Robust weighted Gaussian processes

Ruben Ramirez-Padron, Boris Mederos & Avelino J. Gonzalez

ONLINE FIRST

Computational Statistics

ISSN 0943-4062

Comput Stat
DOI 10.1007/s00180-020-01011-0

Computational Statistics
https://doi.org/10.1007/s00180-020-01011-0

ORIGINAL PAPER

Robust weighted Gaussian processes

Ruben Ramirez-Padron[1,2] · Boris Mederos[3] · Avelino J. Gonzalez[4]

Received: 10 May 2018 / Accepted: 29 June 2020

Abstract
This paper presents robust weighted variants of batch and online standard Gaussian processes (GPs) to effectively reduce the negative impact of outliers in the corresponding GP models. This is done by introducing robust data weighers that rely on robust and quasi-robust weight functions that come from robust M-estimators. Our robust GPs are compared to various GP models on four datasets. It is shown that our batch and online robust weighted GPs are indeed robust to outliers, significantly outperforming the corresponding standard GPs and the recently proposed heteroscedastic GP method GPz. Our experiments also show that our methods are comparable to and sometimes better than a state-of-the-art robust GP that uses a Student-t likelihood.

Keywords Machine learning · Online learning · Robust regression · Outlying data

1 Introduction

Given a dataset $D = \left\{(\mathbf{x}_i, \mathrm{y}_i) : \mathbf{x}_i \in \mathcal{X}, \ \ \mathcal{X} \subseteq \mathbb{R}^l, \ \ \mathrm{y}_i \in \mathbb{R}, \ i = 1, 2, \ldots, N\right\}$, where $l \in \mathbb{N}$, a common problem in statistics and machine learning consists of modeling the functional relationship between the independent variables $\mathbf{x}_i$ and the corresponding dependent variable y_i. A main goal on statistics is inference, while machine learning

✉ Ruben Ramirez-Padron
rramirez@knights.ucf.edu

Boris Mederos
boris.mederos@uacj.mx

Avelino J. Gonzalez
avelino.gonzalez@knights.ucf.edu

[1] Department of Electrical and Computer Engineering, University of Central Florida, Orlando, FL, USA

[2] Present Address: TeleTracking Technologies, Pittsburgh, PA, USA

[3] Department of Physics and Mathematics, Universidad Autónoma de Ciudad Juárez, Ciudad Juárez, Mexico

[4] Department of Computer Science, University of Central Florida, Orlando, FL, USA

typically focuses on obtaining algorithms that learn a regression model from D that can be used to predict the most likely y values corresponding to arbitrary $\mathbf{x}$ values in $\mathcal{X}$.

Gaussian processes (GPs) have been used successfully as a powerful and flexible Bayesian regression tool in many fields. GPs are nonparametric kernel-based function estimation techniques that model a probability distribution over a space $\mathfrak{F}$ of functions $f : \mathcal{X} \rightarrow \mathbb{R}$, where $\mathcal{X} \subseteq \mathbb{R}^l$ is a continuous l-dimensional input space (Bishop 2006; MacKay 1998; Rasmussen and Williams 2006).

The expression for the GP posterior is obtained analytically when the likelihood is Gaussian. This simple case of GP Bayesian regression is called *standard GP regression*. It has been shown (Rasmussen and Williams 2006; Seeger 2004), that standard GPs provide performance gains and theoretical benefits when compared to other successful regression techniques, such as splines (de Boor 2001), support vector machines (Drucker et al. 1997) and relevance vector machines (Tipping 2000). GPs can be obtained from a training dataset D using a batch learning algorithm. Additionally, there are algorithms that allow learning GP models incrementally, such as *Online GP* (Csató and Opper 2002).

In practice, it is commonly found that a regression model with independent normal errors describes the majority of observations, except for a few that are located far from the bulk of the data, which are called *outliers*. Even a single outlier can have a large distorting influence on most regression models. Noteworthy, obtaining effective predictions using standard GP models in the presence of outliers has been shown to be difficult or impossible in many cases (Neal 1997).

A common approach to address outliers within a Bayesian framework is to use a robust non-Gaussian likelihood to obtain robust posterior distributions. For instance, robust pseudo-likelihoods (Greco et al. 2008) and likelihoods corresponding to heavy-tailed distributions, such as Laplace and Student-t distributions, have been used (Geweke 1993; West 1984). The Student-t distribution has also been used to obtain robust GPs, e.g., Tipping and Lawrence (2005), Kuss 2006, and Jylänki et al. (2011). For example, in Jylänki et al. (2011) a modified expectation propagation (EP) (Minka 2001) is proposed as an approximation technique to obtain the GP posteriors, to eliminate numerical instabilities. Ranjan et al. (2016) introduces a robust GP that uses Laplace or Student-t likelihoods using expectation-maximization (EM). In Kuss et al. (2005), a Gaussian mixture likelihood (two-model model) for robust GP regression is proposed. In order to handle the intractability of the non-Gaussian likelihood they use the EP for doing approximate inference.

In Mattos et al. (2015), it is shown that robust GPs that use heavy-tailed likelihoods outperform standard GPs when compared using root mean square errors (RMSE). However, non-Gaussian likelihoods lead to analytically intractable inferences, which require approximation techniques such as Laplace approximation (Williams and Barber 1998) and EP. These approximation techniques are generally complex, computationally expensive and convergence is not generally guaranteed.

In Agostinelli and Greco (2013), an approach is presented to address outliers within the Bayesian framework by means of weighted likelihoods. Given a set of i.i.d. observations $\mathbf{y} = \{\mathrm{y}_1, \mathrm{y}_2, \ldots, \mathrm{y}_N\}$, drawn from a random variable Y with unknown probability (density) function $p(y|\theta)$, which is an element of a parametric family

$\{p(y|\theta) : \boldsymbol{\theta} \in \boldsymbol{\Theta} \subset \mathbb{R}^p\}$. A weight is given to each likelihood term to limit the effect of outliers. The weights are determined by a bounded, differentiable and non-negative function $w(\mathrm{y}; \hat{\boldsymbol{\eta}}, \hat{F}_N)$, where $\hat{\boldsymbol{\eta}}$ denotes estimates of unknown parameters and $\hat{F}_N$ stands for an empirical cumulative distribution function. From this, a weighted likelihood function $L^w(\mathbf{y}|\boldsymbol{\theta})$ is proposed, where the weights w are based on the discrepancy between the observed sample and the assumed model. The estimated $\hat{\boldsymbol{\theta}}$ has a high breakdown point. However, this type of weight function may be expensive to compute, given its dependency on estimating multiple parameters and $\hat{F}_N$.

We introduce an effective approach to obtain robust weighted GP (RWGP) models that is inspired by the framework proposed in Agostinelli and Greco (2013). However, our approach is simpler, given that we only require computing a weight for each observation based on the robust mean and variance of a small neighborhood around the observation, instead of imposing strong sufficient conditions on the weight functions, as done in Agostinelli and Greco (2013). The proposed RWGP does not require approximation techniques thanks to the use of a weighted Gaussian likelihood. Additionally, our weighted Gaussian likelihood is a proper Gaussian likelihood. This guarantees that Gaussian posteriors are obtained analytically. This is done without altering the computational complexity of the corresponding standard GP.

The use of weights in GP regression has been proposed before in the context of learning non-stationary system dynamics (Rottmann and Burgard 2010), where data are heteroscedastic. For instance, the weighted GP proposed in Rottmann and Burgard (2010) has been used to represent the uncertainty in a robot's model of an outdoor environment (Murphy et al. 2012). The approach described in Rottmann and Burgard (2010) consists of assigning a weight to each observation and estimating a noise level for each training point. Weights are used exclusively to estimate the hyperparameters of the GP model using weighted cross-validation (Sugiyama et al. 2007). In Rottmann and Burgard (2010), the set of GP hyperparameters is very large, because the individual noise levels are also considered hyperparameters. In our work, weights are not used outside of the GP model as part of a computationally expensive weighted cross-validation.

Le et al. (2005) also proposes an heteroscedastic GP regression model. It provides non-parametric estimates of the mean and variance. To avoid solving a non-convex optimization problem, they estimate the parameters $\mu(x)/\sigma^2(x)$ and $1/\sigma^2(x)$ of the exponential family representation of the normal distribution, which leads to a convex problem. Another relevant work is GPz (Almosallam et al. 2016), which combines sparseness and heteroscedasticity. The underlying assumption is a basis function model (BFM) $y_i = \boldsymbol{\phi}(x_i)^T \boldsymbol{w} + \epsilon_i$, $i = 1, 2, \ldots, N$, where $\boldsymbol{\phi}(x_i) = (\phi_1(x_i), \phi_2(x_i), \ldots, \phi_m(x_i))^T$, $m << N$ and ϵ_i has a zero-mean Gaussian distribution. GPz achieves sparsity by applying a sparsity-inducing prior over the coefficients $\boldsymbol{w} = (w_1, w_2, \ldots, w_m)^T$. To deal with heteroscedastic noise, the likelihood is subsequently redefined as $p(y|\boldsymbol{w}) = \mathcal{N}(y|\boldsymbol{\Phi}\boldsymbol{w}, \boldsymbol{B}^{-1})$, where $\boldsymbol{\Phi} = [\boldsymbol{\phi}(x_1), \boldsymbol{\phi}(x_2), \ldots \boldsymbol{\phi}(x_m)]^T$ and $\boldsymbol{B}$ is a diagonal matrix with elements $\beta_i = \exp(\boldsymbol{\phi}(x_i)\boldsymbol{u} + b)$. As done with $\boldsymbol{w}$, a sparsity-inducing prior is assigned to $\boldsymbol{u}$ to obtain simple expressions for the precision values β_i.

Our work shows some similarities to the heteroscedastic models described above, i.e., weights are used in a similar way. However, our weights are calculated differently, as they are based on the "outlierness" of the corresponding observations using robust statistics (Hampel et al. 1986; Huber and Ronchetti 1981; Rey 1983), to assess the dissimilarity of each observation from the central tendency of neighboring data. This differs from the heteroscedastic approach where the variance of the data is modeled throughout the space, without considering outliers. These differences make a clear distinction between heteroscedatic models and ours.

Outlier detection is a problem related to our work. It consists of modeling a normal class and subsequently using that model to assign outlier scores to new observations. Various works have proposed GPs for outlier detection, where outlier scores are functions of the GP's mean and variance (Kemmler et al. 2010; Ramirez-Padron et al. 2013; Ramirez-Padron 2015; Wang and Mao 2019). Robust methods such as those mentioned above are usually employed to model the normal class as a first step in outlier detection. However, our focus in this work is to learn robust GP models without the need to determine whether observations are outliers or not.

Summarizing, the main contributions of our work are:

1. A novel method to compute weights called here *robust data weighers*, which rely on robust and quasi-robust weight functions from robust statistics.
2. A weighted Gaussian likelihood that employs robust data weighers in order to provide robustness.
3. A batch RWGP that extends the batch standard GP by relying on our weighted Gaussian likelihood.
4. An extension of our approach to the case of online learning, resulting in online RWGP.
5. Proof that the computational complexities of our RWGPs remain the same as that of the corresponding standard GPs; i.e., $O(N^3)$.
6. Multiple experiments using simulated and real-life data show that our RWGPs outperform the corresponding standard GPs and GPz in all cases where datasets were contaminated with outliers. The experiments also show that RWGPs compare favorably to a state-of-the-art robust GP that uses a Student-t likelihood.

The rest of this paper is structured as follows: Sect. 2 provides an overview of related work, upon which our research was developed. In Sect. 3 we introduce our robust data weighers, the robust weighted Gaussian likelihood, and RWGPs. Section 4 shows that the computational complexities of RWGPs are the same as the corresponding standard GPs. Sections 5 and 6 provide an experimental comparison between RWGPs, standard GPs and other robust GP methods. Section 7 explores why the MML method is not appropriate in general to estimate GP hyperparameters. Conclusions and future research directions are given in Sect. 8.

2 Mathematical preliminaries

In this work, the observation model is $\mathrm{y}(\mathbf{x}) = f(\mathbf{x}) + \varepsilon$, where $f(\mathbf{x})$ is a latent random function that is modeled by a GP and the noise ε is independent and identically distributed (i.i.d.) $\mathcal{N}(0, \sigma^2)$. A GP is defined as follows:

Definition 1 (*Gaussian Process*) (Rasmussen and Williams 2006). A GP is a collection of random variables $\{f(\mathbf{x})\}_{\mathbf{x}\in\mathcal{X}}$, $\mathcal{X} \subseteq \mathbb{R}^l$, such that given any finite subcollection $\{\mathbf{x}'_i\}_{i=1}^{M}$, the random vector $\mathbf{f} = \left[f(\mathbf{x}'_1), f(\mathbf{x}'_2), \ldots, f(\mathbf{x}'_M)\right]^T$ has a multivariate Gaussian distribution.

Given a random vector $\mathbf{f}$, the probability $p_0(\mathbf{f}) = \mathcal{N}(\boldsymbol{\mu}_0(\mathbf{f}), \boldsymbol{K}_0)$, where $\boldsymbol{\mu}_0(\mathbf{f}) = \left[\mu_0(\mathbf{x}'_1), \ldots, \mu_0(\mathbf{x}'_M)\right]^T$, $\boldsymbol{\mu}_0(\mathbf{x}) = \mathbb{E}(\mathbf{f}(\mathbf{x}))$, and $\boldsymbol{K}_0 = \left(k(\mathbf{x}'_i, \mathbf{x}'_j)\right)_{i,j}$ is a $M \times M$ covariance matrix. Any positive-definite kernel function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ (Shawe-Taylor and Cristianini 2004) can be used as the GP's covariance function, which determines the characteristics of the GP's realizations, such as smoothness.

Throughout this paper, the points $\mathbf{x}$ are assumed to be non-random and noise-free, $\boldsymbol{X}$ denotes the $N \times l$ matrix $[\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N]^T$ and $\mathbf{y}$ denotes the vector $\left[\mathrm{y}_1, \mathrm{y}_2, \ldots, \mathrm{y}_N\right]^T$. Given any other collection $(\mathbf{x}'_1, \mathbf{x}'_2, \ldots, \mathbf{x}'_M)$ of points, $\boldsymbol{X}'$ denotes the matrix $\left[\mathbf{x}'_1,\ \mathbf{x}'_2, \ldots, \mathbf{x}'_M\right]^T$.

In the following subsections, we describe batch and online standard GPs, as formulated in Csató and Opper (2002) and Rasmussen and Williams (2006), and provide a review of robust and quasi-robust weight functions (Huber and Ronchetti 1981; Maronna et al. 2006; Rey 1983).

2.1 Batch GP regression

Bayesian GP regression is concerned with estimating the predictive distribution $p(\mathrm{y}(\mathbf{x})|\mathbf{x}, D)$ for any $\mathbf{x} \in \mathcal{X}$, given a GP prior $p_0(\mathbf{f})$. It is computed as

$$p(\mathrm{y}(\mathbf{x})|\mathbf{x}, \mathrm{D}) = \int p(\mathrm{y}(\mathbf{x})|f(\mathbf{x}))p_{post}(f(\mathbf{x}))df(\mathbf{x}), \tag{1}$$

where the GP posterior $p_{post}(f(\mathbf{x}))$ is calculated as

$$p_{post}(f(\mathbf{x})) = \frac{p(D|f(\mathbf{x}))p_o(f(\mathbf{x}))}{p(D)} = \frac{\int p(\mathbf{y}|\mathbf{f}_D)\ p_o(f(\mathbf{x}), \mathbf{f}_D)d\mathbf{f}_D}{\mathbb{E}_0[p(\mathbf{y}|\mathbf{f}_D)]}. \tag{2}$$

In (2), $\mathbf{f}_D = [f(\mathbf{x}_1), f(\mathbf{x}_2), \ldots, f(\mathbf{x}_N)]^T$ and $\mathbb{E}_0$ denotes expected value with respect to the GP prior. Calculating the GP posterior $p_{post}(f(\mathbf{x}))$ generally implies approximating an N-dimensional integral that is analytically intractable if $p(\mathbf{y}|\mathbf{f}_D)$ is not Gaussian. However, in the case of a standard GP, the mean and variance of the posterior at a new input $\mathbf{x}$ have the following expressions (Rasmussen and Williams 2006):

$$\mu_{\text{post}} = \mu_0(\mathbf{x}) + \boldsymbol{k}_{\mathbf{x}}^T\boldsymbol{q}, \tag{3}$$

$$\sigma^2_{post} = k(\mathbf{x}, \mathbf{x}) + \boldsymbol{k}_{\mathbf{x}}^T\boldsymbol{R}\boldsymbol{k}_{\mathbf{x}}, \tag{4}$$

where

$$\boldsymbol{q} = \left(\boldsymbol{K}_D + \sigma^2 I\right)^{-1} (\mathbf{y} - \mu_0(\boldsymbol{X})), \tag{5}$$

$$\boldsymbol{R} = -\left(\boldsymbol{K}_D + \sigma^2 I\right)^{-1}, \tag{6}$$

with $\boldsymbol{\mu}_0(\boldsymbol{X}) := [\mu_0(\mathbf{x}_1), \mu_0(\mathbf{x}_2), \ldots, \mu_0(\mathbf{x}_N)]^T$ and $\boldsymbol{K}_D = \left(k(\mathbf{x}_i, \mathbf{x}_j)\right)_{i,j}$ denotes the $N \times N$ prior covariance matrix.

2.2 Online GP

Online GP (Csató and Opper 2002) is based on the Bayesian online learning framework proposed in Opper (1998). Given a new learning observation $(\mathbf{x}_{t+1}, \mathrm{y}_{t+1})$, its corresponding likelihood $p(\mathrm{y}_{t+1}|f(\mathbf{x}_{t+1}))$ is combined with the GP posterior $p_t(\mathbf{f})$ that was obtained from the previous step t to obtain the GP posterior $p_{t+1}(\mathbf{f})$. Specifically, the following recursive expressions are obtained in Csató and Opper (2002) for the moments of the GP posterior at step $t+1$:

$$\mu_{t+1}(\mathbf{x}) = \mu_t(\mathbf{x}) + k_t(\mathbf{x}, \mathbf{x}_{t+1}) q_{t+1}, \tag{7}$$

$$k_{t+1}(\mathbf{x}, \mathbf{x}') = k_t(\mathbf{x}, \mathbf{x}') + k_t(\mathbf{x}, \mathbf{x}_{t+1}) r_{t+1} k_t(\mathbf{x}_{t+1}, \mathbf{x}'). \tag{8}$$

where the coefficients q_{t+1} and r_{t+1} are obtained as

$$q_{t+1} = \frac{\partial}{\partial \mathbb{E}_t\left[f(\mathbf{x}_{t+1})\right]} \ln \mathbb{E}_t\left[p(\mathrm{y}_{t+1}|f(\mathbf{x}_{t+1}))\right], \tag{9}$$

$$r_{t+1} = \frac{\partial^2}{\partial^2 \mathbb{E}_t\left[f(\mathbf{x}_{t+1})\right]} \ln \mathbb{E}_t\left[p(\mathrm{y}_{t+1}|f(\mathbf{x}_{t+1}))\right], \tag{10}$$

with $\mathbb{E}_t\left[p(\mathrm{y}_{t+1}|f(\mathbf{x}_{t+1}))\right] = \int p(\mathrm{y}_{t+1}|f(\mathbf{x}_{t+1})) p_t(f(\mathbf{x}_{t+1})) df(\mathbf{x}_{t+1})$. A full description of how this result is obtained is given in Csató and Opper (2002).

By unfolding the recursive Eqs. (7) and (8), the iterative formulations for estimating the moments of the online GP at step $t+1$ are written as follows:

$$\mu_{t+1}(\mathbf{x}) = \mu_0(\mathbf{x}) + \boldsymbol{k}_{\mathbf{x},t+1}^T \boldsymbol{\alpha}_{t+1}, \tag{11}$$

$$k_{t+1}(\mathbf{x}, \mathbf{x}') = k(\mathbf{x}, \mathbf{x}') + \boldsymbol{k}_{\mathbf{x},t+1}^T \boldsymbol{C}_{t+1} \boldsymbol{k}_{\mathbf{x}',t+1}, \tag{12}$$

where $\boldsymbol{k}_{\mathbf{x},t+1} := \left[k(\mathbf{x}, \mathbf{x}_1), \ldots, k(\mathbf{x}, \mathbf{x}_{t+1})\right]^T$ and the vector $\boldsymbol{\alpha}_{t+1}$ and the matrix $\boldsymbol{C}_{t+1}$ are computed recursively using the following expressions:

$$\boldsymbol{s}_{t+1} = \begin{bmatrix} \boldsymbol{C}_t \boldsymbol{k}_{x_{t+1},t} \\ 1 \end{bmatrix}, \quad \boldsymbol{\alpha}_{t+1} = \begin{bmatrix} \boldsymbol{\alpha}_t \\ 0 \end{bmatrix} + q_{t+1} \boldsymbol{s}_{t+1},$$

$$\boldsymbol{C}_{t+1} = \begin{bmatrix} \boldsymbol{C}_t & \mathbf{0} \\ \mathbf{0} & 0 \end{bmatrix} + r_{t+1} \boldsymbol{s}_{t+1} \boldsymbol{s}_{t+1}^T,$$

where $\boldsymbol{\alpha}_1 = q_1$ and $\boldsymbol{C}_1 = r_1$. For the particular case of a GP with a Gaussian likelihood, the coefficients q_{t+1} and r_{t+1} are expressed as follows (Csató and Opper 2002):

$$q_{t+1} = \frac{\mathrm{y}_{t+1} - m_{t+1}}{\sigma_{t+1}^2 + \sigma^2}, \quad r_{t+1} = -\frac{1}{\sigma_{t+1}^2 + \sigma^2},$$

where $m_{t+1} = \mu_t(\mathbf{x}_{t+1})$ and $\sigma_{t+1}^2 = k_t(\mathbf{x}_{t+1}, \mathbf{x}_{t+1})$.

2.3 Robust and quasi-robust weight functions

Robust statistics (Huber and Ronchetti 1981; Maronna et al. 2006) yields estimation methods that are not greatly affected by outliers. Given a set of real-valued observations $\mathrm{y}_i, i = 1, 2, \ldots, N$ that depend on an unknown parameter μ, let us assume a location model $\mathrm{y}_i = \mu + \varepsilon_i$, where the additive errors $\varepsilon_i, i = 1, 2, \ldots, N$ are i.i.d. random variables. Frequently, a small percentage of errors do not obey the assumed distribution. A common approach to estimate μ under this condition is to obtain the M-estimate of location

$$\hat{\mu} = \arg\min \sum_{i=1}^{N} \rho(\mathrm{y}_i - \mu), \tag{13}$$

where $\rho : \mathbb{R} \longrightarrow \mathbb{R}^+$ satisfies the following two properties:

$$\text{Symmetry} : \rho(-z) = \rho(z), \tag{14}$$

$$\text{Robustness} : \lim_{z\to+\infty} \frac{\psi(z)}{z} = 0, \tag{15}$$

with $\psi(z) = \rho'(z)$, which is known as the *influence function* (Hampel et al. 1986; Maronna et al. 2006). The solution $\hat{\mu}$ of (13) is obtained by means of the following iterative expression:

$$\mu_{n+1} = \frac{\sum_{i=1}^{N} w(\mathrm{y}_i - \mu_n)\mathrm{y}_i}{\sum_{j=1}^{N} w(\mathrm{y}_j - \mu_n)}, \tag{16}$$

where the following expression is called here *robust weight function*

$$w(z) := \begin{cases} \psi(z)/z, & \textit{if } z \neq 0 \\ \psi'(0), & \textit{if } z = 0 \end{cases}. \tag{17}$$

In general, $w(z)$ is a non-increasing function of $|z|$. Because of the robustness property in (15), outlying observations receive smaller weights. Therefore their contribution to the model is small.

There are many ψ-functions that can be found in the literature, such as Huber's function (Huber 1964), Cauchy's function (Maronna et al. 2006) and the Welsh's function (Dennis and Welsch 1978). For some ψ-functions, $w(z) \to 0$ very rapidly when $z \to +\infty$. Consequently, observations that are not too distant from μ might have a very small influence in the estimation process. Additionally, the robustness property can yield numerical algorithms that are ill-posed (Rey 1983). An effective approach to overcome these two issues is to define ψ so that the robustness property is changed to

$$\lim_{z\to+\infty}\frac{\psi(z)}{z}=\gamma, \qquad \gamma\in(0,1). \tag{18}$$

where γ is a very small value. Functions ρ that satisfies (18) are called *quasi-robust ρ-functions* (Rey 1983) and are denoted here by ρ_Q. Commonly, a quasi-robust ρ-function is expressed as $\rho_Q(z)=(1-\gamma)\rho(z)+0.5\gamma z^2$ (Rey 1983). The corresponding weight function, which we call here *quasi-robust weight function* w_Q are given by

$$w_Q(z)=(1-\gamma)w(z)+\gamma. \tag{19}$$

3 Robust weighted Gaussian processes

This section introduces our approach to obtain RWGPs. It includes our definition of robust data weigher and the expressions for batch and online RWGPs.

3.1 Robust weighted Gaussian likelihood

Here we extend the Gaussian likelihood $p(D|\mathbf{f}_D)$ by making it dependent on a collection of weights $\mathbf{w}_D=[\mathrm{w}_1,\mathrm{w}_2,\ldots,\mathrm{w}_N]$, as follows:

Definition 2 (*Weighted Gaussian likelihood*) Given a dataset $D=\{(\mathbf{x}_i,\mathrm{y}_i)\}$, where $\mathbf{x}_i\in\mathcal{X}$, $\mathcal{X}\subseteq\mathbb{R}^l$, $\mathrm{y}_i\in\mathbb{R}$, $i=1,2,\ldots,N$, and given a collection of weights $\mathbf{w}_D=[\mathrm{w}_1,\mathrm{w}_2,\ldots,\mathrm{w}_N]$, such that $\mathrm{w_i}\in(0,1]$, for $i=1,2,\ldots,N$, we call

$$p(D|\mathbf{f}_D;\mathbf{w}_D)=\frac{1}{\sqrt{(2\pi)^N|\mathbf{W}|}}e^{-\frac{1}{2}\,\mathbf{z}_D^T\mathbf{W}^{-1}\mathbf{z}_D}, \tag{20}$$

a *weighted Gaussian likelihood*, where $\mathbf{W}=\sigma^2\mathrm{diag}\left(\frac{1}{\mathrm{w}_1},\frac{1}{\mathrm{w}_2},\ldots,\frac{1}{\mathrm{w}_N}\right)$, $\mathbf{z}_D=\mathbf{y}-\mathbf{f}_D$, and $|\mathbf{W}|$ is the determinant of $\mathbf{W}$.

To add robustness to a weighted Gaussian likelihood, weights should satisfy the following two conditions: outlying observations receive weights that are closer to zero and non-outlying observations should receive weights that are closer to 1. Informally, we call a weighted Gaussian likelihood that uses weights that satisfy these conditions a *robust weighted Gaussian likelihood*. In our case, the weights $\mathrm{w_i}$ are calculated by evaluating a robust weight function $w(\cdot)$ on a standardized value of y_i with respect to data in a neighborhood of $\mathbf{x}_i$, as described in the following section.

The smaller a weight w_i the more irrelevant is the effect of $\mathrm{y}_i-f(\mathbf{x}_i)$ on the corresponding likelihood term $\frac{\sqrt{w_i}}{\sqrt{2\pi\sigma^2}}e^{-\frac{\mathrm{w}_i}{2\sigma^2}(\mathrm{y}_i-f(\mathbf{x}_i))^2}$, in the sense that $f(x_i)$ has more room to change without greatly impacting the value of the corresponding likelihood term. Consequently, predictions of GP models that employ this robust weighted Gaussian likelihood will be more robust to outliers than standard GPs. This is depicted in Fig. 1 by the plots of likelihood terms for different weight values.

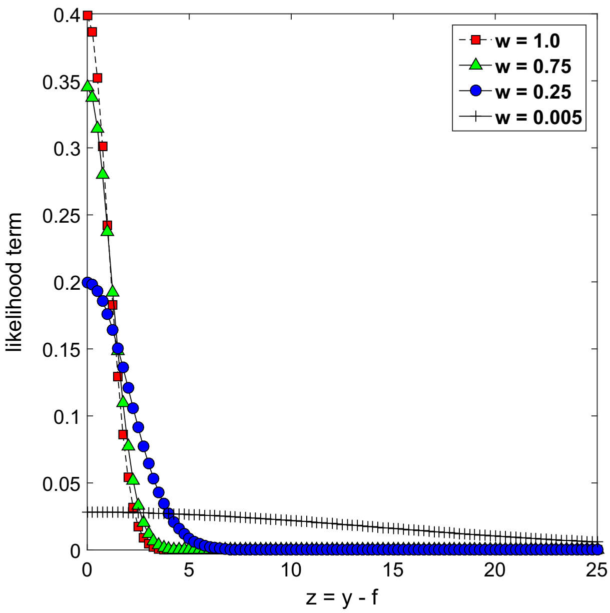

Fig. 1 Plot of likelihood terms from the weighted Gaussian likelihood for different values of w, using $\sigma^2 = 1$

3.2 Robust data weighers

Here we introduce a novel method to compute weights that we call *robust data weighers*. For each observation $(\mathbf{x}_i, \mathrm{y}_i)$ in D, we determine a neighborhood N_i of $\mathbf{x}_i$ in order to compute the corresponding weight w_i, which is calculated based on the relationship of y_i to robust estimations of mean μ_i and variance v_i of the set $\{\mathrm{y}_j : \mathbf{x}_j \in N_i\}$. Small weights are given to observations $(\mathbf{x}_i, \mathrm{y}_i)$ for which y_i significantly deviate from μ_i.

The neighborhood of an observation can be defined based on a distance to the observation or a fixed number of its nearest neighbors. Using the latter approach could lead to neighborhoods where data points are far away from each other. Consequently, those neighborhoods might not properly reflect the context of the observations, i.e., they would not take into account how observations are distributed across the input space. The impact of this issue is greatly reduced when a distance is used to define the neighborhoods. Therefore, we resort to neighborhoods that are defined based on distance. Sometimes a weight function may not be practical for neighborhoods that contain so few data points that weights could not be estimated effectively. To address this issue, we introduce a *default weight* η, $\gamma \leq \eta < 1$, which denotes the uncertainty associated with lack of data in N_i. These considerations lead to the following:

Definition 3 (*Robust data weigher*). Given a training set D, a *robust data weigher* is a function that assigns a weight to each observation in D, according to the expression:

$$\mathrm{w}_{\mathbf{x}_i} = \mathrm{w}_i = \begin{cases} \eta, & \text{if } |N_i| < s \\ w\left(\frac{y_i - \mu_i}{\sqrt{v_i}}\right), & \text{otherwise} \end{cases} \tag{21}$$

where w is either a robust weight function or a quasi-robust weight function; N_i denotes a distance-based neighborhood of $\mathbf{x}_i$; μ_i and v_i are a robust mean and robust variance of the set $\left\{\mathrm{y}_j : \mathbf{x}_j \in N_i\right\}$; and s is a positive integer.

In our implementation of robust data weighers, we use a fixed radius r to construct the neighborhoods N_i of points $(\mathbf{x}_i, \mathrm{y}_i)$. They are defined as the hypersphere $\left\{\mathbf{x}_j : d(\mathbf{x}_i, \mathbf{x}_j) \leq r,\ j = 1, 2, \ldots, N\right\}$, where d stands for a distance that is fitted to the geometry of the data $\mathbf{X}$. In our case, we employed the popular Mahalanobis distance (Mahalanobis 1936), to allow a better fit of the neighborhood to the geometry of the data, in a way that should lead to having more elements for an effective calculation of weights.

The threshold parameter s plays an important role in the definition of robust data weigher. If s is large, it is likely that the vast majority of the data points are given the default weight η. This should make the RWGP model behave similar to the corresponding standard GP. If the value of s was small then most points would be assigned a non-default weight, which should lead to a robust function estimation by our models. This will generally be the case when the neighborhoods have an appropriate radius r. In this work we experimented with various combinations of s and r, which allowed us to obtain good preliminary results from our proposed methods.

The experiments described in this work employed weights that were calculated using a quasi-robust weight function, to avoid the numerical instabilities generally associated to robust weight functions. The default weight η was set to 0.5. We used the highly robust estimator of location and scatter *minimum covariance determinant* (MCD) (Rousseeuw 1984), from the MATLAB library LIBRA (Verboven and Hubert 2005), to calculate μ_i and v_i.

3.3 Robust weighted GPs

Here we propose a simple and computationally efficient framework to make standard GPs robust to outliers without resorting to using an intractable likelihood. We call a standard GP that employs the robust weighted Gaussian likelihood a *robust weighted GP* (RWGP). In the following subsections, we derive the expressions for the posterior moments of batch and online RWGPs.

3.3.1 Batch Robust Weighted GP

To calculate the GP posterior, which is a normal distribution, we need its mean μ_{post} and the covariance σ^2_{post}. The parameterization lemma (Csató 2002; Csató and Opper 2002) allows calculating these posterior moments for any likelihood, based on derivatives of the log marginal likelihood

$$\ln \int p(\mathbf{y}|\mathbf{f}_D;\, \mathbf{w}_D)\, p_o(\mathbf{f}_D) d\mathbf{f}_D.$$

Note that $p(\mathbf{y}|\mathbf{f}_D;\, \mathbf{w}_D)\, p_o(\mathbf{f}_D)$ is the joint Gaussian distribution $p(\mathbf{f}_D, \mathbf{y};\, \mathbf{w}_D)$, which is given by (section 2.3.3 of Bishop (2006)):

$$p(\mathbf{f}_D, \mathbf{y};\, \mathbf{w}_D) = \mathcal{N}\left(\mathbf{f}_D, \mathbf{y}\middle| \begin{bmatrix} \mathbb{E}_0\left[\mathbf{f}_D\right] \\ \mathbb{E}_0\left[\mathbf{f}_D\right] \end{bmatrix}, \begin{bmatrix} \boldsymbol{K}_D & \boldsymbol{K}_D \\ \boldsymbol{K}_D & \boldsymbol{K}_D + \mathbf{W} \end{bmatrix}\right). \tag{22}$$

Employing (22) and properties of multivariate Gaussian distributions, we find an analytic expression for the log marginal likelihood:

$$\ln \int p(\mathbf{y}|\mathbf{f}_D;\, \mathbf{w}_D)\, p_o(\mathbf{f}_D) d\mathbf{f}_D = \ln \int p(\mathbf{y}, \mathbf{f}_D;\, \mathbf{w}_D) d\mathbf{f}_D \tag{23}$$

$$= \ln\left(\mathcal{N}\left(\mathbb{E}_0\left[\mathbf{f}_D\right], K_D + \mathbf{W}\right)\right). \tag{24}$$

Specifically

$$\ln \int p(\mathbf{y}|\mathbf{f}_D;\, \mathbf{w}_D)\, p_o(\mathbf{f}_D) d\mathbf{f}_D = -\frac{1}{2}(\mathbf{y} - \mathbb{E}_0\left[\mathbf{f}_D\right])^T \left[\boldsymbol{K}_D + \mathbf{W}\right]^{-1} (\mathbf{y} - \mathbb{E}_0\left[\mathbf{f}_D\right])$$

$$-\frac{1}{2}\ln\left|\boldsymbol{K}_D + \mathbf{W}\right| - \frac{N}{2}\ln(2\pi). \tag{25}$$

Deriving (25) as specified in the parameterization lemma (Csató 2002; Csató and Opper 2002), we obtain the following expressions for q and R:

$$\boldsymbol{q} = \left[\boldsymbol{K}_D + \mathbf{W}\right]^{-1} (\mathbf{y} - \boldsymbol{\mu}_0(\boldsymbol{X})), \tag{26}$$

$$\boldsymbol{R} = -\left[\boldsymbol{K}_D + \mathbf{W}\right]^{-1} \tag{27}$$

which allow the calculation of the first two moments μ_{post} and σ^2_{post} of the RWGP posterior p_{rob_post}, using (3) and (4) respectively.

Once we have the RWGP posterior, the corresponding predictive distribution is given by

$$p_{rob}(\mathrm{y}(\mathbf{x})|\mathbf{x}, \mathrm{D}) = \int p(\mathrm{y}(\mathbf{x})|f(\mathbf{x});\, \mathrm{w}_{\mathrm{x}})\, p_{rob\_post}(f(\mathbf{x})) df(\mathbf{x}) \tag{28}$$

Applying eqs. (2.109) and (2.110) from section 2.3.3 of Bishop (2006), we obtain that

$$p_{rob}(\mathrm{y}(\mathbf{x})|\mathbf{x}, \mathrm{D}) = \mathcal{N}(\mathrm{y}(\mathbf{x})|\mu_{\text{post}}, \sigma^2_{post} + \sigma^2/\mathrm{w}_{\mathrm{x}}) \tag{29}$$

The expressions for derivatives of the log marginal likelihood that allows the estimation of hyperparameters of RWGP using the MML method are given in the "Appendix".

3.3.2 Robust weighted online GP

The expressions for the online RWGP are the same as in online GP, except that the expressions for the terms q_{t+1} and r_{t+1} are calculated differently, based on the robust weighted Gaussian likelihood. To obtain the values of q_{t+1} and r_{t+1}, the integral that appears in (9) and (10) is rewritten as follows:

$$\begin{aligned}\mathbb{E}_t[p\,(\mathrm{y}_{t+1}|f(\mathbf{x}_{t+1}))] &= \int \mathcal{N}\left(f(\mathbf{x}_{t+1}), \frac{\sigma^2}{\mathrm{w}_{t+1}}\right)\mathcal{N}(m_{t+1}, \sigma_{t+1}^2)df(\mathbf{x}_{t+1})\\ &= \int \mathcal{N}\left(\begin{bmatrix} m_{t+1}\\ m_{t+1}\end{bmatrix}, \begin{bmatrix}\sigma_{t+1}^2 & \sigma_{t+1}^2\\ \sigma_{t+1}^2 & \sigma_{t+1}^2 + \frac{\sigma^2}{\mathrm{w}_{t+1}}\end{bmatrix}\right) df(\mathbf{x}_{t+1})\\ &= \mathcal{N}\left(m_{t+1}, \sigma_{t+1}^2 + \frac{\sigma^2}{\mathrm{w}_{t+1}}\right). \end{aligned} \tag{30}$$

Consequently, q_{t+1} and r_{t+1} are rewritten as:

$$q_{t+1} = \frac{\mathrm{y}_{t+1} - m_{t+1}}{\sigma_{t+1}^2 + \frac{\sigma^2}{\mathrm{w}_{t+1}}}, \quad r_{t+1} = -\frac{1}{\sigma_{t+1}^2 + \frac{\sigma^2}{\mathrm{w}_{t+1}}},$$

where w_{t+1} is the robust weight assigned to observation $(\mathbf{x}_{t+1}, \mathrm{y}_{t+1})$, based on a neighborhood of $\mathbf{x}_{t+1}$ as described previously, but restricted to the set $\{(\mathbf{x}_i, \mathrm{y}_i) : i = 1,\ 2, \ldots, t+1\}$. These expressions for q_{t+1} and r_{t+1} are employed to update model parameters $\boldsymbol{\alpha}$ and $\mathbf{C}$ at each step of the training algorithm in the same way that is done in standard online GP.

There is a subtlety related to how weights are calculated in the online setting: weights for previously learned observations would likely have changed at any learning step if they were recalculated taking the new observations into account. Ideally, $\boldsymbol{\alpha}$ and $\mathbf{C}$ would reflect such change at each learning step t. However, this is very difficult given the recursive nature of the learning algorithm and should be considered future work. Note however, that weights should tend to stabilize after learning a large number of observations if there was not significant concept drift in our data. Therefore, our approach (i.e., not updating the weights of previously learned data) should not impose a great limitation on the effectiveness of online RWGP.

4 Computational complexities

This section shows that our RWGPs can be implemented in a way that their computational complexities are the same as that of the corresponding standard models.

The reliance of RWGPs on neighborhoods of data points leads to employ space-partitioning data structures on D, such as k-d trees (Bentley 1980), which have a computational complexity of $O(N\log N)$ (Wald and Havran 2006). Once a k-d tree is built, the complexity of determining the N neighborhoods is $O(N\log N)$. Given that applying the MCD method to a single neighborhood on m-dimensional data takes

$O(N^{\frac{m(m+3)}{2}})$ (Bernholt and Fischer 2004), using MCD on each neighborhood in our case takes $O(N^2)$ (i.e. $m = 1$). This leads to a final complexity of $O(N^3)$ for applying MCD to N neighborhoods. These complexities are used below to determine the complexities of RWGPs.

4.1 Batch robust weighted GP

The computational complexity associated to training a batch standard GP model is $O(N^3)$ (Rasmussen and Williams 2006). In the case of our batch RWGP, we must add the complexities of constructing a k-d tree, determining the neighborhoods of all points in D and calculating the corresponding N weights. This leads to a combined computational complexity of $O(N^3)$ for processing the training data and calculating weights. The complexities of calculating our posterior RWGPs are not affected by the presence of weights in the likelihoods, therefore, the batch RWGP and the batch standard GP have the same computational complexity.

4.2 Online robust weighted GP

Training an online standard GP model on N data points also has a computational complexity of $O(N^3)$ (Csató and Opper 2002). In the case of online RWGP, a k-d tree should be constructed progressively by processing one observation at a time. Adding an observation to a k-d tree that already contains t observations has $O(\log(t+1))$ complexity. Consequently, iteratively building a k-d tree for N observations sums up to $O(\log(N!))$. Additionally, the complexity of determining the neighborhood of each observation $t+1$ is $O(\log(t+1))$ and it takes $O((t+1)^2)$ to obtain the MCD estimates. Summing up these complexities for N observations, we obtain that determining N neighborhoods takes $O(\log(N!))$ and the complexity of using the MCD estimator on N neighborhoods is $O(N^3)$. Given that these independent processing steps have a computational complexity that is not greater than that of online GP, the computational complexity of online RWGP is also $O(N^3)$.

5 Experimental setup

This section is divided in three parts. The first subsection describes the datasets used in our experiments. The second subsection describes how we determined the parameters of the data weigher and the hyperparameters of the standard and the proposed weighted GPs. The last subsection explains the procedure that we employed to compare the following methods: batch standard GP (batch GP), online GP and our corresponding weighted variants, the robust GP method based on Student-t (Jylänki et al. 2011) and the heteroscedastic GP named GPz (Almosallam et al. 2016).

5.1 Data sets

We designed four experiments. The first experiment used a 1-dimensional simulated dataset. The second used the "motorcycle" real-life dataset (Silverman 1985), which is also 1-dimensional. The third considered the real-life 2-dimensional dataset called "galaxy" (Buta 1987). The fourth experiment employed the "concrete" dataset (Yeh 1998), which has eight attributes. The following paragraphs briefly describe these datasets.

Simulated This is a simple dataset that we built based on the ground-truth function

$$\mathrm{y(x)} = \sin\left(\frac{1}{2}\mathrm{x}\right)\left(\frac{10\log(\mathrm{x}+2)}{\mathrm{x}}\right) + \frac{\mathrm{x}^2}{200}. \tag{31}$$

This function was randomly sampled at 60 points from the interval [0.25, 30], adding a Gaussian noise with variance equal to 0.1 to the corresponding y values.

Motorcycle This dataset was obtained from simulated motorcycle crashes that were carried out to assess the efficacy of helmets (Schmidt et al. 1981). The dataset contains 133 observations of accelerometer readings and their corresponding reading times. The regression problem consists of learning the relationship between time as the independent variable and acceleration as the dependent variable.

Galaxy This dataset contains 323 measurements of the radial velocities of a spiral galaxy, measured in km/s. The measurements were taken over seven lines drawn over the sky, all crossing at the origin of the galaxy. In our experiment we want to learn the relationship between the radial velocities and the two independent variables EastWest and NorthSouth, which describe the location of the readings.

Concrete This dataset contains 1030 observations that relate concrete compressive strength (MPa) of different mixtures of concrete to eight independent attributes. Seven attributes correspond to the amounts of mixture ingredients and the last attribute is the age of the mixture. The eight attributes were normalized by using their standard scores. This was done to achieve the same scale across dimensions, which facilitated determining effective values for the kernel parameters.

5.2 Data weigher parameters and GP hyperparameters

We implemented the standard and weighted variants of batch GP and online GP regression models in MATLAB. The well-known exponential kernel $k(\mathbf{x}, \mathbf{x}') = \mathrm{e}^{-\frac{1}{2}\sum_{i=1}^{l}\theta_i(\mathbf{x}_i - \mathbf{x}'_i)^2}$ was used in our experiments, where the kernel hyperparameters $\theta_i, i = 1, 2, \ldots, l$ are considered scale factors.

The employed robust data weigher used the quasi-robust weight function associated with the Welsh's ρ-function (Dennis and Welsch 1978), i.e., $w_Q(z) = (1-\gamma)e^{-\lambda z^2} + \gamma$, where λ is a scale parameter that was set to 1 in our case because its role is assumed by the robust variance v_i, as can be seen in (21).

Regarding the parameters of the weight functions, we wanted to employ values of r and s for each dataset that would maximize the use of the quasi-robust weights in our experiments. For that purpose, we set $s = 5$ after some preliminary experimentation

Table 1 Optimized hyperparameter values for batch standard GP and batch RWGP methods

Data set	Methods	
	Batch standard GP	Batch RWGP
Simulated	$\sigma^2 = 0.13309, a_1 = 0.11043$	$\sigma^2 = 0.058388, a_1 = 0.10792$
Motorcycle	$\sigma^2 = 0.2, a_1 = 0.03$	$\sigma^2 = 0.008, a_1 = 0.02$
Galaxy	$\sigma^2 = 0.01, a_1 = 0.005, a_2 = 0.001$	$\sigma^2 = 0.007, a_1 = 0.005, a_2 = 0.008$
Concrete	$\sigma^2 = 0.007, a_1 = \cdots = a_8 = 0.6$	$\sigma^2 = 0.002, a_1 \ldots a_8 = 0.5$

across all datasets, expecting that a small s value would keep a local approach. For each dataset, we determined r by following a two-step procedure. First, we obtained a set S that contains the Mahalanobis distances from each observation to the farthest of its s nearest neighbors. Second, we set r equal to a value slightly greater than the maximum element of S. Following this approach, the r values corresponding to the first, second, third and fourth datasets were 0.25, 0.5, 1 and 4, respectively. The parameter γ was set to 0.005 for all experiments, as we were able to obtain good results with this value.

The hyperparameters for the batch standard GPs and batch RWGPs (i.e. σ^2 and $\boldsymbol{\theta}_k$) were estimated initially using the MML method on all datasets. However, this method only provided effective hyperparameter values for the simulated dataset of the first experiment (in Sect. 7 we give some insight into why the MML method failed in general). For the other three datasets the hyperparameter values obtained by the MML method led to inadequate predicted posterior means. This was evidenced by relatively high RMSE values and visualizations of the corresponding posterior means that were just a flat line for the Motorcycle and Galaxy datasets. Consequently, we estimated GP hyperparameters for the last three experiments by using five runs of tenfold cross-validation (CV), minimizing the root mean squared errors (RMSE). Each hyperparameter took values in following range: $[0.001 : 0.001 : 0.01,\ 0.02 : 0.01 : 0.1,\ 0.2 : 0.1 : 1,\ 2 : 1 : 10]$.

Estimating hyperparameters for online GPs is a difficult problem, as a critical mass of training data is generally needed to obtain reliable estimates. Therefore, for online GP and online RWGP we employed the values of hyperparameters estimated for the corresponding batch GP method. Table 1 lists the hyperparameter values estimated for each dataset for batch standard GP and batch RWGP.

5.3 Model comparison

Having estimated GP hyperparameters, we implemented a tenfold CV approach to obtain RMSE values for the GPs under comparison. The CV was run 30 times on each GP, in order to obtain 30 RMSE values for each model. Here we followed a well-accepted convention, that it is reasonable to expect that the average RMSE values would be approximately normally distributed.

The experimental setup just described was applied to the datasets and also to various versions of the datasets that were obtained by randomly contaminating the original data with outliers. To obtain the contaminated versions of each dataset, we randomly selected the following percentages of observations and modified them in order to make them outliers: 5%, 10%, 15% and 20%. Specifically, for each observation ($\mathbf{x}$, y) we randomly subtracted or added to its original y value a random number between three and five times the standard deviation of the y values in the original dataset. The different types of GPs where trained on the contaminated versions of the datasets using the same hyperparameter values that were estimated for the corresponding uncontaminated versions of the data.

The contamination process described above was applied to the original datasets before executing each of the 30 tenfold CV runs. Specifically, the models were compared under exactly the same conditions within each CV run, including the same original or contaminated datasets and exactly the same CV folds. This allowed us to compare the RMSE values of the GP models at each contamination level (including the original data that correspond to 0% contamination) by employing one-way repeated measures ANOVA (Girden 1992). The RMSE values were computed using the y values of the corresponding original datasets as ground-truth. For each dataset (contaminated or not) the CV runs are seen as subjects and the different GPs constitute the within-subjects factors. This approach was used to compare the following models: batch standard GP to batch RWGP; online GP to online RWGP; and GPz to batch RWGP. The significance level α for all ANOVAs was set to 0.01 to compensate for the multiple inferences.

We were unable to use one-way repeated measures ANOVA to compare the Student-*t* GP (Jylänki et al. 2011) implemented in GPStuff to our batch RWGP due to multiple convergence failures experienced by GPStuff. However, we obtained RMSE values for GPStuff by separately running the contamination and CV procedures explained above, discarding all cases of failed convergence, until we reached the desired 30 RMSE values per contamination level corresponding to good hyperparameter estimates. Hence, we resorted to standard two-way ANOVA to compare the performance of Student-*t* GP to our RWGP, also using 0.01 as the significance level.

6 Experimental results

This section summarizes the results from the ANOVAs for each dataset and each contamination level. The results from our experiments are shown in Tables 2, 3, 4 and 5, where average RMSE values have been rounded to three significant digits. The tables include the *p* values that show whether differences in RMSE were significant or not.

Table 2 shows the results for the simulated dataset. In the case of 0% outlier contamination, our batch RWGP performed significantly better than GPz and our online RWGP showed the same performance than online GP. In contrast, batch GP and Student-*t* GP significantly outperformed our batch RWGP in the case of data without outlier contamination. For contamination levels of 5%, 10% and 15% our RWGPs outperformed all other methods, and this result was statistically significant in all cases except for Student-*t* GP. Finally, for 20% contamination, RWGPs significantly outperformed other models except Student-*t* GP. However, the observed better performance of Student-*t* GP when compared to our method was not statistically significant.

Table 2 Results of ANOVAs for different contamination levels; significance level $\alpha = 0.01$. Experiment 1 (simulated)

Contamination (%)	Method1	Method2	Avg RMSE Method 1	Avg RMSE Method 2	AvgErrorDiff	*p* value
0	Batch GP	Batch RWGP	0.399	0.415	− 0.016	< 0.001
0	Online GP	Online RWGP	0.399	0.399	0.000	0.896
0	GPz	Batch RWGP	0.642	0.415	0.227	< 0.001
0	StudentT-EP	Batch RWGP	0.390	0.415	− 0.025	< 0.001
5	Batch GP	Batch RWGP	0.974	0.536	0.438	< 0.001
5	Online GP	Online RWGP	0.974	0.757	0.217	< 0.001
5	GPz	Batch RWGP	1.310	0.536	0.774	< 0.001
5	StudentT-EP	Batch RWGP	1.585	0.536	1.049	0.193
10	Batch GP	Batch RWGP	1.347	0.644	0.703	< 0.001
10	Online GP	Online RWGP	1.347	1.121	0.226	< 0.001
10	GPz	Batch RWGP	1.822	0.644	1.178	< 0.001
10	StudentT-EP	Batch RWGP	2.690	0.644	2.046	0.112
15	Batch GP	Batch RWGP	1.593	0.799	0.794	< 0.001
15	Online GP	Online RWGP	1.593	1.374	0.219	< 0.001
15	GPz	Batch RWGP	2.096	0.799	1.297	< 0.001
15	StudentT-EP	Batch RWGP	2.391	0.799	1.592	0.297
20	Batch GP	Batch RWGP	1.816	1.000	0.816	< 0.001
20	Online GP	Online RWGP	1.816	1.561	0.255	< 0.001
20	GPz	Batch RWGP	2.208	1.000	1.208	< 0.001
20	StudentT-EP	Batch RWGP	0.783	1.000	− 0.217	0.221

Table 3 Results of ANOVAs for different contamination levels; significance level $\alpha = 0.01$. Experiment 2 (motorcycle)

Contamination (%)	Method 1	Method 2	Avg RMSE Method 1	Avg RMSE Method 2	AvgErrorDiff	*p* value
0	Batch GP	Batch RWGP	23.160	24.177	− 1.017	< 0.001
0	Online GP	Online RWGP	23.160	26.486	− 3.326	< 0.001
0	GPz	Batch RWGP	26.777	24.177	2.600	< 0.001
0	StudentT-EP	Batch RWGP	24.747	24.177	0.570	0.061
5	Batch GP	Batch RWGP	26.287	24.428	1.859	< 0.001
5	Online GP	Online RWGP	26.287	27.546	− 1.259	< 0.001
5	GPz	Batch RWGP	39.018	24.428	14.590	< 0.001
5	StudentT-EP	Batch RWGP	25.482	24.428	1.054	0.012
10	Batch GP	Batch RWGP	29.164	24.810	4.354	< 0.001
10	Online GP	Online RWGP	29.164	28.603	0.561	0.070
10	GPz	Batch RWGP	41.275	24.810	16.465	< 0.001
10	StudentT-EP	Batch RWGP	24.807	24.810	− 0.003	0.996
15	Batch GP	Batch RWGP	30.489	24.792	5.697	< 0.001
15	Online GP	Online RWGP	30.489	29.582	0.907	0.078
15	GPz	Batch RWGP	43.529	24.792	18.737	< 0.001
15	StudentT-EP	Batch RWGP	24.851	24.792	0.059	0.878
20	Batch GP	Batch RWGP	33.290	25.010	8.280	< 0.001
20	Online GP	Online RWGP	33.290	30.926	2.364	< 0.001
20	GPz	Batch RWGP	69.649	25.010	44.639	0.063
20	StudentT-EP	Batch RWGP	25.440	25.010	0.430	0.450

Table 4 Results of ANOVAs for different contamination levels; significance level $\alpha = 0.01$. Experiment 3 (galaxy)

Contamination(%)	Method 1	Method 2	Avg RMSE Method 1	Avg RMSE Method 2	AvgErrorDiff	*p* value
0	Batch GP	Batch RWGP	14.026	13.871	0.155	< 0.001
0	Online GP	Online RWGP	14.026	14.279	-0.253	< 0.001
0	GPz	Batch RWGP	16.749	13.871	2.878	< 0.001
0	StudentT-EP	Batch RWGP	94.115	13.871	80.244	< 0.001
5	Batch GP	Batch RWGP	30.560	14.205	16.355	< 0.001
5	Online GP	Online RWGP	30.560	23.490	7.070	< 0.001
5	GPz	Batch RWGP	28.884	14.205	14.679	< 0.001
5	StudentT-EP	Batch RWGP	22.039	14.205	7.834	0.002
10	Batch GP	Batch RWGP	41.011	16.012	24.999	< 0.001
10	Online GP	Online RWGP	41.011	29.844	11.166	< 0.001
10	GPz	Batch RWGP	41.720	16.012	25.708	< 0.001
10	StudentT-EP	Batch RWGP	16.286	16.012	0.274	0.836
15	Batch GP	Batch RWGP	49.974	17.038	32.936	< 0.001
15	Online GP	Online RWGP	49.974	36.899	13.075	< 0.001
15	GPz	Batch RWGP	48.092	17.038	31.054	< 0.001
15	StudentT-EP	Batch RWGP	18.044	17.038	1.006	0.451
20	Batch GP	Batch RWGP	54.355	21.912	32.443	< 0.001
20	Online GP	Online RWGP	54.355	42.905	11.450	< 0.001
20	GPz	Batch RWGP	49.926	21.912	28.014	< 0.001
20	StudentT-EP	Batch RWGP	24.621	21.912	2.709	0.228

Table 5 Results of ANOVAs for different contamination levels; significance level $\alpha = 0.01$. Experiment 4 (concrete)

Contamination (%)	Method 1	Method 2	Avg RMSE Method 1	Avg RMSE Method 2	AvgErrorDiff	p value
0	Batch GP	Batch RWGP	5.222	5.340	−0.118	< 0.001
0	Online GP	Online RWGP	5.222	5.408	−0.186	< 0.001
0	GPz	Batch RWGP	5.912	5.340	0.572	< 0.001
0	StudentT-EP	Batch RWGP	NA	5.340	NA	NA
5	Batch GP	Batch RWGP	14.307	5.969	8.338	< 0.001
5	Online GP	Online RWGP	14.307	6.461	7.846	< 0.001
5	GPz	Batch RWGP	7.691	5.969	1.722	< 0.001
5	StudentT-EP	Batch RWGP	5.899	5.969	−0.070	0.604
10	Batch GP	Batch RWGP	19.843	6.804	13.039	< 0.001
10	Online GP	Online RWGP	19.843	8.131	11.712	< 0.001
10	GPz	Batch RWGP	8.922	6.804	2.118	< 0.001
10	StudentT-EP	Batch RWGP	5.597	6.804	−1.207	< 0.001
15	Batch GP	Batch RWGP	24.148	7.439	16.709	< 0.001
15	Online GP	Online RWGP	24.148	8.936	15.212	< 0.001
15	GPz	Batch RWGP	10.013	7.439	2.574	< 0.001
15	StudentT-EP	Batch RWGP	5.925	7.439	−1.514	< 0.001
20	Batch GP	Batch RWGP	27.737	8.832	18.905	< 0.001
20	Online GP	Online RWGP	27.737	11.716	16.021	< 0.001
20	GPz	Batch RWGP	11.249	8.832	2.417	< 0.001
20	StudentT-EP	Batch RWGP	10.699	8.832	1.867	0.653

The experimental results for the motorcycle dataset are shown in Table 3. As in the first experiment, the standard GPs significantly outperformed RWGPs on the original data. However, RWGPs outperformed GPz and Student-t GP in this case, significantly only when compared with GPz. For 5% contamination, RWGPs outperformed all models except online GP, although this result was not significant for Student-t GP. For 10% contamination, RWGPs outperformed all models except Student-t GP, significantly in the case of GPz and batch GP. For Student-t GP the average RMSE values were almost identical. For contaminations of 15% and 20% RWGPs outperformed all other models, significantly in half of the cases.

The results for the galaxy data are listed in Table 4. When compared on the original dataset, RWGPs significantly outperformed other models except for online GP, which significantly outperformed online RWGP. Our RWGPs outperformed all other models for the rest of contamination levels. This result was significant in all cases except for Student-t GP for contaminations 10%, 15% and 20%.

Table 5 shows results for concrete data. For the original dataset, the standard GPs significantly outperformed RWGPs, while RWGP outperformed GPz. We were unable to obtain hyperparameter values for Student-t GP using GPStuff for the uncontaminated data due to repeated failures to converge. For other contamination levels, RWGPs significantly outperformed standard GPs and GPz. Student-t GP outperformed RWGP for 5%, 10% and 15% contaminations, significantly in the cases of 10% and 15%. In the case of 20% contamination, RWGP outperformed Student-t GP although not significantly.

Taking statistics for all tables and contamination levels from 5% onwards, we note that our proposed method outperformed the other methods 58 times out of 64. From those 58 cases, 45 were statistically significant. Specifically for positive levels of contamination:

– RWGPs outperformed the corresponding batch and online standard GPs in 31 out of 32 comparisons. This result was statistically significant in 29 of the 31 cases.
– RWGPs outperformed Student-t GP in 11 out of 16 cases. Of those 11 cases, only one is statistically significant. Of the 5 cases in which Student-t GP outperformed our method, only 2 are significant.
– RWGPs always outperformed GPz and this outcome was statistically significant in all cases but one.

Looking at regressions on data without contamination, the results show a slightly better average RMSE values in favor of the standard GPs. Our proposed method outperformed Student-t GP in 2 out 3 cases with only one being significant. On the other hand Student-t GP outperformed RWGP significantly in one case. Our proposed method significantly outperformed GPz on all datasets without contamination.

In summary, the benefits of using RWGPs on data that contain outliers have been clearly stated through the statistical analysis of our experimental results, from simple one-dimensional datasets to real-life multidimensional data such as the "concrete" dataset.

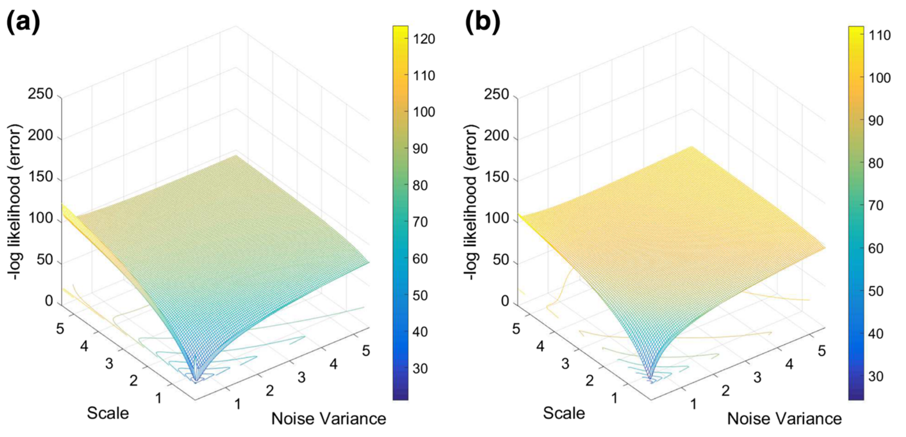

Fig. 2 Optimization surfaces for the first experiment (simulated data). **a** Batch standard GP. **b** Batch RWGP

7 On hyperparameter estimation

This section explores why the MML method gave us ineffective GP hyperparameter estimates in general, which led us to follow a cross-validation approach. Here we plot the surfaces of the negative log marginal likelihood functions corresponding to the simulated and motorcycle datasets, to contrast the effective parameter estimation using MML on the first dataset to the failure of the MML method on the second dataset.

Figure 2a, b show the optimization surfaces for the simulated data within a neighborhood of our GP hyperparameter estimates, for the batch GP and batch RWGP models respectively. In both cases there is only one minimum within the chosen neighborhood, which correspond to the solutions that we found. In general, the log marginal likelihood might have multiple local optima when using exponential kernels; see Rasmussen and Williams (2006), Sec. 5.4.1. Although the existence of a single optimum in our first experiment cannot be fully guaranteed, we repeated the MML estimation procedure 100 times, starting from different points chosen uniformly within the neighborhood plotted in Fig. 2, and we consistently obtained the same estimates. Additionally, we expanded the plots in Fig. 2 to larger neighborhoods, and we again obtained the same unique minimum in all cases.

In the case of the motorcycle data, Fig. 3a, b show the optimization surfaces for the batch GP and batch RWGP models, respectively. In this case both surfaces seem to lack a global minimum within a region that should contain optimal values for the hyperparameters. Figure 3 confirms that the MML method cannot be applied successfully in this case, regardless of the optimization method of choice.

For the sake of completeness, Figs. 4 and 5 show the surfaces of average RMSE values for the hyperparameters values employed by the CV procedure for the simulated and motorcycle datasets. The existence of a minimum average RMSE value within each optimization area was apparent in the two cases.

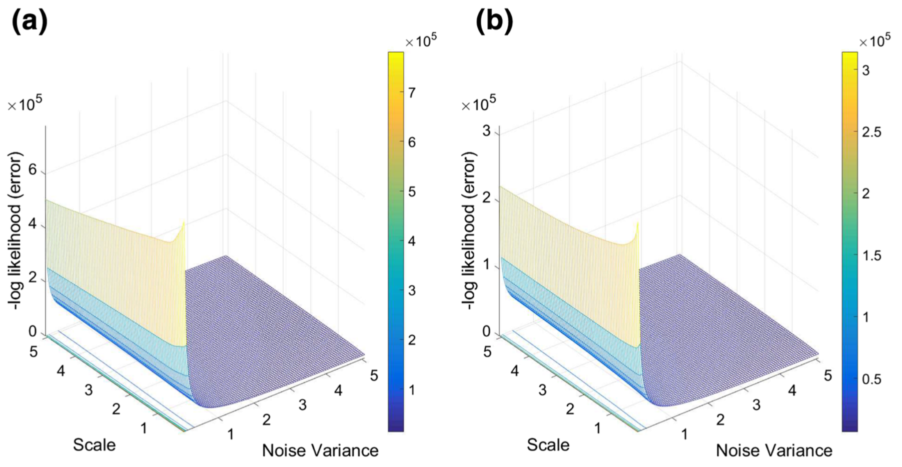

Fig. 3 Optimization surfaces for the second experiment (motorcycle). **a** Batch standard GP. **b** Batch RWGP. Both surfaces show regions that suggest the lack of minimum

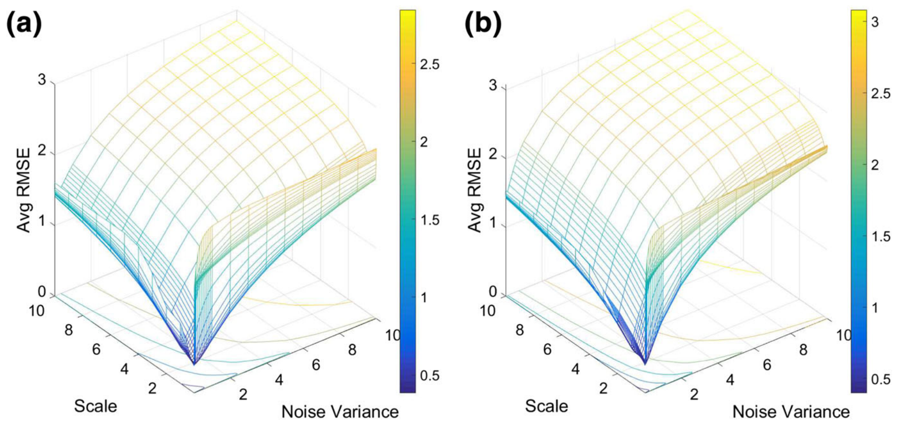

Fig. 4 Surfaces of average RMSE for the first experiment (simulated data). **a** Batch standard GP. **b** Batch RWGP

8 Conclusion

This work has introduced batch and online robust weighted GPs (RWGPs) by using a weighted version of the standard Gaussian likelihood. Weights are calculated relying on either a robust weight function or a quasi-robust weight function. We have presented the mathematical expressions for batch RWGPs and online RWGPs. Remarkably, the computational complexity of our RWGPs are the same as that of the corresponding standard GPs. Robustness is achieved without applying complex approximation techniques that have been proposed in previous approaches.

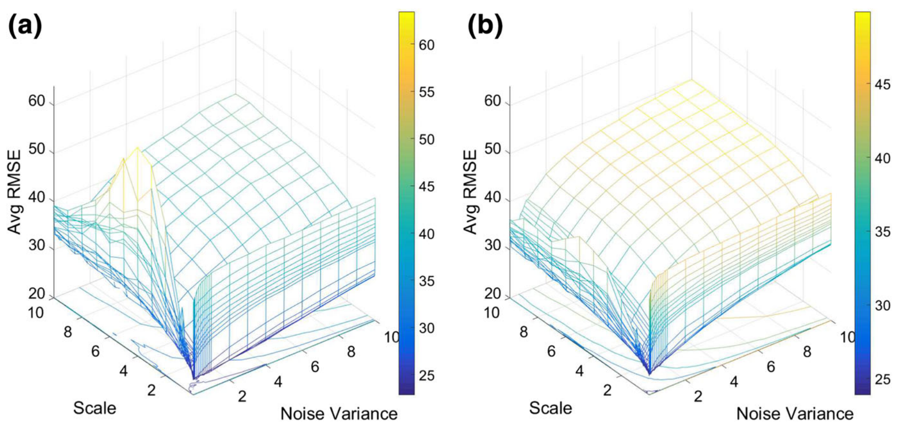

Fig. 5 Surfaces of average RMSE for the second experiment (motorcycle). **a** Batch standard GP. **b** Batch RWGP

The experimental sections show that our methods can outperform standard GPs and a very recent heteroscedastic GP model when trained in data that contain outliers. Additionally, our methods are shown to provide results that are comparable to and sometimes better than a state-of-the-art robust GP that uses Student-*t* likelihood.

Effective estimation of the parameters r and s and γ to calculate the robust weights is difficult in general. As future research, we will explore automatically estimating these parameters, using the Bayesian framework, and how sensitive RWGPs are to variations of these parameters. Similarly, we would like to include an effective approach to automatically estimate the GP hyperparameters by means of defining appropriate priors and applying efficient Bayesian estimation methods. Finally, the relation between percentage of outliers and input dimensions is also an interesting question worth of exploration in future work.

Appendix: Estimation of weighted GP hyperparameters

Here we derive the expressions for the derivatives of the log marginal likelihood of the batch RWGP with respect to the GP hyperparameters $\boldsymbol{\theta} = (\theta_0, \boldsymbol{\theta}_k)^T = (\sigma^2, \boldsymbol{\theta}_k)^T$, where $\boldsymbol{\theta}_k = (\theta_1, \theta_2, \ldots, \theta_l)$, with $l \geq 0$ (i.e. $\boldsymbol{\theta}_k$ might be empty), denotes the kernel hyperparameters. Given the training set D, the MML method consists of finding the hyperparameter values that maximize the log marginal likelihood given by (25):

$$\mathcal{L}(\boldsymbol{\theta}) = -\frac{1}{2}(\mathbf{y} - \mathbb{E}_0\left[\mathbf{f}_D\right])^T \boldsymbol{K}_p^{-1}\left(\mathbf{y} - \mathbb{E}_0\left[\mathbf{f}_D\right]\right),$$

where $\boldsymbol{K}_p = \boldsymbol{K}_D + \boldsymbol{W}$. The derivatives of $\mathcal{L}$ with respect to each $\theta_i, \ i = 0, 1, \dots, l$ are written as:

$$\frac{\partial \mathcal{L}}{\partial \theta_i} = \frac{1}{2}(\mathbf{y} - \mathbb{E}_0\left[\mathbf{f}_D\right])^T \boldsymbol{K}_p^{-1} \frac{\partial \boldsymbol{K}_p}{\partial \theta_i} \boldsymbol{K}_p^{-1} (\mathbf{y} - \mathbb{E}_0\left[\mathbf{f}_D\right]) - \frac{1}{2}\mathrm{tr}\left(\boldsymbol{K}_p^{-1} \frac{\partial \boldsymbol{K}_p}{\partial \theta_i}\right), \tag{32}$$

where the derivatives of $\boldsymbol{K}_p$ are as follows:

$$\frac{\partial \boldsymbol{K}_p}{\partial \sigma^2} = \frac{\partial \mathbf{W}}{\partial \sigma^2} = \mathrm{diag}\left(\frac{1}{\mathrm{w}_1}, \frac{1}{\mathrm{w}_2}, \dots, \frac{1}{\mathrm{w}_N}\right), \tag{33}$$

$$\frac{\partial \boldsymbol{K}_p}{\partial \theta_i} = \frac{\partial \boldsymbol{K}_D}{\partial \theta_i}, \quad i = 1, 2, \dots, l. \tag{34}$$

References

Agostinelli C, Greco L (2013) A weighted strategy to handle likelihood uncertainty in Bayesian inference. Comput Stat 28:319–339

Almosallam IA, Jarvis MJ, Roberts SJ (2016) GPz: non-stationary sparse Gaussian processes for heteroscedastic uncertainty estimation in photometric redshifts. Mon Not R Astron Soc 462(1):726–739

Bentley JL (1980) Multidimensional divide and conquer. Commun ACM 23(4):214–229

Bernholt T, Fischer P (2004) The complexity of computing the MCD-estimator. Theor Comput Sci 326:383–398

Bishop CM (2006) Pattern recognition and machine learning. Springer, New York

Buta R (1987) The structure and dynamics of ringed galaxies, III: surface photometry and kinematics of the ringed nonbarred spiral NGC7531. Astrophys J Suppl Ser 64:1–37

Csató L (2002) Gaussian processes—iterative sparse approximations. PhD thesis. Aston University, Birmingham, UK. http://publications.aston.ac.uk/id/eprint/1327/

Csató L, Opper M (2002) Sparse on-line Gaussian processes. Neural Comput 14(3):641–668

de Boor CA (2001) Practical guide to splines. Applied mathematical sciences, vol 27, revised edn. Springer, New York

Dennis JE Jr, Welsch RE (1978) Techniques for nonlinear least squares and robust regression. Commun Stat Simul Comput 7(4):345–359

Drucker H, Burges CJ, Kaufman L, Smola AJ, Vapnik V (1997) Support vector regression machines. Advances in neural information processing systems, pp 155–161

Geweke J (1993) Bayesian treatment of the independent Student-t linear model. J Appl Econom 8(S1):S19–S40

Girden ER (1992) ANOVA: repeated measures. Sage University Paper Series on Quantitative Applications in the Social Sciences 07-084, Sage University, Newbury Park, CA

Greco L, Racugno W, Ventura L (2008) Robust likelihood functions in Bayesian analysis. J Stat Plan Inference 138(5):1258–1270

Hampel FR, Ronchetti EM, Rousseeuw PJ, Stahel WA (1986) Robust statistics. The approach based on influence functions. Wiley, New York

Huber PJ (1964) Robust estimation of a location parameter. Ann Stat 53(1):73–101

Huber PJ, Ronchetti EM (1981) Robust statistics. Wiley, New York

Jylänki P, Vanhatalo J, Vehtari A (2011) Robust Gaussian process regression with a student-t likelihood. J Mach Learn Res 12:3227–3257

Kemmler M, Rodner E, Denzler J (2010) One-class classification with Gaussian processes. In: Proceedings of the Asian conference on computer vision. Lecture notes in computer science, vol 6493. Springer, pp 489–500

Kuss M (2006) Gaussian process models for robust regression, classification, and reinforcement learning. Doctoral dissertation, Technische Universität Darmstadt, Germany. http://hdl.handle.net/11858/00-001M-0000-0013-D2CD-C

Kuss M, Pfingsten T, CsatóL, Rasmussen CE (2005) Approximate inference for robust Gaussian process regression. Max Planck Institute for Biological Cybernetics, Tübingen, Germany, Technical Report 136. http://hdl.handle.net/11858/00-001M-0000-0013-D703-4

Le QV, Smola AJ, Canu S (2005) Heteroscedastic Gaussian process regression. In: Proceedings of the 22nd international conference on machine learning. ACM, pp 489–496

MacKay DJC (1998) Introduction to Gaussian processes. In: Bishop CM (ed) Neural networks and machine learning. NATO ASI series, vol 168. Springer, Berlin, pp 133–165

Mahalanobis PC (1936) On the generalised distance in statistics. Proc Natl Inst Sci India 2(1):49–55

Maronna RA, Martin DR, Yohai VJ (2006) Robust statistics: theory and methods. Wiley, Chichester

Mattos CLC, Santos JDA, Barreto GA (2015) An empirical evaluation of robust Gaussian process models for system identification. In: International conference on intelligent data engineering and automated learning. Springer, Cham, pp 172–180

Minka TP (2001) Expectation propagation for approximate Bayesian Inference. In: Proceedings of the seventeenth conference on uncertainty in artificial intelligence. Morgan Kaufmann Publishers Inc., pp 362–369

Murphy L, Martin S, Corke P (2012) Creating and using probabilistic cost maps from vehicle experience. In: Proceedings of IEEE/RSJ international conference on intelligent robots and systems, intelligent robots and systems (IROS). IEEE, pp 4689–4694

Neal RM (1997) Monte Carlo implementation of Gaussian process models for Bayesian regression and classification. Technical Report 9702, Department of Statistics and Department of Computer Science, University of Toronto. arXiv:physics/9701026

Opper M (1998) A Bayesian approach to on-line learning. In: Saad D (ed) On-line learning in neural networks. Cambridge University Press, Cambridge

Ramirez-Padron R (2015) Batch and online implicit weighted Gaussian processes for robust Novelty detection. Doctoral dissertation, University of Central Florida. http://purl.fcla.edu/fcla/etd/CFE0005869

Ramirez-Padron R, Mederos B, Gonzalez AJ (2013) Novelty detection using sparse online Gaussian processes for visual object recognition. In: International FLAIRS conference. St. Pete Beach, FL, USA, pp 124–129

Ranjan R, Huang B, Fatehi A (2016) Robust Gaussian process modeling using EM algorithm. J Process Control 42:125–136

Rasmussen CE, Williams C (2006) Gaussian processes for machine learning. MIT Press, Cambridge

Rey WJJ (1983) Introduction to robust and quasi-robust statistical methods. Springer, Berlin

Rottmann A, Burgard W (2010) Learning non-stationary system dynamics online using Gaussian processes. In: Proceedings of 32nd DAGM symposium, Darmstadt, Germany, pp 192–201

Rousseeuw PJ (1984) Least median of squares regression. J Am Stat Assoc 79(388):871–880

Schmidt G, Mattern R, Schüler F (1981) Biomechanical investigation to determine physical and traumatological differentiation criteria for the maximum load capacity of head and vertebral column with and without protective helmet under effects of impact. EEC Research Program on Biomechanics of Impacts, Final Report Phase III, Project 65, Institut für Rechtsmedizin, Universität Heidelberg, Germany

Seeger M (2004) Gaussian processes for machine learning. Int J Neural Syst 14(2):69–106

Shawe-Taylor J, Cristianini N (2004) Kernel methods for pattern analysis. Cambridge University Press, Cambridge

Silverman BW (1985) Some aspects of the spline smoothing approach to non-parametric curve fitting. J R Stat Soc Ser B (Methodol) 47(1):1–52

Sugiyama M, Krauledat M, Müller KR (2007) Covariate shift adaptation by importance weighted cross validation. J Mach Learn Res 8:985–1005

Tipping ME (2000) The relevance vector machine. In: Advances in neural information processing systems, pp 652–658

Tipping ME, Lawrence ND (2005) Variational inference for Student-t models: robust Bayesian interpolation and generalised component analysis. Neurocomputing 69(1–3):123–141

Verboven S, Hubert M (2005) LIBRA: a MATLAB Library for robust analysis. Chemometr Intell Lab Syst 75:127–136

Wald I, Havran V (2006) On building fast kd-trees for ray tracing, and on doing that in O(N log N). In: Proceedings of the 2006 IEEE symposium on interactive ray tracing, pp 61–69

Wang B, Mao Z (2019) Outlier detection based on Gaussian process with application to industrial processes. Appl Soft Comput J 76:505–516

West M (1984) Outlier models and prior distributions in Bayesian linear regression. J R Stat Soc (Ser B) 46(3):431–439

Williams CKI, Barber D (1998) Bayesian classification with Gaussian processes. IEEE Trans Pattern Anal Mach Intell 12(20):1342–1351

Yeh C (1998) Modeling of strength of high performance concrete using artificial neural networks. Cem Concr Res 28(12):1797–1808