	53.114 s	14.564 s	
covRQ	131054	38.550 s	4.478 s	
covMaha	131054	34.072 s	8.941 s	
covMaha>maha	65477	18.399 s	11.976 s	
minimize	1	16.340 s	0.403 s	
gp	100	15.843 s	0.168 s	
infExact	100	15.365 s	0.022 s	

Figura 3.4: Sección del Profiler ilustrando el tiempo consumido por cada función del algoritmo

3.0.4. Elección memoria pasada, M

Por otro lado, resulta necesario fijar un valor de memoria pasada, M. Este valor determina la ventana de muestras pasadas que toma la muestra t para estimar $t+1$. En [7]-[10] se plantea la posibilidad de que este parámetro esté relacionado con la constante de tiempo de la dinámica de la batería (y, en consecuencia, de su química). Sin embargo, tras haberse realizado, como puede verse en la Figura 3.5, un barrido de distintos valores se ha detectado que no existe una variación real en la estimación provocada por este parámetro. Se ha observado que para valores de $M < 15$ los valores de la matriz de covarianza de los datos de entrenamiento $K_{\mathbf{f},\mathbf{f}}$ son tan similares que Matlab no es capaz de aportar una solución exacta a su inversión. Este problema desaparece en caso de hacer uso de la función gp() de la librería GPML que aporta directamente los valores de media y varianza estimados. Pese a esta posibilidad que la

función gp() aporta, al computar todo el cálculo matricial en una misma función, se está recalculando en cada iteración el valor de $K_{\mathbf{f},\mathbf{f}}^{-1}$ siendo que es constante. Es por ello, que pese a la posibilidad de reducir el valor de M, y a la vista de que las métricas son similares con valores $M \geq 15$ se ha preferido no hacer uso de dicha función y realizar los cálculos matriciales sin librería y escoger $M = 15$ como parámetro fijo con el fin de reducir el tiempo de cómputo.

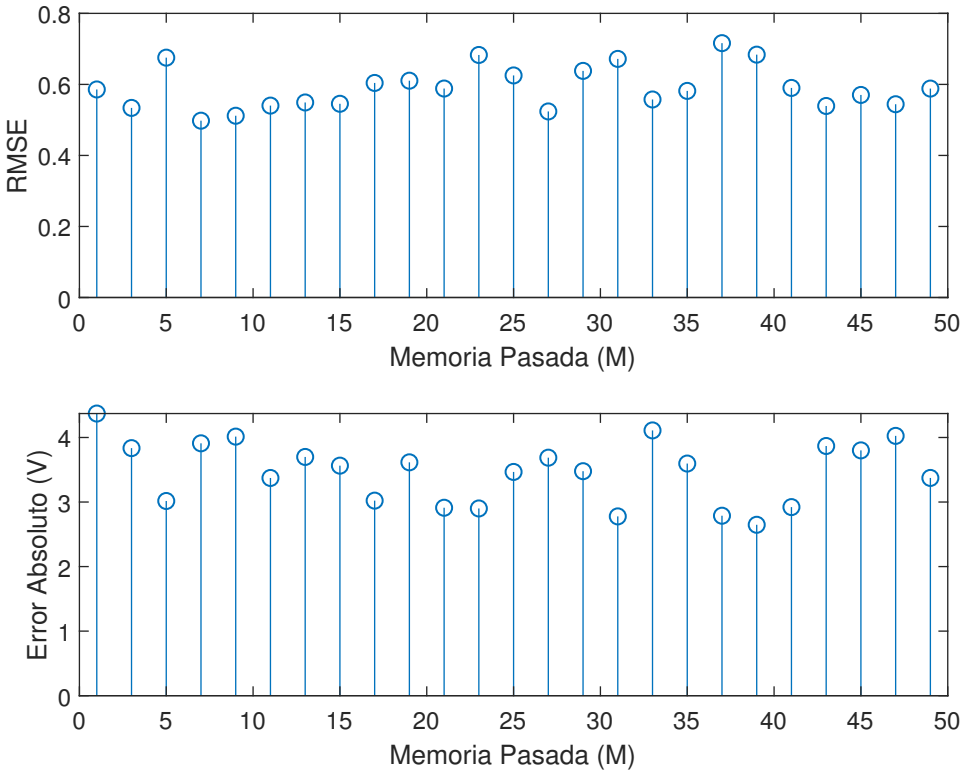

Figura 3.5: Barrido a lo largo de los posibles valores de memoria pasada M a través de la función gp()

3.0.5. Múltiples Expertos GPR como estimador de tensión

Debido a la multitud de tipos de días que existen a lo largo de un año completo, y con la intención de reducir el esfuerzo computacional que realiza un único algoritmo GPR al estimar con un solo entrenamiento una base de datos con amaneceres y anocheceres tan diversos, se ha realizado una implementación diferente a la habitual donde existen expertos GPR. Estos expertos GPR son especialistas en un tipo de día cada uno. Pese a que a priori se pueda pensar que para un año existen 4 tipos de días correspondientes cada uno de ellos con una estación del año, se ha comprobado que esta no es la forma de clasificación más precisa ya que no existe una diferencia real entre días de verano

y días de primavera. Con el objetivo de mejorar esta clasificación, se ha hecho uso de mapas autoorganizados (Anexo **??**) [12] y del algoritmo de K-*means* (Anexo **??**). La clasificación de la base de datos fue desarrollada por el grupo de investigación en [23] y este trabajo se centra en su aplicación al GPR. A través de esta clasificación, se pueden agrupar los distintos tipos de días que realmente afectan al comportamiento de la batería.

Para ello, se hace uso de la base de datos de Sigena de los años 2007 a 2009 y se toman como variables de entrada:

- Tensión de inicio de descarga
- Tensión de final de descarga
- Horas de carga
- Horas de descarga
- Carga almacenada

Tras entrenar el algoritmo, el mapa autoorganizado resultante y las distribuciones de las variables se representan en la Figura 3.6.

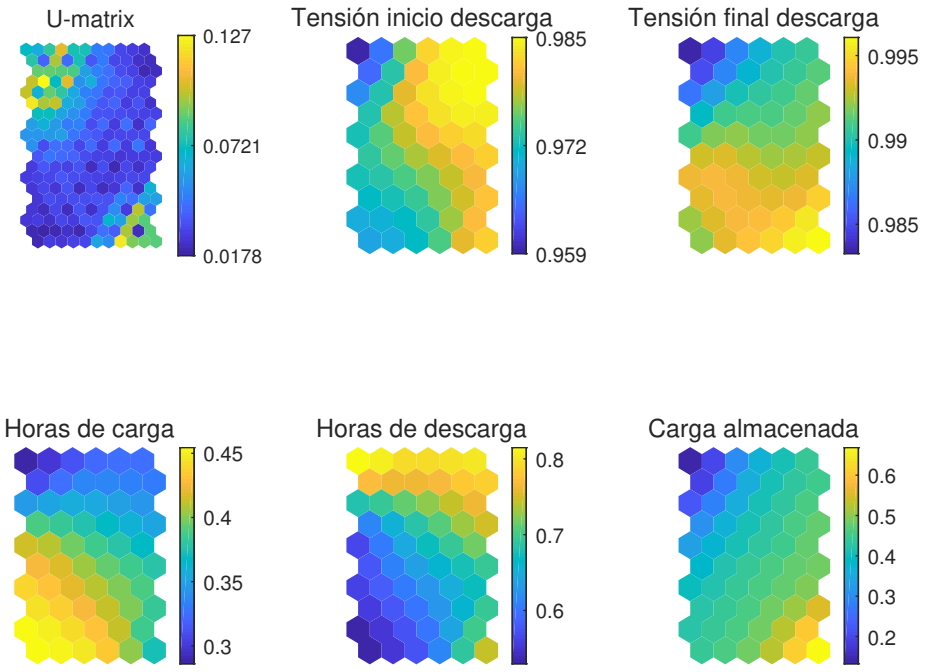

Figura 3.6: SOM de las variables

El algoritmo de agrupación K-*means* requiere que se le fije, a priori, el número de *clusters* a realizar. Este proceso se ha realizado de forma iterativa y se ha comprobado que a partir de 5 *clusters* no existía mucha diferencia en los tipos de días que aportaba. Estos *clusters* se pueden ver representados en la Figura 3.7.

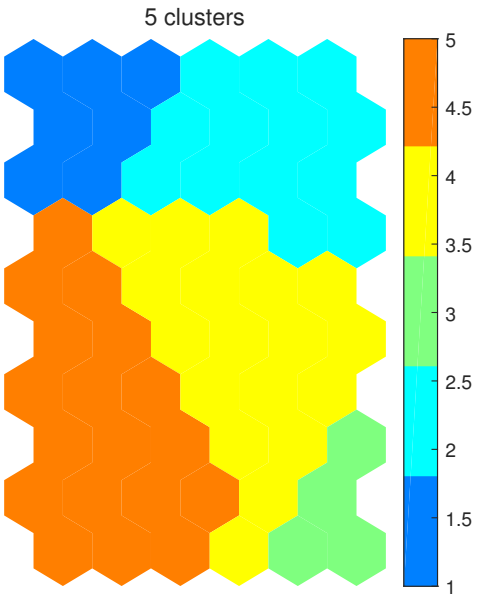

Figura 3.7: Agrupaciones del SOM

Se puede hacer una representación de la forma de onda media de cada agrupación, Figura 3.8.

- Día malo
- Día de Invierno con la batería en flotación
- Día de Invierno sin la batería en flotación
- Día de Primavera con la batería en flotación
- Día de Verano con la tensión en flotación

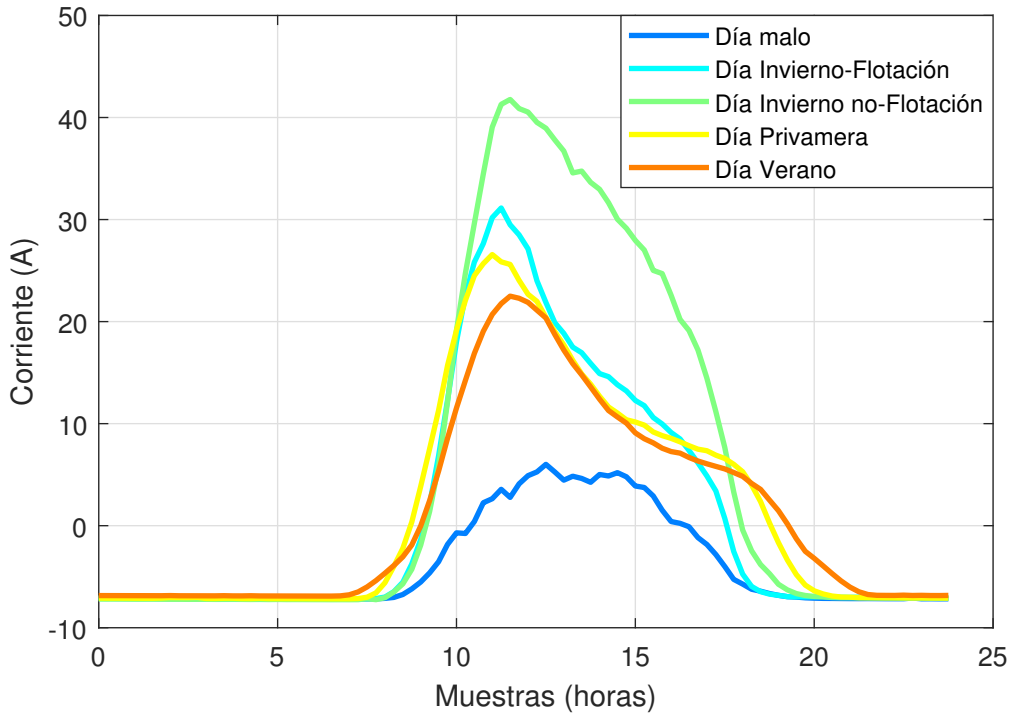

Figura 3.8: Tipos de días en la base de datos obtenidos a partir del uso del SOM y del algoritmo K-*means*

Una vez obtenidas las 5 bases de datos con los diferentes tipos de días, se entrena con cada una de ellas un GPR de forma individual de la misma forma que se hizo en los apartados previos obteniendo así diferentes valores de hiperparámetros para cada uno de los 5 expertos. Todos ellos comparten los parámetros que ya fijamos con anterioridad (Kernel, M, L). En la ejecución de la fase de test, se toma como válido el resultado del experto GPR que obtenga menor varianza en su estimación. Esta forma de elección del valor estimado es la más simple y a su vez la que mejores prestaciones ha aportado. Otra de las aproximaciones planteadas ha sido entrenar una pequeña red neuronal, de tipo perceptrón multicapa, que tomara como entradas las medias y las varianzas de cada experto y diera una salida que fuera tomada como la óptima. Sin embargo, los resultados eran similares y aumentaba el coste computacional.

3.0.6. Sparse GPR como estimador de tensión

Con el objetivo de abordar un mayor número de casuísticas en la fase de entrenamiento y con el fin de mantener un coste computacional reducido se hace uso de la herramienta de aproximación FITC Sparse. El proceso de estimación es equivalente al tratado en la sección 3.0.2 pero haciendo uso de la expresión de media y varianza

obtenida en 2.21. Como parámetros adicionales a las secciones previas, se incluye elegir el número (m) y la posición de las *inducing points* $\mathbf{X_u}$ que representan un subconjunto de la base de datos de entrenamiento. En la literatura, existen diversos métodos para obtener dicho subconjunto. El algoritmo de agrupación de puntos lejanos (FPC, *Farthes Point Clustering*) [19] con un coste computacional de $\mathcal{O}(nm)$ es el más complejo de todos ellos. Sin embargo, la mayor parte de la literatura opta por la selección de puntos o de forma aleatoria o equiespaciados en la base de datos, ambos con un coste de $\mathcal{O}(m)$. Se ha escogido este último por la sencillez de implementación y por su extendido uso en el sector. Una vez escogido el subconjunto, es posible optimizarlo si se pasa como parámetro, junto a los hiperparámetros, al algoritmo de gradiente descendente de tal forma que juntos minimicen la función objetivo.

En lo respectivo a la cantidad m de *inducing points* a seleccionar se ha realizado un barrido para explorar los resultados que FITC Sparse aporta en función de m. Los valores RMSE que se pueden ver en la Figura 3.10 tienden a variar ligeramente en la primera centena de *inducing points* manteniéndose prácticamente constantes de ahí en adelante. En la Figura 3.9 se puede ver el coste computacional tanto de la fase de entrenamiento como de test. Por tener el mínimo valor de error en el rango de la centena a la vez que mantiene un coste computacionalmente reducido, se ha escogido $m = 80$ como parámetro fijo para las pruebas.

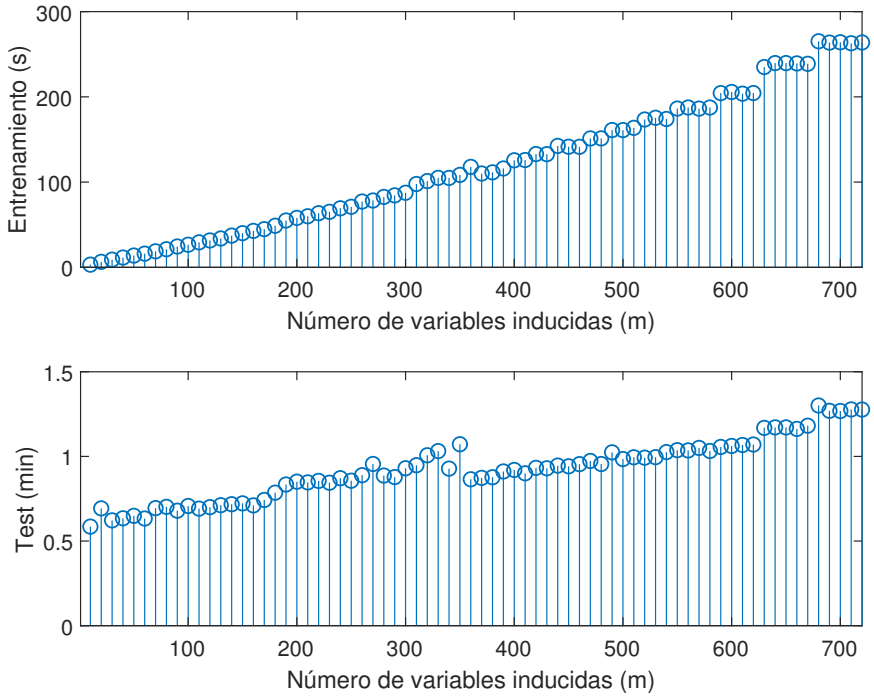

Figura 3.9: Duración de las fases de entrenamiento en segundos y de test en minutos para distinto número de *inducing points*

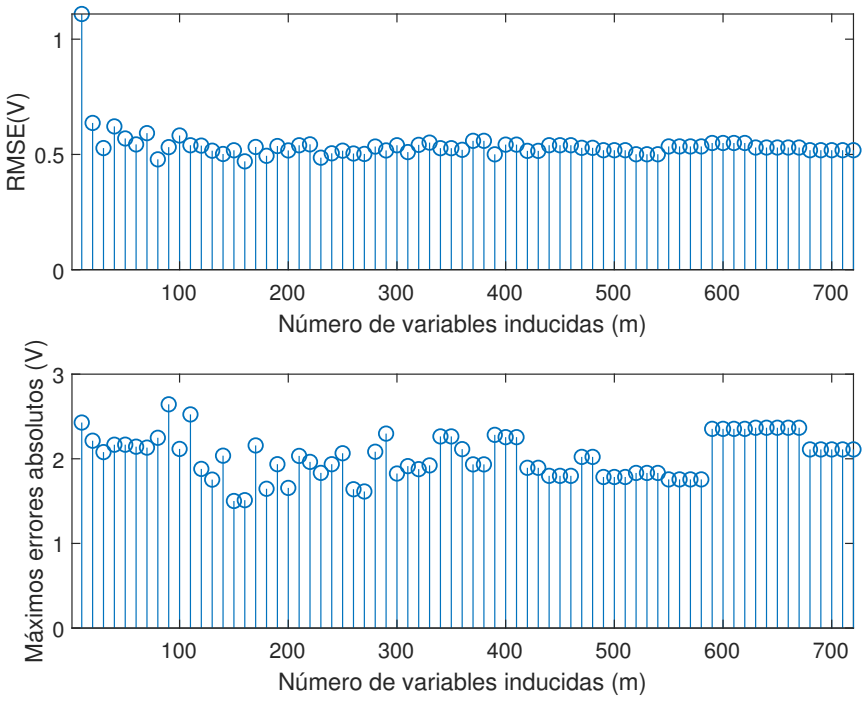

Figura 3.10: La figura superior representa el RMSE promedio en voltios de las estimaciones del mes completo. La figura inferior representa los máximos picos de error en voltios encontrados en la estimación del mes completo. Ambas se representan en función del número de *inducing points*

Algoritmo 4 Flujo del Sparse FITC

Entrenamiento: Datos de entrenamiento. $\mathcal{D} = (\mathbf{x}_{t_1}, y_{t_1}), \ldots, (\mathbf{x}_{t_N}, y_{t_n})$
Paso 1: Elegir el kernel e inicializar los hiperparámetros a cero.
Paso 2: Elección aleatoria de m *inducing points* $\mathbf{X_u}$
Paso 3: Método de gradiente que obtenga los valores óptimos de hiperparámetros y *inducing points* de tal forma que se minimice el $NLL(\mathcal{D}, \mathcal{H}, \mathbf{X_u})$
Test: Se precomputan las matrices que no incluyan los valores de test
$\mathbf{y} = [y_{t1}, y_{t2}, \ldots, y_{tN}]^T$
$K_{\mathbf{u},\mathbf{u}}$=$k(\mathbf{X_u}, \mathbf{X_u})$
$K_{\mathbf{y},\mathbf{u}} = k(\mathbf{x}, \mathbf{X_u})$
$\mathbf{\Sigma} = (K_{\mathbf{u},\mathbf{u}} + K_{\mathbf{u},\mathbf{y}}\mathbf{\Lambda}^{-1}K_{\mathbf{y},\mathbf{u}})^{-1}$
$\mathbf{\Lambda} = \text{diag}[K_{\mathbf{y},\mathbf{y}} - K_{\mathbf{y},\mathbf{u}}K_{\mathbf{u},\mathbf{u}}^{-1}K_{\mathbf{u},\mathbf{y}}]$
$\mathbf{\Lambda^{-1}}$
for $m = 0$ to L **do**
    Reajustar $\mathbf{x}_t^{(m)}$ como en 3.14
    $K_{*,\mathbf{f}} = k(\mathbf{x}_t^{(m)}, \mathbf{x}_{t_i}), i = 1, \ldots, N$
    $K_{*,*} = k(\mathbf{x}_t^{(m)}, \mathbf{x}_t^{(m)})$
    $\mu_t = K_{*,\mathbf{u}}\mathbf{\Sigma}K_{\mathbf{u},\mathbf{y}}\mathbf{\Lambda^{-1}}\mathbf{y}$
    $\sigma_t^2 = K_{*,*} - K_{*,\mathbf{u}}K_{\mathbf{u},\mathbf{u}}^{-1}K_{\mathbf{u},*} + K_{*,\mathbf{u}}\mathbf{\Sigma}K_{\mathbf{u},*}$
end for
Salida: Media y varianza estimada μ_t y σ_t^2

Capítulo 4

Resultados y Métricas

En este capítulo se resumen los resultados obtenidos en los algoritmos desarrollados. Para la verificación, se ha seleccionado un patrón estable de comparación, que permita evaluar cualitativamente y cuantitativamente cada uno de los métodos. Se exponen de forma gráfica y mediante métricas de error los tres métodos: GPR Básico, Múltiples Expertos y Sparse GPR. La estimación se ha realizado con distintos patrones futuros conocidos, los cuales serán comparados en las siguientes secciones.

- **Perfil de corriente y temperatura futura conocido**: Contiene los valores pasados y futuros de tensión, corriente y temperatura.
- **Perfil de corriente futura conocido pero no de temperatura**: Contiene tanto los valores pasados de tensión, corriente y temperatura como los valores futuros de corriente.
- **Perfil de temperatura futura conocido pero no de corriente**: Contiene tanto los valores pasados de tensión y temperatura como los valores futuros de temperatura

Para evaluar los resultados del sistema, resulta preciso hacer uso de métricas de error.

$$\mathrm{RMSE}(m) = \sqrt{\frac{1}{N_{test}} \sum_{1}^{N_{test}} (V_{t+m} - \widehat{V}_{t+m})^2} \tag{4.1}$$

$$\mathrm{MAE}(m) = \underset{i=1,\dots N_{test}}{max} = |V_{t+m} - \widehat{V}_{t+m}| \tag{4.2}$$

donde N_{test} se refiere a el número de muestras de test y $m = 1, \dots L$. Para evaluar el error nos interesa conocer la desviación de la forma de onda estimada frente a la real, tanto para días homogéneos cómo para cambios bruscos en la insolación. Además,

resulta necesario establecer métricas cercanas al problema planteado. Desde un punto de vista de implementación futura, el BMS podría centrar la predicción futura en la muestra de final de día. La pregunta que el operario, o el sistema automático, debe realizarse es qué tensión mínima va a ser capaz de suministrar el sistema en la hipótesis de que ya no se pueda absorber más energía del sol. Es por ello que su medida es de gran importancia.

$$\underset{FinalDia}{\text{RMSE}}(m) = \sqrt{\frac{1}{N_{dias}} \sum_{1}^{N_{test}} (\underset{FinalDia}{V_{t+m}} - \underset{FinalDia}{\widehat{V}_{t+m}})^2} \tag{4.3}$$

$$\underset{FinalDia}{\text{MAE}}(m) = \underset{i=1,...N_{dias}}{max} = |\underset{FinalDia}{V_{t+m}} - \underset{FinalDia}{\widehat{V}_{t+m}}| \tag{4.4}$$

Para poder medir el coste del algoritmo se ha incluido en las métricas de error el tiempo de cómputo de las fases de entrenamiento y de test. Aunque se ha de tener en cuenta que es una medida dependiente del ordenador con el que se ha realizado y sirve cómo método de comparación relativa entre la complejidad entre un modelo y otro.

Especificaciones GPR Básico

- Datos de entrenamiento: 720 horas equiespaciadas de 2008
- Mes de test: Mayo de 2009
- Kernel escogido: Racional Cuadrático (RQ)
- Memoria pasada: M=15
- Horizonte de predicción: L=48

Especificaciones Múltiples Expertos GPR

- Datos de entrenamiento: 720 horas equiespaciadas a cada experto GPR con su especialidad de día
- Mes de test: Mayo de 2009
- Kernel escogido: Racional Cuadrático (RQ)
- Memoria pasada: M=15
- Horizonte de predicción: L=48

Especificaciones Sparse GPR

- Datos de entrenamiento: 2008 completo y optimizado en la fase de entrenamiento
- Mes de test: Mayo de 2009
- Kernel escogido: Racional Cuadrático (RQ)
- Memoria pasada: M=15
- Horizonte de predicción: L=48

4.1. Perfil de corriente y temperatura futura conocido

En este sección se trata el caso de estimación de tensión futura con los perfiles de corriente y temperatura conocidos. Se busca estudiar el impacto que tiene la corriente y la temperatura en la predicción de tensión. Es esperable que, al alimentar el algoritmo con el perfil de corriente futuro, la estimación sea buena ya que en la corriente reside la información de carga de la batería (concepto de *Coulomb Counting* [24]).

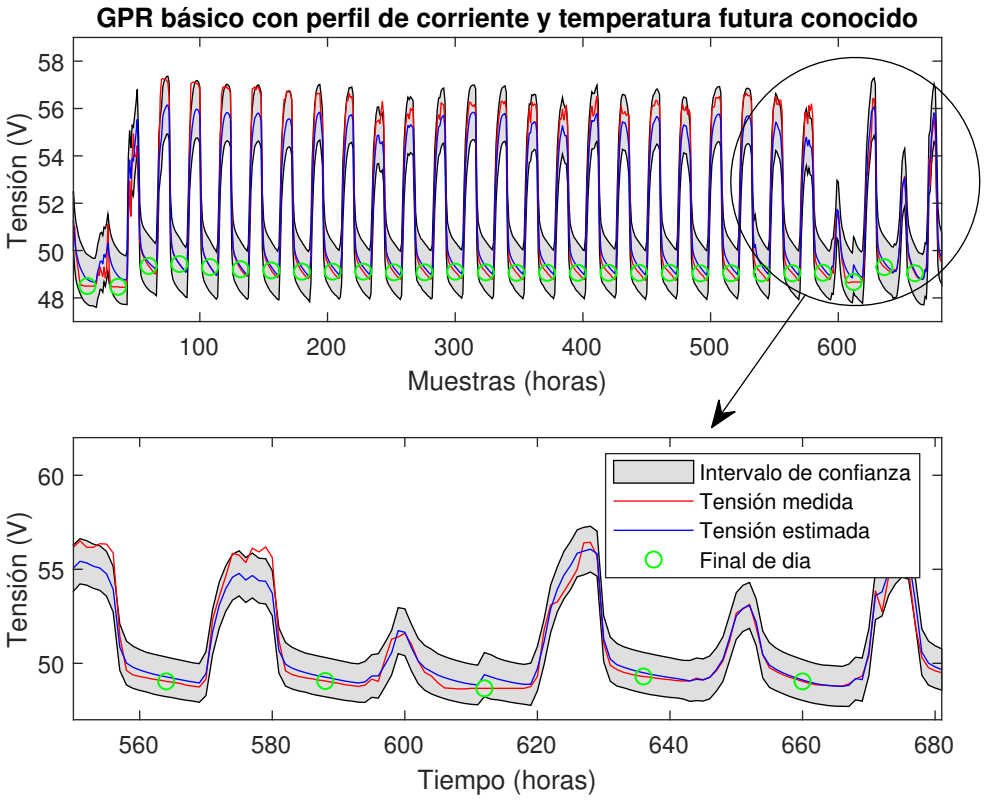

Figura 4.1: Formas de onda del GPR con corriente y temperatura futura conocidos

En la Figura 4.1 se puede ver cómo el modelo GPR ha sido capaz de reproducir la forma de onda. Este modelo ha sido capaz de seguir las caídas de tensión producidos en los días de insolaciones iniciales y finales de forma certera.

Por otro lado, las métricas de error de la Figura 4.2 muestran cómo existe un patrón periódico de error localizado en lo alto de la forma de onda. Este fenómeno es producido ya que es el momento de máxima incertidumbre del algoritmo (máximo intervalo de confianza) y el dato es menos fiable.

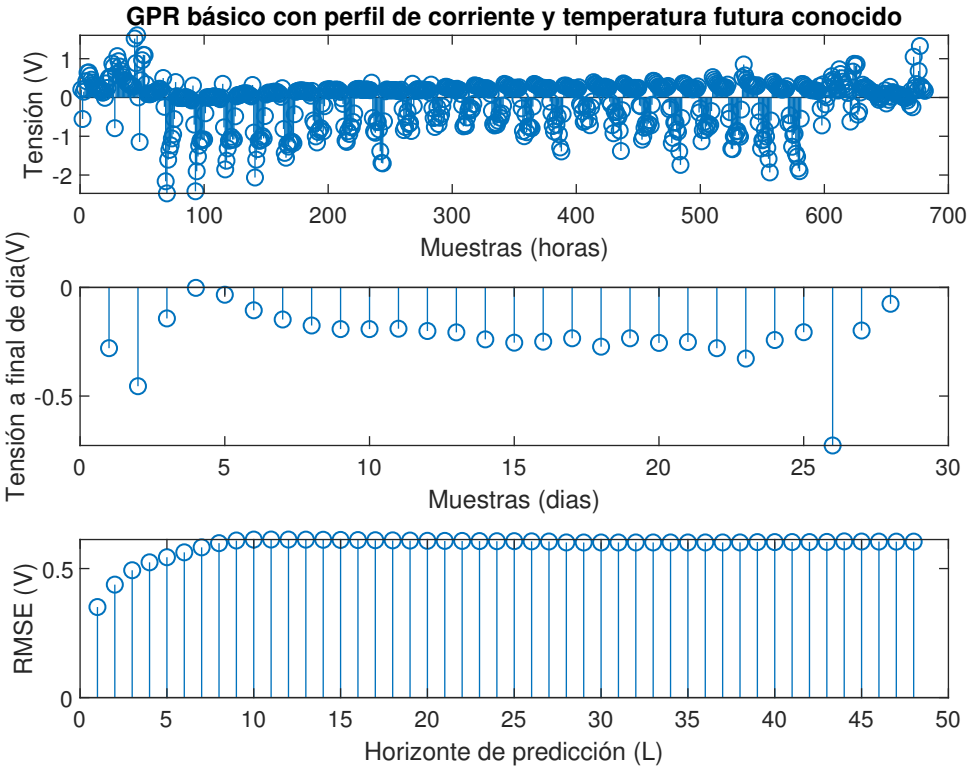

Figura 4.2: Métricas de error del GPR con corriente y temperatura futura conocidos

En el caso de los Múltiples Expertos, la evolución de tensión en la Figura 4.3 también es buena, con defectos y virtudes similares al GPR Básico. Se sigue apreciando cómo el modelo se ha ajustado a la onda real,

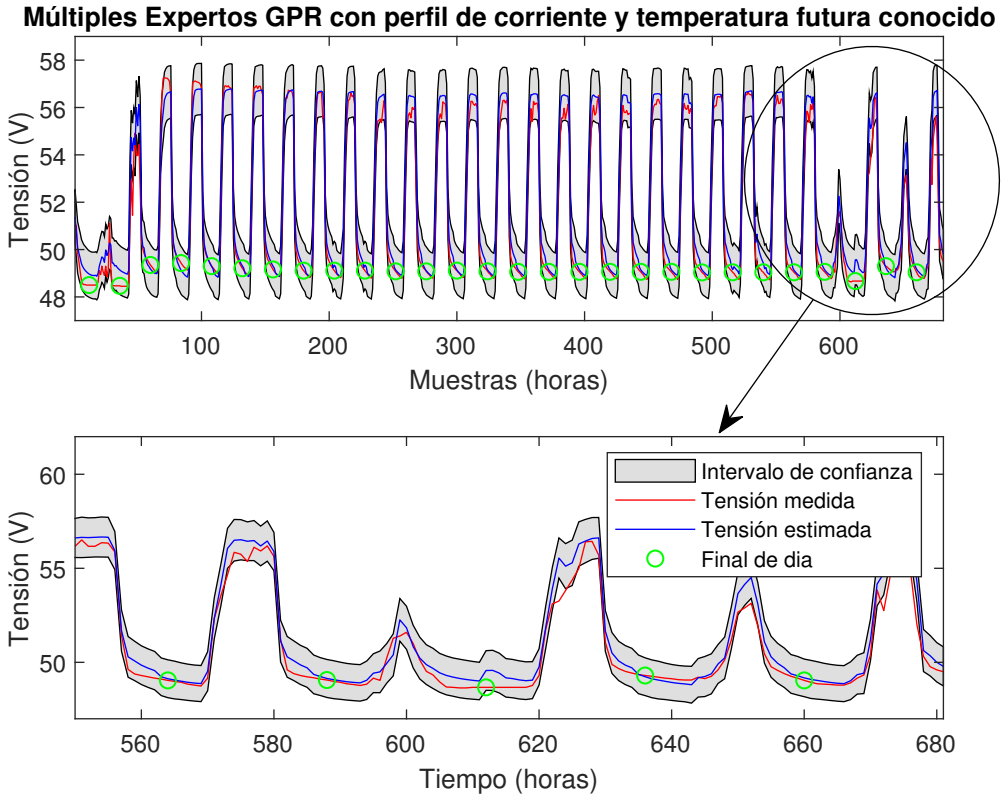

Figura 4.3: Formas de onda del ME con corriente y temperatura futura conocidos

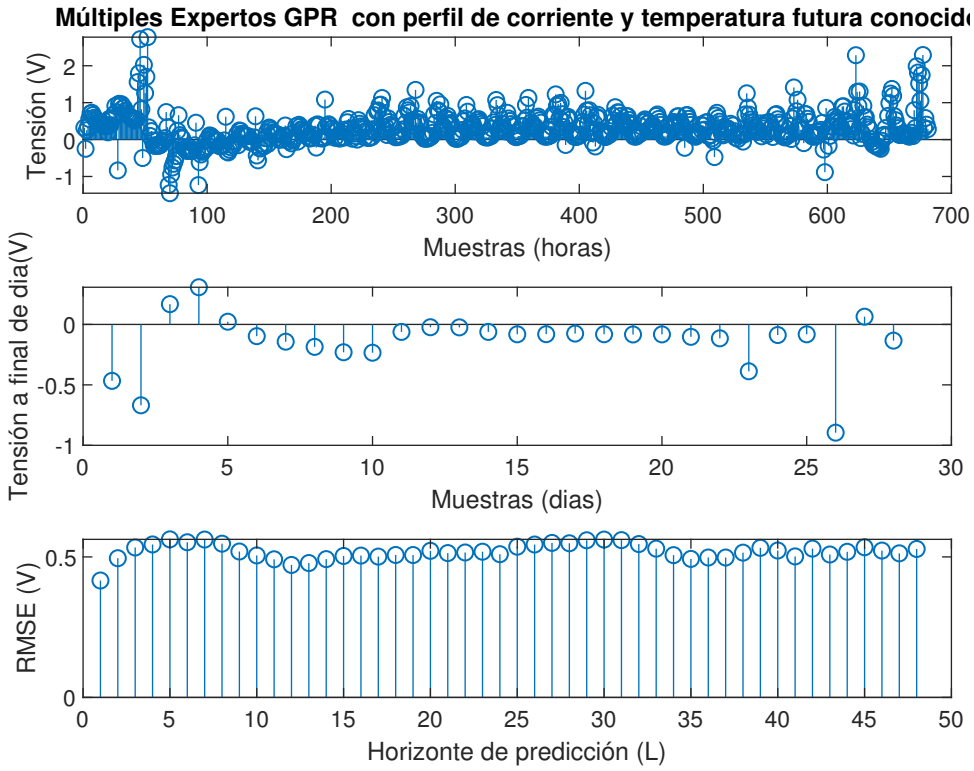

Figura 4.4: Métricas de error del ME GPR con corriente y temperatura futura conocidos

Los problemas de errores periódicos vistos en el caso del GPR Básico se han atenuado tal y como se puede ver en la Figura 4.4. Sin embargo, siguen siendo apreciables y se tendrá que evaluar su importancia en el sistema.

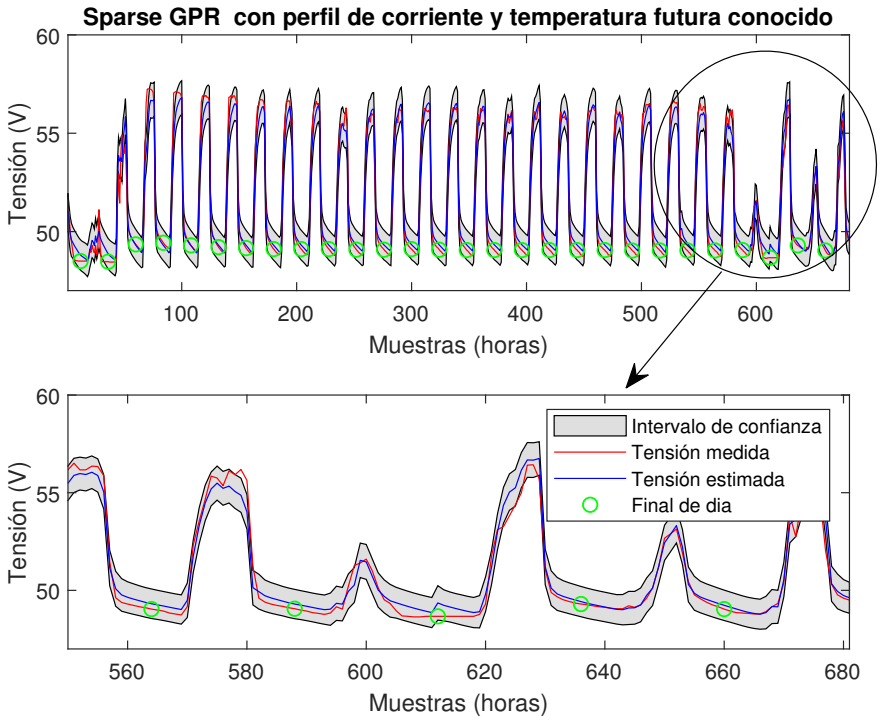

Figura 4.5: Formas de onda del Sparse con corriente y temperatura futura conocidos

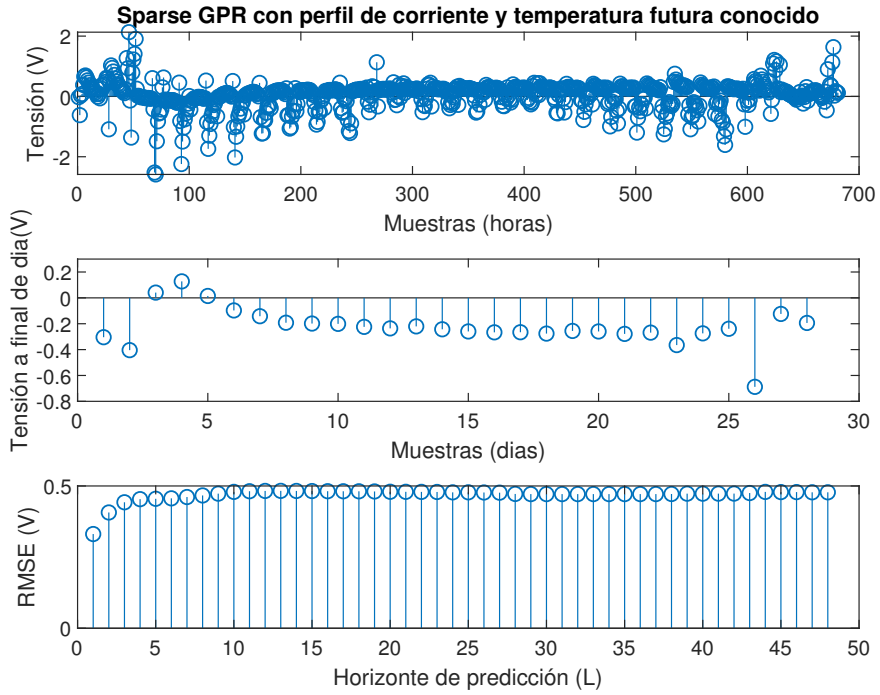

Figura 4.6: Métricas de error del Sparse con corriente y temperatura futura conocidos

En el caso de el Sparse GPR, se puede ver en la Figura 4.5 como la onda de tensión estimada sigue, de forma precisa, la forma de onda original.

Los errores obtenidos en la Figura 4.6 muestra como este método también consigue atenuar los errores observados en el GPR Básico obteniéndose como máximo picos de 2V de error en situaciones de insolación.

	GPR Básico	Múltiples Expertos GPR	Sparse GPR
RMSE (V)	0.5898	0.5193	0.4695
MAE (V)	1.6033	0.2275	2.1305
$\text{RMSE}_{FinalDia}$ (V)	0.2799	0.1803	0.2379
$\text{MAE}_{FinalDia}$ (V)	0.0020	0.3075	0.1275
Duración Entrenamiento (s)	14	62	24
Duración Test (s)	55	446	46

Tabla 4.1: Métricas de error con perfil de corriente y temperatura conocido

Como se puede ver en las figuras, los tres métodos han sido capaces de seguir de forma precisa, en mayor o menor medida, de seguir el patrón tensión en un mes con sucesos heterogéneos como insolaciones. Los máximos puntos de error se encuentran a en el pico de onda ya que es en ese momento cuando la incertidumbre es máxima. La Tabla de métricas de error 4.1 muestra como el GPR Básico ha sido el que mayor error ha cometido en el conjunto de onda. Sin embargo, ha alcanzado gran capacidad para no producir picos de error al final del día y es el que menor coste de entrenamiento ha requerido. Por otro lado, los Múltiples Expertos logran su función de reducir el error del GPR básico pero a costa de un incremento elevado tanto en el entrenamiento como en el test. El Sparse GPR ha sido el método que mejores métricas ha obtenido con un tiempo de test similar al básico y de entrenamiento algo superior.

4.2. Perfil de corriente futura conocido pero no de temperatura

En este sección se trata el caso en el que se cuenta con tanto los valores pasados de tensión, corriente y temperatura como los valores futuros de corriente. Se busca estudiar, con respecto a la sección anterior, el impacto que tiene la corriente y la temperatura en la predicción de tensión.

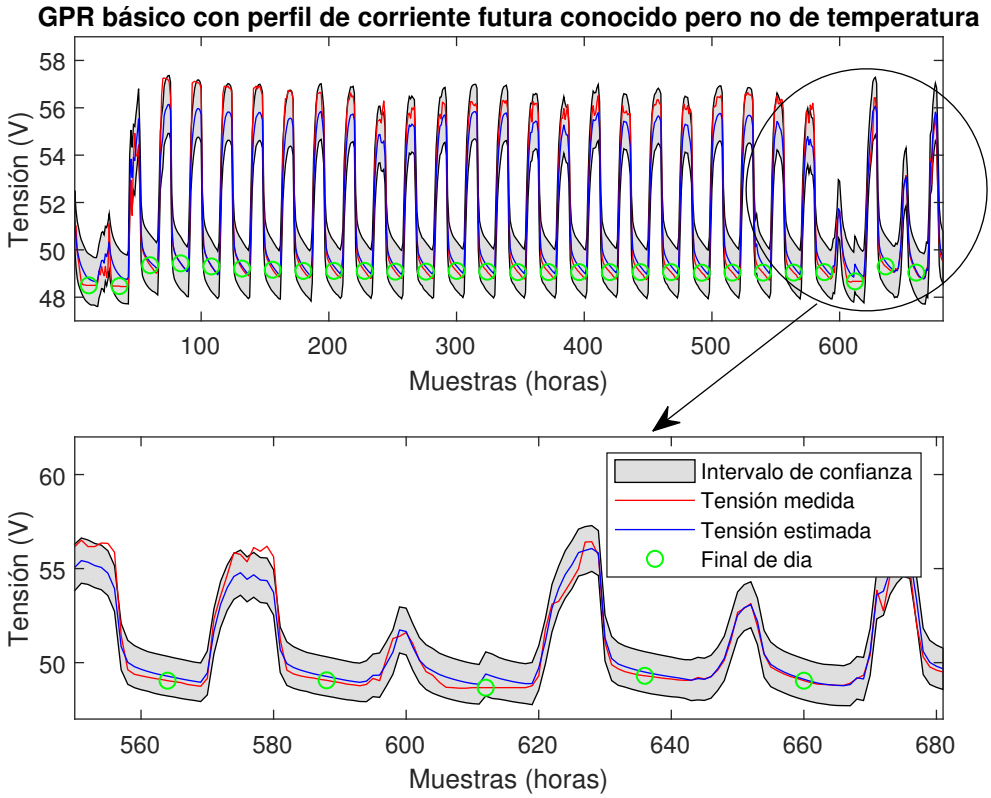

Figura 4.7: Formas de onda del GPR con corriente futura conocida

Los resultados apreciables resultan ser similares a los obtenidos en la sección previa, tanto en la forma de onda como en las métricas representadas.

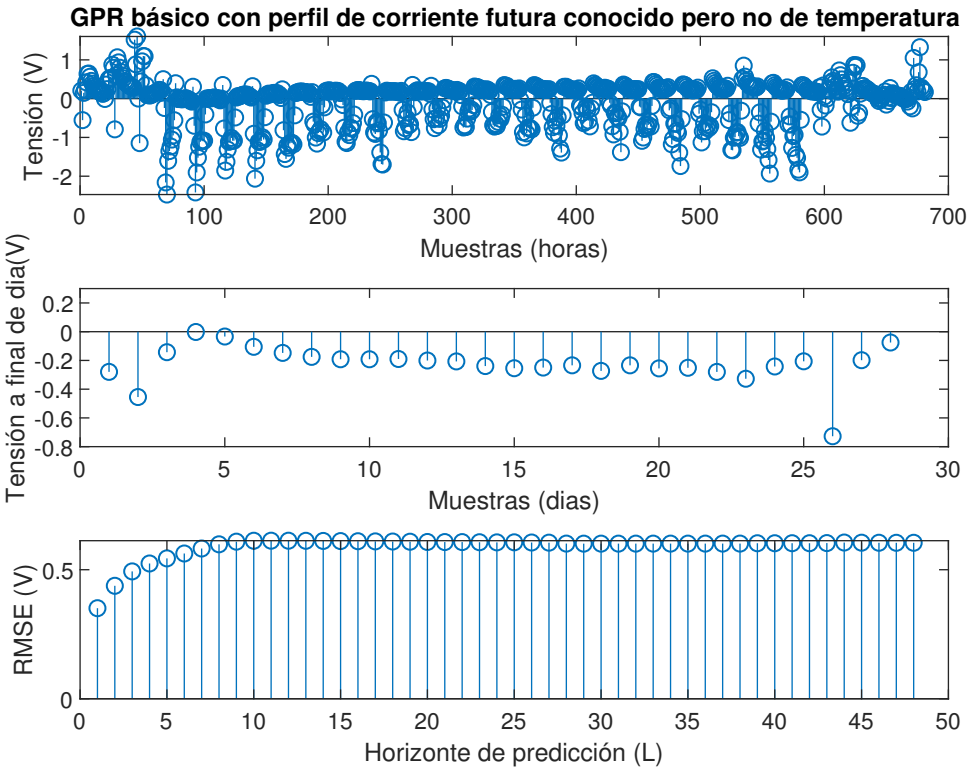

Figura 4.8: Métricas de error del GPR con corriente futura conocida

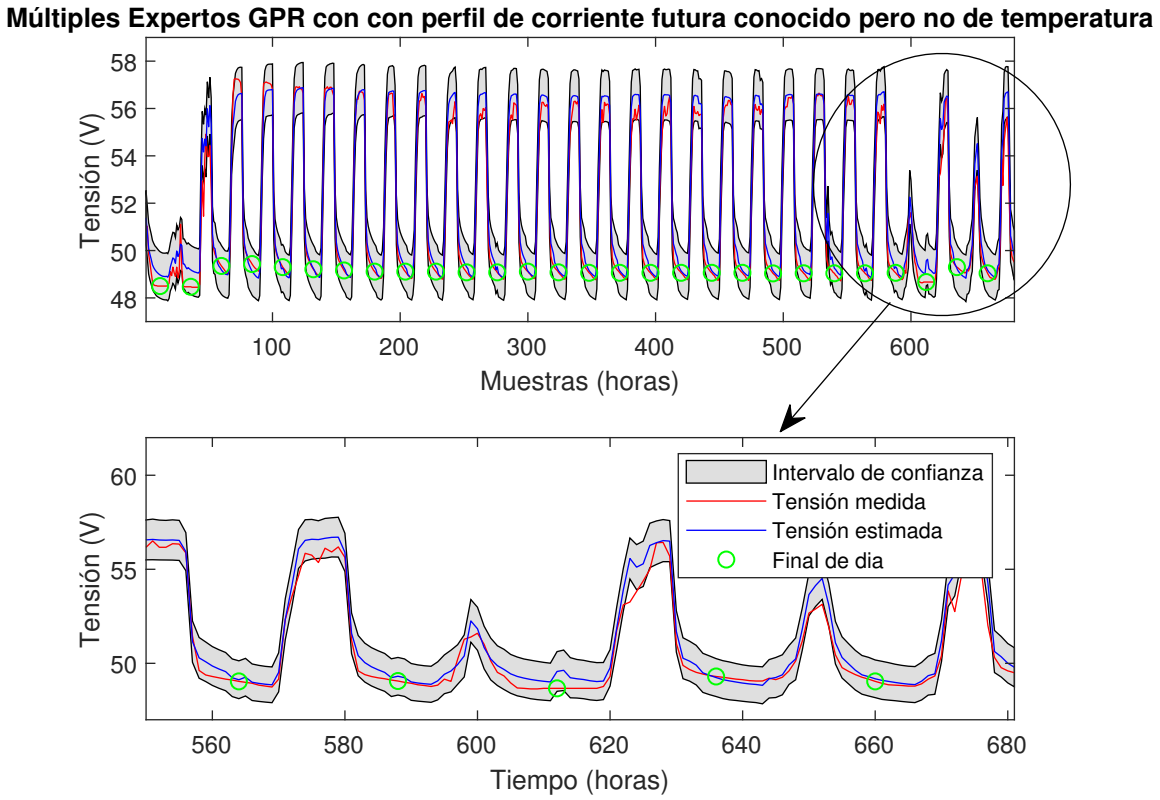

Figura 4.9: Formas de onda del ME GPR con corriente futura conocida

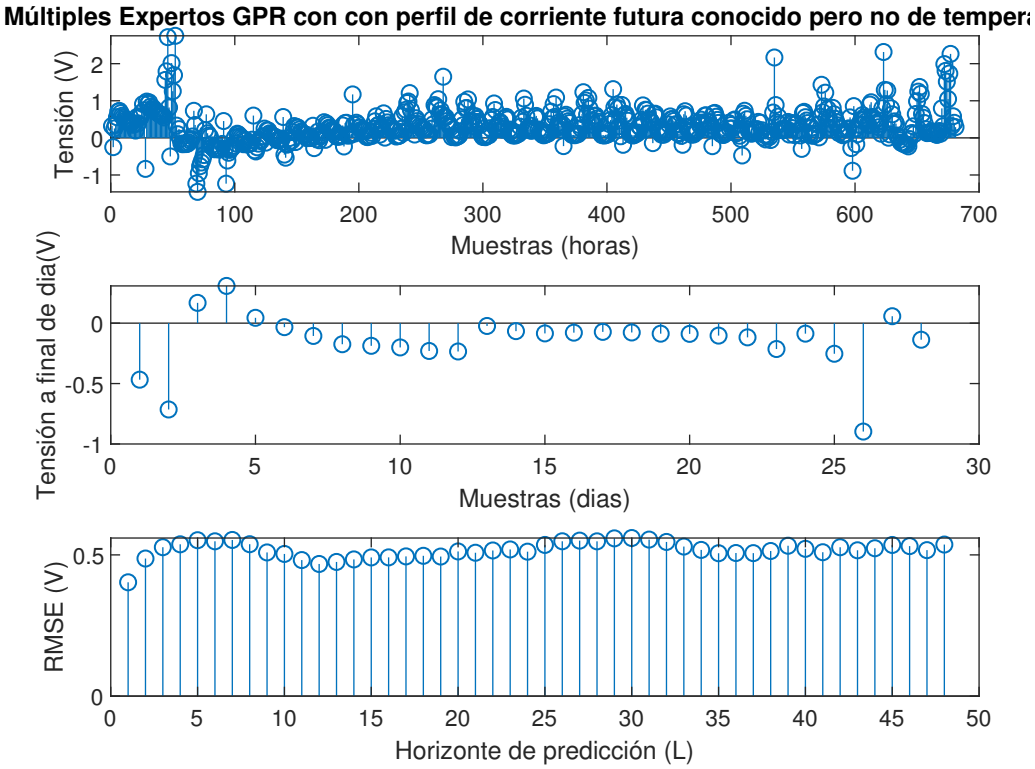

Figura 4.10: Métricas de error del ME GPR con corriente futura conocida

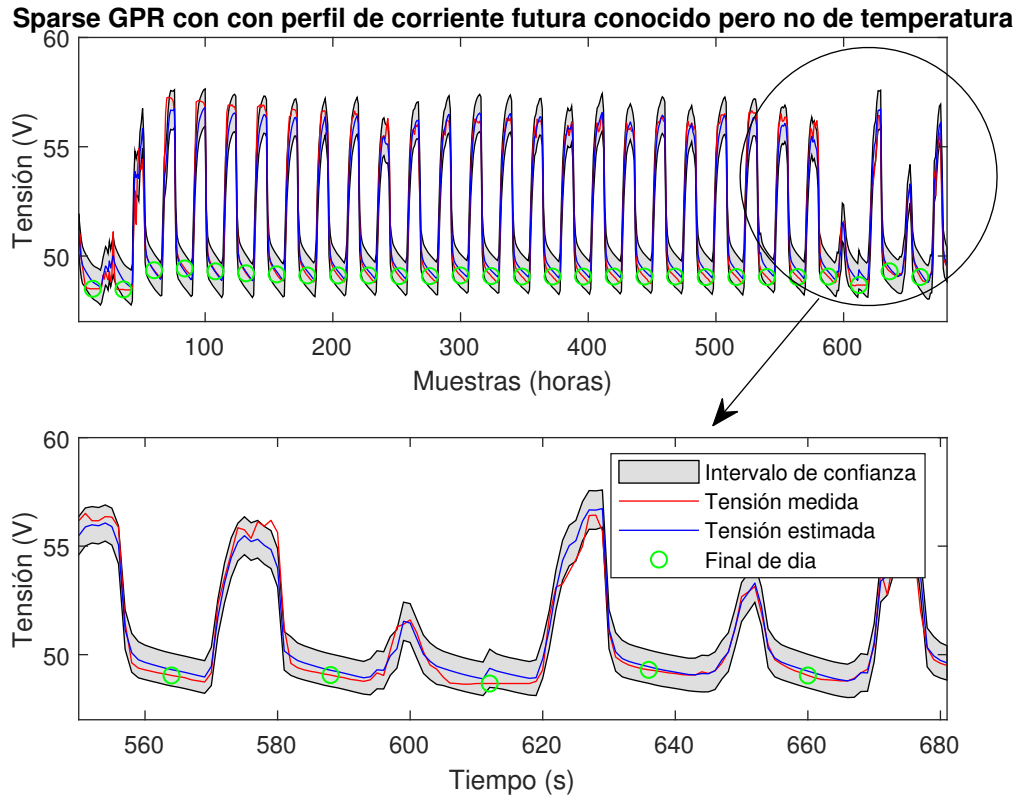

Figura 4.11: Formas de onda del Sparse GPR con corriente futura conocida

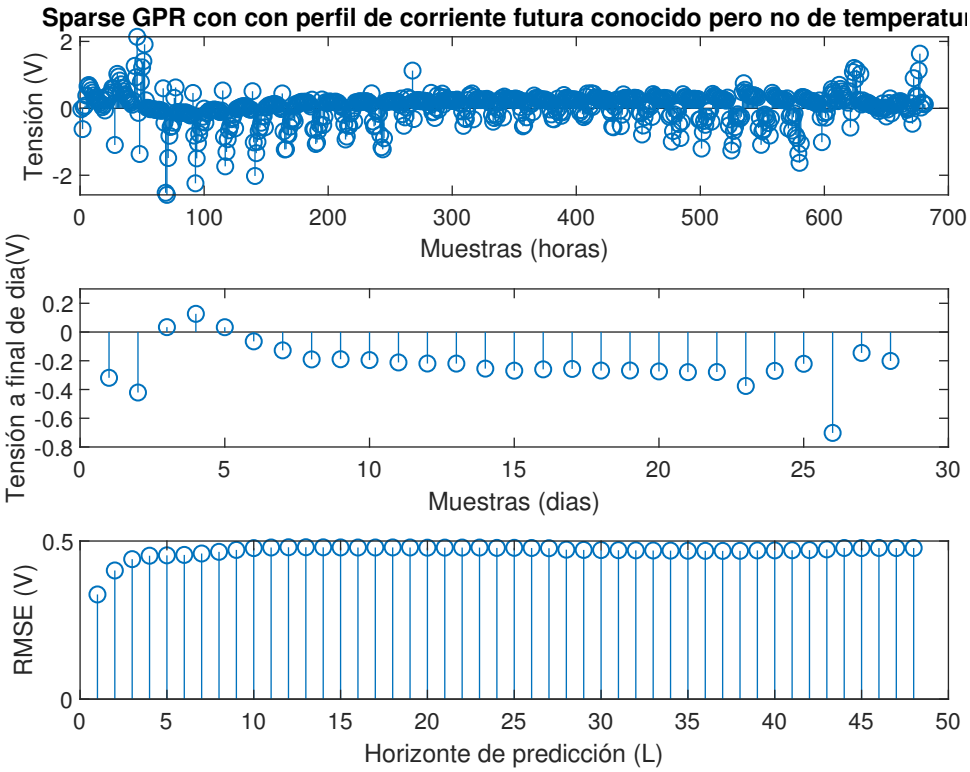

Figura 4.12: Métricas de error del Sparse GPR con corriente futura conocida

	GPR Básico	Múltiples Expertos GPR	Sparse GPR
RMSE (V)	0.5898	0.5173	0.4685
MAE (V)	1.6033	2.7508	2.1378
RMSE $_{FinalDia}$ (V)	0.2799	0.1898	0.2384
MAE $_{FinalDia}$ (V)	0.0020	0.3081	0.1266
Duración Entrenamiento (s)	14	60	24
Duración Test (s)	55	416	47

Tabla 4.2: Métricas de error con perfil de corriente conocido

Como se puede ver tanto en las Figuras previas como en la Tabla de métricas de error 4.2,las evoluciones de las forma de onda, se han mantenido prácticamente invariantes con respecto a la sección previa donde se introducía la temperatura. Es por ello, que es posible extraer la conclusión de que la temperatura no aporta un peso

significativo en la estimación. Las causas de ello podrían estar relacionadas con la alta inercia térmica del pack de baterías, o con la independencia del comportamiento de carga con la temperatura. Este resultado facilita que, en la práctica, puedan llevarse a cabo este tipo de algoritmos mediante solamente dos variables, ya que el BMS puede focalizarse en un perfil de corriente hipotético sin dependencia térmica.

4.3. Perfil de temperatura futura conocido

En este sección se aborda el caso en el que se prescinde de la corriente en la predicción y se hace uso de, exclusivamente, la tensión y temperatura pasadas y el perfil de temperatura futuro.

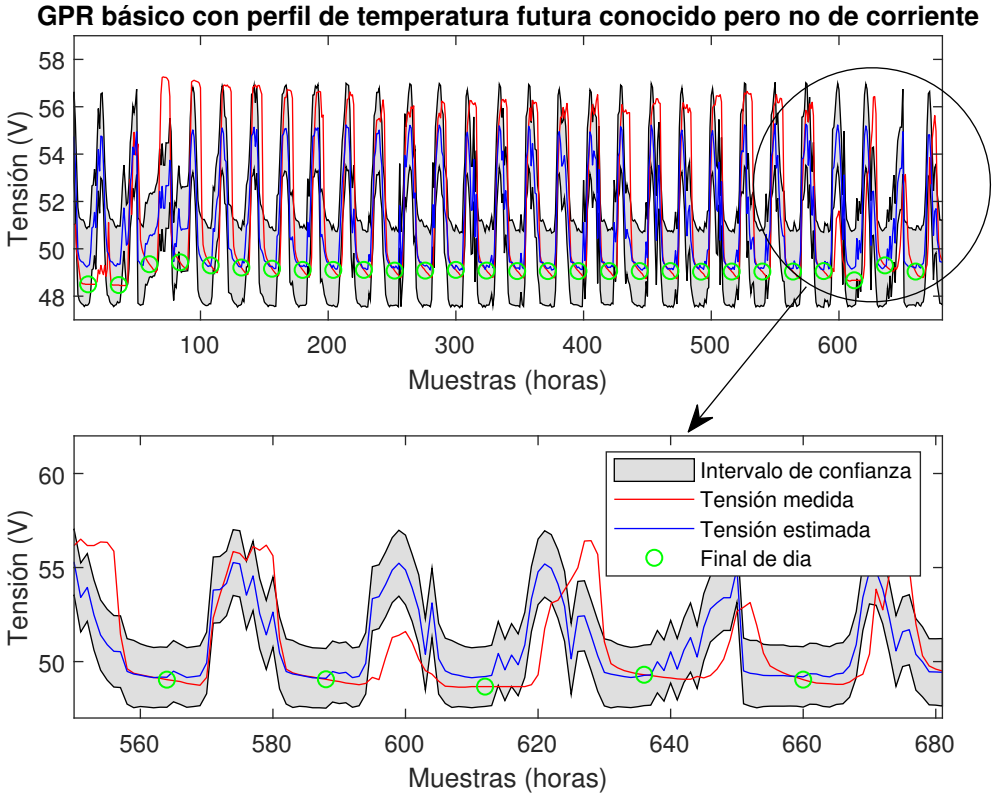

Figura 4.13: Formas de onda del Sparse GPR con temperatura futura conocida

Como se puede apreciar en la Figura 4.13, pese a no contar con el perfil de corriente, el modelo ha sido capaz de reproducir un patrón de onda similar al obtenido en las baterías. Sin embargo, como en la temperatura no reside la información de la carga, no es capaz de reproducir correctamente el modelo.

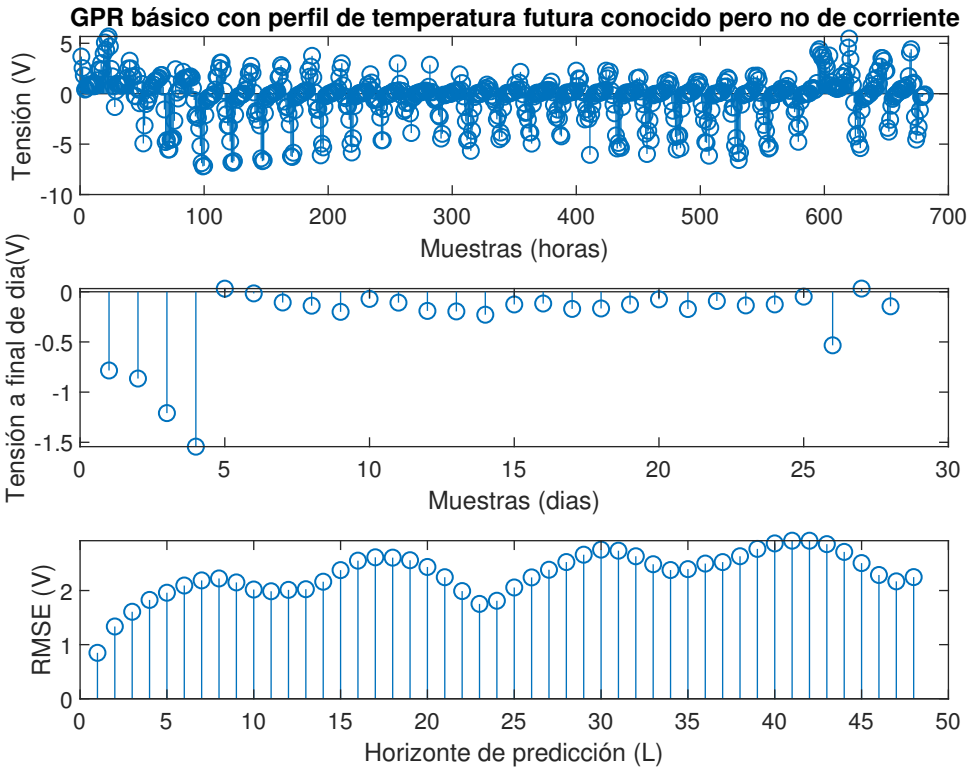

Figura 4.14: Métricas de error del GPR con temperatura futura conocida

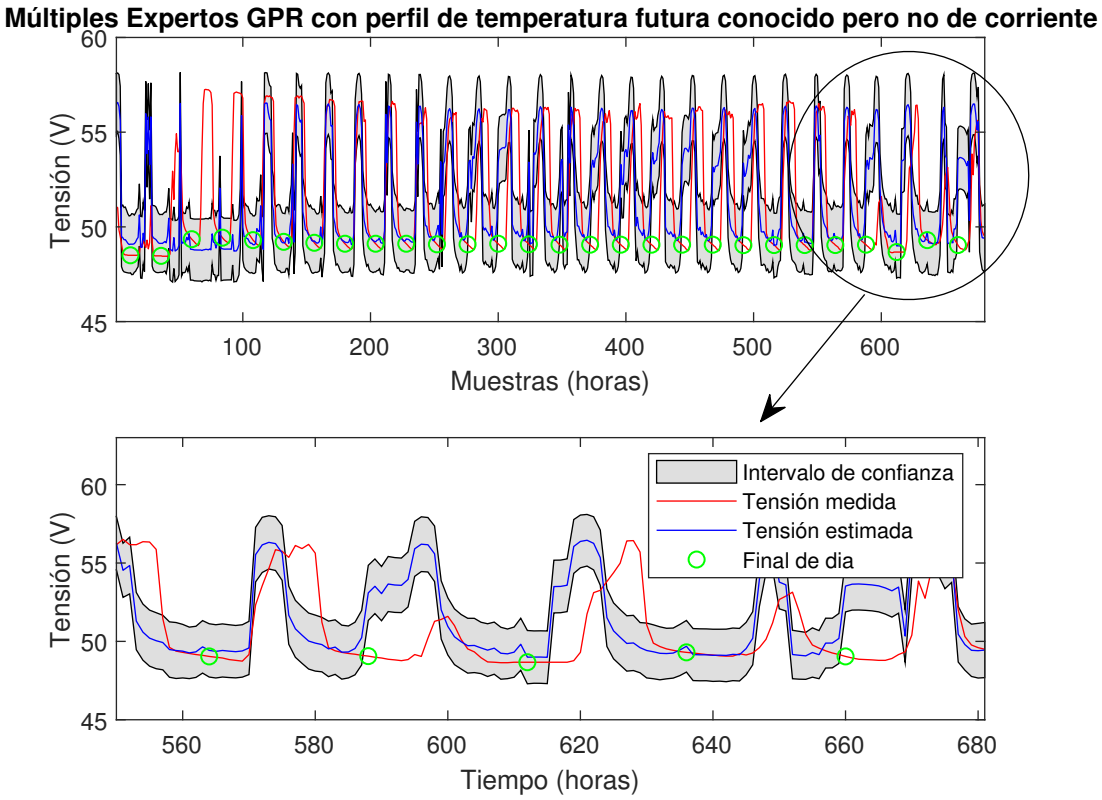

Figura 4.15: Formas de onda del ME GPR con temperatura futura conocida

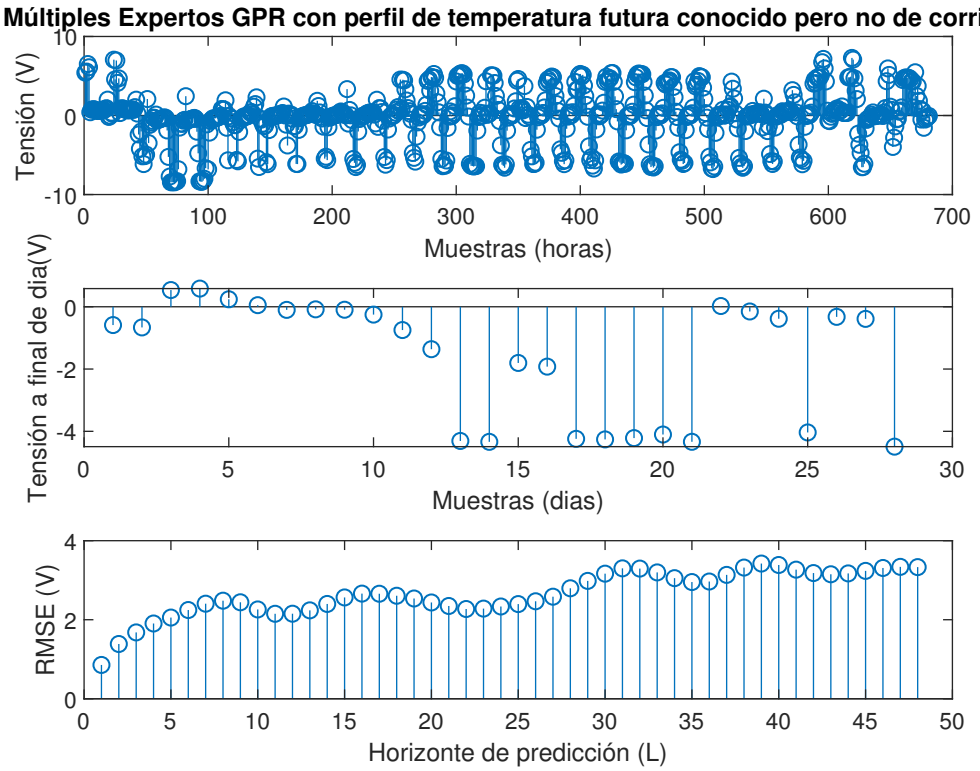

Figura 4.16: Métricas de error del ME GPR con temperatura futura conocida

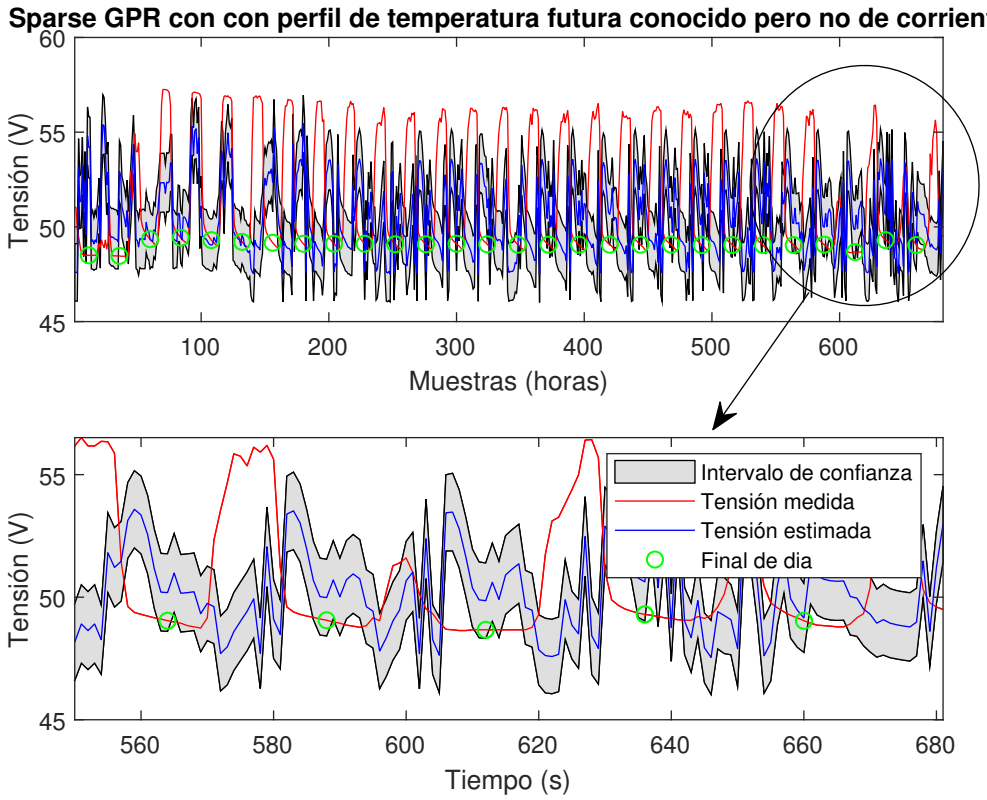

Figura 4.17: Formas de onda del Sparse GPR con temperatura futura conocida

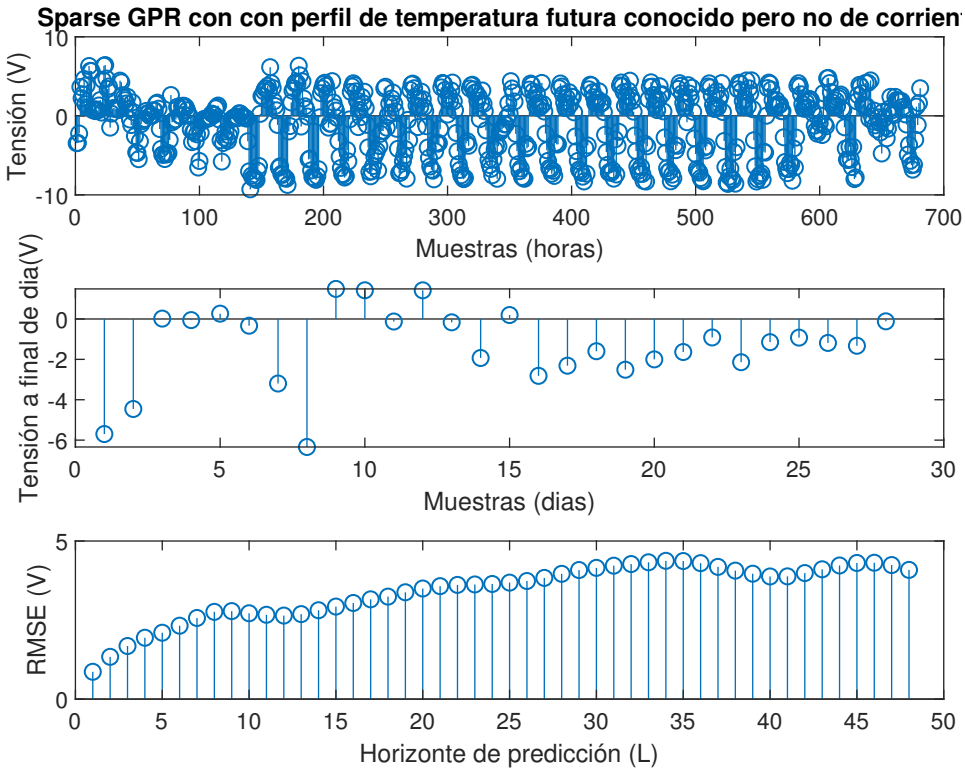

Figura 4.18: Métricas de error del Sparse GPR con temperatura futura conocida

Los Múltiples Expertos 4.15 siguen el mismo patrón que el GPR Básico reproduciendo una forma de onda escasamente correlada con la real. Por otro lado, el resultado resultante del modelo de Sparse GPR no ha sido capaz ni de encontrar el patrón periódico.

	GPR Básico	Múltiples Expertos GPR	Sparse GPR
RMSE (V)	2.3036	2.6616	3.4191
MAE (V)	5.6754	7.2880	6.4276
$\text{RMSE}_{FinalDia}$ (V)	0.2771	1.7369	2.3347
$\text{MAE}_{FinalDia}$ (V)	0.0316	0.5838	6.3401
Duración Entrenamiento (s)	14	58	20
Duración Test (s)	52	391	37

Tabla 4.3: Métricas de error con perfil de temperatura conocido

Como se puede ver tanto en las figuras como en la tabla de errores 4.3 prescindir de la corriente en la estimación hace que no sea posible estimar de forma adecuada la tensión. Sin embargo, frente a los casos previos, el GPR Básico resulta ser capaz de formar una onda similar que consigue un error reducido al final de día.

Capítulo 5

Conclusiones

Se ha abordado el problema de estimación de tensión futura en baterías a través del uso de los Procesos Gaussianos como herramienta de regresión. Para ello se ha partido de un nuevo algoritmo: R-MVSP, que hasta ahora solamente se había ensayado en bases de datos pequeñas y artificiales.

Con el fin de entender qué límites, ventajas y optimizaciones pueden aplicar este algoritmo a la estimación de tensión futura, se han realizado tres implementaciones diferentes de GPR: GPR básico, Múltiples Expertos GPR y Sparse GPR basado en FITC.

Los algoritmos GPR suponen una mejora respecto a otras técnicas, por presentar menores problemas de overfitting y disponer de una parámetro de calidad de la estimación (intervalo de confianza).

Se han desarrollado los algoritmos en Matlab. Para su evaluación se ha utilizado una base de datos realista y de gran tamaño recogida en los repetidores de la red de comunicaciones de la Confederación Hidrográfica de Ebro. El objetivo ha sido determinar si este tipo de algoritmos pueden proporcionar estimaciones fiables a futuro para evitar fallos críticos en sistemas de comunicaciones no conectados a la red eléctrica.

Los resultados muestran como para la estimación futura de tensión es imprescindible el uso del perfil de corriente y que la influencia de la temperatura en la estimación no es representativa. Por otro lado se demuestra que es posible implementar, como continuación de este trabajo, un sistema de gestión del riesgo de las instalaciones basado en las técnicas GPR.

Si bien la carga computacional es relativamente elevada, durante el trabajo

se ha desarrollado una optimización de las librerías iniciales permitiendo así una implementación eficiente del método. Así mismo La utilización de las variantes Sparse y el uso de kernels de estructura simple permiten optimizar el tiempo de ejecución, extendiendo la base de datos de entrenamiento y aumentando así las casuísticas que el modelo es capaz de abordar.

Se está trabajando actualmente en un artículo (JCR-Q1) sobre los resultados obtenidos en este trabajo.

Lineas Futuras

La línea futura más inmediata es la del uso de esta misma base de datos y algoritmos en la estimación del SoH Y del SoC de la misma forma que se ha hecho en otros artículos con métodos similares.

Implementación de los algoritmos en Phyton para cómputo en la nube ya que este tipo de instalaciones de comunicaciones tiene acceso a internet. Otra posibilidad sería la de llevar estos algoritmos a un *Single Board Computer* (SBC) y procesar los datos de forma local.

Utilización de herramientas de kernel automáticas que se ajustan a tu base de datos. Se está desarrollando actualmente un modelo de generación automática de funciones de covarianza que permite modelar de forma óptima tu base de datos [18].

Extender el número de variables que se inyectan al GPR Sparse para poder ver el envejecimiento de las baterías mediante reentrenamientos periódicos. La base de datos crece conforme se tomen nuevas medidas en los sensores.

Realizar una clasificación de tipos de días usando como parámetros de entrada ,del algoritmo K-*means* u otro, valores relacionados con la estimación futura. En definitiva, cambiar la perspectiva a la hora de etiquetar.

Capítulo 6

Bibliografía

[1] A. E. Mejdoubi, A. Oukaour, H. Chaoui, H. Gualous, J. Sabor, and Y. Slamani. State-of-charge and state-of-health lithium-ion batteries' diagnosis according to surface temperature variation. *IEEE Transactions on Industrial Electronics*, 2016.

[2] A. E. Mejdoubi, A. Oukaour, H. Chaoui, J. H. Gualous, Sabor, and Y. Slamani. State-of-charge and state-of-health lithium-ion batteries' diagnosis according to surface temperature variation. *IEEE Industrial Informatics*, 2013.

[3] H. T. Lin, T. J. Liang, and S. M. Chen. Estimation of battery state of health using probabilistic neural network. *IEEE Transactions on Industrial Electronics*, 2015.

[4] Jianan Hu, Juejun Hu, Haibin Lin, X.P.Li, C. L. Jiang, X. H. Qiu, and Weishan Li. State-of-charge estimation for battery management system using optimized support vector machine for regression. *Journal of Power Sources*, 2014.

[5] Datong Liu, Jingyue Pang, Jianbao Zhou, Yu Peng, and Michael Pecht. Prognostics for state of health estimation of lithium-ion batteries based on combination gaussian process functional regression. *Journal of Microelectronics Reliability*, 2013.

[6] Fran Li and Jiuping Xu. A new prognostics method for state of health estimation of lithium-ion batteries based on a mixture of gaussian process models and particle filter. *Journal of Microelectronics Reliability*, 2015.

[7] Gozde O. Sahinoglu, Milutin Pajovic, Zafer Sahinoglu, Yebin Wang, Philip V. Orlik, and Toshihiro Wada. Battery state of charge estimation based on regular/recurrent gaussian process regression. *IEEE Transactions on Industrial Electronics*, 2017.

[8] Gozde Ozcan, Milutin Pajovic, Zafer Sashinoglu, Yebin Wang, Philip V. Orlik, and Toshihiro Wada. Online battery state-of-charge estimation based on sparse gaussian process regression. *IEEE Power and Energy Society General Meeting*, 2016.

[9] Chao Hu, Gaurav Jain, Craig Schmidt, Carrie Strief, and Melani Sullivan. Online estimation of lithium-ion battery capacity using sparse bayesian learning. *Journal of Power Sources*, 2015.

[10] Milutin Pajovic, Zafer Sahinoglu, Yebin Wang, Philip V. Orlik, and Toshihiro Wada. Online data-driven battery voltage prediction. *IEEE Industrial Informatics*, 2017.

[11] Decha Moungsri, Tomoki Koriyama, and Takao Kobayashi. GPR-based thai speech synthesis using multi-level duration prediction. *Speech Communication*, 2018.

[12] Som toolbox. `http://www.cis.hut.fi/somtoolbox/`.

[13] Kolmogorov A. N. Sur l'interpolation et extrapolation des suites stationnaires. *IEEE Transactions on quantum neuroscience electronics*, 1939.

[14] N. Wiener. *Extrapolation, Interpolation, and Smoothing of Stationary Time Series.* MIT Press, 1949.

[15] C. E. Rasmussen and Christopher K. I. Williams. The intrinsic random funcions and their applications. *Advances in Applied Probability*, 1973.

[16] C. E. Rasmussen and Christopher K. I. Williams. Gaussian processes for regression. *MIT Press*, 1996.

[17] C. E. Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning.* Cambridge, 2006.

[18] David Kristjanson Duvenaud. *Automatic Model Construction with Gaussian Processes.* Phd thesis, University of Cambridge, 2014.

[19] Teofilo F.Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science*, 1985.

[20] Joaquín Quiñonero-Candela and C. E. Rasmussen. A unifying view of sparse approximate gaussian process regression. *Machine Learning Research*, 2005.

[21] Matthias Bauer, Mark van der Wilk, and C. E. Rasmussen. Understanding probabilistic sparse gaussian process approximations. *30th Conference on Neural Information Processing Systems*, 2016.

[22] André-Louis Cholesky. Sur la résolution numérique des systèmes d'équations linéaires. *Bulletin de la Sabix*, 1910.

[23] Iván Sanz, Carlos Bernal, Antonio Bono, Milutin Pajovic, and Gabriel Martínez. Gaussian process regression applied to VRLA battery voltage prediction in photovoltaic off-grid systems. *Jornada de Jóvenes Investigadores del I3A*, 2018.

[24] Kong Soon Ng, Chin-Sien Mooa, Yi-Ping Chen, and Yao-Ching Hsieh. Enhanced coulomb counting method for estimating state-of-charge and state-of-health of lithium-ion batteries. *Journal of Applied Energy*, 2009.

[25] Gozde Ozcan, Milutin Pajovic, Zafer Sahinoglu, Yebin Wang, Philip V. Orlik, and Toshihiro Wada. Extremum seeking-based parameter identification for state-of-power prediction of lithium-ion batteries. *IEEE Renewable Energy Research and Applications*, 2016.

[26] Ramírez, K. Pérez, and S. Z. Self conscious robots in induction heating home appliances. *IEEE transactions on anthropomorphic robots*, 2018.

[27] Antonio Bono Nuez. *Redes Neuronales Artificiales para Evaluar la Calidad de Vida*. Tesis doctoral, Universidad de Zaragoza, 2015.

[28] Per Christian Hansen. Detection of near-singularity in cholesky and ldl^t factorizations. *Journal of Computational and Applied Mathematics*, 1987.

Lista de Figuras

Lista de Tablas