# Approximate inference in related multi-output Gaussian Process Regression

Ankit Chiplunkar, Emmanuel Rachelson, Michele Colombo, Joseph Morlier

# Approximate inference in related multi-output Gaussian Process Regression

Ankit CHIPLUNKAR *, Emmanuel RACHELSON **, Michele COLOMBO ***, and Joseph MORLIER †

Airbus Operations S.A.S., Toulouse, 31060, France
{ankit.chiplunkar,michele.colombo}@airbus.com
ISAE-Supaero, Département d'Ingénierie des Systèmes Complexes (DISC), Toulouse, 31055, France
{emmanuel.rachelson,joseph.morlier}@isae-supaero.fr
Université de Toulouse, CNRS, ISAE-SUPAERO, Institut Clément Ader (ICA), 31077 TOULOUSE CEDEX 4
{joseph.morlier}@isae-supaero.fr

**Abstract.** In Gaussian Processes a multi-output kernel is a covariance function over correlated outputs. Using a prior known relation between outputs, joint auto- and cross-covariance functions can be constructed. Realizations from these joint-covariance functions give outputs that are consistent with the prior relation. One issue with gaussian process regression is efficient inference when scaling upto large datasets. In this paper we use approximate inference techniques upon multi-output kernels enforcing relationships between outputs. Results of the proposed methodology for theoretical data and real world applications are presented. The main contribution of this paper is the application and validation of our methodology on a dataset of real aircraft flight tests, while imposing knowledge of aircraft physics into the model.

**Keywords:** Gaussian Process, Kernel Methods, Approximate Inference, Multi-Output Regression, Flight-test data

## 1 Introduction

The main difference between the physical sciences and machine learning can be explained by the difference between deduction and induction. Physical sciences is deduction: where a very general formula is applied to a particular case. The basics of newtonian physics when applied to a particular aircraft geometry give

* PhD Candidate, Flight Physics Airbus Operations, 316 route de Bayonne 31060
** Associate Professor, Université de Toulouse, ISAE-SUPAERO, Département d'Ingénierie des Systèmes Complexes (DISC), 10 Avenue Edouard Belin, 31055 TOULOUSE Cedex 4, France
*** Loads and Aeroelastics Engineer, Flight Physics, 316 route de Bayonne 31060
† Professor, Université de Toulouse, CNRS, ISAE-SUPAERO, Institut Clément Ader (ICA), 10 avenue Edouard Belin 31077 TOULOUSE Cedex 4

us inertial loads. The basics of aerodynamics when applied to particular set of aircraft geometry and aircraft states give out aerodynamic pressures. Physical sciences take global rules and apply them to local configurations, whereas machine learning is induction [9]. It looks at local features and data, tries to find similarity measures between them and gives a global formula for the environment. For this reason machine learning algorithms perform better in presence of more and more data.

Unfortunately, gathering highly accurate data for physical systems is a costly exercise. Highly accurate CFD simulations may run for weeks and flight test campaigns cost millions of dollars. In this regard there is a dichotomy between these two fields of science, where machine learning needs more data for good performance but procuring data from physical systems is a costly exercise. In this work we consider using prior information of relationships between several outputs thereby effectively increasing the number of data-points.

We use multiple-output Gaussian Process (GP) regression [12] to encode the physical laws of the system and effectively increase the amount of training data points. Inference on multiple output data is also known as co-kriging [14], multi-kriging [3] or Gradient Enhanced Kriging. Using a general framework [7] to calculate covariance functions between multiple-outputs, we extend the framework of gradient enhanced kriging to integral enhanced kriging, quadratic enhanced kriging or any functional relationship between inputs.

Let us start by defining a $P$ dimensional input space and a $D$ dimensional output space. Such that $\{(x_i^j, y_i^j)\}$ for $j \in [1; n_i]$ are the training datasets for the $i^{th}$ output. Here $n_i$ is the number of measurement points for the $i^{th}$ output, while $x_i^j \in \mathbb{R}^P$ and $y_i^j \in \mathbb{R}$. We next define $x_i = \{x_i^1; x_i^2; \ldots; x_i^{n_i}\}$ and $y_i = \{y_i^1; y_i^2; \ldots; y_i^{n_i}\}$ as the full matrices containing all the training points for the $i^{th}$ output such that $x_i \in \mathbb{R}^{n_i \times P}$ and $y_i \in \mathbb{R}^{n_i}$. Henceforth we define the joint output vector $Y = [y_1; y_2; y_3; \ldots; y_D]$ such that all the output values are stacked one after the other. Similarly, we define the joint input matrix as $X = [x_1, x_2, x_3, \ldots, x_D]$. If $\Sigma n_i = N$ for $i \in [1, D]$. Hence $N$ represents the total number of training points for all the outputs combined. Then $Y \in \mathbb{R}^N$ and $X \in \mathbb{R}^{N \times P}$.

For simplicity take the case of an explicit relationship between two outputs $y_1$ and $y_2$. Suppose we measure two outputs with some error, while the true physical process is defined by latent variables $f_1$ and $f_2$. Then the relation between the output function, measurement error and true physical process can be written as follows.

$$\begin{aligned} y_1 &= f_1 + \epsilon_{n1} \\ y_2 &= f_2 + \epsilon_{n2} \end{aligned} \tag{1}$$

Where, $\epsilon_{n1}$ and $\epsilon_{n2}$ are measurement error sampled from a white noise gaussian $\mathcal{N}(0, \sigma_{n1})$ and $\mathcal{N}(0, \sigma_{n2})$ respectively. While the physics based relation can be expressed as follows.

$$f_1 = g\left(f_2, x_1\right) \tag{2}$$

Here $g$ is an operator defining the relation between $f_1$ and an independent latent output $f_2$.

While a joint model developed using correlated covariance functions gives better predictions, it incurs a huge cost on memory occupied and computational time. The main contribution of this paper is to apply approximate inference on these models of large datasets and reduce the heavy computational costs incurred. For a multi-output GP as defined earlier the covariance matrix is of size $N$, needing $\mathcal{O}\left(N^3\right)$ calculations for inference and $\mathcal{O}\left(N^2\right)$ for storage. In this work we compare performance of variational inference [15] and distributed GP [8] to approximate the inference in a joint-kernel.

The remaining paper proceeds as follows, section 2 provides the theoretical framework for Gaussian Process regression. Section 3 extends the multi-output GP regression in presence of correlated covariances. In section 4 various methods of approximating inference of a multi-output GP are derived. Finally, in section 5 we demonstrate the approach on both theoretical and flight-test data.

## 2 Gaussian Process Regression

A gaussian process is an infinite dimensional multi-variate gaussian. Such that any subset of the process is a multi-variate gaussian distribution. A gaussian process can be fully parametrized by a mean and covariance function Eq. 3.

$$y(x) = GP(m(x,\theta), k(x,x',\theta)) \tag{3}$$

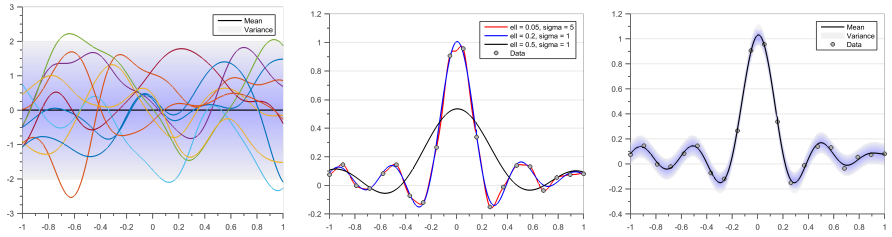
A random draw from a Gaussian Process gives us a random function around the mean function $m(x,\theta)$ and of the shape as defined by covariance function $k(x,x',\theta)$. Hence, Gaussian Process gives us a method to define a family of functions whose shape is defined by its covariance function. A popular choice of covariance function is a squared exponential function Eq. 4, because it defines a family of highly smooth (infinitely differentiable) non-linear functions as shown in Fig. 1(a).

$$k(x,x',\theta) = \theta_1^2 exp[-\frac{(x-x')^2}{2\theta_2^2}] \tag{4}$$

A covariance function is fully parametrized by its hyperparameters $\theta_i$'s. For the case of Squared exponential kernel the hyperparameters are its amplitude $\theta_1$ and length scale $\theta_2$.

Given a dataset $(x,y)$ regression deals with finding latent function $f$ between the inputs $x$ and outputs $y$. While performing polynomial regression we assume that our function $f$ comes from a family of polynomial functions. Since Gaussian Processes are such handy tools to define a family of non-linear functions. In a Gaussian Process Regression (GPR) we start with an initial family of functions defined by a GP called prior Fig. 1(a).

$$\mathbb{P}(f \mid x,\theta) = GP(y|0, K_{xx}) \tag{5}$$




(a) Prior (b) Marginal Likelihood (c) Posterior

**Fig. 1.** Gaussian Process Regression

Due to the bayesian setting of GP we can calculate the posterior mean and variance as shown in Eq. 6 and Eq. 7. This means that we are effectively eliminating all the functions in the prior that do not pass through our data points Fig. 1(c).

$$m(y_*) = k_{x_* x}(K_{xx})^{-1} y \tag{6}$$

$$Cov(y_*) = k_{x_* x_*} - k_{x_* x}\ (K_{xx})^{-1} k_{x x_*} \tag{7}$$

We can also improve our predictions by choosing a better prior. This involves optimizing the Marginal Likelihood (ML) $\mathbb{P}(y \mid x, \theta)$ calculated as Eq. 8. The probability that our dataset $(x, y)$ comes from a family of functions defined by the prior is called the ML [12]. Hence, when we optimize the ML we are actually finding the optimal $\theta$ or family of functions that describe our data Fig. 1(b).

$$\mathbb{P}(y \mid x, \theta) = GP(y|0, K_{xx} + \sigma^2 I) \tag{8}$$

## 3 Multi-output Gaussian Process

Given a dataset for multiple outputs $\{(x_i, y_i)\}$ for $i \in [1; D]$ we define the joint output vector $Y = [y_1; y_2; y_3; \ldots; y_D]$ such that all the output values are stacked one after the other. Similarly, we define the joint input matrix as $X = [x_1; x_2; x_3; \ldots; x_D]$. For the sake of simplicity, suppose we measure two outputs $y_1$ and $y_2$ with some error, while the true physical process is defined by latent variables $f_1$ and $f_2$ equation 2. The operator $g(.)$ can be a known physical equation or a computer code between the outputs, it basically represents a transformation from one output to another.

### 3.1 Related Work

Earlier work developing such joint-covariance functions [2] have focused on building different outputs as a combination of a set of latent functions. GP priors are placed independently over all the latent functions thereby inducing a correlated

covariance function. More recently it has been shown that convolution processes [1] , [3] can be used to develop joint-covariance functions for differential equations. In a convolution process framework output functions are generated by convolving several latent functions with a smoothing kernel function. In the current paper we assume one output function to be independent and evaluate the remaining auto- and cross-covariance functions exactly if the physical relation between them is linear [13] or use approximate joint-covariance for non-linear physics-based relationships between the outputs [7].

### 3.2 Multi-output Joint-covariance kernels

A GP prior in such a setting with 2 output variables is expressed in equation 9.

$$\begin{bmatrix} f_1 \\ f_2 \end{bmatrix} \sim GP\left[\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} K_{11}\ K_{12} \\ K_{21}\ K_{22} \end{pmatrix}\right] \tag{9}$$

$K_{12}$ and $K_{21}$ are cross-covariances between the two inputs $x_1$ and $x_2$. $K_{22}$ is the auto-covariance function of independent output, while $K_{11}$ is the auto-covariance of the dependent output variable. The full covariance matrix $K_{XX}$ is also called the joint-covariance. While, the joint error matrix will be denoted by $\Sigma$;

$$\Sigma = \begin{bmatrix} \sigma_{n1}^2 & 0 \\ 0 & \sigma_{n2}^2 \end{bmatrix} \tag{10}$$

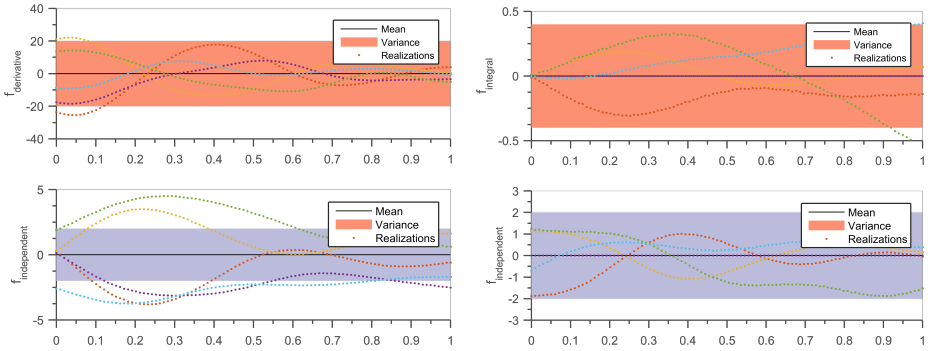
Where, $\epsilon_{n1}$ and $\epsilon_{n2}$ are measurement error sampled from a white noise gaussian $\mathcal{N}(0, \sigma_{n1})$ and $\mathcal{N}(0, \sigma_{n2})$.

For a linear operator $g(.)$ the joint-covariance matrix can be derived analytically [14], due to the affine property of Gaussian's equation 11.

$$\begin{bmatrix} f_1 \\ f_2 \end{bmatrix} \sim GP\left[\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} g(g(K_{22}, x_2), x_1)\ g(K_{22}, x_1) \\ g(K_{22}, x_2)\qquad K_{22} \end{pmatrix}\right] \tag{11}$$

Using the known relation between outputs we have successfully correlated two GP priors from equation 2. This effectively means that when we randomly draw a function $f_2$ it will result in a correlated draw of $f_1$ such that the two draws satisfy the equation 2. We have effectively represented the covariance function $K_{11}$ in terms of the hyperparameters of covariance function $K_{22}$ using the known relation between outputs.

Without loss of generality we can assume that the independent output $f_2$ belongs to a family of functions defined by a Squared Exponential kernel. The joint-covariance between $f_1$ and $f_2$, means that a random draw of independent function $f_2$ will result in a correlated draw of the function $f_1$. The fig: 2(a) shows random draws coming from a differential relationship between $f_1$ (red) and $f_2$ (blue) such that $f_1 = \frac{\partial f_2}{\partial x}$. We can see that the top figure is derivative of the bottom one since $f_{derivative}$ goes to zero where $f_{independent}$ goes to maxima or minima. Similarly, the fig: 2(b) shows random draws coming from an integral relationship between $f_1$ (red) and $f_2$ (blue) such that $f_1 = \int f_2$. We can see that the top figure is integral of the bottom one since $f_{independent}$ goes to zero where $f_{integral}$ goes to maxima or minima.




(a) Joint draws between $f_1$ *red* and $f_2$ *blue* such that $f_1 = \frac{\partial f_2}{\partial x}$.

(b) Joint draws between $f_1$ *red* and $f_2$ *blue* such that $f_1 = \int f_2$.

**Fig. 2.** Multi-Output Gaussian Process Random Draws

### 3.3 GP Regression Using Joint-Covariance

We start with defining a zero-mean prior for our observations and make predictions for $y_1(x_*) = y_{*1}$ and $y_2(x_*) = y_{*2}$. The corresponding prior according to equation 9 and 10 will be:

$$\begin{bmatrix} Y(X)) \\ Y(X_*)) \end{bmatrix} = GP\left[\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} K_{XX} + \Sigma & K_{XX_*} \\ K_{X_*X} & K_{X_*X_*} + \Sigma \end{bmatrix}\right] \tag{12}$$

The posterior distribution is then given as a normal distribution with expectation and covariance matrix given by [12]

$$m(y_*) = K_{X_*X}\left(K_{XX}\right)^{-1} Y \tag{13}$$

$$Cov(y_*) = K_{X_*X_*} - K_{X_*X}\left(K_{XX}\right)^{-1} K_{XX_*} \tag{14}$$

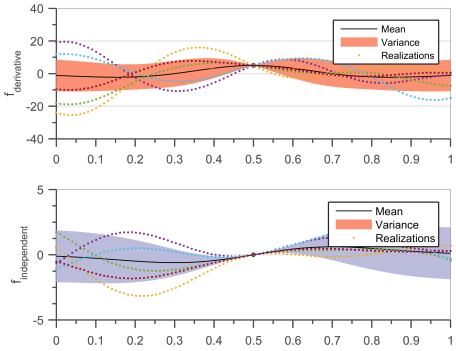
Here, the elements $K_{XX}$, $K_{X_*X}$ and $K_{X_*X_*}$ are block covariances derived from equations 11. Due to the bayesian setting we have basically eliminated all the functions that do not pass through the points defined by the observed data.

The joint-covariance matrix depends on several hyperparameters $\theta$. They define a basic shape of the GP prior. To end up with good predictions it is important to start with a good GP prior. We minimize the negative log-marginal likelihood to find a set of good hyperparameters. This leads to an optimization problem where the objective function is given by equation 15

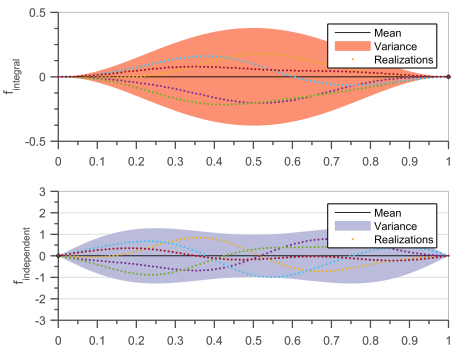
$$\log(\mathbb{P}(y \mid X, \theta)) = \log[GP(Y|0, K_{XX} + \Sigma)] \tag{15}$$

With its gradient given by equation 16

$$\frac{\partial}{\partial\theta}\log(\mathbb{P}(y \mid X, \theta)) = \frac{1}{2}Y^T K_{XX}^{-1}\frac{\partial K_{XX}}{\partial\theta}K_{XX}^{-1}Y - \frac{1}{2}tr(K_{XX}^{-1}\frac{\partial K_{XX}}{\partial\theta}) \tag{16}$$




(a) Joint draws between $f_1$ *red* and $f_2$ *blue* such that $f_1 = \frac{\partial f_2}{\partial x}$. The posterior is conditioned such that $f_1 = 5; x = 0$ and $f_2 = 0; x = 0$. The draws from the posterior follow the relationship between outputs and also the conditioning.




(b) Joint draws between $f_1$ *red* and $f_2$ *blue* such that $f_1 = \int f_2$. The posterior is conditioned such that $f_1 = 0; x = 0$ and $f_1 = 0; x = 1$. The draws from the posterior follow the relationship between outputs and also the conditioning.

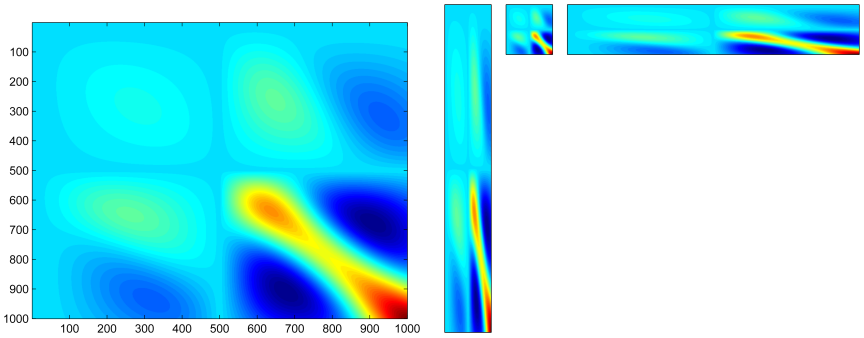
**Fig. 3.** Multi-Output Gaussian Process Regression Predictions

Here the hyperparameters of the prior are $\theta = \left\{l_2, \sigma_2^2, \sigma_{n1}^2, \sigma_{n2}^2\right\}$. These correspond to the hyperparameters of the independent covariance function $K_{22}$ and errors in the measurements $\sigma_{n1}^2$ and $\sigma_{n2}^2$. Calculating the negative log-marginal likelihood involves inverting the matrix $K_{XX} + \Sigma$. The size of the $K_{XX} + \Sigma$ matrix depends on total number of input points $N$, hence inverting the matrix becomes intractable for large number of input points.

The fig: 3(a) shows mean and variance for a differential relationship between independent function $f_2$ (blue) and differential function $f_1$ (red) and such that $f_1 = \frac{\partial f_2}{\partial x}$. We have conditioned the two functions such that $f_1 = 5; x = 0$ and $f_2 = 0; x = 0$. This means that the function $f_2$ passes through 0 and has a derivative equal to 5 at $x = 0$. We have not maximized the marginal likelihood for this case and the hyperparameters of $K_{22}$ are $\theta_1 = 1; \theta_2 = 0.2$. All the corresponding draws from the posterior GP also follow the conditioning.

The fig: 3(b) shows mean and variance for a integral relationship between independent function $f_2$ (blue) and differential function $f_1$ (red) and such that $f_1 = \int f_2.dx$. We have conditioned the two functions such that $f_1 = 0; x = 1$ and $f_1 = 0; x = 0$. This means that the function $f_2$ has an integral 0 at the two points $[0, 1]$ . We have not maximized the marginal likelihood for this case and the hyperparameters of $K_{22}$ are $\theta_1 = 1; \theta_2 = 0.2$. All the corresponding draws from the posterior GP also follow the conditioning. We can observe that the draws will also have an integral 0 between the range $[0, 1]$.

In the next section we describe how to solve the problem of inverting huge $K_{XX} + \Sigma$ matrices using approximate inference techniques.




(a) Full multi-output covariance matrix for fig: 3(b)

(b) Approximated multi-output covariance matrix for fig: 3(b)

**Fig. 4.** Variational approximation of covariance matrix for Gaussian Process Regression

## 4 Approximating Inference

The above GP approach is intractable for large datasets. For a multi-output GP as defined in section 3.2 the covariance matrix is of size $N$, where $\mathcal{O}\left(N^3\right)$ time is needed for inference and $\mathcal{O}\left(N^2\right)$ memory for storage. Thus, we need to consider approximate methods in order to deal with large datasets.

Inverting the covariance matrix takes considerable amount of time and memory during the process. Hence, almost all techniques to approximate inference try and approximate the inversion of covariance matrix $K_{XX}$. If a covariance matrix is diagonal or block-diagonal in nature then methods such as mixture of experts are used eg. distributed GP. Whereas if the covariance matrix is more spread out and has similar terms in its cross diagonals then low-rank approximations are used eg. variational approximation. The remaining section details the two methods for approximating covariance matrix which can be later used to resolve equations 13, 14 and 16.

### 4.1 Variational Approximation on Multi-output GP

Sparse methods use a small set of $m$ function points as support or inducing variables. Suppose we use $m$ inducing variables to construct our sparse GP. The inducing variables are the latent function values evaluated at inputs $x_M$. Learning $x_M$ and the hyperparameters $\theta$ is the problem we need to solve in order to obtain a sparse GP method. An approximation to the true log marginal likelihood in equation 15 can allow us to infer these quantities.

We try to approximate the joint-posterior distribution $p(X|Y)$ by introducing a variational distribution $q(X)$. In the case of varying number of inputs for different outputs, we place the inducing points over the input space and extend the derivation of [15] to multi-output case.

$$q(X) = \mathcal{N}(X|\mu, A) \tag{17}$$

Here $\mu$ and $A$ are parameters of the variational distribution. We follow the derivation provided in [15] and obtain the lower bound of true marginal likelihood.

$$F_V = log(\mathcal{N}[Y|0, \sigma^2 I + Q_{XX}]) - \frac{1}{2\sigma^2} Tr(\tilde{K}) \tag{18}$$

where $Q_{XX} = K_{XX_M} K_{X_M X_M}^{-1} K_{X_M X}$ and $\tilde{K} = K_{XX} - K_{XX_M} K_{X_M X_M}^{-1} K_{X_M X}$. $K_{XX}$ is the joint-covariance matrix derived using equation 11 using the input vector $X$ defined in section 1. $K_{X_M X_M}$ is the joint-covariance function on the inducing points $X_M$, such that $X_M = [x_{M1}, x_{M2}, ..., x_{M2}]$. We assume that the inducing points $x_{Mi}$ will be same for all the outputs, hence $x_{M1} = x_{M2} = ... = x_{M2} = x_M$. While $K_{XX_M}$ is the cross-covariance matrix between $X$ and $X_M$.

Note that this bound consists of two parts. The first part is the log of a GP prior with the only difference that now the covariance matrix has a lower rank of $MD$. This form allows the inversion of the covariance matrix to take place in $\mathcal{O}\left(N(MD)^2\right)$ time. The second part as discussed above can be seen as a penalization term that regularizes the estimation of the parameters.

The bound can be maximized with respect to all parameters of the covariance function; both model hyperparameters and variational parameters. The optimization parameters are the inducing inputs $x_M$, the hyperparameters $\theta$ of the independent covariance matrix $K_{22}$ and the error while measuring the outputs $\sigma$. There is a trade-off between quality of the estimate and amount of time taken for the estimation process. On the one hand the number of inducing points determine the value of optimized negative log-marginal likelihood and hence the quality of the estimate. While, on the other hand there is a computational load of $\mathcal{O}\left(N(MD)^2\right)$ for inference. We increase the number of inducing points until the difference between two successive likelihoods is below a predefined quantity.

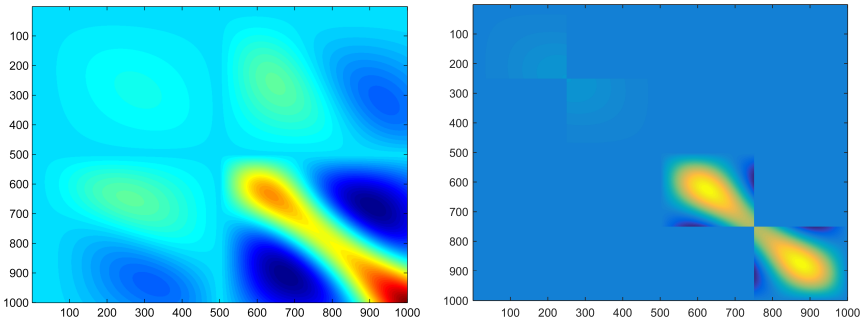
### 4.2 Distributed Inference on Multi-output GP

An alternative to sparse approximations is to learn local experts on subset of data. Traditionally, each subset of data learns a different model from another, this is done to increase the expressiveness in the model [11]. The final predictions are then made by combining the predictions of local experts [5].

An alternative way is to tie all the different experts using one single set of hyperparameters [8]. This is equivalent to assuming one single GP on the whole dataset such that there is no correlation across experts as seen in figure 5(b). This tying of experts acts as a regularization and inhibits overfitting. Although ignoring correlation among experts is a strong assumption, it can be justified if the experts are chosen randomly and with enough overlap.

If we partition the dataset into M subsets such as $\mathcal{D}^{(i)} = X^{(i)}, y^{(i)}, i = 1, \dots M$.

$$\log p(y|X, \theta) \approx \sum_{k=1}^{M} \log p_k(y^{(i)}|X^{(i)}, \theta) \tag{19}$$




(a) Full multi-output covariance matrix for fig: 3(b)

(b) Distributed multi-output covariance matrix for fig: 3(b) the cross-diagonal terms (blue) are zero due to independence between experts

**Fig. 5.** Distributed approximation of covariance matrix for Gaussian Process Regression

The above equation 19 describes the formulation for marginal likelihood. Due to the independence assumption the marginal likelihood can be written as a sum of individual likelihoods and then can be optimized to find the best-fit hyperparameters. After learning the hyperparameters we can combine the predictions of local experts to give mean and variance predictions. The robust Bayesian Commitee Machine (rBCM) model combines the various experts using their confidence on the prediction point [8]. In such manner experts which have high confidence at the prediction points get more weight when compared to experts with low confidence.

$$m(Y_*) = (Cov(X_*))^{-2} \sum \beta_k \sigma_k^{-2} m_k(X_*) \tag{20}$$

$$(Cov_(Y_*))^{-2} = \sum_k \beta_k \sigma_k^{-2} + (1 - \sum_k \beta_k)\sigma_{**}^{-2} \tag{21}$$

In the above equations $m_k(X_*)$ and $\sigma_k$ are the mean and covariance predictions from expert $k$ at point $X_*$. $\sigma_{**}$ is the auto-covariance of the prior at prediction points $X_*$. $\beta_k$ determines the influence of experts on the final predictions [4] and is given as $\beta_k = \frac{1}{2}(\log \sigma_{**}^{-2} - \log \sigma_k^{-2})$.

## 5 Experiments

We empirically assess the performance of distributed Gaussian Process and Variational Inference with respect to the training time and accuracy. We start with a synthetic dataset where we try to learn the model over derivative relationship and compare the two inference techniques. We then evaluate the improvement on real world flight-test dataset.

The basic toolbox used for this paper is GPML provided with [12], we generate covariance functions to handle relationships as described in equations 11 using the "Symbolic Math Toolbox" in MATLAB 2014b. Variational inference is wrapped from gpStuff toolbox [16] and distributed GP is inspired from [8]. All experiments were performed on an Intel quad-core processor with 4Gb RAM.

### 5.1 Experiments on Theoretical Data

We consider a derivative relationship between two output functions as described in equation 2. Such that

$$g(f,x) = \frac{\partial f}{\partial x}$$

Since the differential relationship $g(.)$ is linear in nature we use the equation 11 to calculate the auto- and cross-covariance functions as shown in table 1.

**Table 1.** Auto- and cross-covariance functions for a differential relationship.

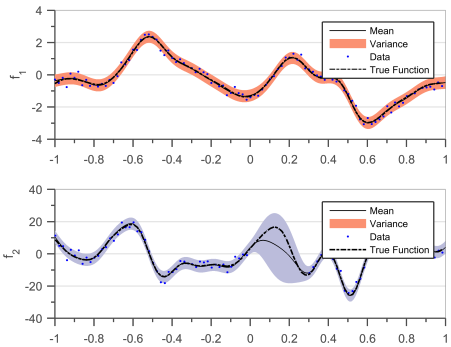
| | | |
|---|---|---|
| Initial Covariance | $K_{22}$ | $\sigma^2 exp(\frac{-1}{2}\frac{d^2}{l^2})$ |
| Cross-Covariance | $K_{12}$ | $\sigma^2 \frac{d}{l^2} exp(\frac{-1}{2}\frac{d^2}{l^2})$ |
| Auto-covariance | $K_{11}$ | $\sigma^2 \frac{d^2-l^2}{l^4} exp(\frac{-1}{2}\frac{d^2}{l^2})$ |

Data is generated from equations 22, a random function is drawn from GP to get $f_2$ whose derivative is then calculated to generate $f_1$. $y_1$ and $y_2$ are then calculated by adding noise according the the equations 22. 10,000 points are generated for both the outputs $y_1$ and $y_2$. Values of $y_2$ are masked in the region $x \in [0, 0.3]$ the remaining points now constitute our training dataset.

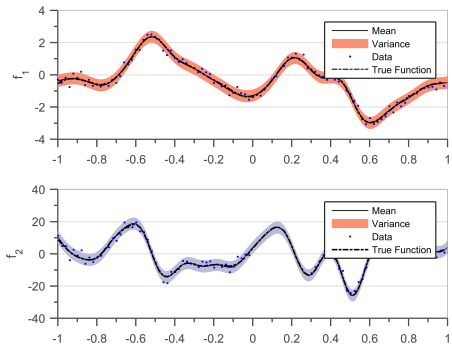
$$\begin{aligned} f_2 &\sim GP[0, K_{SE}(0.1, 1)] \\ \sigma_{n2} &\sim \mathcal{N}[0, 0.2] \\ \sigma_{n1} &\sim \mathcal{N}[0, 2] \end{aligned} \tag{22}$$

$K_{SE}(0.1, 1)$ means squared exponential kernel with length scale 0.1 and variance as 1.

Figure 6 shows comparison between an independent fit GP and a joint multi-output GP whose outputs are related through a derivative relationship described in equation 5.1. For figure 6(a) using variational inference algorithm we optimize the lower bound of log-marginal likelihood, for independent GP's on $y_1$ and $y_2$. For figure 6(b) using variational inference we optimize the same lower bound but with a joint-covariance approach as described in section 4.1 using $y_1$, $y_2$ and $g(.)$. We settled on using 100 equidistant inducing points for this exercise [6] and have only optimized the hyperparameters to learn the model.




(a) Independent fit for two GP's mean is represented by solid black line. $2\Sigma$ confidence band is represented by light red for $f_1$ and light blue for $f_2$.The dashed black line represents the true latent function values; noisy data is denoted by blue dots. Experiment was run on 10,000 points but only 100 data points are plotted to increase readibility. Inference is performed using variational inference algorithm and equidistant 100 inducing points. We can observe the huge difference between the real data and the predicted mean values at zone with no data.

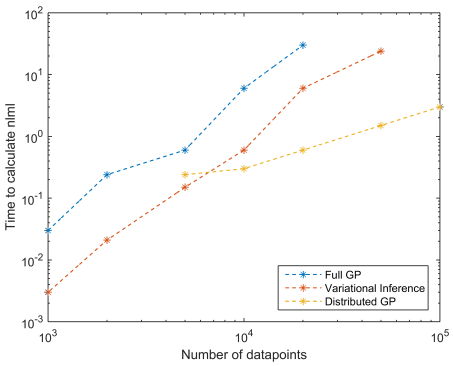



(b) Joint multi-output GP's for two outputs, mean is represented by solid black line. $2\Sigma$ confidence band is represented by light red for $f_1$ and light blue for $f_2$.The dashed black line represents the true latent function values; noisy data is denoted by blue dots. Experiment was run on 10,000 points but only 100 data points are plotted to increase readibility. Inference is performed using variational inference algorithm and equidistant 100 inducing points. We can observe the improved prediction between zone with no data because information is being shared between the two outputs.

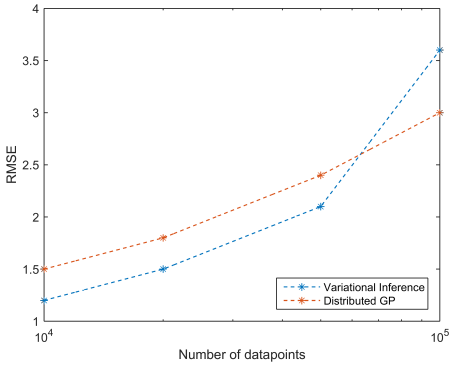
**Fig. 6.** Experimental results for differential relationship while using variational approximation

Figure 6(a) shows the independent fit of two GP for the differential relationship, while figure 6(b) shows the joint GP fit. The GP model with joint-covariance gives better prediction even in absence of data of $y_2$ for $x \in [0, 0.3]$.

For the second experiment we compare the Root Mean Squared Error (RMSE) and run-times of distributed GP and Variational Inference algorithms while performing approximate inference. We progressively generate from 10e3 to 10e5 data-points according to the equations 22 and 5.1. We separated 75% of the data as training set and 25% of the data as the test set, the training and test sets were chosen randomly. The variational inference relationship kernel as described in section 4.1 with 100 equidistant inducing points was used. We learn the optimal values of hyper-parameters for all the sets of training data. The distributed GP algorithm as described in section 4.2 was used with randomly chosen 512 points per expert. We learn the optimal values of hyper-parameters for all the sets of training data. The accuracy is plotted as RMSE values with respect to the test set. The runtime is defined as time taken to calculate negative




(a) Comparison of time to calculate negative log marginal likelihood for a Full GP, Variational inference and Distributed GP with increasing number of datapoints. We observe for datapoints greater that 10e5 the distributed GP algorithm starts outperforming variational inference




(b) Comparison of RMSE for variational inference and distributed GP algorithm. We observe for datapoints greater that 10e4 the distributed GP algorithm starts outperforming variational inference.

**Fig. 7.** Comparison of run time and RMSE between distributed GP and Variational Inference

log marginal likelihood equations 18 and 19. The RMSE values are calculated for only the dependent output $y_1$ and then plotted in the figure 7(a).

In figure 7(a) the time to calculate negative log-marginal likelihood with increasing number of training points is calculated. As expected the full GP takes more time when compared to variational inference or distributed GP algorithms. The Variational inference algorithm has better run-time till 10e4 data-points after that distributed GP takes lesser time. In figure 7(b) the RMSE error with test set is compared between the variational inference and distributed GP algorithm. Here too the variational inference algorithm performs better lesser number of datapoints but distributed GP starts performing better when we reach more than 10e4 datapoints.

One thing to note is that we have fixed the number and position of inducing points while performing this experiment. While a more optimized set of inducing points will have better results, for datasets of the order 10e5 distributed GP algorithm starts outperforming variational inference. Moreover, upon observing figure 5(b) we can say that the covariance matrix in a joint GP setting is not diagonal in nature and hence an approximation technique which can compensate between low-rank approximation and diagonal approximation should be investigated [10].

### 5.2 Experiments on Flight Test Data

We perform experiments on flight loads data produced during flight test phase at Airbus. Loads are measured across the wing span using strain gauges. Shear load

$T_z$ and bending moment $M_x$ as described in figure 8 are used as two outputs for this exercise. $\eta$ or point of action of forces and angle of attack $\alpha$ are the two inputs. The aircraft is in quasi-equillibrium in all conditions and there are no dynamic effects observed throughout this dataset. All data is normalized according to airbus policy.

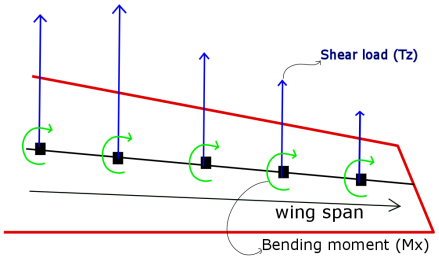



**Fig. 8.** Wing Load Diagram

The relation between $T_z$ and $M_x$ can be written as:

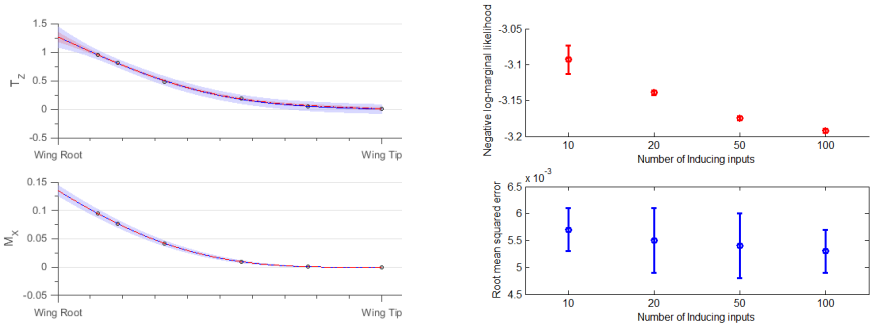
$$M_x(\eta, \alpha) = \int_{\eta}^{\eta_{edge}} T_Z(x, \alpha)(x - \eta)dx \tag{23}$$

.

The equation 23 is applicable for the $\eta$ axis. Here, $\eta_{edge}$ denotes the edge of the wing span. The forces are measured at 5 points on the wing span and at 8800 points on the $\alpha$ dimension. We compare plots of relationship-imposed multioutput GP and independent GP. Then we compare the measures of negative-log marginal likelihood and RMSE for varying number of inducing points.

Figure 9(a) shows the independent (blue shade) and joint fit (red shade) of two GP. The top figure shows $T_Z$ with the variance of dependent GP plotted in red and variance of independent GP plotted in blue. Bottom figure shows plots for $M_X$. Since the number of input points is less than 10e5 we use variational inference. 100 inducing points in the input space are used to learn and plot the figure. The variance of red is smaller than that of blue showing the improvement in confidence when imposing relationships in the GP architecture. The relationship between $T_Z$ and $M_X$ gives rise to better confidence during the loads prediction. This added information is very useful when identifying faulty sensor data since equation 23 will push data points which do not satisfy the relationship out of the tight confidence interval.

Figure 9(b) shows improvement in the negative log-marginal likelihood and RMSE plots upon increasing number of inducing points. 10 sets of experiments were run on 75% of the data as training set and 25% of the data as the test set, the training and test sets were chosen randomly. We learn the optimal values of hyper-parameters and inducing points for all the 10 sets of experiments




(a) $2\sigma$ confidence interval and mean of the dependent GP are represented in red shade and solid red line. $2\sigma$ confidence interval and mean of the independent GP are represented in blue shade and solid blue line. Experiment was run on 8800 data points Noisy data is denoted by circles only 1 $\alpha$ step is plotted. Confidence interval improves upon adding the relationship kernel.

(b) Progression of RMSE and log-likelihood upon increasing number of inducing points. Top plot shows the value of mean and variance of negative log-marginal likelihood. The bottom figure in blue shows the mean and variance of root mean squared error. 10 sets of experiments were run on 75% of the data as training set and 25% of the data as the test set, the training and test sets were chosen randomly.

**Fig. 9.** Experimental results for aircraft flight loads

of training data. Finally, RMSE values are evaluated with respect to the test set and negative log-marginal likelihood are evaluated for each learned model. The RMSE and log-likelihood improve upon increasing the number of inducing points.

## 6 Conclusions and Future Work

This paper presents approximate inference methods for physics-based multiple output GP's. We extend the variational inference and distributed GP inference techniques to be applied on physics-based multi-output GP's and reduce the computational load for inference.

Section 5.1 demonstrates the advantage of using multi-output GP in presence of prior known relationships on a theoretical dataset. Significant improvement in prediction can be observed in presence of missing data due to transfer of information occuring due to the prior relationship. Then, we compare the difference in accuracy and run times upon using distributed GP or variational inference to approximate inference. Although variational inference performs better for datasets of size less than 10e4, distributed GP becomes significantly advantageous for datasets greater than 10e5.

Section 5.2 shows real world application of joint kernel on flight loads data. We demonstrate that adding prior physics-based relationships allows us to create

a robust, physically consistent and interpretable surrogate model for these loads. Aircraft flight domain consists of various maneuvers, this is further complicated by several relationships between outputs. In the future we wish to develop better strategies to exploit the clustered nature of flight-domain and more than two relationships.

## References

1. Alvarez, M.A., Luengo, D., Lawrence, N.D.: Latent force models. In: Dyk, D.A.V., Welling, M. (eds.) AISTATS. JMLR Proceedings, vol. 5, pp. 9–16. JMLR.org (2009)
2. Bonilla, E., Chai, K.M., Williams, C.: Multi-task gaussian process prediction. In: Platt, J., Koller, D., Singer, Y., Roweis, S. (eds.) Advances in Neural Information Processing Systems 20, pp. 153–160. MIT Press, Cambridge, MA (2008)
3. Boyle, P., Frean, M.: Dependent gaussian processes. In: In Advances in Neural Information Processing Systems 17. pp. 217–224. MIT Press (2005)
4. Cao, Y., Fleet, D.J.: Generalized product of experts for automatic and principled fusion of gaussian process predictions. CoRR abs/1410.7827 (2014), `http://arxiv.org/abs/1410.7827`
5. Chen, T., Ren, J.: Bagging for gaussian process regression. Neurocomputing 72(7), 1605–1610 (2009)
6. Chiplunkar, A., Rachelson, E., Colombo, M., Morlier, J.: Sparse physics-based gaussian process for multi-output regression using variational inference. In: Proceedings of the 5th International Conference on Pattern Recognition Applications and Methods. pp. 437–445 (2016)
7. Constantinescu, E.M., Anitescu, M.: Physics-based covariance models for gaussian processes with multiple outputs. International Journal for Uncertainty Quantification 3 (2013)
8. Deisenroth, M.P., Ng, J.W.: Distributed gaussian processes. arXiv preprint arXiv:1502.02843 (2015)
9. Domingos, P.: A few useful things to know about machine learning. Communications of the ACM 55(10), 78–87 (2012)
10. March, W.B., Xiao, B., Biros, G.: Askit: Approximate skeletonization kernel-independent treecode in high dimensions. SIAM Journal on Scientific Computing 37(2), A1089–A1110 (2015)
11. Rasmussen, C.E., Ghahramani, Z.: Infinite mixtures of gaussian process experts. Advances in neural information processing systems 2, 881–888 (2002)
12. Rasmussen, C.E., Williams, C.K.I.: Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning). The MIT Press (2005)
13. Solak, E., Murray-smith, R., Leithead, W.E., Leith, D.J., Rasmussen, C.E.: Derivative observations in gaussian process models of dynamic systems. In: Becker, S., Thrun, S., Obermayer, K. (eds.) Advances in Neural Information Processing Systems 15, pp. 1057–1064. MIT Press (2003)
14. Stein, M.L.: Interpolation of Spatial Data: Some Theory for Kriging. Springer, New York (1999)
15. Titsias, M.K.: Variational learning of inducing variables in sparse gaussian processes. In: In Artificial Intelligence and Statistics 12. pp. 567–574 (2009)
16. Vanhatalo, J., Riihimäki, J., Hartikainen, J., Jylänki, P., Tolvanen, V., Vehtari, A.: Gpstuff: Bayesian modeling with gaussian processes. J. Mach. Learn. Res. 14(1), 1175–1179 (Apr 2013)