Physics-Aware Gaussian Processes in Remote Sensing

Article *in* Applied Soft Computing · March 2018

DOI: 10.1016/j.asoc.2018.03.021

CITATION
1

READS
432

8 authors, including:

Gustau Camps-Valls
University of Valencia
622 PUBLICATIONS **13,323** CITATIONS
SEE PROFILE

Luca Martino
King Juan Carlos University
129 PUBLICATIONS **1,113** CITATIONS
SEE PROFILE

Daniel Heestermans Svendsen
University of Valencia
6 PUBLICATIONS **22** CITATIONS
SEE PROFILE

Manuel Campos-Taberner
University of Valencia
51 PUBLICATIONS **330** CITATIONS
SEE PROFILE

Applied Soft Computing 68 (2018) 69–82

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Physics-aware Gaussian processes in remote sensing

Gustau Camps-Valls[a,*], Luca Martino[a], Daniel H. Svendsen[a], Manuel Campos-Taberner[b], Jordi Muñoz-Marí[a], Valero Laparra[a], David Luengo[c], Francisco Javier García-Haro[b]

[a] Image Processing Laboratory (IPL), Universitat de València, Spain
[b] Earth Physics and Thermodynamics Department, Faculty of Physics, Universitat de València, Dr. Moliner, 46100 Burjassot, València, Spain
[c] Signal Processing and Communications Dep., Univ. Politécnica de Madrid, Spain

ARTICLE INFO

Article history:
Received 8 January 2018
Received in revised form 13 March 2018
Accepted 14 March 2018
Available online 22 March 2018

Keywords:
Earth observation
Remote sensing
Vegetation
Kernel methods
Gaussian processes (GPs)
Inverse modeling
Geosciences
Radiative transfer models (RTMs)

ABSTRACT

Earth observation from satellite sensory data poses challenging problems, where machine learning is currently a key player. In recent years, Gaussian Process (GP) regression has excelled in biophysical parameter estimation tasks from airborne and satellite observations. GP regression is based on solid Bayesian statistics, and generally yields efficient and accurate parameter estimates. However, GPs are typically used for inverse modeling based on concurrent observations and *in situ* measurements only. Very often a *forward model* encoding the well-understood physical relations between the state vector and the radiance observations is available though and could be useful to improve predictions and understanding. In this work, we review three GP models that respect and learn the physics of the underlying processes in the context of both *forward and inverse modeling*. After reviewing the traditional application of GPs for parameter retrieval, we introduce a Joint GP (JGP) model that combines *in situ* measurements and simulated data in a single GP model. Then, we present a latent force model (LFM) for GP modeling that encodes ordinary differential equations to blend data-driven modeling and physical constraints of the system governing equations. The LFM performs multi-output regression, adapts to the signal characteristics, is able to cope with missing data in the time series, and provides explicit latent functions that allow system analysis and evaluation. Finally, we present an Automatic Gaussian Process Emulator (AGAPE) that approximates the forward physical model using concepts from Bayesian optimization and at the same time builds an optimally compact look-up-table for inversion. We give empirical evidence of the performance of these models through illustrative examples of vegetation monitoring and atmospheric modeling.

1. Introduction

Solving inverse problems is a recurrent topic of research in Engineering and Physics in general, and in Remote Sensing and Earth Observation (EO) in particular. A very relevant inverse problem is that of estimating vegetation properties from remotely sensed images. Accurate inverse models help to determine the phenological stage and health status (e.g. development, productivity, stress) of crops and forests [1], which has important societal, environmental and economical implications. Leaf area index (LAI) defined as half the total intercepting leaf area per unit ground surface area [2], leaf chlorophyll content (*Chl*), fraction of absorbed photosynthetically active radiation (fAPAR), and fractional vegetation cover (FVC) are among the most important vegetation parameters to retrieve from space observations [3,4].

In general, physical models implement the laws of Physics and allow us to compute the observation values given a state and a model [5]. Sometimes, and depending on the body of literature, they are known as process-based models or mechanistic models. In remote sensing, we refer to them as *radiative transfer models* as they implement the equations of energy (radiation) transfer. This is known as the *forward modeling* problem. In the *inverse modeling* problem, the aim is to reconstruct the system state from a set of measurements (observations), see Fig. 1. Notationally, a forward model describing the system is generally expressed as $\mathbf{x} = g(\mathbf{y}, \boldsymbol{\omega})$, where $\mathbf{x}$ is a measurement obtained by the satellite (e.g. radiance); the vector $\mathbf{y}$ represents the state of the biophysical variables on the Earth (which we desire to infer or predict and are often referred to as *outputs* in the inverse modeling approach); $\boldsymbol{\omega}$ contains a set of controllable conditions (e.g. wavelengths, viewing direction, time, Sun position, and polarization); and $g(\cdot)$ is a function which relates $\mathbf{y}$ with $\mathbf{x}$. Such a function g is typically considered to be nonlinear, smooth and continuous. Our goal is to obtain an inverse model, $f(\cdot) \approx g^{-1}(\cdot)$, parameterized by $\boldsymbol{\theta}$, which approxi-

* Corresponding author.
E-mail address: gustau.camps@uv.es (G. Camps-Valls).

https://doi.org/10.1016/j.asoc.2018.03.021
1568-4946/

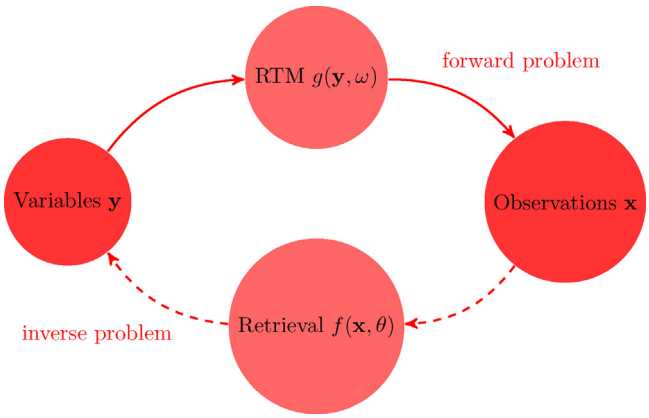

Fig. 1. Forward (solid lines) and inverse (dashed lines) problems in remote sensing.

mates the bio-geo-physical variables $\mathbf{y}$ given the data $\mathbf{x}$ received by the satellite, i.e., $\hat{\mathbf{y}} = f(\mathbf{x}, \boldsymbol{\theta})$. Radiative transfer models (RTMs) are typically used to implement the forward direction [6,7]. However, inverting RTMs directly is very complex because the number of unknowns is generally larger than the number of independent radiometric information [8]. Also, estimating physical parameters from RTMs is hampered by the presence of high levels of uncertainty and noise, primarily associated to atmospheric conditions, sensor calibration, sun angle, viewing geometry, as well as the poor sampling of the parameter space in most of the applications. All these issues translate into inverse problems where deemed similar spectra may correspond to very diverse solutions. This gives raise to undetermination and ill-posed problems.

Methods for model inversion and parameter retrieval can be roughly separated in three main families: statistical, physical and hybrid methods [9]. *Statistical inversion* predicts a biogeophysical parameter of interest using a training dataset of input-output data pairs coming from concurrent measurements of the parameter of interest (e.g. leaf area index -LAI-) and the corresponding satellite observations (e.g. radiances or reflectances). Statistical methods typically outperform other approaches, but ground truth measurements involving a terrestrial campaign are necessary. On the contrary, *physical inversion* reverses RTMs by searching for similar spectra in look-up-tables (LUTs), and assigning the parameter state corresponding to the most similar observed spectrum. This requires selecting an appropriate cost function, and generating a rich, representative LUT from the RTM. The use of RTMs to generate data sets is a common practice, and especially convenient because acquisition campaigns are very costly in terms of time, money, and human resources, and usually limited in terms of parameter combinations. Finally, *hybrid inversion* exploits the input–output data generated by RTM simulations and train statistical regression models to invert the RTM model. Hybrid models combine the flexibility and scalability of machine learning while respecting the physics encoded in the RTMs. Currently, kernel machines in general [10], and Bayesian non-parametric approaches such as Gaussian Process (GP) regression [11] in particular, are among the preferred regression models [12,13].

These GP models have been implemented in Earth observation operational chains for the derivation of biophysical variables at global scale, such as LAI through a hybrid approach. In addition, multitemporal LAI retrievals were derived with similar methodology at local scale taking the advantage of remote sensing data at decametric spatial resolutions and short revisit time, such as Sentinel-2 data [14,15]. These features allow spatial and temporal interpretation of GP estimates and their associated uncertainties at field level which can be related with remote sensing artifacts (e.g. clouds) and crop heterogeneity (e.g. crop damages).

While hybrid inversion is practical when no *in situ* data is available, intuitively it makes sense to let predictions be guided by actual measurements whenever they are present. Likewise, when only very few real *in situ* measurements are available, it is sensible to incorporate simulated data from RTMs to properly ground the models. This is a novel approach considered in this work, which extends the hybrid inversion by proposing a statistical method that performs nonlinear and nonparametric inversion blending both real and simulated data. The so-called joint GP (JGP) essentially learns how to trade off noise variance in the real and simulated data [16].

A second topic covered in this work follows an alternative pathway to *learn* latent functions that generated the observations using GP models. We introduce a *latent force model* (LFM) for GP modelling [17,18]. The proposed LFM-GP combines the ordinary differential equations of the forward model (through smoothing kernels) and empirical data (from *in situ* campaigns). The LFM presented here performs multi-output structured regression, adapts to the signal characteristics, is able to cope with missing data in the time series, and provides explicit latent functions that allow system analysis and evaluation.

Finally, we deal with the important issue of *emulation* of RTMs, that is *learning* surrogate GP models to approximate costly RTMs. The proposed Automatic Gaussian Process Emulator (AGAPE) methodology combines the interpolation capabilities of Gaussian processes (GPs) with the accurate design of an acquisition function that favours sampling in low density regions and flatness of the interpolation function. AGAPE allows building compact sets to perform efficient inverse modelling while respecting the complex physical rules encoded in RTMs.

All in all, in this paper we will illustrate the use of GPs in standard retrieval applications. In particular, we will introduce GPs to tackle problems of hybrid modeling, extending the naive application of previous works. We formalize a full framework for Earth observation with GPs. The framework incorporates different GPs models, and extend our previous works on including temporal information in GP modeling [19,20], incorporating both simulated and real data [16], advancing in the incorporation of physical rules in the modeling through the generation of kernel functions out of differential equations [17,21], multiple output GPs to assess consistency of the predictions [21], and to learn compact look-up-tables (LUT) and emulators of RTMs using GPs in a Bayesian optimization procedure [22–24]. This work improves the previous survey in [25] with an improved literature review and contextualization, as well as new experimental results on the use of GPs in precision agriculture (see Section 2), new results and application to transfer learning of the joint GP (see Section 3), new results for the latent force model in gap-filling problems originally introduced in [21] (see Section 4), as well as more details, new formulations and experiments for the automatic emulator model, which is now fully automatic and works for multioutput problems, see Section 5.

The remainder of the paper is organized as follows. We first briefly introduce the standard GP for regression in Section 2. Then a Joint Gaussian Process (JGP) is proposed in Section 3 that exploits the regularities between real and simulated data, and provides a simple framework for incorporating physical knowledge into a GP model. We introduce LFMs in Section 4 for vegetation monitoring across time, and then an automatic emulator based on GPs is presented in Section 5. We conclude in Section 6 with some remarks and an outline of future work.

2. Gaussian Process models for inverse modeling

GPs are state-of-the-art tools for regression and function approximation, and have been recently shown to excel in biophysical variable retrieval by following both statistical [12,13] and hybrid approaches [26,19].

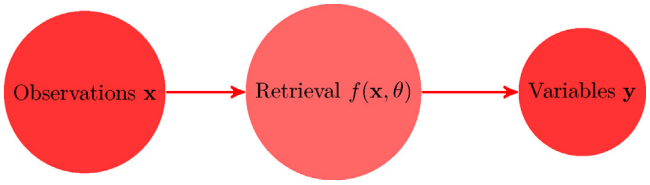

Fig. 2. Statistical inverse modelling. Given a set of observations $\mathbf{x}$ and set of parameters $\boldsymbol{\theta}$, the statistical model $f(\mathbf{x}, \boldsymbol{\theta})$ provides estimations of the variables $\mathbf{y}$. In this case the model performs the inverse function of a physical model, which starting from the variables $\mathbf{y}$ provides the observations $\mathbf{x}$.

2.1. A brief overview of GPs

Let us consider a set of n pairs of observations or measurements, $\mathcal{D}_n := \{\mathbf{x}_i, y_i\}_{i=1}^n$, where $\mathbf{x} \in \mathbb{R}^d$ and $\mathbf{y} \in \mathbb{R}$, which are perturbed by an additive independent noise. The input data pairs ($\mathbf{X} \in \mathbb{R}^{n \times d}$, $\mathbf{y} \in \mathbb{R}^{n \times 1}$) used to fit the inverse machine learning model $f(\cdot)$ come from either *in situ* field campaign data (statistical approach) or simulations by means of an RTM (hybrid approach). We assume the following model,

$$y_i = f(\mathbf{x}_i) + e_i,\ e_i \sim \mathcal{N}(0, \sigma_e^2), \tag{1}$$

where $f(\mathbf{x})$ is an unknown latent function, $\mathbf{x} \in \mathbb{R}^d$, and σ_e^2 stands for the noise variance. Defining $\mathbf{y} = [y_1, \ldots, y_n]^\top$ and $\mathbf{f} = [f(\mathbf{x}_1), \ldots, f(\mathbf{x}_n)]^\top$, the conditional distribution of $\mathbf{y}$ given $\mathbf{f}$ becomes $p(\mathbf{y}|\mathbf{f}) = \mathcal{N}(\mathbf{f}, \sigma_e^2\mathbf{I})$, where $\mathbf{I}$ is the $n \times n$ identity matrix. Now, in the GP approach, we assume that $\mathbf{f}$ follows a n-dimensional Gaussian distribution $\mathbf{f} \sim \mathcal{N}(\mathbf{0}, \mathbf{K})$ [27] (Fig. 2).

The covariance matrix $\mathbf{K}$ of this distribution is determined by a kernel function with entries $\mathbf{K}_{ij} = k(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/(2\sigma^2))$, encoding similarity between the input points [11]. The intuition here is the following: the more similar input i and j are, according to some metric, the more correlated output i and j ought to be. Thus, the marginal distribution of $\mathbf{y}$ can be written as

$$p(\mathbf{y}) = \int p(\mathbf{y}|\mathbf{f})p(\mathbf{f})d\mathbf{f} = \mathcal{N}(\mathbf{0}, \boldsymbol{C}_n),$$

where $\boldsymbol{C} = \mathbf{K} + \sigma_e^2\mathbf{I}$. Now, what we are really interested in is predicting a new output y_*, given an input $\mathbf{x}_*$. The GP framework handles this by constructing a joint distribution over the training and test points,

$$\begin{bmatrix} \mathbf{y} \\ y_* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \boldsymbol{C} & \mathbf{k}_*^\top \\ \mathbf{k}_* & c_* \end{bmatrix}\right),$$

where $\mathbf{k}_* = [k(\mathbf{x}_*, \mathbf{x}_1), \ldots, k(\mathbf{x}_*, \mathbf{x}_n)]^\top$ is an $n \times 1$ vector and $c_* = k(\mathbf{x}_*, \mathbf{x}_*) + \sigma_e^2$. Then, using standard GP manipulations, we can find the distribution over y_* conditioned on the training data, which is a normal distribution with predictive mean and variance given by

$$\begin{aligned} \mu_{\mathrm{GP}}(\mathbf{x}_*) &= \mathbf{k}_*^\top(\mathbf{K} + \sigma_e^2\mathbf{I}_n)^{-1}\mathbf{y}, \\ \sigma_{\mathrm{GP}}^2(\mathbf{x}_*) &= c_* - \mathbf{k}_*^\top(\mathbf{K} + \sigma_e^2\mathbf{I}_n)^{-1}\mathbf{k}_*. \end{aligned} \tag{2}$$

Thus, GPs yield not only predictions μ_{GP^*} for test data, but also the so-called "error-bars", σ_{GP^*}, assessing the uncertainty of the mean prediction. The hyperparameters $\boldsymbol{\theta} = [\sigma, \sigma_e]$ to be tuned in the GP determine the width of the squared exponential kernel function and the noise on the observations. This can be done by marginal likelihood maximization or simple grid search, attempting to minimize the squared prediction errors. In the next section we describe some practical cases regarding the use of GPs both μ_{GP^*} and σ_{GP^*} in EO.

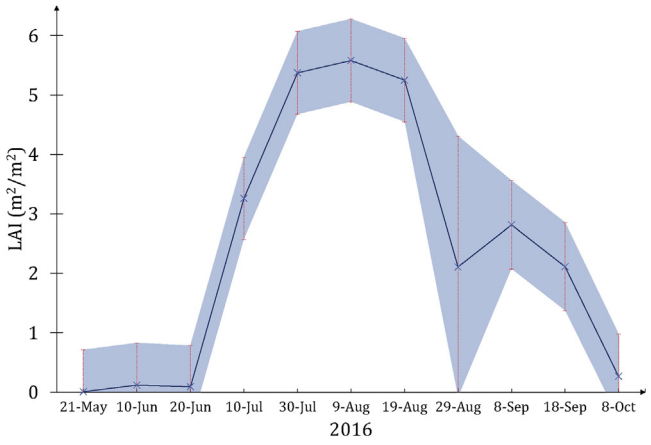

Fig. 3. Temporal evolution of GPR LAI estimates and associated uncertainty (shaded space) over a rice pixel.

2.2. GPs for model inversion in precision agriculture

In this section, we show examples of the GP regression (GPR) model utility in real world applications related to precision farming/agriculture from remote sensing data. In this case, GPR was used for inverting the PROSAIL radiative transfer model (thus following a hybrid approach). PROSAIL simulates leaf reflectance for the optical spectrum, from 400 to 2500 nm with a 1 nm spectral resolution, as a function of biochemistry and structure of the canopy, its leaves, the background soil reflectance and the system geometry. The leaf and canopy variables as well as the soil brightness parameter, were generated following a PROSAIL site-specific parameterization to constrain the model to Mediterranean rice areas [19]. Firstly, PROSAIL was run in forward mode in order to build a database composed of pairs of simulated Sentinel-2 spectra and associated LAI values. A total number of 2000 simulations were computed in such a way the obtained spectra and LAI values covered the expected season of rice crops as well as their management (agricultural practice). Then, the database (often called look-up table) was used for training the GPR model, which was then used for estimating LAI using real Sentinel-2 imagery. Hence, every time a Sentinel-2 image was available, the corresponding LAI map was derived. This procedure was conducted between mid-May until early-October thus completely covering the rice season. As result we derived 11 Sentinel-2 LAI maps.

Fig. 3 shows the LAI evolution over a rice field which is in accordance with rice plant evolution. It is worth mentioning that an unexpected drop was detected on August 29th. LAI decreased too much on this date: a LAI decrease about 3 in a 10-day period does not correctly characterize the typical rice LAI behaviour. Moreover, if we observe the temporal evolution of the GP predictive variance, σ_{GP}, values remain virtually constant at a value about 0.8. However, σ_{GP} dramatically increases (up to 3) on the date that the unexpected drop was observed. Fig. 4 provides a Sentinel-2 a map for the predictive variance on August 29th where undetected clouds presented very high values as compared with cloud free rice fields. The lower confidence (higher prediction uncertainty) is associated with spectra non-represented in the PROSAIL training database. Therefore, non-vegetated surfaces, such as clouds, present higher prediction uncertainty (lower confidence). This assessment of σ_{GP} can be useful to properly weight estimates with low confidence when used by crop modelers, and also to improve cloud masks.

The spatio-temporal detail of the derived maps due to the use of Sentinel-2 data (10 m spatial and 10-day temporal resolution), allows intra-field and multi-temporal analysis useful for crop assessment [14]. These features allow the identification of

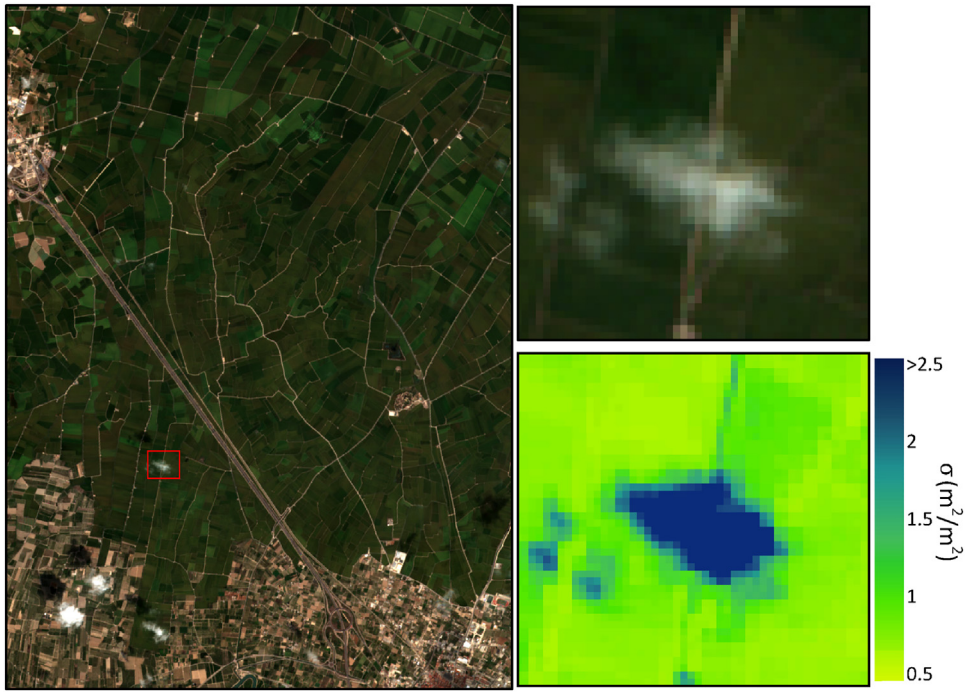

Fig. 4. Sentinel-2 RGB composite over rice fields near Valencia (Spain) on August, 29th 2016 and the corresponding LAI uncertainty over an undetected cloud.

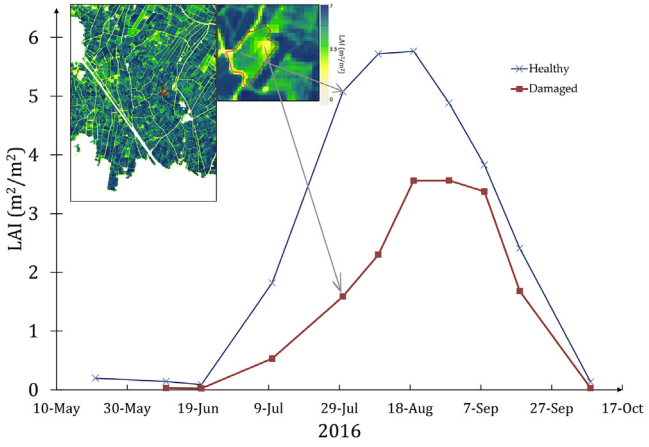

Fig. 5. Temporal evolution of GPR LAI estimates and Sentinel-2 LAI map on July, 29 2016. Red line corresponds to a damaged pixel while the blue one corresponds to a healthy pixel within the same field. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

significant different values within the same rice field. Intra-field LAI differences are mainly due to the heterogeneity of the field related with non-homogeneous agro-practices. The retrieved high-resolution LAI estimates can be used to continuously monitor the cropping season and to detect crop growth anomaly linked with potential crop damage. In particular, Fig. 5 exhibits the temporal evolution of two pixels within the same rice field. The blue line corresponds to the LAI evolution of a healthy pixel and the red one describes the temporal behaviour of a pixel located in the same rice field but affected by a rice disease. According to the temporal profiles, the anomalous LAI behaviour started on the beginning of the season impacting in the LAI values mainly in the rice development stage. This information was corroborated by *in situ* observations. Overall, this kind of analysis and assessment can be used to early derive anomalies maps related with crop damages which could be used by farmers in order to apply agro-practices for mitigating yield loss.

3. Forward and inverse joint GP models

3.1. Notation and formulation

Let us now assume that the previous dataset $\mathcal{D}_n$ is formed by two disjoint sets: one set of r real data pairs, $\mathcal{D}_r = \{(\mathbf{x}_i, y_i)\}_{i=1}^{r}$, and one set of s RTM-simulated pairs $\mathcal{D}_s = \{(y_j, \mathbf{x}_j)\}_{j=r+1}^{n}$, so that $n = r + s$ and $\mathcal{D}_n = \mathcal{D}_r \cup \mathcal{D}_s$. In matrix form, we have $\mathbf{X}_r \in \mathbb{R}^{r\times d}$, $\mathbf{y}_r \in \mathbb{R}^{r\times 1}$, $\mathbf{X}_s \in \mathbb{R}^{s\times d}$ and $\mathbf{y}_s \in \mathbb{R}^{s\times 1}$, containing all the inputs and outputs of $\mathcal{D}_r$ and $\mathcal{D}_s$, respectively. Finally, the $n \times 1$ vector $\mathbf{y}$ contains all the n outputs, sorted with the real data first, followed by the simulated data. Now, we define a different model where the observation noise depends on the origin of the data: σ_e^2 for real observations ($\mathbf{x}_i \in \mathcal{D}_r$) or σ_e^2/γ for RTM simulations ($\mathbf{x}_i \in \mathcal{D}_s$), where the parameter $\gamma > 0$ accounts for the importance of the two sources of information relative to each other.

The resulting distribution of $\mathbf{y}$ given $\mathbf{f}$ is only slightly different from that of the regular GP, namely $p(\mathbf{y}|\mathbf{f}) = \mathcal{N}(\mathbf{f}, \sigma_e^2\mathbf{V})$ where $\mathbf{V}$ is an $n \times n$ diagonal matrix in which the first r diagonal elements are equal to 1 and the remaining s are equal to γ^{-1}: $\mathbf{V} = \text{diag}(1, \ldots, 1, \gamma^{-1}, \ldots, \gamma^{-1})$. The predictive mean and variance of a test output y_*, conditioned on the training data, then becomes

$$\begin{aligned} \mu_{\text{JGP}}(\mathbf{x}_*) &= \mathbf{k}_*^\top (\mathbf{K} + \sigma_e^2\mathbf{V})^{-1}\mathbf{y}, \\ \sigma^2_{\text{JGP}}(\mathbf{x}_*) &= c_* - \mathbf{k}_*^\top (\mathbf{K} + \sigma_e^2\mathbf{V})^{-1}\mathbf{k}_*. \end{aligned} \tag{3}$$

Note that when $\gamma = 1$ the standard GP formulation is obtained. Otherwise γ acts as an extra regularization term accounting for the relative importance of the real and the simulated data points. Selection of the hyperparameters of the JGP, $\boldsymbol{\theta} = [\sigma, \sigma_n, \gamma]$, is central to the effectiveness of the model, since what we are really interested is in performing predictions on the real data. We therefore maximize the *pseudo*-likelihood [11] of the *real* data only:

$$L(\mathbf{X}, \mathbf{y}, \boldsymbol{\theta}) = \sum_{i=1}^{r} \log p(y_i|\mathbf{X}_{\setminus i}, \mathbf{y}_{\setminus i}, \boldsymbol{\theta}), \tag{4}$$

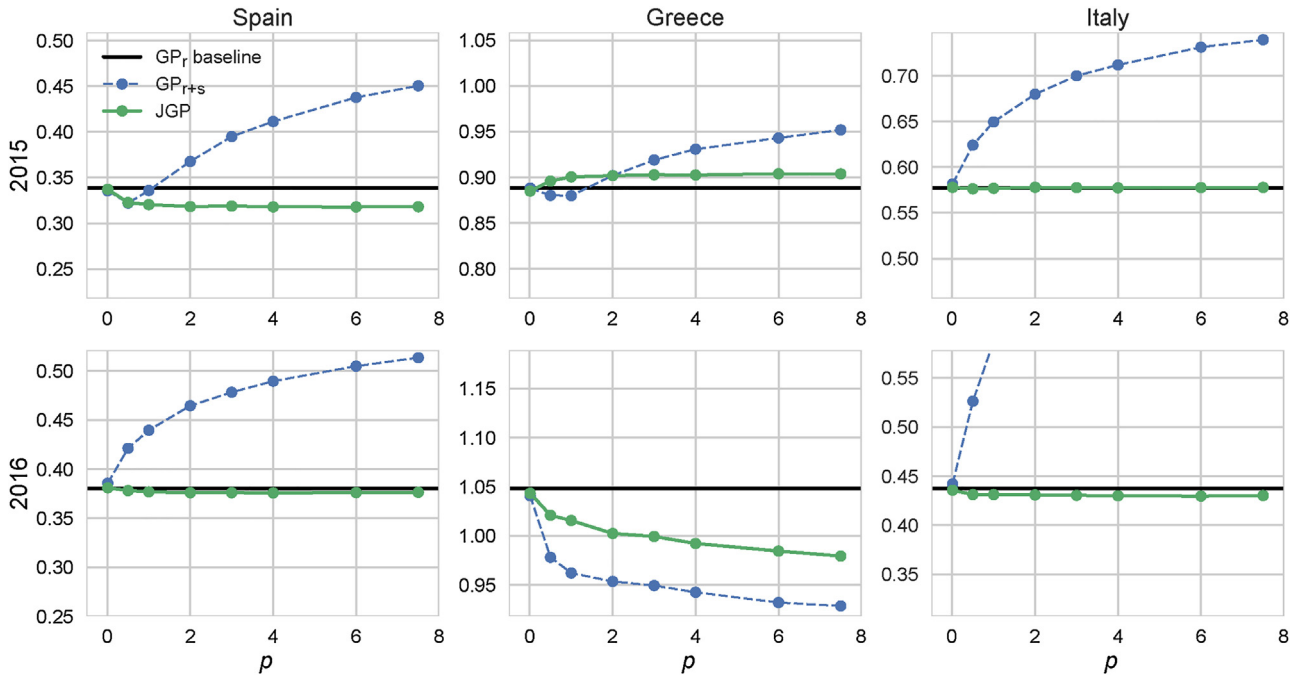

Fig. 6. Performance comparison (RMSE) for different ways of including simulated data. $p = s/r$ is the ratio between simulated (s) and real (r) data. The JGP and the regular GP, trained on a dataset of real and simulated data pooled together (i.e., the GP$_{r+s}$), are compared to the base line of the GP trained exclusively on real data. RMSE is shown for the different sites, campaign dates and simulated-to-real data ratios. As the scale is constant over the plots for better comparison, it was omitted how the GP$_{r+s}$ RMSE monotonically increases and reaches 0.85 for $p = 8$ from the plot in Italy 2016.

where we sum over the log-likelihood of each *real* data point given the remaining training data. The sub-index $\backslash i$ represents the remaining training data. The log-likelihood of a single point i given the remaining data is

$$\log p(y_i|\mathbf{X}_{\backslash i}, \mathbf{y}_{\backslash i}, \boldsymbol{\theta}) = -\frac{1}{2}\log 2\pi\sigma_i^2 - \frac{(y_i - \mu_i)^2}{\sigma_i^2},$$

where μ_i and σ_i^2 are computed using (3) with all $r+s$ datapoints except the ith. By optimizing hyperparameters in this way, γ becomes a measure of how useful the simulated data is in predicting the real data.

3.2. Experimental results

We are concerned about the prediction of leaf area index (LAI) parameter from space, a parameter that characterizes plant canopies and is roughly defined as the total needle surface area per unit ground area. Non-destructive real LAI data were acquired over Elementary Sampling Units (ESUs) within rice fields in Spain, Italy and Greece during field campaigns in 2015 and 2016, i.e., 6 datasets. The temporal frequency of the campaigns was approximately 10 days starting from the very beginning of rice emergence (early-June) up to the maximum rice green LAI development (mid-August). LAI measurements were acquired using a dedicated smartphone app (PocketLAI), which uses both the smartphone's accelerometer and camera to acquire images at 57.5° below the canopy and computes LAI through an internal segmentation algorithm [26]. The center of the ESU was geo-located for later matching and association of the mean LAI estimate with the corresponding satellite spectra. We used Landsat 8 surface reflectance data over each area corresponding to the dates of measurements' acquisition. The resulting datasets contain a number of *in situ* measurements in the range of 70–300 depending on the country and year. On the other hand, three simulated datasets of $s = 2000$ pairs of Landsat 8 spectra and LAI, with characteristics corresponding to the relevant rice area, were obtained running the PROSAIL RTM in forward mode following the similar procedure described in Section 2.2, but in this case we simulated Landsat-8 spectra instead of Sentinel-2.

Two types of experiments were conducted. In the first one, we investigate, for each of the 6 datasets, how including simulated data might improve predictions in a regular 10-fold cross-validation scheme. In the second experiment, we explore how simulated data might help prediction of LAI in one site, given that one only has access to data from a different site. This is a quite habitual situation, often referred to as *domain adaptation* or *transfer learning* in machine learning [9,28]; or simply as *model transferability* in remote sensing applications. In this case, we use datasets from 2016. We shall refer to these experiment types as *same-site* and *cross-site* respectively.

3.2.1. Same-site experiments

We assessed the performance of the JGP for different amounts of simulated data. We compare to a regular GP model (see Section 2) which only has access to real data, which we will refer to as GP$_r$, and a different regular GP model which has access to a training set of both simulated and real data. This, more naive approach of including simulated data, which does not distinguish between data sources, is referred to as GP$_{r+s}$.

Fig. 6 shows the effect of the ratio between simulated and real data points, $p = s/r$, on the RMSE evaluated using 10-fold crossvalidation. The JGP behaves in different ways on different datasets. There are cases where a γ value close to 0 is fitted, the simulated data is largely ignored, and it follows the GP$_r$ baseline. For the datasets where this does not happen, we see that a $p \sim 1$ is enough to produce an effect. This is worth noting as the inversion of the kernel matrix, needed to train the JGP, scales in time complexity with the number of samples cubed, $\mathcal{O}(n^3) = \mathcal{O}((r+s)^3)$.

In the case of Greece 2015, an average increase in RMSE is observed which, percentage-wise is around ∼1%. In Spain 2015 and Greece 2016, a decrease in RMSE of around ∼5% can be observed. Interestingly, we see that the naive inclusion of simulated data (the

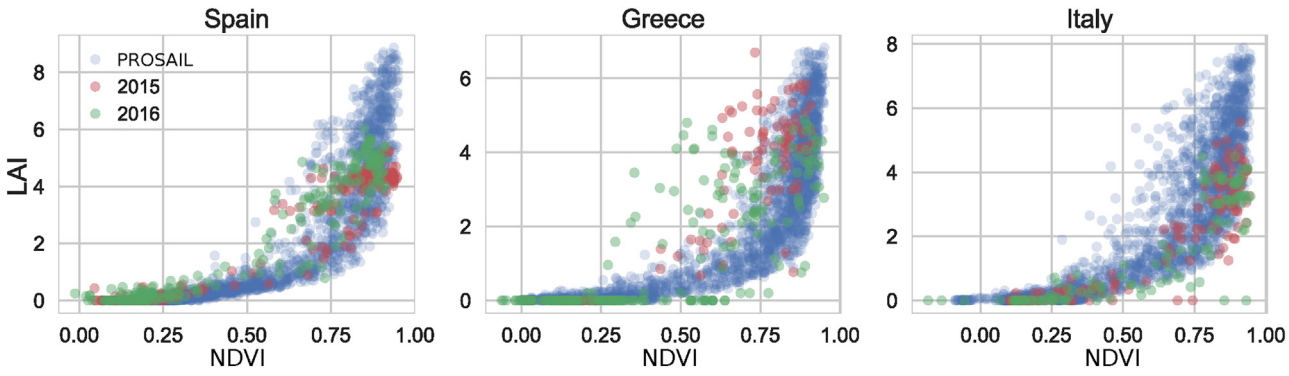

Fig. 7. Scatterplots in the NDVI-LAI representation space of the real and RTM-simulated data for all sites and acquisition campaigns (2015, 2016).

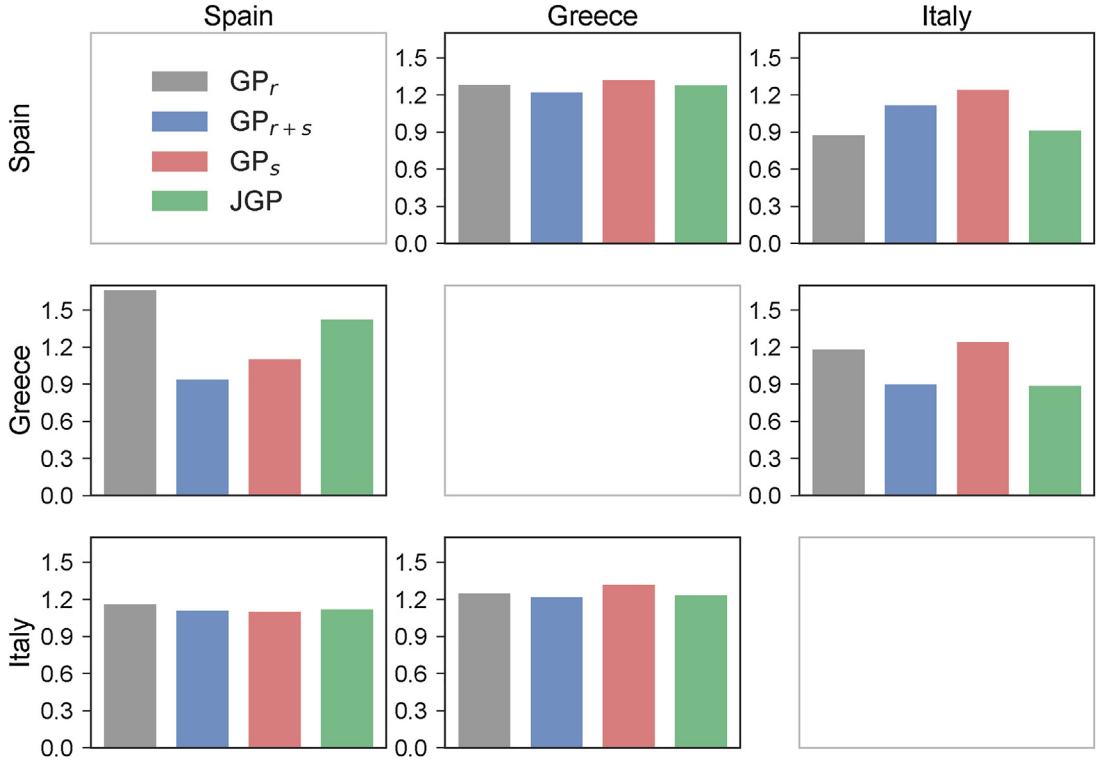

Fig. 8. Performance (RMSE) of the different approaches to cross-site learning, where rows and columns indicate the source and target datasets respectively.

GP_{r+s} scheme) generally leads to an increase in error, except for the case of Greece 2016. This hints towards the fact that the GP_{r+s} can perform better than the JGP approach when simulated data is of high quality, as it is less conservative. Overall, the JGP appears to be a safe way to include simulated data, at worst increasing RMSE by ∼1% in one dataset, and at best decreasing it by ∼5%. This is made possible by the hyperparameter fitting procedure which attempts to assess whether the simulated data is useful or confusing for prediction.

3.2.2. Cross-site experiments

We turn now to the question of whether a regression model, trained on data from one particular site, can be useful for prediction at a different one. The experiment is not incidental, but of practical concern and implications, as it is expensive to collect data, limiting the amount of real data available. It comes down to a question of how similar the distributions over the output variable LAI, given the input variables, are across sites.

In order to visualize the 6 datasets we plot them in the NDVI-LAI representation space in Fig. 7, along with their simulated counterparts. The PROSAIL simulated data exhibits the well known exponential relation between NDVI and LAI, and a similar trend is visible in the real datasets. The data distribution that stands the most out, both with respect to simulated and other real datasets, is that pertaining to the Greek site. This is, as we shall see, what determines how well a model trained on that data will perform on other datasets.

We consider the following strategies for predicting on a *target* site, having only data from a *source* site. One might train a GP with the available *real* source data, denoted GP_r in this section. Otherwise, we might have some knowledge about the target site that we can use to create an RTM-simulated database. This strategy of training only on simulated data will be referred to as GP_s here. Finally, one can try to combine the real data from the source site, with simulated target site data. This could be attempted through training a normal GP on the union of these two datasets, i.e., a GP_{r+s} model, or through the JGP.

Fig. 8 shows how these methods compare, where row names indicate the train/source site and the column denotes the test/target site. Thus, the RMSE of the GP_s is constant across

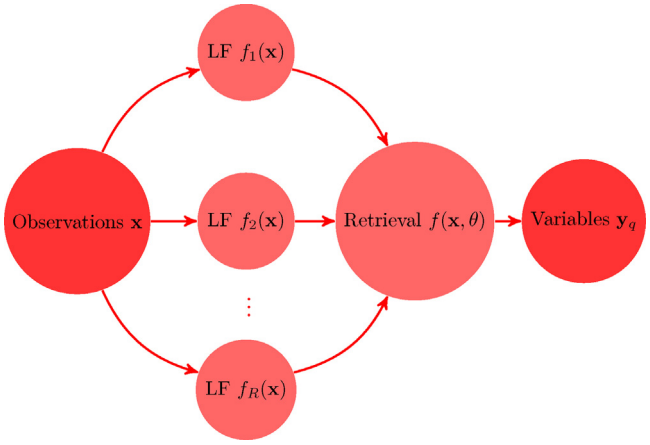

Fig. 9. Inverse modeling with latent forces. In this case, the statistical inversion model does not depend directly on the inputs, but on a set of *a priori* unknown independent latent functions that describe the underlying physical model.

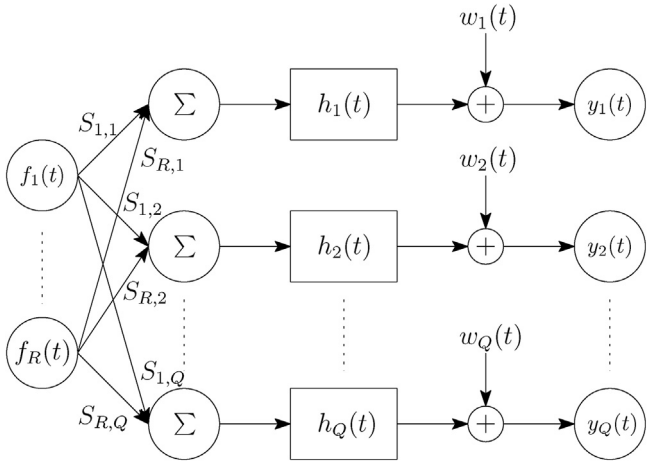

Fig. 10. GP-LFM relating inputs (latent forces) and outputs (observations).

columns. We see how the fact that the simulated data distribution poorly matches that of the real data in Greece and Italy (see Fig. 7) is reflected in the RMSE for the GP_s approach in the two rightmost columns of Fig. 8. Conversely, we see from the second row how inclusion of simulated data in some form is very useful for predicting in other sites when having access only to the real dataset from Greece. We note here, about the JGP, that it is the only method that consistently performs better than the simple GP_r strategy. In conclusion, the JGP can be said to be a safe approach to include simulated data for non-linear regression.

4. Inverse modelling with latent force models

In this second case study, we are interested in inverse modelling from real *in situ* data, *learning* not only an accurate retrieval model but also the physical mechanism that generated the input-output observed relations without even accessing any RTM, see Fig. 9. Here, we assume that our observations correspond simply to the temporal variable, $\mathbf{x} \sim t$, so the latent functions are defined in the time domain, $f_r(t)$. Nevertheless, extension to multidimensional objects such as radiances is straightforward by using different kernels. Notationally, let us consider a multi-output scenario with Q correlated observed time series, $y_q(t)$ for $1 \le q \le Q$, and let us assume that we have n samples available for each of these signals, taken at sampling points t_i, s.t. $y_q[i] = y_q(t_i)$ for $1 \le i \le n$. This is the *training set*, which is composed of an input vector, $\mathbf{t} = [t_1, \ldots, t_n]^\top$, and an output matrix, $\mathbf{Y} = [\mathbf{y}_1, \ldots, \mathbf{y}_Q]$ with $\mathbf{y}_q = [y_q[1], \ldots, y_q[n]]^\top$. We aim to build a GP model for the Q outputs that can be used to perform inference on the *test set*: $\tilde{\mathbf{t}} = [\tilde{t}_1, \ldots, \tilde{t}_m]^\top$ and $\tilde{\mathbf{Y}} = [\tilde{\mathbf{y}}_1, \ldots, \tilde{\mathbf{y}}_Q]$ with $\tilde{\mathbf{y}}_q = [\tilde{y}_q[1], \ldots, \tilde{y}_q[m]]^\top$ and $\tilde{y}_q[m'] = y_q(\tilde{t}_{m'})$ for test inputs at $t_{m'}$.

4.1. Formulation

Let us assume that a set of R independent latent functions (LFs), $f_r(t)$ with $1 \le r \le R$, are responsible for the observed correlation between the outputs. Then, the cross-correlation between the outputs arises naturally as a result of the coupling between the set of independent LFs, instead of being imposed directly on the set of outputs. Let us define the form of these latent functions and the coupling mechanism between them. In this work, we model the LFs as zero-mean Gaussian processes (GPs), and the coupling system emerges through a linear convolution operator described by an *impulse response*, $h_q(t)$, as follows:

$$y_{r,q}(t) = \mathrm{L}_q[t]\{f_r(t)\} = f_r(t) * h_q(t) = \int_0^t f_r(\tau) h_q(t-\tau)\mathrm{d}\tau, \tag{5}$$

where $\mathrm{L}_q[t]\{f_r(t)\}$ indicates the linear operator associated to the linear convolution of the latent force $f_r(t)$ with the *smoothing kernel* $h_q(t)$. As shown in Fig. 10, the outputs are finally obtained as a linear weighted combination of these pseudo-outputs plus an additive white Gaussian noise (AWGN) term:

$$y_q(t) = \sum_{r=1}^{R} S_{r,q} y_{r,q}(t) + w_q(t), \tag{6}$$

where $S_{r,q}$ represents the coupling strength between the rth LF and the qth output, and $w_q(t) \sim \mathcal{N}(0, \eta_q^2)$ is the AWGN term. In practice, we consider only the squared exponential auto-covariance function for the LFs, $k_{f_r f_r}(t'-t) \propto \exp(-\frac{(t'-t)^2}{2\ell_r^2})$, where the hyperparameter ℓ_r controls the length-scale of the process.

The smoothing kernel encodes our knowledge about the linear system (that relates the unobserved LFs and the outputs), and can be based on basic physical principles of the system at hand (as in [17,18]) or selected arbitrarily (as in [29,30]). In this paper, we consider the Gaussian smoothing kernel, $h_q(t) \propto \exp(-\frac{t^2}{2\nu_q^2})$. Since the LFs are zero-mean GPs, the noise is zero-mean and Gaussian, and all the operators involved are linear, the joint LFs-output process is also a GP. Therefore, the mean function of the qth output is $\mu_{y_q}(t) = 0$, whereas the cross-covariance function between two outputs is

$$k_{y_p y_q}(t,t') = \sum_{r=1}^{R} S_{r,p} S_{r,q} \mathrm{L}_p[t] \left\{ \mathrm{L}_q[t']\{k_{f_r f_r}(t,t')\} \right\} + \eta_q^2 \delta[p-q]\delta[t'-t], \tag{7}$$

where the term $\mathrm{L}_p[t] \left\{ \mathrm{L}_q[t']\{k_{f_r f_r}(t,t')\} \right\}$ denotes the application of the convolutional operator twice to the autocorrelation function of the LFs, which results in the following double integral:

$$\mathrm{L}_p[t] \left\{ \mathrm{L}_q[t']\{k_{f_r f_r}(t,t')\} \right\} = \int_0^t \int_0^{t'} h_p(t-\tau) h_q(t'-\tau') k_{f_r f_r}(\tau,\tau')\mathrm{d}\tau'\mathrm{d}\tau.$$

Finally, the cross-correlation between the LFs and the outputs readily gives $k_{f_r y_q}(t,t') = S_{r,q}\mathrm{L}_q[t']\{k_{f_r f_r}(t,t')\}$, which involves a single one-dimensional integral. All integrals can be solved analytically when both the LFs and the smoothing kernel have a Gaussian shape.

Learning hyperparameters through marginal log-likelihood maximization is very challenging because of its complicated dependence on the hyperparameters $\boldsymbol{\theta} = [\nu_q, l_r, \sigma, \sigma_n, \eta_q]$. We propose to

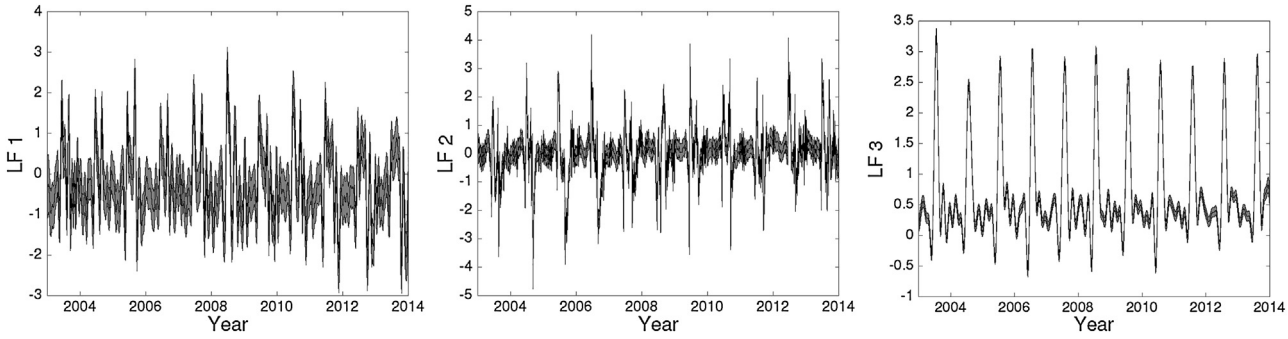

Fig. 11. Inferred LFs (black line) and uncertainty measured by ±2 standard deviations about the mean predicted value (grey shaded area), obtained from the full LAI and fAPAR dataset from Spain and Italy with $R = 3$.

Table 1
Absolute and normalised MSE using the full dataset for $R \in \{1, 2, 3\}$ LFs.

	MSE (NMSE)			
R	LAI (ES)	LAI (IT)	fAPAR (ES)	fAPAR (IT)
1	0.1139 (2.08%)	0.2422 (5.97%)	0.0080 (4.02%)	0.0046 (2.49%)
2	0.0548 (1.00%)	0.1636 (4.03%)	0.0013 (0.67%)	0.0039 (2.11%)
3	0.0012 (0.02%)	0.1657 (4.09%)	0.0002 (0.10%)	0.0025 (1.38%)

solve the problem through a stochastic gradient descent technique, the scaled conjugate gradient [31]. Once the hyperparameters $\boldsymbol{\theta}$ of the model have been learned, inference proceeds by applying standard GP regression formulas [11] (cf. Section 2). Now, since the conditional PDF is Gaussian, the minimum mean squared error (MMSE) prediction is simply given by the conditional mean:

$$\hat{\mathbf{y}} = \boldsymbol{\mu}_{\bar{\mathbf{y}}|\mathbf{y}} = \mathbf{K}_{\bar{\mathbf{y}}\mathbf{y}}\mathbf{K}_{\mathbf{y}\mathbf{y}}^{-1}\mathbf{y}, \tag{8}$$

where $\hat{\mathbf{y}} = [\hat{\mathbf{y}}_1^\top, \ldots, \hat{\mathbf{y}}_Q^\top]$ is the vectorized version of the inferred outputs, which can be expressed in matrix form as $\hat{\mathbf{Y}} = [\hat{\mathbf{y}}_1, \ldots, \hat{\mathbf{y}}_Q]$ with $\hat{\mathbf{y}}_q = [\hat{y}_q[1], \ldots, \hat{y}_q[m]]^\top$ and $\hat{y}_q[m'] = \hat{y}_q(\tilde{t}'_m)$.

4.2. Experimental results

We are concerned about multiple time series of two (related) biophysical parameters, LAI and fraction of Absorbed Photosynthetically Active Radiation (fAPAR), in the locations of the experiments described in Section 3.2. We focus on a set of representative rice pixels of each area, thus allowing us to observe the inter-annual variability of rice from 2003 to 2013 at a coarse spatial resolution (2 km), which is useful for regional vegetation modelling.

4.2.1. Learning the LAI-fAPAR relationships

In this section, we explore the LAI vs. fAPAR relationship, which is usually modeled using the following exponential model, as largely observed in the literature [32]:

$$\text{fAPAR} = 1 - \exp(\alpha \times \text{LAI}). \tag{9}$$

In order to determine whether the GP-LFM is able to capture this well-known relationship, we train the model using the multi-output time series composed of all the available LAI and fAPAR data from the MODIS sensor for Spain and Italy from the beginning of 2003 until the end of 2013 (i.e., $N = 506$ and the number of outputs is $Q = 4$). After removing truly missing data (marked with negative values in the original time series) this results in 2006 training samples (506 samples for each of the time series from Spain and 497 samples for the Italian ones). We have experimented with a variable number of latent forces, $R \in \{1, 2, 3\}$. Table 1 shows the quantitative results in terms of mean squared error (MSE),

$$\text{MSE}_q = \frac{1}{N_q}\sum_{n=0}^{N_q-1}(y_q[n] - \hat{y}_q[n])^2, \tag{10}$$

and normalised MSE,

$$\text{NMSE}_q(\%) = \frac{\text{MSE}_q}{\frac{1}{N_q}\sum_{n=0}^{N_q-1} y_q^2[n]} \times 100, \tag{11}$$

where $y_q[n]$ denotes the true value of the nth sample from the qth time series, $\hat{y}_q[n]$ is the value predicted by the model, $N_q \leq N$ is the number of samples available for the qth time series ($N_1 = N_2 = 506$ and $N_3 = N_4 = 497$, as mentioned above) and $q = 1, \ldots, Q = 4$.

Noting the substantial decrease in MSE, in the rest of this section we set $R = 3$. Fig. 11 shows the three LFs inferred and Fig. 12 displays the four output time series. In Fig. 12 we can see the good modeling accuracy for all the time series (as evidenced by the low NMSE values displayed in Table 1), whereas in Fig. 11 we see that LF3 captures the smooth and periodic component of the output and the other two LFs focus on the noisier part (albeit with an important residual periodical component). Finally, Fig. 13 displays the LAI vs. fAPAR scatter plot, obtained from the modeled time series both for Spain and Italy. The shaded area corresponds to the uncertainty (that appears now in both axis, as a consequence of the modeling uncertainty in both LAI and fAPAR), whereas the continuous red line shows the exponential model in (9), that has been fitted to the data by performing a simple least squares regression in the log-domain. Note the good fit in both cases of the expected LAI vs. fAPAR relationship, given by (9) with $\alpha = -0.4047$ and $\alpha = -0.4593$ for Spain and Italy respectively, and the scatter plot obtained from the GP-LFM.

4.2.2. Dealing with missing data

In this second example, we show the ability of the model to recover the missing samples (i.e., to perform gap filling) that typically appear in this kind of datasets. We use again all the LAI data from the MODIS sensor for Spain from the beginning of 2003 until the end of 2013 (i.e., $N_1 = 506$), but only the first half (years 2003–2009) of the LAI data from Italy and the two fAPAR time series (i.e., $N_2 = 275$, $N_3 = 276$ and $N_4 = 275$). In order to show that the GP-LFM is able to capture the underlying dynamics of the multi-output time series with a single LF, we set $R = 1$. The three time series with missing data are displayed in Fig. 14, whereas the single LF (not shown) is very similar to the smooth LF (LF3) in Fig. 11. Note the good fit of the three time series, even though the last four years of data are removed from all of them.

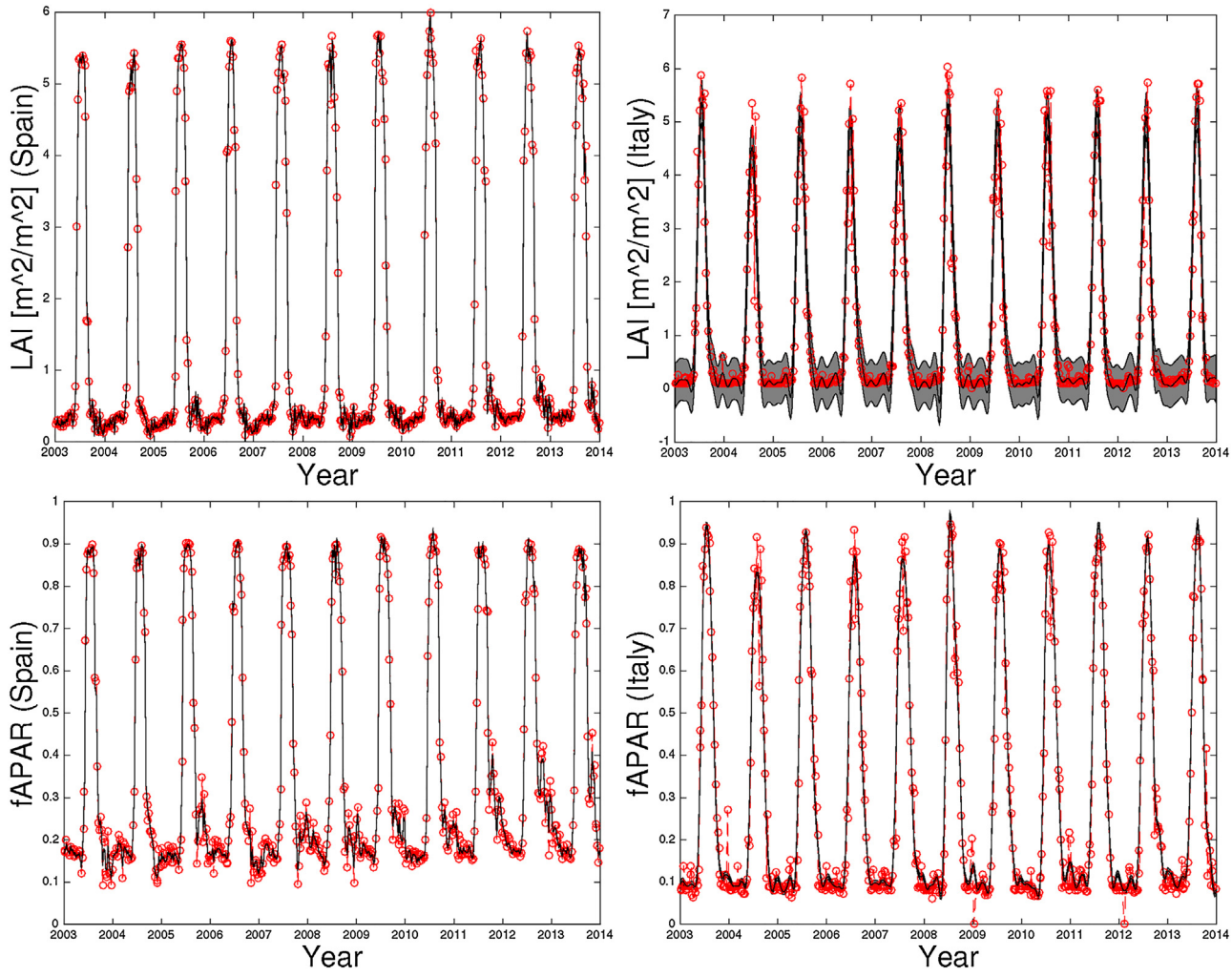

Fig. 12. Training data (red circles), predicted time series (black line) and uncertainty measured by ±2 standard deviations about the mean predicted value (grey shaded area), obtained from the full LAI and fAPAR dataset from Spain and Italy when $R = 3$. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

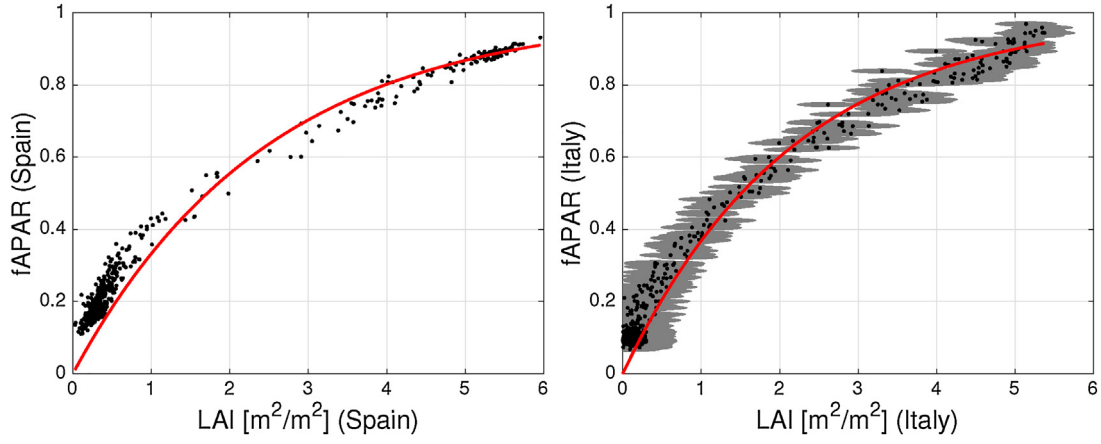

Fig. 13. LAI vs. fAPAR for the data learned using $R = 3$ LFs and all the available data for years 2003–2013.

5. Automatic emulation

Emulation deals with the challenging problem of building statistical models for complex physical RTMs. The emulators are also called *surrogate* or *proxy* models, and try to learn from data the equations encoded in the RTM. Namely, an emulator is a statistical model that tries to reproduce the behavior of a deterministic and often very costly physical model. Emulators built with GPs are gaining popularity in remote sensing and geosciences, since they allow efficient data processing and sensitivity analysis [33,34,13]. Emulators also allow model tractability, as model-data integration, fast inference, analytical Jacobian calculation, and derivation of confi-

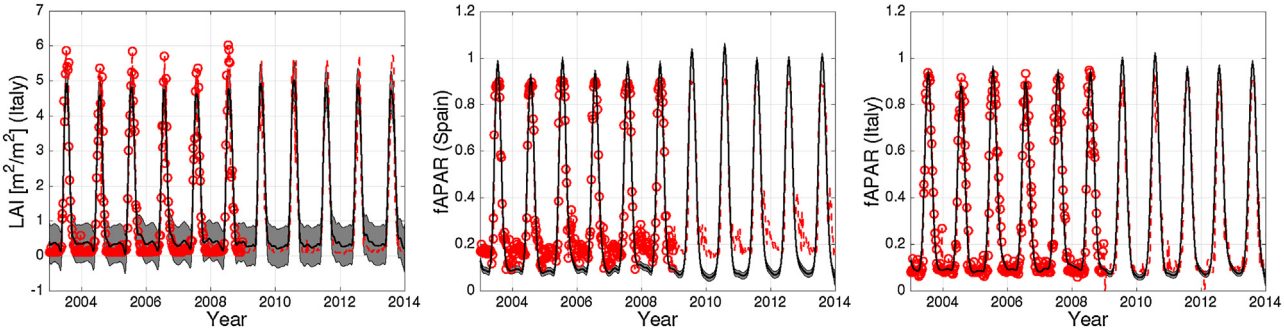

Fig. 14. Gap filling example using a single LF (i.e., $R = 1$). Training used all the LAI data from Spain (years 2003–2013) and the first half (years 2003–2009) of the other three time series: LAI (IT), fAPAR (ES) and fAPAR (IT). The second half constitutes the test set of such time series. Training data (red circles), test data (red dashed line), predicted time series (black line) and uncertainty measured by ±2 standard deviations about the mean predicted value (gray shaded area). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

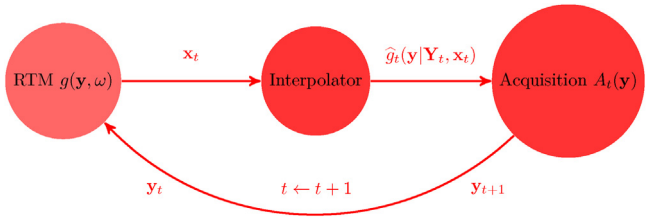

Fig. 15. Scheme of an automatic emulator. The goal of the model is to emulate the RTM as best as possible using a minimum number of runs. This is achieved by an iterative process which starts with a reduced input set of variables $\mathbf{Y}_{t=0}$. Then the RTM provides the corresponding observations $\mathbf{x}_{t=0}$. An interpolator is then fitted, forming part of the acquisition function that is optimized to select new variables to add to the initial input set. The process is iterated until the stop condition is fulfilled.

dence intervals for the estimates and parameters becomes easier and analytical when GPs are used.

Here, we are interested in optimizing emulators such that a minimal number of simulations is run (for a given approximation error). We describe a general framework, called *Automatic Emulation* (AE) technique which is related to Bayesian optimization and active learning techniques. We first define the generic elements of the AE methodology and then describe a specific implementation based on GPs. This yields the Automatic Gaussian Process Emulator (AGAPE) model for automatic emulation and creation of a compact and informative look-up-table.

The goal is to emulate (i.e., interpolate/mimic) a costly function $g(\mathbf{y})$ choosing adequately the nodes, in order to reduce the error in the interpolation with the smallest possible number of evaluation of $g(\mathbf{y})$. Given an input matrix of nodes (used for the interpolation) at the tth iterations, $\mathbf{Y}_t = [\mathbf{y}_1 \ldots \mathbf{y}_{m_t}]$, of dimension $d \times m_t$ (where d is the dimension of each $\mathbf{y}_i$ and m_t is the number of points), we have a vector of outputs, $\mathbf{x}_t = [x_1, \ldots, x_{m_t}]^\top$, where $x_t = g(\mathbf{y}_t)$ is the estimation of the observations (e.g. radiances) at iteration $t \in \mathbb{N}^+$ of the algorithm. Figs. 15 and 16 show two graphical representations of a generic automatic emulator. At each iteration t one performs an *interpolation/regression*, obtaining $\hat{g}_t(\mathbf{y}|\mathbf{Y}_t, \mathbf{x}_t)$, followed by an *optimization* step that updates the acquisition function, $A_t(\mathbf{y})$, updates the set $\mathbf{Y}_{t+1} = [\mathbf{Y}_t, \mathbf{y}_{m_t+1}]$ adding a new node, set $m_t \longleftarrow m_t + 1$ and $t \longleftarrow t+1$. Note that the acquisition function $A_t(\mathbf{y})$ is the core of the automatic emulation method. It plays the role of an oracle, suggesting in which regions it is more convenient to introduce new nodes, used in the next interpolation step. Clearly, the design of $A_t(\mathbf{y})$ is a key point for the success of the automatic emulation. The procedure is repeated until a suitable stopping condition is met, such as a certain maximum number of points is included or a desired precision error ϵ is achieved. Since g is unknown, we could compute $\|\hat{g}_t(\mathbf{y}) - \hat{g}_{t-1}(\mathbf{y})\| \leq \epsilon$.

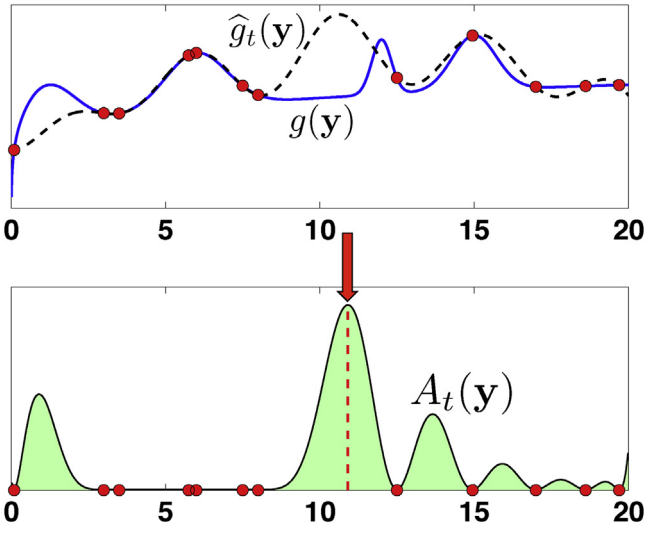

Fig. 16. General sketch of an Automatic Emulation (AE) procedure. The RTM model $g(\mathbf{y})$ (top – solid line), its approximation $\hat{g}_t(\mathbf{y})$ (top – dashed line) and an acquisition function $A_t(\mathbf{y})$. Its maximum suggests where adding a new node/point to the LUT.

5.1. Theoretical formulation

The acquisition function, $A_t(\mathbf{y})$, encodes useful information for proposing new nodes to build the emulator. Namely, the acquisition function $A_t(y)$ suggests where introducing new support points for the next interpolation step. For this reason, the best new possible node, according to the designed acquisition function $A_t(y)$, is exactly the argmax$A_t(y)$. Therefore, at each iteration, a new node is added maximizing $A_t(\mathbf{y})$, i.e.,

$$\mathbf{y}_{m_t+1} = arg\max A_t(\mathbf{y}),$$

and set $\mathbf{Y}_{t+1} = [\mathbf{Y}_t, \mathbf{y}_{m_t+1}]$, $m_{t+1} = m_t + 1$. Observe that the acquisition function $A_t(\mathbf{y})$ must take into account the locations of the current nodes and the geometry of the underlying function g. Hence, we propose to factorize the acquisition function as product of a *geometry* $H_t(\mathbf{y})$ and a *diversity* $D_t(\mathbf{y})$ terms. The distribution of the previous nodes is encoded into the function $D_t(\mathbf{y})$, whereas the geometric information is included in $H_t(\mathbf{y})$. More specifically, we define the acquisition function as

$$A_t(\mathbf{y}) = [H_t(\mathbf{y})]^{\beta_t} D_t(\mathbf{y}), \quad \beta_t \in [0, 1], \tag{12}$$

where $A_t(\mathbf{y}) : \mathcal{Y} \mapsto \mathbb{R}$, and β_t is an increasing function with respect to t, with $\lim_{t\to\infty} \beta_t = 1$ (or $\beta_t = 1$ for $t > t'$). Function $H_t(\mathbf{y})$ captures the geometrical information in g, while function $D_t(\mathbf{y})$ depends on the

distribution of the points in the current vector $\mathbf{Y}_t$. More specifically, $D_t(\mathbf{y})$ presents a greater value around empty areas within $\mathcal{Y}$, whereas $D_t(\mathbf{y})$ will be approximately zero close to the support points and exactly zero at the support points, i.e., $D_t(\mathbf{y}_i)=0$, for $i=1,\ldots,m_t$ and $\forall t \in \mathbb{N}$. Since g is unknown, the function $H_t(\mathbf{y})$ can be only derived from information acquired in advance or by considering the approximation $\hat{g}$. The tempering value, β_t, helps to downweight the likely less informative estimates in the very first iterations. If $\beta_t=0$, we disregard $H_t(\mathbf{y})$ and $A_t(\mathbf{y})=D_t(\mathbf{y})$, whereas, if $\beta_t=1$, we have $A_t(\mathbf{y})=H_t(\mathbf{y})D_t(\mathbf{y})$.

5.2. Specific implementations

An automatic emulation (AE) procedure above described is completely defined by the following elements:

1. the choice of the interpolator providing the approximation $\hat{g}_t(\mathbf{y}|\mathbf{Y}_t,\mathbf{x}_t)$,
2. the choice of the function $D_t(\mathbf{y})$,
3. the choice of the function $H_t(\mathbf{y})$,
4. and the choice of the tempering function β_t.

Furthermore, the stopping condition can be considered as an additional element. For the interpolation, we have to take into account the ability of the interpolator of building the approximation in high dimensional spaces and the differentiability of $\hat{g}_t$ (in the support domain with the exception of the a set of null measure). The conceptual set of elements $\{\hat{g}_t, D_t, H_t, \beta_t\}$ defines an AE method. Different combinations of these elements produces different AE techniques, each one yielding different performance.

5.3. Automatic Gaussian Process Emulator (AGAPE)

We consider a GP technique as interpolator with $\mathbf{y}_t$ as inputs and $\mathbf{x}_t$ as outputs (i.e., reversed with respect to the previous section). In addition, note that interpolation forces to zero the noise standard deviation, i.e., $\sigma_e=0$. Therefore, the AGAPE predictive mean and variance at iteration t for a new point $\mathbf{y}_*$ becomes simply

$$\begin{aligned}\hat{g}_t(\mathbf{y}_*) &= \mathbf{k}_*^\top \mathbf{K}^{-1}\mathbf{x} = \mathbf{k}_*^\top \boldsymbol{\alpha},\\ \sigma^2(\mathbf{y}_*) &= k(\mathbf{y}_*,\mathbf{y}_*) - \mathbf{k}_*^\top \mathbf{K}^{-1}\mathbf{k}_*,\end{aligned}$$

where now $\mathbf{k}_* = [k(\mathbf{y}_*,\mathbf{y}_1),\ldots,k(\mathbf{y}_*,\mathbf{y}_{m_t})]^\top$ contains the similarities between the input point $\mathbf{x}_*$ and the observed ones at iteration t, $\mathbf{K}$ is an $m_t \times m_t$ kernel matrix with entries $\mathbf{K}_{ij} := k(\mathbf{y}_i,\mathbf{y}_j)$, and $\boldsymbol{\alpha} = \mathbf{K}_{nn}^{-1}\mathbf{x}_t$ is the coefficient vector for interpolation. The interpolation for $\mathbf{y}_*$ can be simply expressed as a linear combination of $\hat{g}_t(\mathbf{y}_*) = \mathbf{k}_*^\top \boldsymbol{\alpha} = \sum_{i=1}^{m_t} \alpha_i k(\mathbf{y}_*,\mathbf{y}_i)$. We consider the standard squared exponential kernel function now working with state vectors $\mathbf{y}$, i.e., $k(\mathbf{y},\mathbf{y}') = \exp(-\|\mathbf{y}-\mathbf{y}'\|^2/(2\delta^2))$. The training of the hyperparameter δ of kernel function k can be performed with standard procedure, as cross-validation or marginal likelihood optimization [35].

Note that $\sigma^2(\mathbf{y}_i)=0$ for all $i=1,\ldots,m_t$ and $\sigma^2(\mathbf{y})$ depends on the distance among the support points $\mathbf{y}_t$, and the chosen kernel function k and associated hyper-parameter σ. For this reason, the function $\sigma^2(\mathbf{y})$ is a good candidate to represent the distribution of the $\mathbf{y}_t$'s since it is zero at each $\mathbf{y}_i$, and higher far from the points $\mathbf{y}_i$'s. Moreover, $\sigma^2(\mathbf{y})$ takes into account the information of the GP interpolator. Therefore, we consider as the diversity term

$$D(\mathbf{y}) := \sigma^2(\mathbf{y}),$$

i.e., $D(\mathbf{y})$ is induced by the GP interpolator. As geometric information, we consider enforcing flatness on the interpolation function, and thus aim to minimize the norm of the gradient of the interpolating function $\hat{g}_t$ w.r.t. the input data $\mathbf{y}$, i.e.,

$$H(\mathbf{y}) = \|\nabla_y \hat{g}_t(\mathbf{y}|\mathbf{Y}_t,\mathbf{x}_t)\| = \left\|\sum_{i=1}^{m_t}\alpha_i \nabla_y k(\mathbf{y},\mathbf{y}_i)\right\|.$$

This intuitively makes wavy regions of $\hat{g}_t$ require more support points than flat regions. The gradient vector for the squared exponential kernel $k(\mathbf{y},\mathbf{y}') = \exp(-\|\mathbf{y}-\mathbf{y}'\|^2/(2\delta^2))$ with $\mathbf{y}=[y_1,\ldots,y_d]^\top$, can be computed in closed-form, $\nabla_y k(\mathbf{y},\mathbf{y}') = -\frac{k(\mathbf{y},\mathbf{y}')}{\delta^2}[(y_1-y_1'),\ldots,(y_d-y_d')]^\top$. Therefore, the acquisition function can be readily obtained by defining $\beta_t = 1-\exp(-\gamma t)$, where $\gamma \geq 0$, and $A_t(\mathbf{y}) = [H_t(\mathbf{y})]^{\beta_t} D_t(\mathbf{y})$. We optimized $A_t(\mathbf{x})$ using interacting parallel simulated annealing methods, for the sake of simplicity [35,36]. Indeed, these techniques do not require the knowledge of the gradient of the acquisition function, and thus more sophisticated schemes can be employed. In our experiments, simulated annealing schemes provide good performance, reaching a solution close to the global maximum in few iterations [37,38]. However, as the dimension of the input space grows, the performance becomes worse.

5.4. Multi-output automatic emulation

So far we have considered an RTM model of type $x=g(\mathbf{y})+e$, where $e \sim \mathcal{N}(0,\sigma_e^2)$ (with $\sigma_e=0$ in AGAPE). Let us now denote the following RTM model represented by the equation

$$\mathbf{x} = \mathbf{g}(\mathbf{y}) + \mathbf{e}, \tag{13}$$

with $\mathbf{x} = [x^{(1)},\ldots,x^{(K)}] \in \mathbb{R}^K$ and $\mathbf{e} \sim \mathcal{N}(\mathbf{0},\sigma_e^2\mathbf{I}_K)$ where $\mathbf{I}_K$ is a $K\times K$ identity matrix. Then, we have K outputs for each $\mathbf{y}$, i.e.,

$$\mathbf{g}(\mathbf{y}) = [g^{(1)}(\mathbf{y}),\ldots,g^{(K)}(\mathbf{y})] : \mathcal{Y} \to \mathbb{R}^K. \tag{14}$$

In order to design an automatic emulator of $g(\mathbf{y})$, we have to extend the strategy described above. We consider that at the tth iteration the current matrix of nodes $\mathbf{Y}_t = [\mathbf{y}_1,\ldots,\mathbf{y}_{m_t}]$ is shared by all outputs. Therefore, we will design a multi-output emulator with a unique acquisition function $A_t(\mathbf{y})$. The multi-output emulator is completely defined by choosing the following elements:

1. A *multi-output* interpolation/regression scheme, considering the same input matrix $\mathbf{Y}_t = [\mathbf{y}_1,\ldots,\mathbf{y}_{m_t}]$ (for all the outputs) and the output matrix $\mathbf{X}_t = [\mathbf{x}_1,\ldots,\mathbf{x}_{m_t}]$, providing an approximation

$$\hat{\mathbf{g}}_t(\mathbf{y}|\mathbf{Y}_t,\mathbf{X}_t) = [\hat{g}^{(1)}(\mathbf{y}|\mathbf{Y}_t,\mathbf{X}_t),\ldots,\hat{g}^{(K)}(\mathbf{y}|\mathbf{Y}_t,\mathbf{X}_t)].$$

   In [17,39,13,40], different multi-output schemes are described. The simplest procedure consists in applying one independent interpolator for each output.
2. Given the matrix of current nodes $\mathbf{Y}_t = [\mathbf{y}_1,\ldots,\mathbf{y}_{m_t}]$ (shared by all outputs), we obtain the diversity function $D_t(\mathbf{y})$. Generally, we can have a diversity term $D_t^{(k)}(\mathbf{y})$ for each output (at least they can differ for hyper-parameters learned in different interpolator/regressor), hence we can define

$$D_t(\mathbf{y}) = \prod_{k=1}^{K} D_t^{(k)}(\mathbf{y}).$$

3. Given each $\hat{g}^{(k)}(\mathbf{y}|\mathbf{Y}_t,\mathbf{X}_t)$, we obtain the functions $G_t^{(k)}(\mathbf{y})$, for $k=1,\ldots,K$.

Table 2
Averaged number of nodes m_t.

Random	Latin Hypercube	AGAPE
28.43	16.69	9.16

Therefore, we can define the complete acquisition function as

$$A_t(\mathbf{y}) = \left[\prod_{k=1}^{K} G_t^{(k)}(\mathbf{y})\right]^{\beta_t} \prod_{k=1}^{K} D_t^{(k)}(\mathbf{y}). \quad (15)$$

Optimizing $A_t(\mathbf{y})$, we find the next node $\mathbf{y}_{m_{t+1}}$ to incorporate in the next iteration, $\mathbf{Y}_{t+1} = [\mathbf{Y}_t, \mathbf{y}_{m_{t+1}}]$. Other automatic emulator can be designed considering multiple acquisition functions $A_t^{(k)}(\mathbf{y})$, one for each output.

5.5. Emulation of costly radiative transfer codes

We show empirical evidence of performance on the optimization of selected points for a complex and computationally expensive RTM: the MODTRAN5-based LUT. MODTRAN5 is considered as the *de facto* standard atmospheric RTM for atmospheric correction applications [41]. In our test application, and for the sake of simplicity, we have considered $d = 2$ with the Aerosol Optical Thickness at 550 nm (τ) and ground elevation (h) as key input parameters. The underlying function $g(\mathbf{y})$ consists therefore on the execution of MODTRAN5 at given values of τ and h and wavelength of 760 nm. The input parameter space is bounded to 0.05–0.4 for τ and 0–3 km for h. In order to test the accuracy of the different schemes, we have evaluated $g(\mathbf{y})$ at all the possible 1750 combinations of 35 values of τ and 50 values of h. Namely, this thin grid represents the ground-truth in this example.

We tested (a) a standard, yet suboptimal, random approach choosing points uniformly within $\mathcal{Y} = [0.05, 0.4] \times [0, 3]$, (b) the Latin Hypercube sampling [33], and (c) the proposed AGAPE. We start with $m_0 = 5$ points $\mathbf{y}_1 = [0.05, 0]^\top$, $\mathbf{y}_2 = [0.05, 3]^\top$, $\mathbf{y}_3 = [0.4, 0]^\top$, $\mathbf{y}_4 = [0.4, 3]^\top$ and $\mathbf{y}_5 = [0.2, 1.5]^\top$ for all the techniques. We compute the final number of nodes m_t required to obtain an ℓ_2 distance between g and $\hat{g}$ smaller than $\epsilon = 0.03$, with the different methods. The results, averaged over 10^3 runs, are shown in Table 2. AGAPE requires the addition of $\approx$4 new points to obtain a distance smaller than 0.03.

5.6. Example of multi-output emulation

We consider a multi-output toy example with scalar inputs where we can easily compare the achieved approximation $\hat{\mathbf{g}}_t(y)$ with the underlying function $\mathbf{g}(y)$, which is unknown in the real-world applications. In this way, we can exactly check the true accuracy of the obtained approximation using different schemes. For the sake of simplicity, we consider the following multi-output mapping

$$\mathbf{g}(y) = [\log(y), 0.5\log(3y)], \quad y \in (0, 10], \quad (16)$$

then $d = 1$ and $K = 2$. Even in this simple scenario, the procedure used for selecting new points is relevant as confirmed by the results provided below. We start with $m_0 = 4$ support points, $\mathbf{Y}_0 = [0.1, 3.4, 6.7, 10]$. We apply one independent GP for each output. We add to $\mathbf{Y}_t$ sequentially 20 additional points, using different sampling strategies:

(a) the multi-output version of AGAPE (denoted as AMOGAPE),
(b) uniform points randomly generated in (0, 10],
(c) a sequential Sobol sequence,

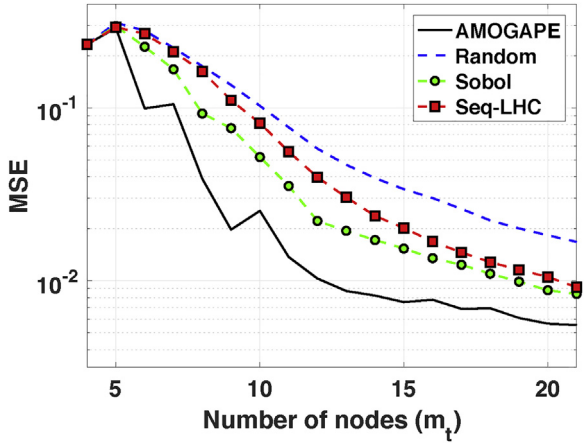

Fig. 17. MSE (in log-scale) between $\mathbf{g}(y)$ and $\hat{\mathbf{g}}_t(y)$ versus the number of the number of support points m_t, that is $m_t = t + 4$ in this example.

(d) and a sequential version of the Latin Hypercube procedure (Seq-LHC).

In this last case, i.e., for Seq-LHC, 20 points are generated following the LHC procedure and then one of them is added to $\mathbf{Y}_t$ at each iteration (without replacement). Note that, at each run, the results can vary even for the deterministic procedure due to the optimization of the hyperparameters (we use a parallel simulated annealing approach that is a stochastic optimization technique [35,42,43]). We average all the results over 500 independent runs.

We compute the L_2 distance, i.e., the Mean Square Error (MSE), between $\hat{\mathbf{g}}_t(y)$ and $\mathbf{g}(y)$ at each iteration, obtained by the different method. We show the evolution of the averaged MSE versus the number of support points m_t (that is $m_t = t + m_0$) in Fig. 17. We can observe that the AMOGAPE scheme outperforms the other methods, providing the smallest MSEs between $\mathbf{g}(y)$ and $\hat{\mathbf{g}}_t(y)$.

6. Conclusions

This paper treated the problems of forward and inverse modeling in remote sensing using advanced machine learning methods. We presented the field formally, revised the concept of inverting or emulating radiative transfer models, and identified the main current shortcomings and trends when using machine learning in these settings.

The first section of the paper introduced the theory and use of GP models in real world applications, including retrieval of biophysical parameters for assessment and decision making in crop management and filling gaps in time series of regional EO products. GP models provide high prediction accuracy and error bars for the predictions that can be useful for masking poor predictions or detecting anomalies. The models have now been adopted or are being considered for implementation in operation processing chains.

In the following sections we payed attention to advanced GP models. All of the presented methods were motivated by observing that two (apparently contradictory) philosophies are typically adopted: either trusting the physical rules encoded in the physical model (for both simulation or inversion), or directly relying on data and thus following data science approaches (for both emulation and retrieval). We posit here that a richer, more appropriate approach to these problems emerges by developing machine learning algorithms that respect the Physics, that can incorporate prior knowledge, and that provide not only accurate but also credible predictions. Three types of physics-aware GP models were introduced: a simple approach to combine *in situ* measurements and

simulated data in a single GP model, a latent force model that incorporates ordinary differential equations, and an automatic compact emulator of physical models through GPs. The developed models demonstrated good performance, adaptation to the signal characteristics and transportability to unseen situations. We analyze the main features, pros and cons of the proposed models in what follows.

The joint GP model is a very useful approach whenever one has access to real and simulated data by a physical model. By definition, simulations can cover a large parameter space. This is why the JGP model can help to extrapolate into the regions where real data is scarce. The JGP model is actually quite conservative and tends to perform as well or better than using real data alone. Two main shortcomings were identified though: (1) the more simulated data added, the better performance is obtained in general but the training process will be slower; and (2) There needs to be simulated data in the region of the real data, otherwise the model is not going to trust that the simulated data can be effectively used to improve predictions. In our experiments we illustrated the performance of the JGP in extrapolation across sites, thus demonstrating that the model learns a sort of domain adaptation by grounding on the simulated data that should be equally relevant in different sites.

A second approach for inverse modeling (retrieval) presented here relies on latent force models, that is, in models that encode differential equations that model the generating system well into GPs. The GP-LFM allows us to model the correlation among multiple related outputs automatically by mixing the black-box, data-driven approach characteristic of machine learning approaches with the physical information used to derive purely mechanistic models. The GP-LFM should be used when: (1) data for multiple correlated outputs is available; (2) input–output relationships in the form of differential equations are known (either exactly or approximately). The GP-LFM has the following advantages: (1) it combines the rigour in the incorporation of the available information of data-driven Bayesian approaches with the problem description accuracy of purely mechanistic models; (2) it can be cast as a generative model where the input-output relationships are not enforced directly (as in other multi-output/task approaches), but through physically interpretable latent forces (GPs) that are automatically learned from data; (3) some physical interpretability can be gained from the learned hyperparameters of the latent GPs and the smoothing kernels used to model input-output relationships; (4) the LFM-GP can be directly applied to outputs with very different characteristics, dimensions and sampling rates; and (5) the model showed very good extrapolation capabilities, thus being able to deal with missing data and to make predictions in regions not covered by the available data. Some disadvantages can be identified though: (1) the model needs to work out the expressions of the output kernels for each combination of input kernels and smoothing kernels; (2) high computational cost of GPs, which is increased by the fact that output kernels are more complex than standard kernels used in GP regression; (3) cannot incorporate nonlinear differential equations, but only their linearized versions, as the whole input-output process would not be a GP any more.

The last novel methodology presented here has to do with the emulation of costly RTMs. For this we presented AGAPE, an automatic sequential interpolator/regressor that iteratively selects the parameter region to sample from, and build an optimal (compact) emulator. Additional advantages of AGAPE are that provides the information where possible new nodes should be incorporated to better approximate the unknown, underlying function encoded in the RTM. The performance of AGAPE depends on a good choice of the starting points (and, as consequence, of an enough number of them), and a well-designed tempering function. These two considerations are connected: indeed, the key point is how much we believe/trust the estimation of the gradient from the previous function approximation. For the same reasons, as the dimensionality of the problem grows, AGAPE requires a greater number of starting nodes, in order to ensure the reliability of the initial approximation of the unknown function.

The framework that we presented here for attaining Physics-aware machine learning was illustrated using Gaussian processes because of the solid grounds on Bayesian inference, properties and mathematical tractability. However, it has not escaped our notice that other machine learning algorithms could be equally applied. These issues will be subject of future research.

Acknowledgements

The research was funded by the European Research Council (ERC) under the ERC-CoG-2014 SEDAL project (grant agreement 647423), and the Spanish Ministry of Economy and Competitiveness (MINECO) and the European Fund for Regional Development (ERDF) through the project TIN2015-64210-R. The authors would like to thank the Institute for Electromagnetic Sensing of the Environment, the Cereal Institute of DEMETER, and the Aristotle University of Thessaloniki for providing the Italian and Greek field data acquired under the ERMES FP7 project.

References

[1] T. Hilker, N.C. Coops, M.A. Wulder, T.A. Black, R.D. Guy, The use of remote sensing in light use efficiency based models of gross primary production: a review of current status and future requirements, Sci. Tot. Environ. 404 (2–3) (2008) 411–423.
[2] J. Chen, T. Black, Defining leaf area index for non-flat leaves, Plant Cell Environ. 15 (1992) 421–429.
[3] R.H. Whittaker, P.L. Marks, Methods of assessing terrestrial productivity, Prim. Product. Biosph. (1975) 55–118.
[4] H.K. Lichtenthaler, Chlorophylls and carotenoids: pigments of photosynthetic biomembranes, Methods Enzymol. 148 (1987) 350–382.
[5] R. Snieder, J. Trampert, Inverse Problems in Geophysics, Springer Vienna, Vienna, 1999, pp. 119–190.
[6] S. Jacquemoud, C. Bacour, H. Poilvé, J.-P. Frangi, Comparison of four radiative transfer models to simulate plant canopies reflectance: direct and inverse mode, Remote Sens. Environ. 74 (3) (2000) 471–481.
[7] W. Verhoef, H. Bach, Simulation of hyperspectral and directional radiance images using coupled biophysical and atmospheric radiative transfer models, Remote Sens. Environ. 87 (2003) 23–41.
[8] S. Liang, Advances in Land Remote Sensing: System, Modeling, Inversion and Applications, Springer Verlag, Germany, 2008.
[9] G. Camps-Valls, D. Tuia, L. Gómez-Chova, S. Jiménez, J. Malo (Eds.), Remote Sensing Image Processing, Morgan & Claypool Publishers, LaPorte, CO, USA, 2011.
[10] G. Camps-Valls, L. Bruzzone, Kernel Methods for Remote Sensing Data Analysis, John Wiley and Sons, 2009.
[11] C.E. Rasmussen, C.K.I. Williams, Gaussian Processes for Machine Learning, The MIT Press, New York, 2006.
[12] J. Verrelst, L. Alonso, G. Camps-Valls, J. Delegido, J. Moreno, Retrieval of vegetation biophysical parameters using Gaussian process techniques, IEEE Trans. Geosci. Remote Sens. 50 (5/P2) (2012) 1832–1843, cited By 26.
[13] G. Camps-Valls, J. Verrelst, J. Muñoz-Marí, V. Laparra, F. Mateo-Jiménez, J. Gomez-Dans, A survey on Gaussian processes for earth observation data analysis, IEEE Geosci. Remote Sens. Mag. (6) (2016), http://dx.doi.org/10.1109/MGRS.2015.2510084 http://ieeexplore.ieee.org/document/7487896/.
[14] M. Campos-Taberner, F.J. García-Haro, G. Camps-Valls, G. Grau-Muedra, F. Nutini, L. Busetto, D. Katsantonis, D. Stavrakoudis, C. Minakou, L. Gatti, M. Barbieri, F. Holecz, D. Stroppiana, M. Boschetti, Exploitation of SAR and optical sentinel data to detect rice crop and estimate seasonal dynamics of leaf area index, Remote Sens. 9 (3) (2017) 248, http://dx.doi.org/10.3390/rs9030248.
[15] L. Busetto, S. Casteleyn, C. Granell, M. Pepe, M. Barbieri, M. Campos-Taberner, R. Casa, F. Collivignarelli, R. Confalonieri, A. Crema, et al., Downstream services for rice crop monitoring in Europe: from regional to local scale, IEEE J. Select. Top. Appl. Earth Observ. Remote Sens. 10 (12) (2017) 5423–5441.
[16] D. Heestermans Svendsen, L. Martino, M. Campos-Taberner, F.J. García-Haro, G. Camps-Valls, Joint Gaussian processes for biophysical parameter retrieval, IEEE Trans. Geosci. Remote Sens. 1 (1) (2017) 1.
[17] M.A. Alvarez, D. Luengo, N.D. Lawrence, Latent force models, International Conference on Artificial Intelligence and Statistics (2009) 9–16.
[18] M.A. Álvarez, D. Luengo, N.D. Lawrence, Linear latent force models using Gaussian processes, IEEE Trans. Pattern Anal. Mach. Intell. 35 (11) (2013) 2693–2705.
[19] M. Campos-Taberner, F.J. García-Haro, G. Camps-Valls, G. Grau-Muedra, F. Nutini, A. Crema, M. Boschetti, Multitemporal and multiresolution leaf area

index retrieval for operational local rice crop monitoring, Remote Sens. Environ. 187 (2016) 102–118, http://dx.doi.org/10.1016/j.rse.2016.10.009.
[20] M. Campos-Taberner, F.J. García-Haro, R. Confalonieri, B. Martńez, A. Moreno, S. Sánchez-Ruiz, M. Gilabert, F. Camacho, M. Boschetti, L. Busetto, Multitemporal monitoring of plant area index in the Valencia rice district with pocketlai, Remote Sens. 8 (3) (2016) 202, http://dx.doi.org/10.3390/rs8030202.
[21] D. Luengo-Garcia, M. Campos-Taberner, G. Camps-Valls, Latent force models for earth observation time series prediction, in: 2016 IEEE International Workshop on Machine Learning for Signal Processing (MLSP 2016), Salerno, Italy, 2016.
[22] G. Camps-Valls, J. Verrelst, L. Martino, J. Vicent, Advanced machine learning emulators of radiative transfer models, in: American Geophysical Union (AGU) Fall meeting 2017, New Orleans, USA, 11–15 December 2017, 2017.
[23] L. Martino, J. Vicent, G. Camps-Valls, Automatic emulator and optimized look-up table generation for radiative transfer models, in: 2017 IEEE International Geoscience and Remote Sensing Symposium, Fort Worth, TX, USA, 2017.
[24] L. Martino, J. Vicent, G. Camps-Valls, Automatic emulation by adaptive relevance vector machines, in: Scandinavian Conference on Image Analysis (SCIA), Tromsø, Norway, 2017.
[25] G. Camps-Valls, D. Svendsen, L. Martino, J. Muñoz-Marí, V. Laparra, M. Campos-Taberner, D. Luengo, Physics-aware Gaussian processes for Earth observation, in: Scandinavian Conference on Image Analysis (SCIA), Tromsø, Norway, 2017.
[26] M. Campos-Taberner, F.J. García-Haro, A. Moreno, M. Gilabert, S. Sanchez-Ruiz, B. Martinez, G. Camps-Valls, Mapping leaf area index with a smartphone and Gaussian processes, Geosci. Remote Sens. Lett. IEEE 12 (12) (2015) 2501–2505.
[27] C.M. Bishop, Pattern recognition, Mach. Learn. 128 (2006) 1–58.
[28] D. Tuia, G. Camps-Valls, Kernel manifold alignment for domain adaptation, PLoS ONE (6) (2016), http://dx.doi.org/10.1371/journal.pone.0148655.
[29] D. Higdon, et al., Space and space-time modeling using process convolutions, Quantitative methods for current environmental issues 3754.
[30] P. Boyle, M. Frean, Dependent Gaussian processes, Advances in Neural Information Processing Systems (2004) 217–224.
[31] I. Nabney, NETLAB: Algorithms for Pattern Recognition, Springer Science & Business Media, 2002.
[32] R. Myneni, S. Hoffman, Y. Knyazikhin, J. Privette, J. Glassy, Y. Tian, Y. Wang, X. Song, Y. Zhang, G. Smith, A. Lotsch, M. Friedl, J. Morisette, P. Votava, R. Nemani, S. Running, Global products of vegetation leaf area and fraction absorbed par from year one of modis data, Remote Sens. Environ. 83 (1) (2002) 214–231, http://dx.doi.org/10.1016/S0034-4257(02)00074-3.
[33] D. Busby, Hierarchical adaptive experimental design for Gaussian process emulators, Reliab. Eng. Syst. Saf. 94 (2009) 1183–1193.
[34] J. Rivera, J. Verrelst, J. Gómez-Dans, J. Muñoz-Marí, J. Moreno, G. Camps-Valls, An emulator toolbox to approximate radiative transfer models with statistical learning, Remote Sens. 7 (7) (2015) 9347–9370.
[35] L. Martino, V. Elvira, D. Luengo, J. Corander, F. Louzada, Orthogonal parallel MCMC methods for sampling and optimization, Digital Signal Process. 58 (2016) 64–84.
[36] J. Read, L. Martino, D. Luengo, Efficient Monte Carlo optimization for multi-label classifier chains, IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP) (2013) 1–5.
[37] L. Martino, V. Elvira, D. Luengo, A. Artes, J. Corander, Smelly parallel MCMC chains, IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP) (2015) 1–5.
[38] L. Martino, V. Elvira, D. Luengo, J. Corander, Interacting parallel Markov adaptive importance sampling, European Signal Processing Conference (EUSIPCO) (2015) 1–5.
[39] M.A. Alvarez, D. Luengo, M.K. Titsias, N.D. Lawrence, Efficient multioutput gaussian processes through variational inducing kernels, International Conference on Artificial Intelligence and Statistics (2010) 25–32.
[40] D. Tuia, J. Verrelst, L. Alonso, F. Pérez-Cruz, G. Camps-Valls, Multioutput support vector regression for remote sensing biophysical parameter estimation, IEEE Geosci. Remote Sens. Lett. 8 (4) (2011) 804–808.
[41] A. Berk, G. Anderson, P. Acharya, L. Bernstein, L. Muratov, J. Lee, M. Fox, S. Adler-Golden, J. Chetwynd, M. Hoke, R. Lockwood, J. Gardner, T. Cooley, C. Borel, P. Lewis, E. Shettle, MODTRAN5: 2006 Update, The International Society for Optical Engineering, 2006.
[42] L. Martino, V. Elvira, D. Luengo, F. Louzada, Parallel Metropolis chains with cooperative adaptation, International Conference on Acoustics, Speech, and Signal Processing (ICASSP) (2016) 1–5.
[43] S. Kirkpatrick, C.D. Gelatt Jr., M.P. Vecchi, Optimization by simulated annealing, Science 220 (4598) (1983) 671–680.