Short-term Prediction and Filtering of Solar Power Using State-Space Gaussian Processes

Sean Nassimiha*
Department of Computer Science
University College London
sean.nassimiha.18@ucl.ac.uk

Peter Dudfield
Open Climate Fix
peter@openclimatefix.org

Jack Kelly
Open Climate Fix
jack@openclimatefix.org

Marc Peter Deisenroth
UCL Centre for Artificial Intelligence
University College London
m.deisenroth@ucl.ac.uk

So Takao
UCL Centre for Artificial Intelligence
University College London
so.takao@ucl.ac.uk

Abstract

Short-term forecasting of solar photovoltaic energy (PV) production is important for powerplant management. Ideally these forecasts are equipped with error bars, so that downstream decisions can account for uncertainty. To produce predictions with error bars in this setting, we consider Gaussian processes (GPs) for modelling and predicting solar photovoltaic energy production in the UK. A standard application of GP regression on the PV timeseries data is infeasible due to the large data size and non-Gaussianity of PV readings. However, this is made possible by leveraging recent advances in scalable GP inference, in particular, by using the state-space form of GPs, combined with modern variational inference techniques. The resulting model is not only scalable to large datasets but can also handle continuous data streams via Kalman filtering.

1 Introduction

On Tuesday, 19 July 2022, Britain paid the highest electricity price ever recorded, at a staggering £9724 per megawatt hour. This (quite horrible) achievement was due to a deep imbalance in the supply and demand of the national power grid, which was under severe stress caused by the hottest UK day on record, and by the recent surge in natural gas prices [1]. It is apparent that our continual reliance on natural gas had enabled this perfect storm; it can be argued that our energy system may have been more resilient had there been a larger proportion of renewables in the mix.

However, managing supply and demand through renewables is not easy. Solar power, among most other renewables, is intermittent, making it necessary to back it up with spinning reserves to provide energy when the sun is not shining. These spinning reserves are, unfortunately, expensive and emit large amounts of CO2 [12]. Thus, by increasing the accuracy of forecasts for renewable electricity supply, one can hope to reduce the amount of spinning reserves and in turn, decrease emissions and costs associated with the use of clean energy sources.

*Also with Balyasny Asset Management

Tackling Climate Change with Machine Learning: workshop at NeurIPS 2022.

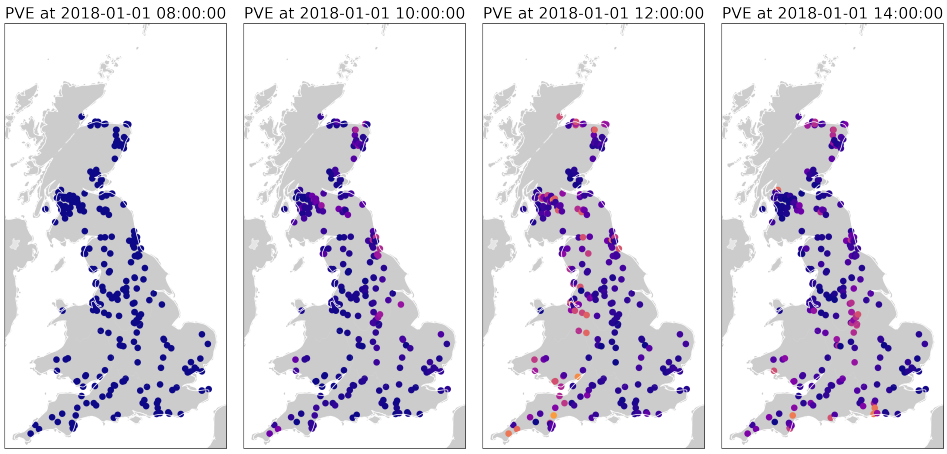


Figure 1: PV output at stations distributed across the UK at different times.

Naturally, there has been a surge of recent works applying machine learning to the task of photovoltaic energy (PV) nowcasting. For example in [18], a comparative study of several deep learning models for the task of one-minute ahead PV value prediction is conducted. Similarly, in [10], a physics-constrained LSTM architecture is considered for one-hour ahead PV prediction (see also [3, 15] for comprehensive reviews on the application of ML models for PV nowcasting). While these deep learning models outperform simple baselines, such as persistence, they do not quantify uncertainty, which we believe is necessary for high-stakes applications such as this.

In this paper, we propose the use of Gaussian processes (GPs) for PV nowcasting as alternative to deep-learning based approaches. These are nonparametric Bayesian regression method that can produce predictions equipped with uncertainty estimates [17], which could be useful in our setting.

2 Data

The dataset that we use in this work was provided to us by Open Climate Fix [2], consisting of solar PV energy (PVE) readings at 5 minute intervals for 1311 PV stations scattered across Great Britain, between January 2018 and November 2021 (see Figure 1).

We preprocess the data by first clipping the PV values between 0 W and 5×10^7 W to remove outliers. This was then re-scaled by dividing each observation by the total capacity of the corresponding PV station, resulting in readings between $[0, 1]$. Data from stations that showed non-zero production overnight or contained missing observations were removed. Finally, the time series were sliced to only include data between 8:00 and 16:00. The resulting timeseries $(y_i)_{i=1}^N$ at a single station is displayed in Figure 2, which shows strong annual and daily seasonality.

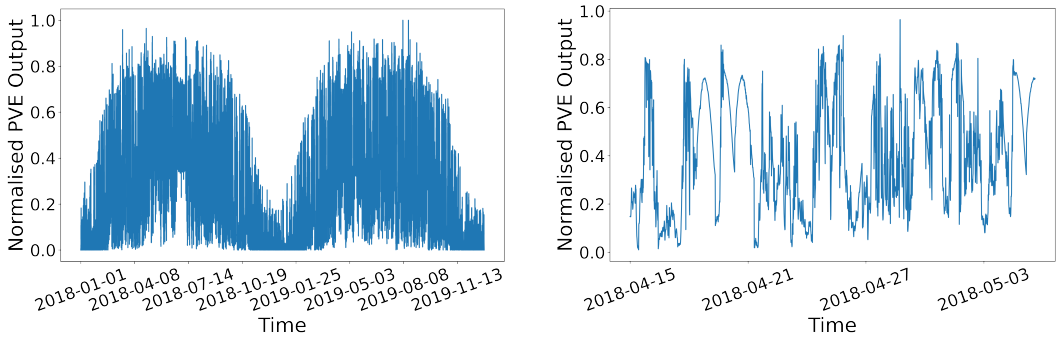


Figure 2: PV timeseries plot at annual and daily scales. The seasonality at the two scales are apparent.

[2] `https://huggingface.co/datasets/openclimatefix/uk_pv`

3 Methodology

To keep this paper self-contained, we supplement the readers with necessary background on GPs and details regarding our methodology in Appendix A.

3.1 Model

To benchmark our results on the PV nowcasting task, we consider two GPs to model the latent timeseries $f(t)$: the first is a plain Matérn-3/2 GP and in the second, we add a quasi-periodic component, which itself is a product between a periodic kernel and a Matérn-3/2 kernel:

$$k_{\text{model 1}}(t,t') = k_{\text{matérn-3/2}}(t,t'), \tag{1}$$

$$k_{\text{model 2}}(t,t') = k_{\text{matérn-3/2}}(t,t') + k_{\text{matérn-3/2}}(t,t')\, k_{\text{periodic}}(t,t'). \tag{2}$$

Given the latent timeseries $f(\cdot) \sim \mathcal{GP}(0, k(\cdot,\cdot))$, we consider a beta likelihood

$$p(y|f) = \prod_{i=1}^{N} \mathcal{B}\big(y_i \,|\, \alpha(f(t_i)), \beta(f(t_i))\big), \tag{3}$$

$$\text{where} \quad \alpha(f) = \Phi^{-1}(f) S_B, \quad \beta(f) = S_B - \alpha(f). \tag{4}$$

Here, $\Phi^{-1} : \mathbb{R} \to \mathbb{R}$ is the probit transformation and S_B is the scale hyperparameter. We chose this likelihood since the beta distribution $\mathcal{B}$ is defined over the interval $[0,1]$ and is capable of modelling a wide variety of shapes. This is suitable in our setting where the preprocessed data $(y_i)_{i=1}^N$ take values in $[0,1]$ and can have a skew towards 0 at certain times of the day.

We also consider a variety of commonly used baselines to assess the performance of our GP models (1)–(2) on the nowcasting task. These include the persistence, yesterday, hourly smoothing, exponential smoothing and vector autoregressive models. Details can be found in Appendix B.1.

3.2 State-space representation

One year of PV measurements from a single station contains $\mathcal{O}(10^5)$ data points, which is too large for a straight-forward application of GP regression. To deal with this, we follow the works [5, 13] to reformulate the system (1)+(3) or (2)+(3) as a *state-space model*

$$\boldsymbol{x}_{t+1} = \mathbf{A}_t \boldsymbol{x}_t + \boldsymbol{\varepsilon}_t, \quad \boldsymbol{\varepsilon}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_t), \tag{5}$$

$$y_t = \mathbf{H}_t \boldsymbol{x}_t + \eta_t, \quad \eta_t \sim \mathcal{N}(\mathbf{0}, R_t), \tag{6}$$

which enables us to use the Kalman filter (KF) to infer the posterior $p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t})$. Since our likelihood (3) is non-Gaussian, we use conjugate-computation variational inference [8] to approximately represent it in the form (6), which we explain in Appendix A.3. The cost of KF is linear in data size as opposed to the cubic cost of vanilla GP regression, making the inference tractable. Moreover, this has the added advantage of being able to consume new data on-the-fly without having to retrain.

4 Results

We now demonstrate our results on the solar PV nowscasting task. We used the `BayesNewton` package[3] to implement the experiments using state-space GP. All experiments were conducted on a MacBook Pro with Apple M1 chip and 16 GB memory.

We assess the performance of our model on a set of 27 PV stations in South-East England by cross-validating on 78 different dates between 2018-2019 (see Appendix B.3 for details on the cross-validation). In each CV fold, we use a period of 100 days for training and the subsequent two hours for prediction. For the metric, we use the mean absolute error and the negative log predictive density (see Appendix B.2), averaged over the 27 systems. We report the average values and variability of the MAE and NLPD across the 78 CV folds in Table 1.

Our cross-validation results suggest that model 2 performs the best overall, both in terms of the MAE and NLPD, beating the seasonal exponential smoothing baseline by a margin. On the other hand,

[3] https://github.com/AaltoML/BayesNewton

Model	MAE ↓ (mean ± std)	NLPD ↓ (median ± m.a.d.)
Persistence	0.119 ± 0.060	N/A
Yesterday	0.152 ± 0.091	N/A
Hourly smoothing	0.125 ± 0.061	N/A
Simple exponential smoothing	0.117 ± 0.058	-11.1 ± 11.1
Seasonal exponential smoothing	0.110 ± 0.049	-12.2 ± 10.4
Vector autoregression	0.129 ± 0.071	N/A
SS-GP model 1	0.134 ± 0.056	-4.93 ± 16.6
SS-GP model 2	$\mathbf{0.109 \pm 0.050}$	$\mathbf{-12.9 \pm 13.8}$

Table 1: Results of the GP models and baselines on the two-hour-ahead PV prediction task. We display the mean ± standard deviation of the MAE and median ± median absolute deviation of the NLPD across the 78 CV folds. The median was used for the NLPD due to the presence of outliers.

our control model 1 performs poorly, beating only the worst-performing 'yesterday' baseline. This indicates the importance of including periodicity to get good predictions.

In Figure 3, we plot the results from model 2 at two different dates, one on a particularly cloudy day (2018-01-29) and another on a day with scattered clouds (2018-02-01). We observe that the weather affects the performance dramatically, with the model predicting significantly worse on a day with scattered clouds due to the high volatility of PV output. Despite this, our model is still capable of producing 95% credible intervals that capture the ground truth PVE values. This results in large variability in the NLPD, as seen in Table 1.

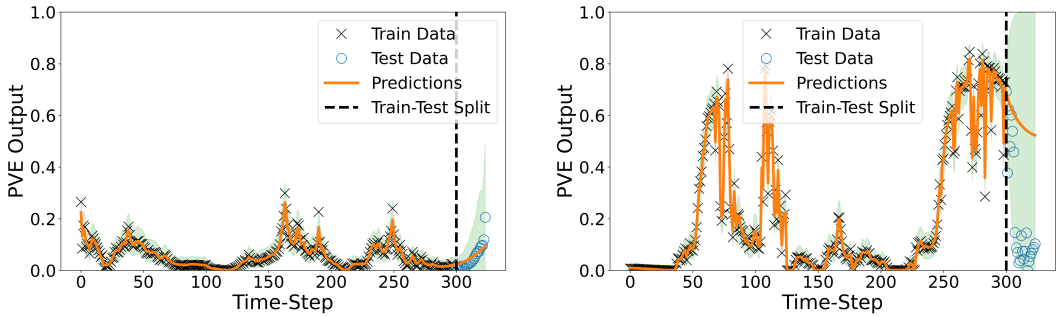


Figure 3: Example predictions from model 2 equipped with 95% credible intervals on two dates: 2018-01-29 (left) and 2018-02-01 (right) at a given PV station. Predictions are poor when cloud cover is sporadic (right), due to higher PVE volatility. See also Figure 8 for additional plots.

The above observation suggests that in order to improve predictions, we need to take into account exogeneous regressors, such as weather data and satellite images, which is beyond the scope of this work. Preliminary work using GPs to predict PV from satellite images was conducted in [9], which could be useful if we can combine it with our model. Taking into account the spatial correlations between different PV locations, as seen in Figure 1, may also help to improve predictions; however our initial experiments using spatio-temporal state-space GPs [4] demonstrated otherwise (see Appendix C for more details). We plan on investigating these directions further in future work.

5 Conclusion

In this exploratory work, we investigated the use of GPs as a tool for predicting solar PV with uncertainties. By recasting this process as a state-space model, inference using a large database of historical PV readings becomes numerically tractable. Moreover, the method allows us to incorporate new data on the fly, making it amenable to online data assimilation, which is key to a successful model in weather forecasting/nowcasting. While we only considered the PV timeseries as inputs to our model, in future, we aim to further investigate the inclusion of (1) spatial correlations between PV stations, and (2) exogenous regressors such as weather data and satellite images, within this framework to further improve predictions.

Acknowledgments and Disclosure of Funding

We would like to thank William Wilkinson and Ollie Hamelijnck for their generous help with the `BayesNewton` package, which we used in our experiments.

References

[1] London narrowly avoided post-heatwave blackout. `https://www-bbc-co-uk.cdn.ampproject.org/c/s/www.bbc.co.uk/news/uk-england-london-62296443.amp`, 2022.

[2] Milton Abramowitz and Irene A Stegun. *Handbook of mathematical functions with formulas, graphs, and mathematical tables*. US Government printing office, 1964.

[3] Kunal Sandip Garud, Simon Jayaraj, and Moo-Yeon Lee. A review on modeling of solar photovoltaic systems using artificial neural networks, fuzzy logic, genetic algorithm and hybrid models. *International Journal of Energy Research*, 2021.

[4] Oliver Hamelijnck, William Wilkinson, Niki Loppi, Arno Solin, and Theodoros Damoulas. Spatio-temporal variational Gaussian processes. *Advances in Neural Information Processing Systems*, 2021.

[5] Jouni Hartikainen and Simo Särkkä. Kalman filtering and smoothing solutions to temporal Gaussian process regression models. In *IEEE International Workshop on Machine Learning for Signal Processing*. IEEE, 2010.

[6] Rob J Hyndman and George Athanasopoulos. Forecasting: Principles and practice. *Online at https://otexts.com/fpp3/*, 2014.

[7] Sanket Kamthe, So Takao, Shakir Mohamed, and Marc Deisenroth. Iterative state estimation in non-linear dynamical systems using approximate expectation propagation. *Transactions on Machine Learning Research*, 2022.

[8] Mohammad Khan and Wu Lin. Conjugate-computation variational inference: Converting variational inference in non-conjugate models to inferences in conjugate models. In *Artificial Intelligence and Statistics*. PMLR, 2017.

[9] Yahya Al Lawati, Jack Kelly, and Dan Stowell. Short-term prediction of photovoltaic power generation using Gaussian process regression. *arXiv preprint arXiv:2010.02275*, 2020.

[10] Xing Luo, Dongxiao Zhang, and Xu Zhu. Deep learning based forecasting of photovoltaic power generation by incorporating domain knowledge. *Energy*, 2021.

[11] Helmut Lütkepohl. *New introduction to multiple time series analysis*. Springer Science & Business Media, 2005.

[12] Open Climate Fix. Nowcasting. `https://www.openclimatefix.org/projects/nowcasting/`.

[13] Simo Särkkä, Arno Solin, and Jouni Hartikainen. Spatiotemporal learning via infinite-dimensional Bayesian filtering and smoothing: A look at Gaussian process regression through Kalman filtering. *IEEE Signal Processing Magazine*, 2013.

[14] Arno Solin and Simo Särkkä. Explicit link between periodic covariance functions and state space models. In *Artificial Intelligence and Statistics*. PMLR, 2014.

[15] Di Su, Efstratios Batzelis, and Bikash Pal. Machine learning algorithms in forecasting of photovoltaic power generation. In *2019 International Conference on Smart Energy Systems and Technologies (SEST)*. IEEE, 2019.

[16] William J Wilkinson, Simo Särkkä, and Arno Solin. Bayes-Newton methods for approximate Bayesian inference with PSD guarantees. *arXiv preprint arXiv:2111.01721*, 2021.

[17] Christopher KI Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*. MIT press Cambridge, MA, 2006.

[18] Jinsong Zhang, Rodrigo Verschae, Shohei Nobuhara, and Jean-François Lalonde. Deep photovoltaic nowcasting. *Solar Energy*, 2018.

A Background

In this Appendix, we provide background materials for the discussions in §3.

A.1 Gaussian processes

A *Gaussian process (GP)* on $\mathbb{R}^d$ is a random function $f : \mathbb{R}^d \to \mathbb{R}$ such that for any finite collection of points $\mathbf{X} = (x_1, \ldots, x_N) \in \mathbb{R}^d \times \cdots \times \mathbb{R}^d$, the random variable $\boldsymbol{f} := (f(x_1), \ldots, f(x_N)) \in \mathbb{R}^N$ is jointly Gaussian. They are uniquely characterised by a mean function $\mu : \mathbb{R}^d \to \mathbb{R}$ and a covariance kernel $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$, where the former satisfies $\mu(x) = \mathbb{E}[f(x)]$ and the latter satisfies $k(x, x') = \mathrm{Cov}(f(x), f(x'))$ for all $x, x' \in \mathbb{R}^d$.

In statistical modelling, they can be used to specify priors over latent functions, whose distributions are updated as we acquire direct or indirect observations of this function. For example, when the dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$ consists of direct observations of f corrupted by i.i.d. Gaussian noise, i.e., $y_i = f(x_i) + \epsilon_i$ where $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$, we can compute our updated belief on f as

$$\mathbb{E}[f(\cdot)|\boldsymbol{y}] = \boldsymbol{k}_{\mathbf{X}}(\cdot)^T(\mathbf{K}_{\mathbf{XX}} + \sigma^2\mathbf{I})^{-1}\boldsymbol{y}, \tag{7}$$

$$\mathrm{Cov}\left(f(\cdot)|\boldsymbol{y}\right) = k(\cdot,\cdot) - \boldsymbol{k}_{\mathbf{X}}(\cdot)^T(\mathbf{K}_{\mathbf{XX}} + \sigma^2\mathbf{I})^{-1}\boldsymbol{k}_{\mathbf{X}}(\cdot). \tag{8}$$

Here, $\mathbf{K}_{\mathbf{XX}} = k(\mathbf{X}, \mathbf{X}) \in \mathbb{R}^{N\times N}$ is the covariance matrix of f evaluated at all the training inputs and $\boldsymbol{k}_{\mathbf{X}}(\cdot) \in \mathbb{R}^N$ is the cross-covariance of f evaluated at the training inputs and at a query point.

A.2 SDE representation of Gaussian processes

Computing the moment updates (7)–(8) pose several practical challenges that need to be considered. Firstly, we observe that the computation entails storing and inverting an $N \times N$ matrix $\mathbf{K}_{XX} + \sigma^2\mathbf{I}$, which has $\mathcal{O}(N^2)$ memory and $\mathcal{O}(N^3)$ compute cost. When the dataset is large (for example in the order of millions), the problem becomes intractable. Secondly, equations (7)–(8) do not allow for efficient assimilation of new datapoints – whenever a new datapoint (x_*, y_*) is acquired, (7)–(8) has to be recomputed every time with the augmented dataset $\mathcal{D}' = \mathcal{D} \cup \{(x_*, y_*)\}$.

In [5], a novel reformulation of GPs in one-dimension is presented, framing them as stochastic differential equations (SDEs). Specifically, given a zero-mean GP $f \sim \mathcal{GP}(0, k(\cdot,\cdot))$ defined over the real line, we can identify f with an SDE

$$\mathrm{d}\boldsymbol{x}(t) = \mathbf{F}\boldsymbol{x}(t)\,\mathrm{d}t + \mathbf{L}\,\mathrm{d}\boldsymbol{W}_t, \quad \boldsymbol{x}(0) \sim \mathcal{N}(\mathbf{0}, \mathbf{P}_\infty), \tag{9}$$

where $\boldsymbol{x}(t)$ is an abstract state vector related to the original process $f(t)$ via $f(t) = \mathbf{H}\boldsymbol{x}(t)$ for some linear operator $\mathbf{H}$. Also, $\mathbf{F}, \mathbf{L} \in \mathbb{R}^{n\times n}$ are real matrices and $\boldsymbol{W}_t \in \mathbb{R}^n$ is a vector-valued Wiener process. The matrices $\mathbf{H}, \mathbf{F}, \mathbf{L}$ and the initial state covariance $\mathbf{P}_\infty$ are determined (up to approximations) by the choice of kernel k.

Example 1 (Matérn-3/2 GP)**.** *We look at the SDE representation of the Matérn GP with $\nu = 3/2$, which can be found in [5, 13]. This kernel has the form*

$$k_{3/2}(t, t') = \sigma^2\left(1 + \frac{\sqrt{3}|t - t'|}{l}\right)\exp\left(-\frac{\sqrt{3}|t - t'|}{l}\right), \tag{10}$$

where l is the lengthscale hyperparameter and σ is the amplitude hyperparameter. By following the arguments in [5], we can represent it as the SDE

$$\mathrm{d}\begin{bmatrix} f(t) \\ f'(t)\end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\lambda^2 & -2\lambda\end{bmatrix}\begin{bmatrix} f(t) \\ f'(t)\end{bmatrix}\mathrm{d}t + \begin{bmatrix} 0 & 0 \\ 0 & \sqrt{q}\end{bmatrix}\mathrm{d}\mathbf{W}_t, \tag{11}$$

where $\lambda = \sqrt{3}/l$, $q = 4\sigma^2\lambda^3$ and $f'(t)$ denotes the first derivative of $f(t)$. Moreover, the initial condition for the system is given by

$$\begin{bmatrix} f(0) \\ f'(0) \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} \sigma^2 & 0 \\ 0 & -\sigma^2\lambda^2 \end{bmatrix} \right). \tag{12}$$

The solution $f(t)$ to the system (11)–(12) *is exactly equivalent to the Matérn-3/2 GP in distribution, that is, $f(\cdot) \sim \mathcal{GP}(0, k_{3/2}(\cdot,\cdot))$. In short, the following matrices determine the state-space representation of the Matérn-3/2 GP:*

$$\mathbf{H} = [1, 0], \quad \mathbf{F} = \begin{bmatrix} 0 & 1 \\ -\lambda^2 & -2\lambda \end{bmatrix}, \quad \mathbf{L} = \begin{bmatrix} 0 & 0 \\ 0 & \sqrt{q} \end{bmatrix}, \quad \mathbf{P}_\infty = \begin{bmatrix} \sigma^2 & 0 \\ 0 & -\sigma^2\lambda^2 \end{bmatrix}. \tag{13}$$

Example 2 (Periodic GP)**.** *The periodic kernel has the explicit form*

$$k_p(t, t') = \sigma^2 \exp\left(-\frac{2\sin^2\left(\omega_0 \frac{t-t'}{2}\right)}{l^2} \right), \tag{14}$$

where l is the lengthscale hyperparameter, σ is the amplitude hyperparameter and ω_0 is the frequency scale hyperparameter. In [14], it is shown that this kernel can be approximately expressed as a system of first-order differential equations

$$\frac{\mathrm{d}x_j(t)}{\mathrm{d}t} = -j\omega_0 y_j(t) \tag{15}$$

$$\frac{\mathrm{d}y_j(t)}{\mathrm{d}t} = j\omega_0 x_j(t), \tag{16}$$

for $j = 0, \ldots, J$, where $J \in \mathbb{N}$ is the approximation order. As we can see, the stochasticity does not appear in the equations themselves, but only in the initial conditions

$$\begin{bmatrix} x_j(0) \\ y_j(0) \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, q_j^2 \mathbf{I}\right), \quad j = 0, \ldots, J, \tag{17}$$

Here, the variance has the expression

$$q_j^2 = \begin{cases} \frac{I_0(l^{-2})}{\exp(l^{-2})}, & \text{for } j = 0 \\ \\ \frac{2I_j(l^{-2})}{\exp(l^{-2})}, & \text{otherwise} \end{cases}, \tag{18}$$

where I_j is the modified Bessel function of the first kind [2]. From the system of random differential equations (15)–(17)*, we obtain an approximation to the periodic GP by $f(t) \approx \sum_{j=0}^{J} x_j(t)$. Putting this together, we see that the following matrices determine the state-space representation of the Periodic GP:*

$$\mathbf{H} = [\mathbf{H}_1, \ldots, \mathbf{H}_J], \quad \mathbf{F} = \texttt{block\_diag}(\mathbf{F}_1, \ldots, \mathbf{F}_J), \tag{19}$$

$$\mathbf{L} = \texttt{block\_diag}(\mathbf{L}_1, \ldots, \mathbf{L}_J), \quad \mathbf{P}_\infty = \texttt{block\_diag}(\mathbf{P}_{\infty,1}, \ldots, \mathbf{P}_{\infty,J}), \tag{20}$$

where

$$\mathbf{H}_j = [1, 0], \quad \mathbf{F}_j = \begin{bmatrix} 0 & -j\omega_0 \\ j\omega_0 & 0 \end{bmatrix}, \quad \mathbf{L}_j = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}, \quad \mathbf{P}_{\infty,j} = q_j^2 \mathbf{I}. \tag{21}$$

We can also combine two or more kernels together to form a new kernel, either by summing them or taking products. Below, we summarise the state-space representation of such kernels.

Example 3 (Sum and product kernels)**.** *Let $k_1(\cdot,\cdot)$ and $k_2(\cdot,\cdot)$ be two kernels, whose corresponding GPs have the SDE representation $\{(\mathbf{H}_i, \mathbf{F}_i, \mathbf{L}_i, \mathbf{P}_{\infty,i})\}_{i\in\{1,2\}}$. The sum kernel $k(t,t') = k_1(t,t') + k_2(t,t')$ gives us a GP with the following SDE representation:*

$$\mathbf{H} = [\mathbf{H}_1, \mathbf{H}_2], \quad \mathbf{F} = \mathbf{F}_1 \oplus \mathbf{F}_2, \quad \mathbf{L} = \mathbf{L}_1 \oplus \mathbf{L}_2, \quad \mathbf{P}_\infty = \mathbf{P}_{\infty,1} \oplus \mathbf{P}_{\infty,2}, \tag{22}$$

where $\oplus$ denotes the direct sum $\mathbf{F}_1 \oplus \mathbf{F}_2 := \texttt{block\_diag}(\mathbf{F}_1, \mathbf{F}_2)$. This amounts to solving the two SDEs independently and aggregating only at the end, i.e., $f(t) = \mathbf{H}_1\boldsymbol{x}_1 + \mathbf{H}_2\boldsymbol{x}_2 = f_1(t) + f_2(t)$. The SDE representation of the product kernel $k(t,t') = k_1(t,t') \times k_2(t,t')$ on the other hand is given by [14]:

$$\mathbf{H} = \mathbf{H}_2 \otimes \mathbf{H}_1, \quad \mathbf{F} = \mathbf{F}_2 \otimes \mathbf{I}_1 + \mathbf{I}_2 \otimes \mathbf{F}_1, \quad \mathbf{L} = \mathbf{L}_2 \otimes \mathbf{L}_1, \quad \mathbf{P}_\infty = \mathbf{P}_{\infty,2} \otimes \mathbf{P}_{\infty,1}, \tag{23}$$

where $\otimes$ is the Kronecker product. This allows us to compute the SDE representation of a wide class of GPs, such as the quasi-periodic model 2 we considered in this work, from simple building blocks.

A.3 Variational inference

In cases where we have non-Gaussian likelihood, direct inference becomes intractable. Hence, we resort to approximate inference methods. In this work, we consider the conjugate-computation variational inference (CVI) scheme [8], which, as demonstrated in [4], works well with spatio-temporal GPs, although we may have also used other inference methods such as approximate EP / power EP [7, 16].

The goal of variational inference is to approximate an intractable posterior distribution $p(\boldsymbol{f}|\boldsymbol{y})$ by a tractable distribution $q(\boldsymbol{x})$, typically Gaussian with trainable mean and covariance, by minimising the KL-divergence $\mathcal{KL}(q(\cdot)\,||\,p(\cdot|\boldsymbol{y}))$. In CVI, this essentially boils down to finding an *approximate likelihood* $p(\boldsymbol{y}|\boldsymbol{f}) \approx \mathcal{N}(\widetilde{\boldsymbol{y}}|\boldsymbol{f}, \widetilde{\mathbf{R}})$ (note: this is seen as a function in $\boldsymbol{f}$, not $\boldsymbol{y}$!) such that

$$q(\boldsymbol{f}) \propto \mathcal{N}(\widetilde{\boldsymbol{y}}|\boldsymbol{f}, \widetilde{\mathbf{R}})p(\boldsymbol{f}), \tag{24}$$

where $p(\boldsymbol{f})$ is the GP prior. Denoting by $\widetilde{\boldsymbol{\lambda}} = (\widetilde{\boldsymbol{\lambda}}^{(1)}, \widetilde{\boldsymbol{\lambda}}^{(2)})$ the natural parameters of the approximate likelihood, that is, $\widetilde{\boldsymbol{\lambda}}^{(1)} = \widetilde{\mathbf{R}}^{-1}\widetilde{\boldsymbol{y}}$ and $\widetilde{\boldsymbol{\lambda}}^{(2)} = \widetilde{\mathbf{R}}^{-1}$, CVI proceeds via the following update rule [4]:

$$\widetilde{\boldsymbol{\lambda}} \leftarrow (1-\beta)\widetilde{\boldsymbol{\lambda}} + \beta \frac{\partial \mathbb{E}_{q(\boldsymbol{f})}[\log p(\boldsymbol{y}|\boldsymbol{f})]}{\partial[\boldsymbol{m}, \boldsymbol{m}\boldsymbol{m}^T + \mathbf{P}]}, \tag{25}$$

where $\boldsymbol{m}, \mathbf{P}$ are the current mean and covariance of the approximate posterior $q(\boldsymbol{f})$ and β is the learning rate. In practice, we can compute the gradients in (25) by using the Monte-Carlo method and the reparameterisation trick. Converting this back to the moment parameterisation, we get

$$\widetilde{\boldsymbol{y}}_{\text{new}} = (\widetilde{\boldsymbol{\lambda}}^{(2)}_{\text{new}})^{-1}\widetilde{\boldsymbol{\lambda}}^{(1)}_{\text{new}} \quad \text{and} \quad \widetilde{\mathbf{R}}_{\text{new}} = (\widetilde{\boldsymbol{\lambda}}^{(2)}_{\text{new}})^{-1}. \tag{26}$$

When the likelihood is separable, that is,

$$p(\boldsymbol{y}|\boldsymbol{f}) = \prod_{n=1}^{N} p(y_n|f_n), \tag{27}$$

we can apply updates on the marginals $q(f_n)$ to compute approximate likelihoods $p(y_n|f_n) \approx \mathcal{N}(\widetilde{y}_n|f_n, \widetilde{R}_n)$ at each time step independently, resulting in a diagonal $\widetilde{\mathbf{R}}$ matrix in (24).

A.4 State-space formulation

Given the above discussions, we can now express our model

$$f \sim \mathcal{GP}(0, k(\cdot,\cdot)) \quad y_n \sim p(y_n|f(t_n)), \quad n = 1, \ldots, N, \tag{28}$$

in *state-space form* to perform inference using the Kalman filter. This reads

$$\boldsymbol{x}_{n+1} = \mathbf{A}_n\boldsymbol{x}_n + \boldsymbol{\varepsilon}_n, \quad \boldsymbol{\varepsilon}_n \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_n), \tag{29}$$

$$\widetilde{y}_{n+1} = \mathbf{H}\boldsymbol{x}_{n+1} + \eta_{n+1}, \quad \eta_{n+1} \sim \mathcal{N}(\mathbf{0}, \widetilde{R}_{n+1}), \tag{30}$$

for $n = 0, 1, \ldots, N-1$, where $\mathbf{A}_n := \exp(\mathbf{F}\Delta t_n)$, $\mathbf{Q}_n := \mathbf{L}(\Delta t_n)\mathbf{L}^T$ (recall the matrices $\mathbf{F}, \mathbf{L}$ from §A.2), $\widetilde{\boldsymbol{y}} = (\widetilde{y}_1, \ldots, \widetilde{y}_N)$, $\widetilde{\mathbf{R}} = \texttt{diag}(\widetilde{R}_1, \ldots, \widetilde{R}_N)$ are the mean and covariance respectively of the approximate likelihood computed in §A.3, and Δt_n is taken to be the distance between two consecutive input data points $\Delta t_n = t_{n+1} - t_n$. The initial state distribution is $\boldsymbol{x}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{P}_\infty)$.

The full filtering algorithm is shown in Algorithm 1. We denoted the mean and covariance of the filtering distribution $p(\boldsymbol{x}_n|\boldsymbol{y}_{1:n})$ by $\boldsymbol{m}_{n|n}, \mathbf{P}_{n|n}$, respectively, and likewise, the mean and covariance of the predictive distribution $p(\boldsymbol{x}_{n+h}|\boldsymbol{y}_{1:n})$ by $\boldsymbol{m}_{n+h|n}, \mathbf{P}_{n+h|n}$ for $h = 1, 2, \ldots$. By using the Kalman filter, we can perform GP inference that only scales linearly in N as opposed to the cubic cost in vanilla GP inference. Moreover, it has the added advantage of being able to assimilate new data on-the-fly, allowing for applications in online settings.

Algorithm 1 Filtering algorithm for model (28)

Init: $\boldsymbol{m}_{0|0} = \mathbf{0}$, $\mathbf{P}_{0|0} = \mathbf{P}_\infty$
for $n = 0$ to $N-1$ **do**
    1. Prediction:

$$\boldsymbol{m}_{n+1|n} = \mathbf{A}_n \boldsymbol{m}_{n|n} \tag{31}$$

$$\mathbf{P}_{n+1|n} = \mathbf{A}_n \mathbf{P}_{n|n} \mathbf{A}_n^T + \mathbf{Q}_n \tag{32}$$

    while Stopping criterion not met **do**
        2. Update $(\widetilde{\boldsymbol{y}}_{n+1}, \widetilde{\mathbf{R}}_{n+1})$ using CVI (25)–(26).
        3. Compute Kalman gain:

$$\mathbf{K}_{n+1} = \mathbf{P}_{n+1|n}\mathbf{H}^T(\mathbf{H}\mathbf{P}_{n+1|n}\mathbf{H}^T + \widetilde{\mathbf{R}}_{n+1})^{-1} \tag{33}$$

        4. Update moments:

$$\boldsymbol{m}_{n+1|n+1} = \boldsymbol{m}_{n+1|n} + \mathbf{K}_{n+1}(\widetilde{\boldsymbol{y}}_{n+1} - \mathbf{H}\boldsymbol{m}_{n+1|n}) \tag{34}$$

$$\mathbf{P}_{n+1|n+1} = (\mathbf{I} - \mathbf{K}_{n+1}\mathbf{H})\mathbf{P}_{n+1|n} \tag{35}$$

$$q(f_{n+1}) \leftarrow \mathcal{N}(f_{n+1} \,|\, \mathbf{H}\boldsymbol{m}_{n+1|n+1}, \mathbf{H}\mathbf{P}_{n+1|n+1}\mathbf{H}^T) \tag{36}$$

    end while
end for
return $(\boldsymbol{m}_{n|n}, \mathbf{P}_{n|n})_{n=0}^N$

A.5 Hyperparameter optimisation

We can also train our model on the dataset to find optimal hyperparameters $\boldsymbol{\Theta}$ using the CVI framework. The hyperparameters in this case include the GP hyperparemeters (such as the lengthscale l) and hyperparemeters in the likelihood distribution. This typically proceeds by maximizing the marginal likelihood

$$p(\boldsymbol{y}_{1:N}|\boldsymbol{\Theta}) = \int p(\boldsymbol{y}_{1:N}|\boldsymbol{f}, \boldsymbol{\Theta})\, p(\boldsymbol{f}|\boldsymbol{\Theta}) \mathrm{d}\boldsymbol{f}, \tag{37}$$

which, for non-Gaussian $p(\boldsymbol{y}_{1:N}|\boldsymbol{f})$, does not have a closed form. Instead, what we do is to maximise the so-called evidence lower bound (ELBO)

$$\mathcal{L}_{\text{ELBO}}(\boldsymbol{\Theta}, q) = \mathbb{E}_{q(\boldsymbol{f})}\left[\log \frac{p(\boldsymbol{y}|\boldsymbol{f}, \boldsymbol{\Theta}) p(\boldsymbol{f}|\boldsymbol{\Theta})}{q(\boldsymbol{f})}\right], \tag{38}$$

for some tractable distribution q, which satisfies

$$\log p(\boldsymbol{y}_{1:N}|\boldsymbol{\Theta}) = \mathcal{L}_{\text{ELBO}}(\boldsymbol{\Theta}, q) + \mathcal{KL}(q(\boldsymbol{f}) \,||\, p(\boldsymbol{f}|\boldsymbol{y}, \boldsymbol{\Theta})). \tag{39}$$

Thus, for $q(\boldsymbol{f}) \approx p(\boldsymbol{f}|\boldsymbol{y}, \boldsymbol{\Theta})$, and therefore $\mathcal{KL}(q(\boldsymbol{f}) \,||\, p(\boldsymbol{f}|\boldsymbol{y}, \boldsymbol{\Theta})) \approx 0$, the ELBO acts as a lower-bound proxy to the log marginal-likelihood. To this end, we can do an EM-style update to find the optimal hyperparameters $\boldsymbol{\Theta}^*$, by alternating between updating the approximate posterior $q(\boldsymbol{f})$ (E-step) and doing a gradient-based maximisation of $\mathcal{L}_{\text{ELBO}}(\boldsymbol{\Theta}, q)$ with respect to $\boldsymbol{\Theta}$ for q held fixed (M-step).

Assuming (24), we can neatly perform this optimisation by combining it with the Kalman filter/smoother, as shown in [4]. In this case, the ELBO (38) can be re-expressed as:

$$\mathcal{L}_{\text{ELBO}}(\boldsymbol{\Theta}) = \sum_{n=1}^N \mathbb{E}_{q(f_n)}\left[\log p(y_n|f_n, \boldsymbol{\Theta})\right] - \sum_{n=1}^N \mathbb{E}_{q(f_n)}\left[\log \mathcal{N}(\widetilde{y}_n|f_n, \widetilde{\mathbf{R}}_n)\right] \tag{40}$$

$$+ \sum_{n=1}^N \mathbb{E}_{p(\boldsymbol{x}_n|\boldsymbol{y}_{1:n-1})}\left[\mathcal{N}(\widetilde{y}_n|\mathbf{H}\boldsymbol{x}_n, \widetilde{\mathbf{R}}_n)\right]. \tag{41}$$

The E-step in this setting becomes equivalent to updating the approximate marginal posteriors $\{q(f_n)\}_{n=1}^N$ so that $q(f_n) \approx p(f_n|\boldsymbol{y}_{1:N}, \boldsymbol{\Theta})$, which we can achieve via Gaussian smoothing (Algorithm 2). We can also approximate the predictive distribution $p(\boldsymbol{x}_n|\boldsymbol{y}_{1:n-1})$ in the third term by $\mathcal{N}(\boldsymbol{x}_n \,|\, \boldsymbol{m}_{n|n-1}, \boldsymbol{P}_{n|n-1})$ using the filtering algorithm (Algorithm 1).

Algorithm 2 Smoothing algorithm for model (28)

Init: Run Algorithm 1 to compute the filtering distributions $(\boldsymbol{m}_{n|n}, \mathbf{P}_{n|n})$ for $n = 0, \ldots, N$
for $n = N$ to 1 **do**
    1. Prediction:

$$\boldsymbol{m}_{n|n-1} = \mathbf{A}_{n-1}\boldsymbol{m}_{n-1|n-1} \tag{42}$$

$$\mathbf{P}_{n|n-1} = \mathbf{A}_{n-1}^T\mathbf{P}_{n-1|n-1}\mathbf{A}_{n-1} + \mathbf{Q}_{n-1} \tag{43}$$

    2. Compute smoother gain:

$$\mathbf{J}_{n-1} = \mathbf{P}_{n-1|n-1}\mathbf{A}_{n-1}\mathbf{P}_{n|n-1}^{-1} \tag{44}$$

    3. Update:

$$\boldsymbol{m}_{n-1|N} = \boldsymbol{m}_{n-1|n-1} + \mathbf{J}_{n-1}(\boldsymbol{m}_{n|N} - \boldsymbol{m}_{n|n-1}) \tag{45}$$

$$\mathbf{P}_{n-1|N} = \mathbf{P}_{n-1|n-1} + \mathbf{J}_{n-1}(\mathbf{P}_{n|N} - \mathbf{P}_{n|n-1})\mathbf{J}_{n-1}^T \tag{46}$$

$$q(f_{n-1}) \leftarrow \mathcal{N}(f_{n-1} \,|\, \mathbf{H}\boldsymbol{m}_{n-1|N}, \mathbf{H}\mathbf{P}_{n-1|N}\mathbf{H}^T) \tag{47}$$

end for
return $(\boldsymbol{m}_{n|N}, \mathbf{P}_{n|N})_{n=0}^N$

B Experimental details

Here, we provide further details on the experiments conducted in Section 4.

B.1 Baselines

We consider a variety of baselines to assess the performance of the GP models applied to the PV nowcasting task.

Persistence. The most simple and trivial benchmark is the persistence model, defined as:

$$\hat{y}_{n+1} = y_n. \tag{48}$$

In other words, it predicts the next observations as the most recent value of the time series. This simple model is surprisingly hard to beat, and is a good first benchmark to compare against.

Yesterday. A slight variation of the persitence is the yesterday model, defined as:

$$\hat{y}_{n+1} = y_{n+1-\#24\text{hr}}, \tag{49}$$

where '#24hr' indicates the number of timesteps corresponding to 24 hours in physical time. This is a good baseline for data that shows strong daily seasonality.

Hourly smoothing. This is the first model here that performs any degree of smoothing (i.e., averaging of past values). This is given by (assuming that the time interval between each observations is 5 minutes):

$$\hat{y}_{n+1} = \frac{1}{12}\sum_{i=1}^{12} y_{n+1-i}. \tag{50}$$

We take averages of the past 12 data points since 1 hour = 5 minutes $\times$ 12 and we are smoothing over data in the past hour, as the name suggests.

Simple Exponential Smoothing. The simple exponential smoothing (EST) is one of the most effective methods of forecasting when no clear trend or seasonality are present [6]. This is similar to hourly smoothing, except that instead of cutting off the averaging at 1 hour and using uniform weights, the weights decay exponentially with time:

$$\hat{y}_{n+1} = \sum_{j=0}^{J-1} \alpha(1-\alpha)^j y_{n-j} + (1-\alpha)^n \ell_0, \tag{51}$$

where $\alpha \in [0, 1]$ is the smoothing exponent, and ℓ_0 is the initial value, also treated as a hyperparameter. We can also write this in "component form" as:

$$\begin{cases} \hat{y}_{n+1} = \ell_n, \\ \ell_n = \alpha y_n + (1 - \alpha)\ell_{n-1}. \end{cases} \tag{52}$$

Defining the "error" $\varepsilon_n := y_n - \ell_{n-1}$, which we assume to be distributed according to $\mathcal{N}(0, \sigma^2)$ (again, σ is a hyperparameter to be optimised), we can further write this in state-space form:

$$\begin{cases} y_n = \ell_{n-1} + \varepsilon_n, \\ \ell_n = \ell_{n-1} + \alpha\varepsilon_n. \end{cases} \tag{53}$$

This is now a stochastic model, which allows us to get uncertainty estimates alongside predictions by means of Kalman filtering/smoothing.

Seasonal Exponential Smoothing. The seasonal EST method, also called Holt-Winters' seasonal method [6], is an extension of the EST algorithm to capture trend and seasonality in data. The additive model is defined by three smoothing equations: one for the level term ℓ_n, one for the trend term b_n, and one for the seasonal term s_n, with parameters α, β^*, and γ respectively. Letting m be the period (in timesteps) of said seasonality, the model is given by:

$$\begin{cases} \hat{y}_{n+1} = \ell_n + b_n + s_{n+1-m} \\ \ell_n = \alpha(y_n - s_{n-m}) + (1 - \alpha)(\ell_{n-1} + b_{n-1}) \\ b_n = \beta^*(\ell_n - \ell_{n-1}) + (1 - \beta^*)b_{n-1} \\ s_n = \gamma(y_n - \ell_{n-1} - b_{n-1}) + (1 - \gamma)s_{n-m}. \end{cases} \tag{54}$$

Similarly as before, defining the error $\varepsilon_n = y_n - \ell_{n-1} - b_{n-1} \sim \mathcal{N}(0, \sigma^2)$, we can express this in state-space form:

$$\begin{cases} y_n = \ell_{n-1} + b_{n-1} + s_{n-m} + \varepsilon_n \\ \ell_n = \ell_{n-1} + b_{n-1} + \alpha\varepsilon_n \\ b_n = b_{n-1} + \beta\varepsilon_n \\ s_n = s_{n-m} + \gamma\varepsilon_n, \end{cases} \tag{55}$$

where $\beta := \alpha\beta^*$. Again, this enables us to get uncertainty estimates via Kalman filtering/smoothing.

Vector Autoregression. The Vector Autoregressive model (VAR) is the only baseline that we considered capable of learning correlations between time series at different PV stations [11]. With VAR, the spatio-temporal process is modelled as a multivariate time series $\mathbf{Y} = (\boldsymbol{y}_1, \ldots, \boldsymbol{y}_{n_t}) \in \mathbb{R}^{n_t \times n_s}$, given by:

$$\boldsymbol{y}_n = \boldsymbol{\nu} + \sum_{j=1}^{J} \mathbf{A}_j \boldsymbol{y}_{n-j} + \boldsymbol{\varepsilon}_t, \quad \boldsymbol{\varepsilon}_n \sim \mathcal{N}(0, \boldsymbol{\Sigma}), \tag{56}$$

where $\boldsymbol{\nu}$ is a vector of intercepts to allow for non-zero means, J is the order of the lag, $(\mathbf{A}_i)_{i=1}^J$ are $(n_s \times n_s)$ matrices, and $\boldsymbol{\Sigma} \in \mathbb{R}^{n_s \times n_s}$ is the error covariance. Since this is a stochastic model, we can also get uncertainty estimates via Kalman filtering/smoothing.

We implemented the baselines using the `statsmodels` package and selected the hyperparameters using Bayesian optimisation.

B.2 Evaluation metrics

We evaluate the performance of our models on a hold-out set $\mathcal{D}^* = \{(t_i^*, y_i^*)\}_{i=1}^{N^*}$, by comparing the mean absolute error (MAE) and the negative log predictive density (NLPD), respectively, defined as:

$$\text{MAE} := \frac{1}{N^*} \sum_{i=1}^{N^*} \left| y_i^* - \mathbb{E}[f(t_i^*) \,|\, \mathcal{D}] \right|, \tag{57}$$

$$\text{NLPD} := -\sum_{i=1}^{N^*} \log p(y_i^* | \mathcal{D}) \tag{58}$$

$$= -\sum_{i=1}^{N^*} \log \int p(y_i^* | \boldsymbol{f}^*) p(\boldsymbol{f}^* | \mathcal{D}) \mathrm{d}\boldsymbol{f}^*, \tag{59}$$

where f is our model, $\mathcal{D}$ is the training set, and $\boldsymbol{f}^*$ denotes the vector of f evaluated at the test input locations $(t_1^*, \ldots, t_{N^*}^*)$. The former assesses only the quality of the predictive mean, while the latter also assesses the quality of the predictive uncertainty. In practice, we approximate the integral in (59) using Monte-Carlo method when we have a non-conjugate likelihood.

When the model f is deterministic, we only evaluate the MAE, replacing the conditional mean $\mathbb{E}[f(t_i^*) \,|\, \mathcal{D}]$ in (57) by $f(t_i^*)$.

B.3 Cross-validation method

We evaluated our models on 27 PV systems located in the South-East of England, which we plot in Figure 4, and used a walk-forward Cross-Validation (CV) with rolling window to assess their performance. Walk-forward CV must be used in time series data to avoid data leakage, as it ensures that the model never uses data that follows chronologically after the prediction window. In Figure 5 we illustrate a typical time series split used in walk-forward CV.

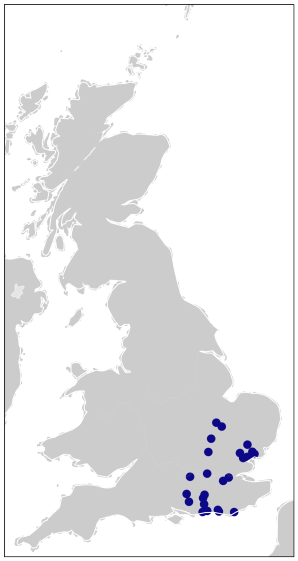


Figure 4: Plot of PV systems scattered in the south-east of England. This sub-sample of PV systems was used during the experimentation phase to be able to run more experiments with lower computational requirements.

In our CV scheme, we used training windows of 100 days and test windows of 2 hours. The models were trained and tested for 78 CV folds, each fold shifted by a day, but ensuring that the time of day at which the forecasts were executed differed between each fold. We also set our folds so that the forecast time always fell between 10:00 and 14:00, so that there are at least two hours of observations within the same day both before and after the forecast (recall that in our preprocessing step, we sliced

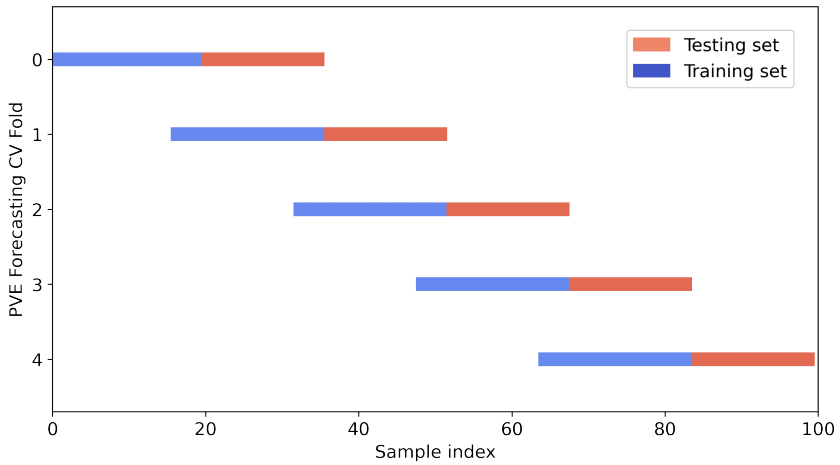


Figure 5: Illustration of our cross-validation scheme with rolling windows.

the timeseries to include only data between 8:00 and 16:00). This was to ensure that the models were tested only on data within the same day to avoid discontinuity in the prediction time window.

To speed up the computation, we also initialised trainable parameters in each fold based on computations in the previous fold. In particular, model hyperparameters in each fold (except the first) were initialised with the final value in the previous fold and the variational parameters $(\widetilde{\boldsymbol{y}}, \widetilde{\mathbf{R}})$ in the approximate likelihood (see A.3) were reused at coinciding points in the training windows.

C Results from spatio-temporal GPs

We have also done some initial experiments using spatio-temporal state-space GPs [13], to take into account spatial correlations between timeseries at different PV stations. We used separable kernels to model the spatio-temporal data, i.e.,

$$k((\boldsymbol{s}, t), (\boldsymbol{s}', t')) = k_s(\boldsymbol{s}, \boldsymbol{s}')\, k_t(t, t'), \tag{60}$$

where the same temporal kernels were used from §3.1 and the Matérn-3/2 kernel was used for the spatial component. Following [4], the same inference method that we described in Appendix A hold for such kernels with slight modification, only that we get a coupled system of 27 stochastic differential equations, instead of 27 independent ones.

In Table 2, we demonstrate the results of spatio-temporal GPs, one that is coupled with a simple Matérn temporal component, which we called 'model 1' in §3.1, and the other that is coupled with a quasi-periodic temporal kernel, which we called 'model 2' in §3.1.

Model	MAE ↓ (mean ± std)	NLPD ↓ (median ± m.a.d.)
Temporal Simple GP	0.134 ± 0.056	-4.93 ± 16.6
Temporal Quasi-Periodic GP	$\mathbf{0.109 \pm 0.050}$	$\mathbf{-12.9 \pm 13.8}$
Spatio-temporal Simple GP	0.175 ± 0.056	-6.88 ± 13.0
Spatio-temporal Quasi-Periodic GP	0.140 ± 0.058	-6.89 ± 29.9

Table 2: Comparison of purely temporal GPs vs. spatio-temporal GPs. Our initial results suggest that including the spatial component deteriorates results rather than improving it, both in terms of the MAE and the NLPD.

The results in Table 2 suggest that the spatial correlation between stations does not help improve our predictions as we had hoped, but rather deteriorates it. A possible reason for this is simply that we could not train our models until convergence as a result of the increased computational cost. It could also be suggesting that since we are predicting solar power directly, which can be affected by external factors such as the make of solar panels, orientation of solar panels, etc., it is better to model each

timeseries individually. However, at first glance, there appears to be substantial spatial correlations between neighbouring systems, as we can see from Figure 1, so we may be able to improve on this upon further work.

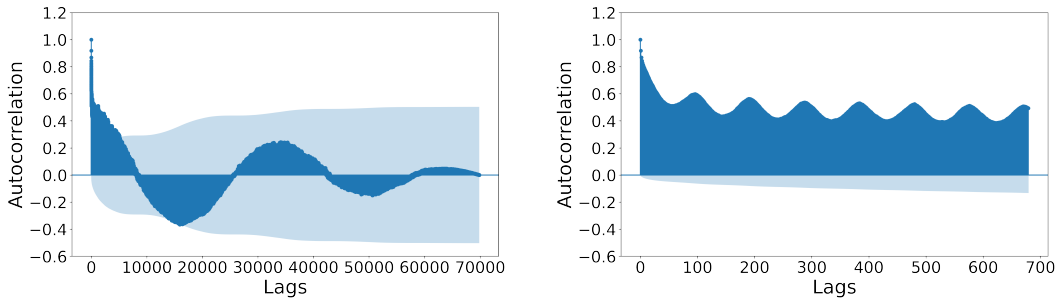
D Additional plots

D.1 Exploratory data analysis

Here, we plot the results of our exploratory analysis on the PV timeseries data.

In our exploratory analysis, we first studied the seasonality and autocorrelation properties of the PV timeseries, as seen for example in Figure 2. A random PV system was selected for the analysis, however every timeseries showed very similar characteristics.

We used the autocorrelation function (ACF) to detect seasonality in our timeseries. The ACF is defined as $\rho(k) = \frac{\gamma(k)}{\gamma(0)}$ where $\gamma(k) = \text{Cov}(y_i, y_i + k)$ is the autocovariance, and $\gamma(0)$ is the variance of the timeseries. An ACF close to 1 or -1 represents a high degree of autocorrelation, which can help identify seasonality in data. The ACF is shown in Figure 6 for a two-year window and a one-week window respectively, where 96 lags correspond to one day of observations.



(a) ACF for two-year window

(b) ACF for one-week window

Figure 6: ACF plots at windows of (a) two years, and (b) one-week. A clear daily and yearly seasonality is seen.

From the above plot, we see a clear daily and yearly seasonality. Moreover, the high values observed in the ACF at the first lag suggests high correlation with the most recent observation.

In the second step of our analysis, we studied the correlation between the PV values and available information on each PV station, including orientation, tilt, and capacity of each system. Intuitively, a system that is oriented and tilted towards the sun could yield more power than a system that is oriented and tilted away from the sun. Capacity was also studied to understand if there is any effect related to the size of the system and its expected production. The PV outputs were averaged over the entire time frame for each system. We plot the correlation matrices in Figure 7.

The correlation matrices do not indicate any relationship between the three explanatory variables and the average PV generation. In an effort to further identify any such relationship, an ordinary least square regression model was fitted, however none of the three explanatory variables were found to have a significant relationship with the response variable. For this reason, we deduce that none of the available explanatory system-specific variables will be useful for predicting PV output.

D.2 Additional predictions

In Figure 8, we plot additional predictions at different PV stations from our best-performing model at two different dates: 2018-01-29 and 2018-02-01. The former was a cloudy day, resulting in overall low PV readings with little-to-moderate variance. The latter was a day with scattered clouds resulting in wildly varying PV readings.

We see that in all stations, the model performs well in the former case. On the other hand, it suffers greatly in the latter case, resulting in high predictive variance, which, while still capturing the ground truth values, has uncertainties too large for it to be useful in downstream tasks.

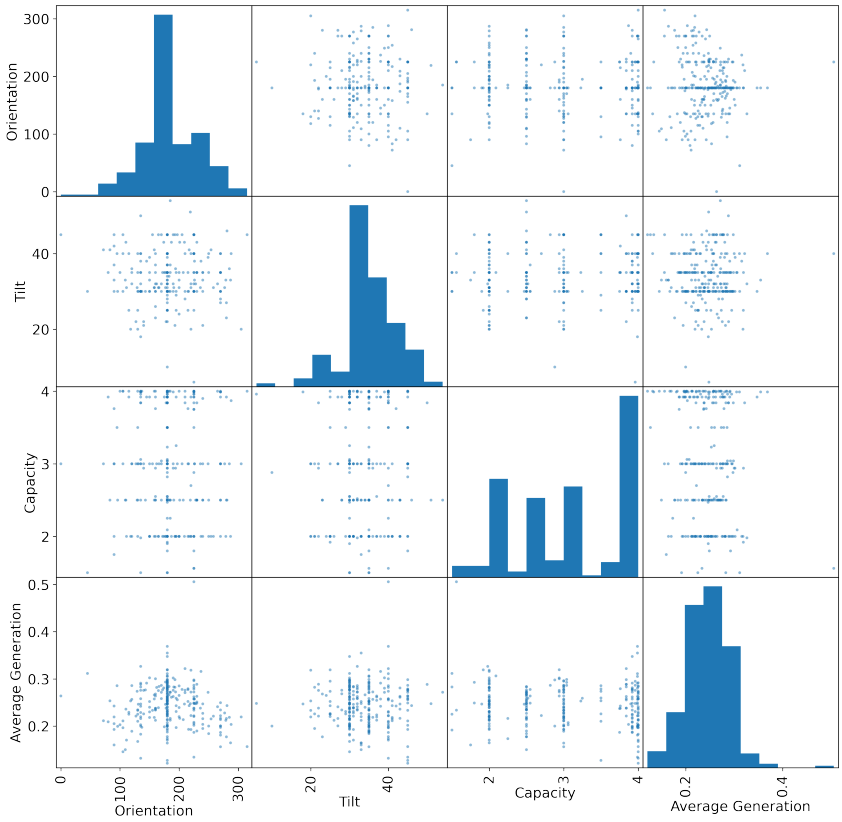
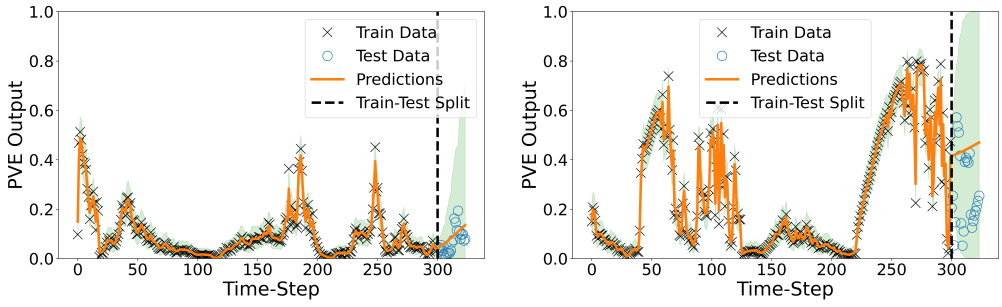
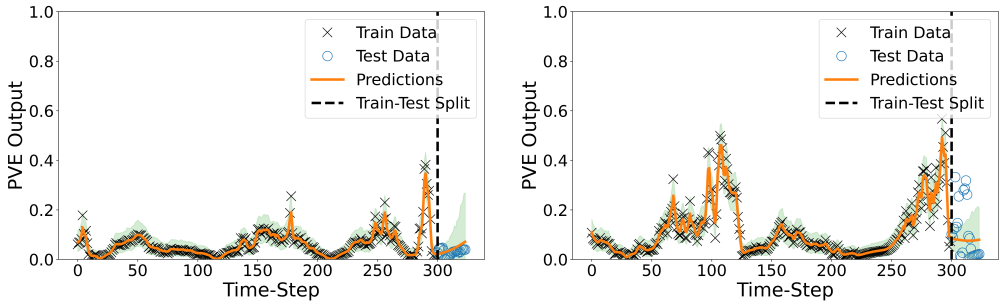


Figure 7: Scatter matrix of three explanatory variables - orientation, tilt, and capacity - and the average generation. No strong relationship can be observed between the explanatory variables and average PV generation.



(a) Predictions at 2018-01-29



(b) Predictions at 2018-01-29

(c) Predictions at 2018-02-01

(d) Predictions at 2018-02-01

Figure 8: Predictions from our best performing model at three different PV stations at (a) 2018-01-29, and (b) 2018-02-01. Predictions are generally poor when the timeseries is highly stochastic due to cloud motion and other possible external factors.