# Faster variational inducing input Gaussian process classification

***P. A. Izmailov and D. A. Kropotov***
izmailovpavel@gmail.com; dmitry.kropotov@gmail.com
Lomonosov Moscow State University, 1 Leninskie Gory, Moscow, Russia

**Background**: Gaussian processes (GP) provide an elegant and effective approach to learning in kernel machines. This approach leads to a highly interpretable model and allows using the Bayesian framework for model adaptation and incorporating the prior knowledge about the problem. The GP framework is successfully applied to regression, classification, and dimensionality reduction problems. Unfortunately, the standard methods for both GP-regression and GP-classification scale as $\mathcal{O}(n^3)$, where $n$ is the size of the dataset, which makes them inapplicable to big data problems. A variety of methods have been proposed to overcome this limitation both for regression and classification problems. The most successful recent methods are based on the concept of inducing inputs. These methods reduce the computational complexity to $\mathcal{O}(nm^2)$ where $m$ is the number of inducing inputs with $m$ typically much less than $n$. The present authors focus on classification. The current state-of-the-art method for this problem is based on stochastic optimization of an evidence lower bound (ELBO) that depends on $\mathcal{O}(m^2)$ parameters. For complex problems, the required number of inducing points $m$ is fairly big, making the optimization in this method challenging.
**Methods**: The structure of variational lower bound that appears in inducing input GP classification has been analyzed. First, it has been noted that using quadratic approximation of several terms in this bound, it is possible to obtain analytical expressions for optimal values of most of the optimization parameters, thus sufficiently reducing the dimension of optimization space. Then, two methods have been provided for constructing necessary quadratic approximations: one is based on Jaakkola–Jordan bound for logistic function and the other is derived using Taylor expansion.
**Results**: Two new variational lower bounds have been proposed for inducing input GP classification that depend on a number of parameters. Then, several methods have been suggested for optimization of these bounds and the resulting algorithms have been compared with the state-of-the-art approach based on stochastic optimization. Experiments on a bunch of classification datasets show that the new methods perform the same or better results than the existing one. However, new methods do not require any tunable parameters and can work in settings within a big range of $n$ and $m$ values, thus significantly simplifying training of GP classification models.
**Keywords**: *Gaussian process; classification; variational inference; big data; inducing inputs; optimization; variational lower bound*

**DOI:** 10.21469/22233792.3.1.02

## 1 Introduction

Gaussian processes provide a prior over functions and allow finding complex regularities in data. Gaussian processes are successfully used for classification/regression problems and dimensionality reduction [1]. In this work, only the classification problem is considered.

Standard methods for GP-classification scale as $\mathcal{O}(n^3)$ where $n$ is the size of the training dataset. This complexity makes them inapplicable to big data problems. Therefore, a variety of methods were introduced to overcome these limitations [2–4]. The focus of the paper is on the methods based on so-called inducing inputs. Paper [5] introduces the inducing inputs

approach for training GP models for regression. This approach is based on variational inference and proposes a particular lower bound for marginal likelihood (evidence). This bound is then maximized with regard to parameters of kernel function of the GP, thus fitting the model to data. The computational complexity of this method is $\mathcal{O}(nm^2)$ where $m$ is the number of inducing inputs used by the model and is assumed to be substantially smaller than $n$. Paper [6] develops these ideas by showing how to apply stochastic optimization to the ELBO similar to the one used in [5]. However, a new lower bound depends on $\mathcal{O}(m^2)$ variational parameters that makes optimization in the case of big $m$ challenging.

Paper [7] shows how to apply the approach from [6] to the GP-classification problem. It provides a lower bound that can be optimized with regard to kernel parameters and variational parameters using stochastic optimization. However, the lower bound derived in [7] is intractable and has to be approximated via Gauss–Hermite quadratures or other integral approximation techniques. This lower bound is also fit for stochastic optimization and depends on $\mathcal{O}(m^2)$ parameters.

In this work, a new approach was developed for training inducing input GP models for classification problems. Here, a structure of variational lower bound from [7] was analyzed. It has been noted that using quadratic approximation of several terms in this bounds, it is possible to obtain analytical expressions for optimal values of the most of optimization parameters, thus sufficiently reducing the dimension of optimization space. So, two methods have been provided for constructing necessary quadratic approximations: one based on Jaakkola–Jordan bound for logistic function and the other derived using Taylor expansion.

The paper is organized as follows. In section 2, the standard GP-classification framework and its main limitations are described. In section 3, the concept of inducing inputs is introduced and the ELBO of [7] is derived. Section 4 contains the main contribution — two new tractable ELBO and different methods for their optimization. Section 5 provides experimental comparison of new methods with the existing approach from [7], The last section concludes the paper.

## 2 Gaussian processes classification model

In this section, classic GP framework and its application for classification problems (for detailed discussion, see [1]) are reviewed.

### 2.1 Gaussian process definition

A GP is a collection of random variables, any finite number of which has a joint Gaussian distribution.

Here, only the processes that take place in a finite-dimensional real space $\mathbb{R}^d$ are considered. In this case, $f$ is the GP if for any $k$, for any $\boldsymbol{t}_1, \ldots, \boldsymbol{t}_k \in \mathbb{R}^d$, the joint distribution

$$(f(\boldsymbol{t}_1), \ldots, f(\boldsymbol{t}_k))^\mathsf{T} \sim \mathcal{N}(\boldsymbol{m}_t, \boldsymbol{K}_t)$$

for some $\boldsymbol{m}_t \in \mathbb{R}^k$ and $\boldsymbol{K}_t \in \mathbb{R}^{k\times k}$.

Mean $\boldsymbol{m}_t$ of this distribution is defined by the mean function $m : \mathbb{R}^d \to \mathbb{R}$ of the GP:

$$\boldsymbol{m}_t = (m(\boldsymbol{t}_1), \ldots, m(\boldsymbol{t}_k))^\mathsf{T}.$$

Similarly, the covariance matrix $\boldsymbol{K}_t$ is defined by the covariance function $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$:

$$\boldsymbol{K}_t = \begin{pmatrix} k(\boldsymbol{t}_1, \boldsymbol{t}_1) & k(\boldsymbol{t}_1, \boldsymbol{t}_2) & \cdots & k(\boldsymbol{t}_1, \boldsymbol{t}_n) \\ k(\boldsymbol{t}_2, \boldsymbol{t}_1) & k(\boldsymbol{t}_2, \boldsymbol{t}_2) & \cdots & k(\boldsymbol{t}_2, \boldsymbol{t}_n) \\ \vdots & \vdots & \vdots & \vdots \\ k(\boldsymbol{t}_n, \boldsymbol{t}_1) & k(\boldsymbol{t}_n, \boldsymbol{t}_2) & \cdots & k(\boldsymbol{t}_n, \boldsymbol{t}_n) \end{pmatrix}. \tag{1}$$

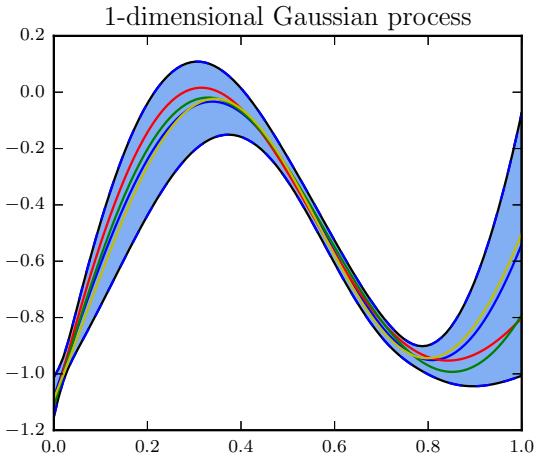


**Figure 1** One-dimensional GP

Then, it is straightforward that a GP is completely defined by its mean and covariance functions. Let us use the following notation:

$$f \sim \mathcal{GP}(m(\cdot), k(\cdot,\cdot)) \,.$$

While the mean function $m$ can be an arbitrary real-valued function, the covariance function $k$ has to be a kernel, so that the covariance matrices (1) it implies are symmetric and positive definite.

Figure 1 shows an example of a one-dimensional GP. The dark blue line is the mean function of the process, the light blue region is the $3\sigma$-region, and different color curves are the samples from the process.

### 2.2 Gaussian process classification

Now, let us apply GP to a binary classification problem. Suppose, one has a dataset $\{(\boldsymbol{x}_i, y_i) \,|\, i = = 1, \ldots, n\}$ where $\boldsymbol{x}_i \in \mathbb{R}^d$, $y_i \in \{-1, 1\}$. Denote the matrix comprised of points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ by $\boldsymbol{X} \in \mathbb{R}^{n \times d}$ and the vector of corresponding class labels $y_1, \ldots, y_n$ by $\boldsymbol{y} \in \{-1, 1\}^n$. The task is to predict the class label $y_* \in \{-1, 1\}$ at a new point $\boldsymbol{x}_* \in \mathbb{R}^d$.

Let us consider the following model. First, let us introduce a latent function $f : \mathbb{R}^d \to \mathbb{R}$ and put a zero-mean GP prior over it:

$$f \sim \mathcal{GP}(0, k(\cdot,\cdot))$$

for some covariance function $k(\cdot,\cdot)$. For now, the covariance function is supposed to be fixed.

Then, let us consider the probability of the object $\boldsymbol{x}_*$ belonging to positive class to be equal to $\sigma(f(\boldsymbol{x}_*))$ for the chosen sigmoid function $\sigma$:

$$p(y_* = +1 \,|\, \boldsymbol{x}_*) = \sigma(f(\boldsymbol{x}_*)). \tag{2}$$

In this work, the logistic function $\sigma(z) = (1 + \exp(-z))^{-1}$ is used; however, one could use other sigmoid functions as well.

The probabilistic model for this setting is given by

$$p(\boldsymbol{y}, \boldsymbol{f} \,|\, \boldsymbol{X}) = p(\boldsymbol{y} \,|\, \boldsymbol{f}) p(\boldsymbol{f} \,|\, \boldsymbol{X}) = p(\boldsymbol{f} \,|\, \boldsymbol{X}) \prod_{i=1}^{n} p(y_i \,|\, f_i) \tag{3}$$

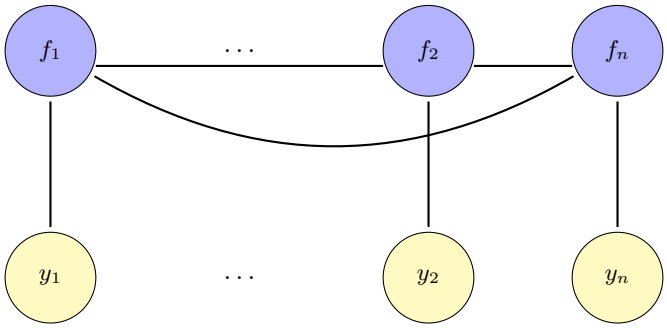


**Figure 2** Gaussian process classification graphical model

where $p(y_i \mid f_i)$ is the sigmoid likelihood (2) and $p(\boldsymbol{f} \mid \boldsymbol{X}) = \mathcal{N}(\boldsymbol{f} \mid \boldsymbol{0}, K(\boldsymbol{X}, \boldsymbol{X}))$ is the GP prior. The corresponding probabilistic graphical model is given in Fig. 2.

Now, inference in model (3) can be done in two steps. First, for new data point $\boldsymbol{x}_*$, one should find the conditional distribution of the corresponding value of the latent process $f_*$. This can be done as follows:

$$p(f_* \mid \boldsymbol{y}, \boldsymbol{X}, \boldsymbol{x}_*) = \int p(f_* \mid \boldsymbol{f}, \boldsymbol{X}, \boldsymbol{x}_*) p(\boldsymbol{f} \mid \boldsymbol{y}, \boldsymbol{X})\, d\boldsymbol{f}\,. \tag{4}$$

Second, the probability that $\boldsymbol{x}_*$ belongs to the positive class is obtained by marginalizing over the latent variable $f_*$:

$$p(y_* = +1 \mid \boldsymbol{y}, \boldsymbol{X}, \boldsymbol{x}_*) = \int \sigma(f_*) p(f_* \mid \boldsymbol{y}, \boldsymbol{X}, \boldsymbol{x}_*)\, df_*\,. \tag{5}$$

Unfortunately, both integrals (4) and (5) are intractable since they involve a product of sigmoid functions and normal distributions. Thus, one has to use some integral-approximation techniques to estimate the predictive distribution.

For example, one can use Laplace approximation method, which builds a Gaussian approximation $q(\boldsymbol{f} \mid \boldsymbol{y}, \boldsymbol{X})$ to the true posterior $p(\boldsymbol{f} \mid \boldsymbol{y}, \boldsymbol{X})$. Substituting this Gaussian approximation back into (4), one obtains a tractable integral. The predictive distribution (5) remains intractable but since this is a one-dimensional integral, it can be easily estimated by quadratures or other techniques. The more detailed derivation of this algorithm and another algorithm, based on Expectation Propagation, can be found in [1].

Computational complexity of computing the predictive distribution both for the Laplace approximation method and Expectation Propagation scales as $\mathcal{O}(n^3)$ since they both require to invert $n \times n$ matrix $K(\boldsymbol{X}, \boldsymbol{X})$. In section 3, the concept of inducing points aimed to reduce this complexity is described.

### 2.3 Model adaptation

In the previous subsection, it was described how to fit a GP to the data in the classification problem. However, only GP with fixed covariance functions have been considered. This model can be rather limiting.

Most of the popular covariance functions have a set of parameters, which are referred to here as covariance (or kernel) hyperparameters. For example, the squared exponential covariance function

$$k_{\mathrm{SE}}(\boldsymbol{x}, \boldsymbol{x}'; \boldsymbol{\theta}) = \sigma^2 \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{x}'\|^2}{l^2}\right)$$

has two parameters $\boldsymbol{\theta}$: variance $\sigma$ and length-scale $l$. An example of a more complicated popular covariance function is the Matern function, given by

$$k_{\text{Matern}}(\boldsymbol{x}, \boldsymbol{x}'; \boldsymbol{\theta}) = \frac{2^{1-\nu}}{\Gamma(\nu)} \left( \frac{\sqrt{2\nu}\|\boldsymbol{x} - \boldsymbol{x}'\|}{l} \right)^{\nu} K_{\nu} \left( \frac{\sqrt{2\nu}}{\|\boldsymbol{x} - \boldsymbol{x}'\|} l \right),$$

with two positive parameters $\boldsymbol{\theta} = (\nu, l)$. Here, $K_{\nu}$ is the modified Bessel function.

In order to get a good model for the data, one should find a good set of kernel hyperparameters $\boldsymbol{\theta}$. Bayesian paradigm provides a way of tuning the kernel hyperparameters of the GP-model through maximization of the model evidence (marginal likelihood) that is given by

$$p(\boldsymbol{y} \mid \boldsymbol{X}, \boldsymbol{\theta}) = \int p(\boldsymbol{y} \mid \boldsymbol{f}) p(\boldsymbol{f} \mid \boldsymbol{X}, \boldsymbol{\theta}) \, d\boldsymbol{f} \to \max_{\boldsymbol{\theta}} . \tag{6}$$

However, this integral is intractable for the model (3) since it involves a product of sigmoid functions and normal distribution. In subsequent sections, several methods to construct a variational lower bound to the marginal likelihood will be described. Maximizing this lower bound with respect to kernel hyperparameters $\boldsymbol{\theta}$, one could fit the model to the data.

## 3 Variational inducing point Gaussian process classification

In the previous section, it was shown how GP can be applied to solve classification problems. The computational complexity of GP for classification scales as $\mathcal{O}(n^3)$ that makes this method inapplicable to big data problems.

A number of approximate methods have been proposed in the literature for both GP-regression and GP-classification [2–4]. In this paper, the methods based on the concept of inducing inputs are considered. These methods construct an approximation based on the values of the process at some $m < n$ points. These points are referred to as inducing points. The idea is the following. The hidden GP $f$ corresponds to some smooth low-dimensional surface in $\mathbb{R}^d$. This surface can, in fact, be well approximated by another GP with properly chosen $m$ training points $\boldsymbol{Z} = (\boldsymbol{z}_1, \ldots, \boldsymbol{z}_m)^{\mathsf{T}} \in \mathbb{R}^{m \times d}$ and process values at that points $\boldsymbol{u} = (u_1, \ldots, u_m)^{\mathsf{T}}$ (inducing inputs). Then, predictions of this new process at training points are used for constructing approximate posterior distribution for $p(\boldsymbol{f} \mid \boldsymbol{y}, \boldsymbol{X})$. The positions $\boldsymbol{Z}$ of inducing inputs can be learned within the training procedure. However, for simplicity, in the following, the dataset $\boldsymbol{X}$ will be clusterized into $m$ clusters using $K$-means and $\boldsymbol{Z}$ will be chosen to be the cluster centers. In practice, it is observed that this approach works well almost in all the cases.

### 3.1 Evidence lower bound

In the following, a variational approach will be used for solving maximum evidence problem (6). In this approach, an ELBO is introduced that is simpler to compute than the evidence itself. Then, this lower bound is maximized with regard to kernel hyperparameters $\boldsymbol{\theta}$ and additional variational parameters used for constructing the lower bound.

Let us consider the following augmented probabilistic model:

$$p(\boldsymbol{y}, \boldsymbol{f}, \boldsymbol{u} \mid \boldsymbol{X}, \boldsymbol{Z}) = p(\boldsymbol{y} \mid \boldsymbol{f}) p(\boldsymbol{f}, \boldsymbol{u} \mid \boldsymbol{X}, \boldsymbol{Z}) = \prod_{i=1}^{n} p(y_i \mid f_i) p(\boldsymbol{f}, \boldsymbol{u} \mid \boldsymbol{X}, \boldsymbol{Z}). \tag{7}$$

The graphical model for the model (7) is shown in Fig. 3. Note that marginalizing the model (7) with regard to $\boldsymbol{u}$ gives the initial model (3).

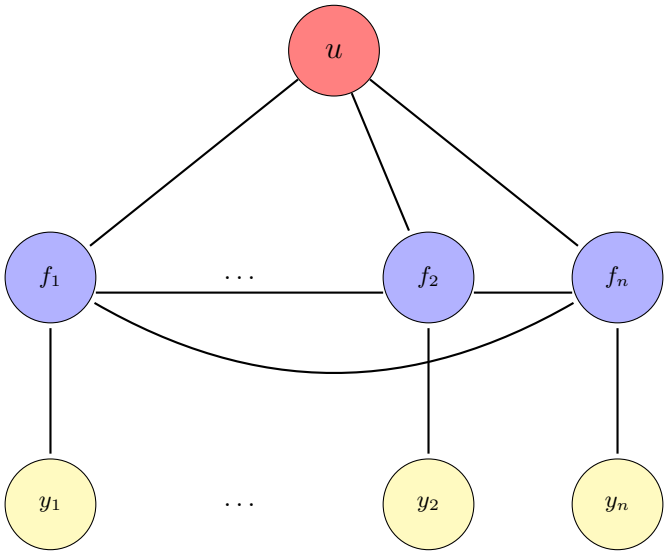


**Figure 3** Gaussian process classification graphical model

Let us denote the covariance matrix comprised of pairwise values of the covariance function $k(\cdot,\ \cdot)$ on the points $\boldsymbol{Z}$ by $K(\boldsymbol{Z},\boldsymbol{Z}) = \boldsymbol{K}_{mm} \in \mathbb{R}^{m\times m}$. Similarly, let us define $\boldsymbol{K}_{nn} = K(\boldsymbol{X},\boldsymbol{X}) \in \mathbb{R}^{n\times n}$ and $\boldsymbol{K}_{nm} = K(\boldsymbol{X},\boldsymbol{Z}) = \boldsymbol{K}_{mn}^{\mathsf{T}} \in \mathbb{R}^{n\times m}$.

As $\boldsymbol{u}$ and $\boldsymbol{f}$ are generated from the same GP with zero-mean prior:

$$\begin{aligned} p(\boldsymbol{f},\boldsymbol{u}\,|\,\boldsymbol{X},\boldsymbol{Z}) &= \mathcal{N}([\boldsymbol{f},\boldsymbol{u}]\,|\,[\boldsymbol{0},\boldsymbol{0}], K([\boldsymbol{X},\boldsymbol{Z}],[\boldsymbol{X},\boldsymbol{Z}]))\,; \\ p(\boldsymbol{u}\,|\,\boldsymbol{Z}) &= \mathcal{N}(\boldsymbol{u}\,|\,\boldsymbol{0},\boldsymbol{K}_{mm})\,; \\ p(\boldsymbol{f}\,|\,\boldsymbol{u},\boldsymbol{X},\boldsymbol{Z}) &= \mathcal{N}(\boldsymbol{f}\,|\,\boldsymbol{K}_{nm}\boldsymbol{K}_{mm}^{-1}\boldsymbol{u},\tilde{\boldsymbol{K}}) \end{aligned} \tag{8}$$

where $\tilde{\boldsymbol{K}} = \boldsymbol{K}_{nn} - \boldsymbol{K}_{nm}\boldsymbol{K}_{mm}^{-1}\boldsymbol{K}_{mn}$. In the following, for simplicity, the dependence on $\boldsymbol{X}$ and $\boldsymbol{Z}$ will be omitted in all formulas. Note that here, optimization is not considered with regard to these values.

Applying the standard variational lower bound (see, for example, [8]) to the augmented model (7), one obtains the following inequality:

$$\log p(\boldsymbol{y}) \geqslant \mathsf{E}_{q(\boldsymbol{u},\boldsymbol{f})} \log \frac{p(\boldsymbol{y},\boldsymbol{u},\boldsymbol{f})}{q(\boldsymbol{u},\boldsymbol{f})} = \mathsf{E}_{q(\boldsymbol{u},\boldsymbol{f})} \log p(\boldsymbol{y}\,|\,\boldsymbol{f}) - \mathrm{KL}\left(q(\boldsymbol{u},\boldsymbol{f}) \| p(\boldsymbol{u},\boldsymbol{f})\right)$$

for any distribution $q(\boldsymbol{u},\boldsymbol{f})$. This inequality becomes equality for the true posterior distribution $q(\boldsymbol{u},\boldsymbol{f}) = p(\boldsymbol{u},\boldsymbol{f}\,|\,\boldsymbol{y})$. Next, let us restrict the variational distribution $q(\boldsymbol{u},\boldsymbol{f})$ to be of the form

$$q(\boldsymbol{u},\boldsymbol{f}) = p(\boldsymbol{f}\,|\,\boldsymbol{u})q(\boldsymbol{u}) \tag{9}$$

where $q(\boldsymbol{u}) = \mathcal{N}(\boldsymbol{u}\,|\,\boldsymbol{\mu},\boldsymbol{\Sigma})$ for some $\boldsymbol{\mu} \in \mathbb{R}^m$, $\boldsymbol{\Sigma} \in \mathbb{R}^{m\times m}$, and $p(\boldsymbol{f}\,|\,\boldsymbol{u})$ is determined by (8). This is the key approximation step in inducing points approach for GP. The chosen family (9) subsumes that with large enough $m$, all information about the hidden process values $\boldsymbol{f}$ at training points can be successfully restored from the values $\boldsymbol{u}$ at inducing inputs, i. e., $p(\boldsymbol{f}\,|\,\boldsymbol{u},\boldsymbol{y}) \approx p(\boldsymbol{f}\,|\,\boldsymbol{u})$.

Form (9) of the variational distribution implies a Gaussian marginal distribution:

$$q(\boldsymbol{f}) = \int p(\boldsymbol{f}\,|\,\boldsymbol{u})q(\boldsymbol{u})\,d\boldsymbol{u} = \mathcal{N}\left(\boldsymbol{f}\,|\,\boldsymbol{K}_{nm}\boldsymbol{K}_{mm}^{-1}\boldsymbol{\mu},\boldsymbol{K}_{nn} + \boldsymbol{K}_{nm}\boldsymbol{K}_{mm}^{-1}(\boldsymbol{\Sigma} - \boldsymbol{K}_{mm})\boldsymbol{K}_{mm}^{-1}\boldsymbol{K}_{mn}\right).$$

As $\log p(\boldsymbol{y} \mid \boldsymbol{f})$ depends on $\boldsymbol{u}$ only through $\boldsymbol{f}$, the expectation

$$\mathsf{E}_{q(\boldsymbol{u},\boldsymbol{f})} \log p(\boldsymbol{y} \mid \boldsymbol{f}) = \mathsf{E}_{q(\boldsymbol{f})} \log p(\boldsymbol{y} \mid \boldsymbol{f}) = \sum_{i=1}^{n} \mathsf{E}_{q(f_i)} \log p(y_i \mid f_i)$$

where $q(f_i)$ is the marginal distribution of $q(\boldsymbol{f})$:

$$q(f_i) = \mathcal{N}\left(f_i \mid \boldsymbol{k}_i^{\mathrm{T}} \boldsymbol{K}_{mm}^{-1} \boldsymbol{\mu}, \boldsymbol{K}_{ii} + \boldsymbol{k}_i^{\mathrm{T}} \boldsymbol{K}_{mm}^{-1} (\boldsymbol{\Sigma} - \boldsymbol{K}_{mm}) \boldsymbol{K}_{mm}^{-1} \boldsymbol{k}_i\right) = \mathcal{N}\left(f_i \mid m_i, S_i^2\right) \tag{10}$$

and $\boldsymbol{k}_i$ is the $i$th column of matrix $\boldsymbol{K}_{mn}$.

Finally,

$$\mathrm{KL}\left(q(\boldsymbol{u}, \boldsymbol{f}) \| p(\boldsymbol{u}, \boldsymbol{f})\right) = \mathrm{KL}\left(q(\boldsymbol{u}) p(\boldsymbol{f} \mid \boldsymbol{u}) \| p(\boldsymbol{u}) p(\boldsymbol{f} \mid \boldsymbol{u})\right) = \mathrm{KL}\left(q(\boldsymbol{u}) \| p(\boldsymbol{u})\right).$$

Combining everything back together, one obtains the ELBO:

$$\log p(\boldsymbol{y}) \geqslant \sum_{i=1}^{n} \mathsf{E}_{q(f_i)} \log p(y_i \mid f_i) - \mathrm{KL}\left(q(\boldsymbol{u}) \| p(\boldsymbol{u})\right). \tag{11}$$

Note that the KL-divergence term in the lower bound (11) can be computed analytically since it is a KL-divergence between two normal distributions. In order to compute the expectations $\mathsf{E}_{q(f_i)} \log p(y_i \mid f_i)$, one has to use integral approximating techniques.

The ELBO (11) can be maximized with respect to variational parameters $\boldsymbol{\mu}$, $\boldsymbol{\Sigma}$, and kernel hyperparameters. Using the optimal distribution $q(\boldsymbol{u})$, one can perform predictions for new data point $\boldsymbol{x}_*$ as follows:

$$\begin{aligned} p(f_* \mid \boldsymbol{y}) = \int p(f_* \mid \boldsymbol{u}, \boldsymbol{f}) p(\boldsymbol{u}, \boldsymbol{f} \mid \boldsymbol{y})\, d\boldsymbol{u} d\boldsymbol{f} &\approx \int p(f_* \mid \boldsymbol{u}, \boldsymbol{f}) q(\boldsymbol{u}, \boldsymbol{f})\, d\boldsymbol{u} d\boldsymbol{f} \\ &= \int p(f_* \mid \boldsymbol{u}, \boldsymbol{f}) p(\boldsymbol{f} \mid \boldsymbol{u}) q(\boldsymbol{u})\, d\boldsymbol{u} d\boldsymbol{f} = \int p(f_* \mid \boldsymbol{u}) q(\boldsymbol{u})\, d\boldsymbol{u}. \end{aligned}$$

The last integral is tractable since both terms $p(f_* \mid \boldsymbol{u})$ and $q(\boldsymbol{u})$ are the normal distributions.

Note that in case of regression with Gaussian noise, the distributions $p(y_i \mid f_i)$ are the Gaussians and, thus, the expectations $\mathsf{E}_{q(f_i)} \log p(y_i \mid f_i)$ are tractable. Paper [6] suggests maximization of the lower bound (11) with respect to $\boldsymbol{\mu}$, $\boldsymbol{\Sigma}$, and covariance hyperparameters with stochastic optimization techniques for GP-regression.

### 3.2 Stochastic Variational Inference method

In case of classification, one cannot analytically compute the expectations $\mathsf{E}_{q(f_i)} \log p(y_i \mid f_i)$ in the lower bound (11). However, the expectations are the one-dimensional Gaussian integrals and can thus be effectively approximated with a range of techniques. In paper [7], Gauss–Hermite quadratures are used for this purpose. Note that the lower bound (11) has the form "sum over training objects." Hence, this bound can be maximized using stochastic optimization techniques. Paper [7] suggests to maximize the lower bound (11) with respect to the variational parameters $\boldsymbol{\mu}, \boldsymbol{\Sigma}$, and kernel hyperparameters $\boldsymbol{\theta}$ using stochastic optimization. This method is referred to as `svi` (Stochastic Variational Inference) method. The lower bound (11) and all its derivatives can be computed in $\mathcal{O}(nm^2 + m^3)$. This complexity has a linear dependence on $n$; hence, `svi` method can be applied for the case of big training data.

## 4 Tractable evidence lower bound for Gaussian process classification

In the previous section, the `svi` method has been described. It is based on stochastic optimization of the lower bound (11) for marginal likelihood and the lower bound itself is computed in $\mathcal{O}(nm^2)$. But the bound depends on $\mathcal{O}(m^2)$ parameters which makes the optimization problem hard to solve when a big number of inducing points is needed.

For GP-regression, the situation is similar. Paper [6] describes a method analogical to the `svi` method for classification. The only difference is that the lower bound becomes tractable in case of regression. Then, the paper [5] tries to solve the problem of big $\mathcal{O}(m^2)$ number of parameters in the algorithm from [6] in the following way. In case of regression, the lower bound (11) can be analytically optimized with respect to variational parameters $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$. Doing so and substituting the optimal values back into the lower bound, one can obtain a new lower bound to the marginal likelihood that depends solely on kernel hyperparameters $\boldsymbol{\theta}$. This simplifies the optimization problem by dramatically reducing the number of optimization parameters. Unfortunately, this new bound does not have a form of "sum over objects;" hence, stochastic optimization methods are no longer applicable here. However, in their experiments, the present authors have found that even for fairly big datasets, the method from [5] outperforms [6] despite the lack of stochastic optimization.

In the following subsection, an approach similar to the method of [5] is devised for the case of classification. A tractable ELBO is provided and it is analytically maximized with respect to variational parameters $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$. Substituting the optimal values of these parameters back into the lower bound, one obtains a new lower bound that depends only on kernel hyperparameters $\boldsymbol{\theta}$.

### 4.1 Global evidence lower bound

In order to derive a tractable lower bound for (11), let us seek a quadratic approximation to the log-logistic function $\log p(y_i \,|\, f_i) = \log \sigma(y_i f_i)$ where $\log \sigma(t) = -\log(1 + \exp(-t))$. Paper [9] provides a global parametric quadratic lower bound for this function:

$$\log \sigma(t) \geqslant \frac{t}{2} - \frac{\xi_t}{2} + \log \sigma(\xi_t) - \lambda(\xi_t)(t^2 - \xi_t^2), \;\; \forall t$$

where $\lambda(\xi_t) = \tanh(\xi_t)/(4\xi_t)$ and $\xi_t \in \mathbb{R}$ is the parameter of the bound. This bound is tight when $t^2 = \xi_t^2$.

Substituting this bound back to (11) with separate values $\boldsymbol{\xi} = \{\xi_i \,|\, i = 1, \ldots, n\}$ for every data point, one obtains a tractable lower bound:

$$\begin{aligned}
\log p(y) &\geqslant \sum_{i=1}^{n} \mathsf{E}_{q(f_i)} \log p(y_i|f_i) - \mathrm{KL}\left(q(\boldsymbol{u}) \| p(\boldsymbol{u})\right) = \sum_{i=1}^{n} \mathsf{E}_{q(f_i)} \log \sigma(y_i f_i) - \mathrm{KL}\left(q(\boldsymbol{u}) \| p(\boldsymbol{u})\right) \\
&\geqslant \sum_{i=1}^{n} \left( \mathsf{E}_{q(f_i)} \left[ \log \sigma(\xi_i) + \frac{y_i f_i - \xi_i}{2} - \lambda(\xi_i) \left( f_i^2 - \xi_i^2 \right) \right] \right) - \mathrm{KL}\left(q(\boldsymbol{u}) \| p(\boldsymbol{u})\right) \\
&= \sum_{i=1}^{n} \left( \log \sigma(\xi_i) - \frac{\xi_i}{2} + \lambda(\xi_i)\xi_i^2 \right) + \frac{1}{2} \boldsymbol{\mu}^{\mathsf{T}} \boldsymbol{K}_{mm}^{-1} \boldsymbol{K}_{mn} \boldsymbol{y} \\
&\quad - \operatorname{tr}\left( \boldsymbol{\Lambda}(\boldsymbol{\xi}) \left( \boldsymbol{K}_{nn} + \boldsymbol{K}_{nm} \boldsymbol{K}_{mm}^{-1} \left( \boldsymbol{\Sigma} - \boldsymbol{K}_{mm} \right) \boldsymbol{K}_{mm}^{-1} \boldsymbol{K}_{mn} \right) \right) \\
&\quad - \boldsymbol{\mu}^{\mathsf{T}} \boldsymbol{K}_{mm}^{-1} \boldsymbol{K}_{mn} \boldsymbol{\Lambda}(\boldsymbol{\xi}) \boldsymbol{K}_{nm} \boldsymbol{K}_{mm}^{-1} \boldsymbol{\mu} - \mathrm{KL}\left(q(\boldsymbol{u}) \| p(\boldsymbol{u})\right) = J(\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\xi}, \boldsymbol{\theta})
\end{aligned}$$

where

$$\boldsymbol{\Lambda}(\boldsymbol{\xi}) = \begin{pmatrix} \lambda(\xi_1) & 0 & \cdots & 0 \\ 0 & \lambda(\xi_2) & \ddots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda(\xi_n) \end{pmatrix}.$$

Differentiating $J$ with respect to $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ and setting the derivatives to zero, one obtains:

$$\hat{\boldsymbol{\Sigma}}(\boldsymbol{\xi}) = \left(2\boldsymbol{K}_{mm}^{-1}\boldsymbol{K}_{mn}\boldsymbol{\Lambda}(\boldsymbol{\xi})\boldsymbol{K}_{nm}\boldsymbol{K}_{mm}^{-1} + \boldsymbol{K}_{mm}^{-1}\right)^{-1}; \tag{12}$$

$$\hat{\boldsymbol{\mu}}(\boldsymbol{\xi}) = \frac{1}{2}\hat{\boldsymbol{\Sigma}}(\boldsymbol{\xi})\boldsymbol{K}_{mm}^{-1}\boldsymbol{K}_{mn}\boldsymbol{y}. \tag{13}$$

Substituting the optimal values of variational parameters back to the lower bound $J$ and omitting the terms not depending on $\boldsymbol{\theta}$ and $\boldsymbol{\xi}$, one obtains a compact lower bound:

$$\hat{J}(\boldsymbol{\theta}, \boldsymbol{\xi}) = \sum_{i=1}^{n}\left(\log\sigma(\xi_i) - \frac{\xi_i}{2} + \lambda(\xi_i)\xi_i^2\right) + \frac{1}{8}\boldsymbol{y}^{\mathrm{T}}\boldsymbol{K}_{nm}\boldsymbol{B}^{-1}\boldsymbol{K}_{mn}\boldsymbol{y} + \frac{1}{2}\log|\boldsymbol{K}_{mm}| - \frac{1}{2}\log|\boldsymbol{B}| - \operatorname{tr}\left(\boldsymbol{\Lambda}(\boldsymbol{\xi})\tilde{\boldsymbol{K}}\right)$$

where

$$\tilde{\boldsymbol{K}} = \boldsymbol{K}_{nn} - \boldsymbol{K}_{nm}\boldsymbol{K}_{mm}^{-1}\boldsymbol{K}_{mn};$$
$$\boldsymbol{B} = 2\boldsymbol{K}_{mn}\boldsymbol{\Lambda}(\boldsymbol{\xi})\boldsymbol{K}_{nm} + \boldsymbol{K}_{mm}.$$

In the following, three different methods will be considered for maximizing the lower bound $\hat{J}(\boldsymbol{\theta}, \boldsymbol{\xi})$.

Note that given the values of $\boldsymbol{\mu}$, $\boldsymbol{\Sigma}$, and $\boldsymbol{\theta}$, one can maximize $J(\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\xi}, \boldsymbol{\theta})$ with respect to $\boldsymbol{\xi}$ analytically. The optimal values for $\boldsymbol{\xi}$ are given by

$$\xi_i^2 = \mathsf{E}_{q(\boldsymbol{f})} f_i^2 = m_i^2 + S_i^2. \tag{14}$$

The values $m_i$ and $S_i$ were defined in (10). In the first method, the analytical formulas (14) are used to recompute the values of $\boldsymbol{\xi}$ and the gradient-based optimization is used to maximize the bound with respect to $\boldsymbol{\theta}$. The pseudocode is given in Algorithm 1. Let us refer to this method as `vi-JJ` where `JJ` stands for Jaakkola and Jordan, the authors of [9]. Note that the computational complexity of one iteration of this method is $\mathcal{O}(nm^2)$, the same as for the `svi` method.

The second method uses gradient-based optimization to maximize $\hat{J}$ with respect to both $\boldsymbol{\theta}$ and $\boldsymbol{\xi}$. Note that in this method, it is not necessary to recompute $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ at each iteration which makes the methods iterations empirically faster for big values of $m$. Let us refer to this method as `vi-JJ-full`.

Finally, `vi-JJ-hybrid` is a combination of the two methods described above. The general scheme of this method is the same as `vi-JJ`. In the `vi-JJ-hybrid` method, analytical formulas are used to recompute $\boldsymbol{\xi}$ as is done in the `vi-JJ` method at stage 1 but at stage 2, gradient-based optimization is used with respect to both $\boldsymbol{\xi}$ and $\boldsymbol{\theta}$. The virtues of this method will be described in the experiments section.

**Algorithm 1** vi-JJ method

**Input:** $n_{\text{upd}}$, $n_{\text{fun}}$
**Output:** $\boldsymbol{\theta}$, $\boldsymbol{\mu}$, $\boldsymbol{\Sigma}$
  $\boldsymbol{\mu}$, $\boldsymbol{\Sigma} \leftarrow \mathbf{0}$, $\boldsymbol{I}$
  **repeat**
    $\tilde{\boldsymbol{\mu}}$, $\tilde{\boldsymbol{\Sigma}} \leftarrow \boldsymbol{\mu}$, $\boldsymbol{\Sigma}$
    **for** $j \leftarrow 1, \ldots, n_{\text{upd}}$ : *// stage 1: updating $\boldsymbol{\mu}$, $\boldsymbol{\Sigma}$, $\boldsymbol{\xi}$*
      $m_t$, $S_t \leftarrow \boldsymbol{k}_t^{\mathsf{T}} \boldsymbol{K}_{mm}^{-1} \tilde{\boldsymbol{\mu}}$, $\boldsymbol{K}_{tt} + \boldsymbol{k}_t^{\mathsf{T}} \boldsymbol{K}_{mm}^{-1} \left( \tilde{\boldsymbol{\Sigma}} - \boldsymbol{K}_{mm} \right) \boldsymbol{K}_{mm}^{-1} \boldsymbol{k}_t, \quad t = 1, \ldots, n$
      $\tilde{\xi}_t^{\,2} \leftarrow m_t^2 + S_t^2, \quad t = 1, \ldots, n$
    $\tilde{\boldsymbol{\mu}}$, $\tilde{\boldsymbol{\Sigma}} \leftarrow \hat{\boldsymbol{\mu}} \left( \tilde{\boldsymbol{\xi}} \right)$, $\hat{\boldsymbol{\Sigma}} \left( \tilde{\boldsymbol{\xi}} \right)$ *// see (12), (13)*
    $\boldsymbol{\mu}$, $\boldsymbol{\Sigma}$, $\boldsymbol{\xi} \leftarrow \tilde{\boldsymbol{\mu}}$, $\tilde{\boldsymbol{\Sigma}}$, $\tilde{\boldsymbol{\xi}}$
    $\boldsymbol{\theta} = \text{minimize} \left( \hat{J} \left( \cdot, \boldsymbol{\xi} \right) \right)$, method='L-BFGS-B', maxfun=$n_{\text{fun}}$ *// stage 2: updating $\boldsymbol{\theta}$*
  **until** convergence

### 4.2 Tractable local approximation to the evidence lower bound

Another way to obtain a tractable approximation to the lower bound (11) is to use a local quadratic approximation for the log-logistic function $\log p(y_i \,|\, f_i)$. In this way, let us perform a second-order Taylor expansion of this function at points $\boldsymbol{\xi} = \{\xi_i \,|\, i = 1, \ldots, n\}$:

$$\log p(y_i \,|\, f_i) \approx -\log(1 + \exp(-y_i \xi_i)) + \frac{y_i}{1 + \exp(y_i \xi_i)} (f_i - \xi_i) - \frac{y_i^2 \exp(y_i \xi_i)}{2(1 + \exp(y_i \xi_i))^2} (f_i - \xi_i)^2. \quad (15)$$

The following derivation is analogical to the derivation in the previous section. Substituting approximation (15) into the lower bound (11), one obtains

$$\begin{aligned}
\log p(y) &\geqslant \sum_{i=1}^{n} \mathsf{E}_{q(f_i)} \log p(y_i | f_i) - \mathrm{KL}\left(q(\boldsymbol{u}) \| p(\boldsymbol{u})\right) \\
&\approx -\sum_{i=1}^{n} \log(1 + \exp(-y_i \xi_i)) + \varphi(\boldsymbol{\xi})^{\mathsf{T}} \left( \boldsymbol{K}_{nm} \boldsymbol{K}_{mm}^{-1} \boldsymbol{\mu} - \boldsymbol{\xi} \right) \\
&\quad - \operatorname{tr}\left( \boldsymbol{\Psi}(\boldsymbol{\xi}) \left( \boldsymbol{K}_{nn} + \boldsymbol{K}_{nm} \boldsymbol{K}_{mm}^{-1} (\boldsymbol{\Sigma} - \boldsymbol{K}_{mm}) \boldsymbol{K}_{mm}^{-1} \boldsymbol{K}_{mn} \right) \right) \\
&\quad - \left( \boldsymbol{K}_{nm} \boldsymbol{K}_{mm}^{-1} \boldsymbol{\mu} - \boldsymbol{\xi} \right)^{\mathsf{T}} \boldsymbol{\Psi}(\boldsymbol{\xi}) \left( \boldsymbol{K}_{nm} \boldsymbol{K}_{mm}^{-1} \boldsymbol{\mu} - \boldsymbol{\xi} \right) - \frac{1}{2} \left( \log \frac{|\boldsymbol{K}_{mm}|}{|\boldsymbol{\Sigma}|} - m + \operatorname{tr}\left( \boldsymbol{K}_{mm}^{-1} \boldsymbol{\Sigma} \right) + \boldsymbol{\mu}^{\mathsf{T}} \boldsymbol{K}_{mm}^{-1} \boldsymbol{\mu} \right).
\end{aligned}$$

Here, $\boldsymbol{\Psi}(\boldsymbol{\xi})$ is the diagonal matrix

$$\boldsymbol{\Psi}(\boldsymbol{\xi}) = \begin{pmatrix} \psi(\xi_1) & 0 & \cdots & 0 \\ 0 & \psi(\xi_2) & \ddots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & \psi(\xi_n) \end{pmatrix}$$

where

$$\psi(\xi_i) = \frac{y_i^2 \exp(y_i \xi_i)}{2(1 + \exp(y_i \xi_i))^2}.$$

Differentiating the approximate bound with respect to $\boldsymbol{\mu}$, $\boldsymbol{\Sigma}$, and $\boldsymbol{\xi}$ and setting the derivatives to zero, one obtains the following formulas for optimal values of these parameters:

$$\hat{\boldsymbol{\Sigma}}(\boldsymbol{\xi}) = \left(2\boldsymbol{K}_{mm}^{-1}\boldsymbol{K}_{mn}\boldsymbol{\Psi}(\boldsymbol{\xi})\boldsymbol{K}_{nm}\boldsymbol{K}_{mm}^{-1} + \boldsymbol{K}_{mm}^{-1}\right)^{-1}; \tag{16}$$

$$\hat{\boldsymbol{\mu}}(\boldsymbol{\xi}) = \hat{\boldsymbol{\Sigma}}(\boldsymbol{\xi})\boldsymbol{K}_{mm}^{-1}\boldsymbol{K}_{mn}\boldsymbol{v}(\boldsymbol{\xi}); \tag{17}$$

$$\xi_i = m_i.$$

Here,

$$\boldsymbol{v}(\boldsymbol{\xi}) = \boldsymbol{\varphi}(\boldsymbol{\xi}) + 2\boldsymbol{\Psi}(\boldsymbol{\xi})\boldsymbol{\xi}$$

and $\boldsymbol{\varphi}(\boldsymbol{\xi})$ is the vector composed of

$$\boldsymbol{\varphi}(\boldsymbol{\xi})_i = \frac{y_i}{1 + \exp(y_i\xi_i)}.$$

Substituting the optimal values for $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ given by (16) and (17) back into the approximate bound and omiting the terms that do not depend on $\boldsymbol{\theta}$, one obtains the following approximate lower bound:

$$\tilde{J}_{\boldsymbol{\xi}} = \frac{1}{2}\boldsymbol{v}(\boldsymbol{\xi})^{\intercal}\boldsymbol{K}_{nm}\boldsymbol{B}^{-1}\boldsymbol{K}_{mn}\boldsymbol{v}(\boldsymbol{\xi}) + \frac{1}{2}\log|\boldsymbol{K}_{mm}| - \frac{1}{2}\log|\boldsymbol{B}| - \operatorname{tr}\left(\boldsymbol{\Psi}(\boldsymbol{\xi})\tilde{\boldsymbol{K}}\right) \tag{18}$$

where

$$\boldsymbol{B} = 2\boldsymbol{K}_{mn}\boldsymbol{\Psi}(\boldsymbol{\xi})\boldsymbol{K}_{nm} + \boldsymbol{K}_{mm}.$$

Note that the lower bound (18) is not a global lower bound for the log-evidence $\log p(y)$. However, locally, a good approximation of the ELBO (11) has been got.

For maximizing the approximate lower bound (18), let us consider a method, analogical to `vi-JJ`. In order to specify this method, let us simply substitute the bound $\hat{J}(\cdot, \boldsymbol{\xi})$ by $\tilde{J}_{\boldsymbol{\xi}}$ in the second stage in Algorithm 1. Let us refer to this method as `vi-Taylor`. The computational complexity of one iteration of this method is, once again, $\mathcal{O}(nm^2)$.

## 5 Experiments

In this section, the derived `vi-JJ`, `vi-Taylor`, `vi-JJ-full`, and `vi-JJ-hybrid` methods will be empirically compared with `svi`. Below, the setting of the experiments is described and their results are discussed.

### 5.1 Experimental setting

In the experiments, there have been compared 5 methods for variational inducing point GP-classification:

- `svi-AdaDelta` uses the AdaDelta optimization method for maximization of the lower bound (11) as it is done in paper [7];
- `vi-JJ` was described in subsection 4.1;
- `vi-Taylor` was described in subsection 4.2;
- `vi-JJ-full` was described in subsection 4.1; and
- `vi-JJ-hybrid` was described in subsection 4.1.

Also, the present authors have made an attempt to use deterministic L-BFGS-B optimization method for maximizing ELBO (11), but it worked substantially worse than all the other methods. Note that all the methods have the same complexity of epochs $\mathcal{O}(nm^2)$. The table

Methods outline

| Method | Numerically optimized variables | Analytically optimized variables |
|---|---|---|
| `svi-AdaDelta` | $\boldsymbol{\theta}$, $\boldsymbol{\mu} \in \mathbb{R}^m$, $\boldsymbol{\Sigma} \in \mathbb{R}^{m \times m}$ | |
| `vi-JJ`, `vi-Taylor` | $\boldsymbol{\theta}$ | $\boldsymbol{\mu} \in \mathbb{R}^m$, $\boldsymbol{\Sigma} \in \mathbb{R}^{m \times m}$, $\boldsymbol{\xi} \in \mathbb{R}^n$ |
| `vi-JJ-hybrid` | $\boldsymbol{\theta}$, $\boldsymbol{\xi} \in \mathbb{R}^n$ | $\boldsymbol{\mu} \in \mathbb{R}^m$, $\boldsymbol{\Sigma} \in \mathbb{R}^{m \times m}$, $\boldsymbol{\xi} \in \mathbb{R}^n$ |
| `vi-JJ-full` | $\boldsymbol{\theta}$, $\boldsymbol{\xi} \in \mathbb{R}^n$ | $\boldsymbol{\mu} \in \mathbb{R}^m$, $\boldsymbol{\Sigma} \in \mathbb{R}^{m \times m}$ |

shows which variables are optimized numerically and which are optimized analytically for each method.

In the present experiments, the lower bound was not optimized with respect to the positions $\boldsymbol{Z}$ of the inducing points. Instead, $K$-means clustering procedure with $K$ equal to the number $m$ of inducing inputs was used and clusters centers were taken as $\boldsymbol{Z}$. Also, the squared exponential covariance function (see section 2) was used in all experiments with a Gaussian noise term.

The stochastic method `svi-AdaDelta` requires the user to manually specify the learning rate and the batch size for the optimization method. For the former, it was necessary to run the method with different learning rates and to choose the value that resulted in the fastest convergence. The learning rates have been used from a fixed grid with a step of 0.1. It always happened that for the largest value from the grid, the method diverged and for the smallest, the method converged slower than for some medium value, verifying that the optimal learning rate was somewhere in the range. To choose the batch size, the following convention was used. For small *german* and *svmguide* datasets, the batch size was set to 50. For other datasets, approximately $n/100$ was used as the batch size where $n$ is the size of the training set.

For the `vi-JJ`, `vi-Taylor`, and `vi-JJ-hybrid` in all of the experiments on every iteration, the values of $\boldsymbol{\xi}$, $\boldsymbol{\mu}$, and $\boldsymbol{\Sigma}$ were recomputed three times ($n_{\text{upd}} = 3$ in Algorithm 1). To tune $\boldsymbol{\theta}$, on every iteration, L-BFGS-B optimization method was run constrained to do not more than 5 evaluations of the lower bound and its gradient. It was found that these values of the parameters work well for all the experimented datasets.

For the `svi-AdaDelta` method, optimization with regard to Cholesky factor of the matrix $\boldsymbol{\Sigma}$ was used to maintain its positive definiteness as described in [7]. AdaDelta optimization method implementation from the *climin* toolbox [10] was used as is done in the original paper.

For every dataset, the present authors experimented with a number of inducing points to verify that the results of the methods are close to the optimal.

The methods were evaluated plotting the accuracy of their predictions on the test data against time. All of the plots have the titles of the following format:

[name of the dataset], $n$ = [number of objects in the training set],
$d$ = [number of features], $m$ = [number of inducing inputs].

Also, all the datasets have been preprocessed by normalizing the features setting the mean of all features to 0 and the variance to 1. For datasets without available test data, 20% of the data have been used as a test set and 80% as a train set.

### 5.2 Results and discussion

The methods' performance was compared on 7 datasets. Here, the results are discussed.

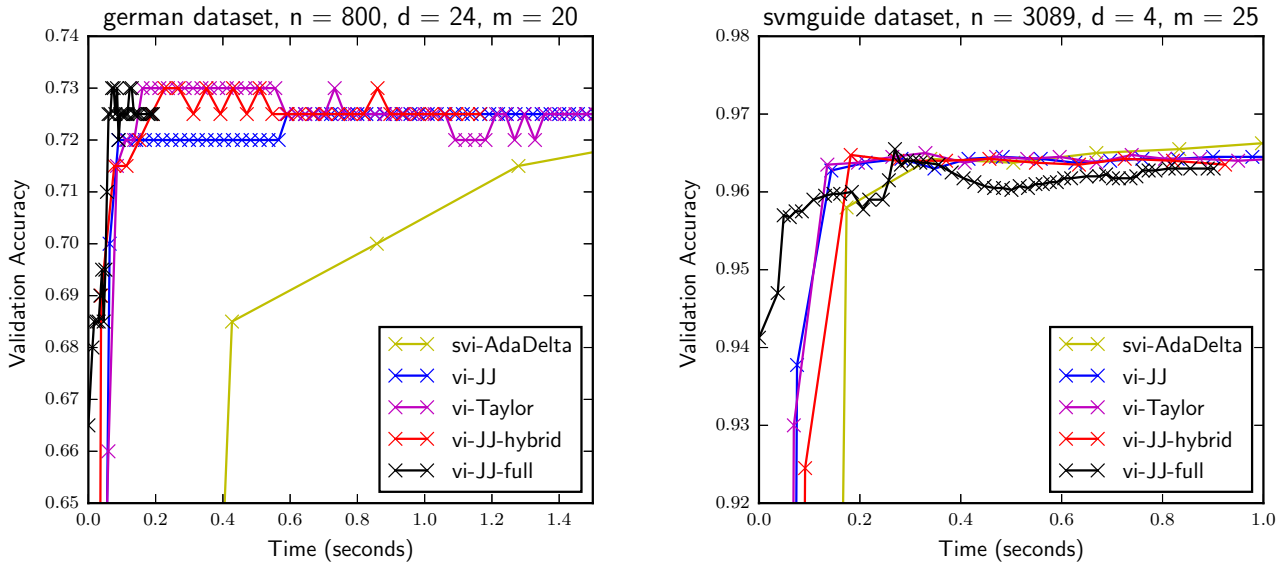


**Figure 4** Methods performance on small datasets

Figure 4 provides the results for *german* nd *svmguide* datasets. As one can see, on the small *german* dataset, the stochastic `svi-AdaDelta` method struggles and it takes it longer to achieve the optimal quality than for all the other methods which show similar results. On the *svmguide* dataset, it takes `vi-JJ-full` and `svi-AdaDelta` a little bit longer to converge, while the other three methods show roughly the same performance.

The results on *magic telescope* and *ijcnn* datasets are provided in Fig. 5. On the *magic telescope* dataset, `vi-JJ` and `vi-Taylor` show poor quality on the first iterations but still manage to converge faster than `svi-AdaDelta`. On both datasets, the `vi-JJ-hybrid` method works similar to `vi-JJ` and `vi-Taylor` but shows better quality on the first iterations on the *magic telescope* data. The method `vi-JJ-full` cannot converge to a reasonable quality on both datasets.

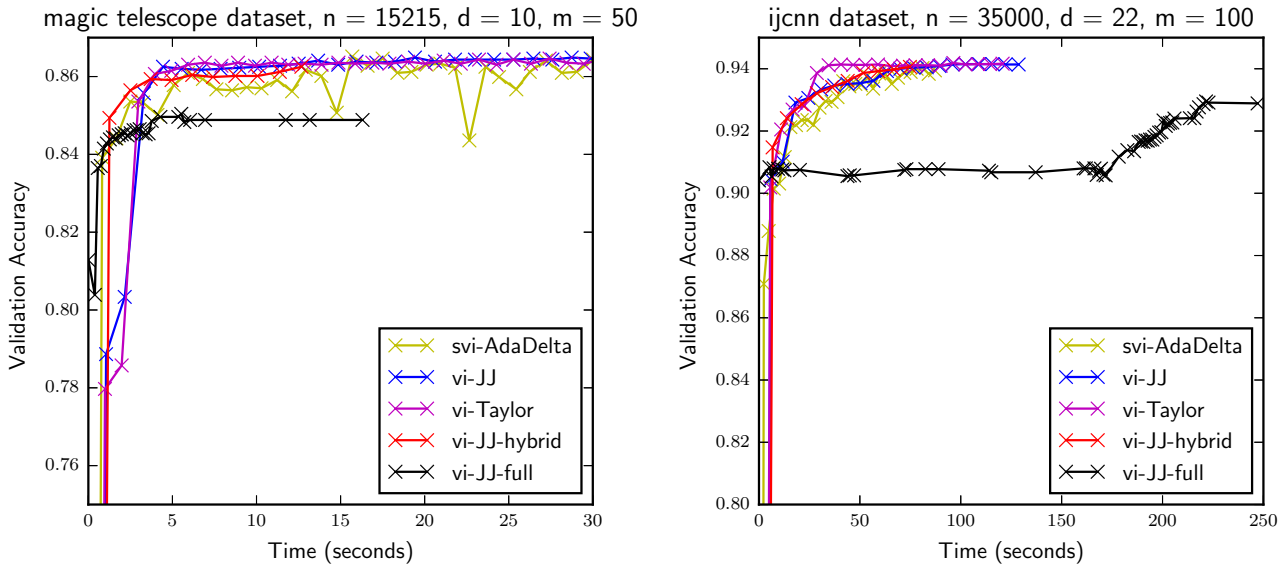


**Figure 5** Methods performance on medium datasets

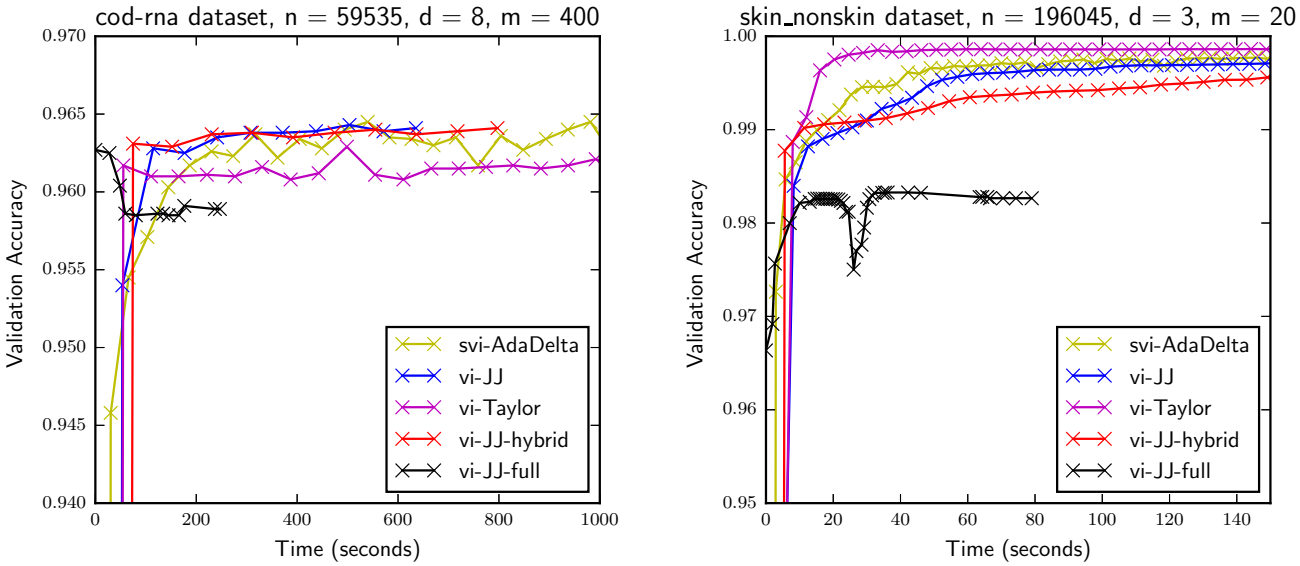


**Figure 6** Methods performance on big datasets

Figure 6 provides the results on big *cod-rna* and *skin_nonskin* datasets. On these datasets, `vi-JJ-full` once again fails to achieve a reasonable quality, while the other methods work similarly.

Finally, the results on *a8a* data are provided in Fig. 7. Here, a rather big amount of $m = 500$ inducing inputs were used. As one can see, `vi-JJ-full` and `vi-JJ-hybrid` are the fastest to achieve the optimal quality. The `svi-AdaDelta` method also converges reasonably fast, while `vi-JJ` and `vi-Taylor` struggle a little bit.

In general, the `vi-JJ`, `vi-Taylor`, and `vi-JJ-hybrid` methods perform similar to the `svi-AdaDelta` method. On the big dataset *skin_nonskin* with only three features, `vi-JJ-hybrid` is a little bit slower than the stochastic `svi-AdaDelta` method but on all other datasets, it is

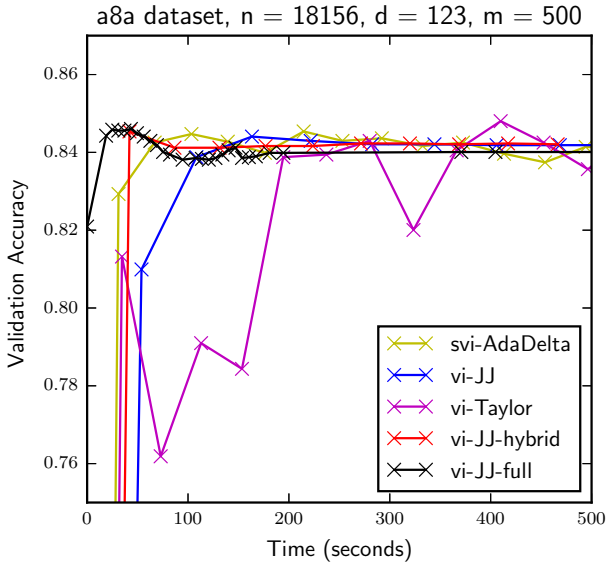


**Figure 7** Methods performance on the *a8a* dataset

better. The `vi-Taylor` and `vi-JJ` methods struggle with *a8a* but are, otherwise, comparable to `vi-JJ-hybrid`. The stochastic `svi-AdaDelta` method performs poorly on small datasets and even on the big *skin_nonskin* data does not manage to substantially outperform the other methods, even provided a good value of the learning rate. Finally, `vi-JJ-full` works well on small data and on the *a8a* but on all other datasets, it does not manage to achieve a reasonable quality.

## 6 Concluding remarks

In this paper, a new approach to training variational inducing input GP classification is presented. Two new tractable ELBO are derived and several ways to maximize them are described. The resulting methods `vi-JJ`, `vi-JJ-full`, `vi-JJ-hybrid`, and `vi-Taylor` are similar to the method of [5] for GP-regression.

An experimental comparison of the suggested methods with the current state-of-the-art method `svi-AdaDelta` of [7] is provided. In experimental setting, the present approach proved to be more practical as it converges to the optimal quality as fast as the `svi-AdaDelta` method without requiring the user to manually choose the parameters of the optimization method.

The four described `vi` methods showed similar performance and it is hard to distinguish them. However, note that the `vi-Taylor` approach is more general and can be applied to the likelihood functions that are not logistic. Also, a method, similar to `vi-JJ-hybrid` and `vi-JJ-full` for the nonlogistic case, can be easily derived but it is out of the scope of this paper.

## References

[1] Rasmussen, C. E., and C. K. I. Williams. 2006. *Gaussian processes for machine learning*. Cambridge, MA: The MIT Press. 266 p.

[2] Smola, A., and P. Bartlett. 2001. Sparse greedy Gaussian process regression. *Advances in neural information processing systems*. Eds. T. K. Lean, T. G. Dietterich, and V. Tresp. The MIT Press. 13:619–625.

[3] Csato, L., and M. Opper. 2002. Sparse online Gaussian processes. *Neural Comput*. 14:641–668.

[4] Quinonero-Candela, J., and C. Rasmussen. 2005. A unifying view of sparse approximate Gaussian process regression. *J. Mach. Learn. Res*. 6:1939–1959.

[5] Titsias, M. 2009. Variational learning of inducing variables in sparse Gaussian processes. *Proc. Machine Learning Res*. 5:567–574.

[6] Hensman, J., N. Fusi, and N. D. Lawrence. 2013. Gaussian processes for Big data. *29th Conference on Uncertainty in Artificial Intelligence Proceedings*. Eds. A. Nicholson and P. Smyth. Bellevue, WA. 282–290.

[7] Hensman, J., A. Matthews, and Z. Ghahramani. 2015. Scalable variational Gaussian process classification. *Proc. Machine Learning Res*. 38:351–360.

[8] Jaakkola, T. 2001. Tutorial on variational approximation methods. *Advanced mean field methods: Theory and practice*. Eds. M. Opper and D. Saad. Neural information processing ser. Cambridge, MA – London: The MIT Press. 129–159.

[9] Jaakkola, T., and M. Jordan. 1997. A variational approach to Bayesian logistic regression models and their extensions. *Conference on Artificial Intelligence and Statistics Proceedings*. Fort Lauderdale, FL.

[10] Climin. Available at: `http://github.com/BRML/climin` (accessed December 29, 2017).

*Received November 17, 2016*

# Быстрый метод обучения модели гауссовских процессов для задач классификации

***П. А. Измаилов, Д. А. Кропотов***

`izmailovpavel@gmail.com; dmitry.kropotov@gmail.com`

МГУ им. М. В. Ломоносова, Россия, г. Москва, Ленинские горы, 1

Предлагается новый подход к настройке моделей гауссовских процессов для задач классификации. Стандартные методы для данной задачи имеют сложность $\mathcal{O}(n^3)$, где $n$ — размер обучающей выборки. Данное обстоятельство не позволяет применять эти методы к задачам с большим объемом данных. В связи с этим в литературе был предложен ряд подходов, основанных на использовании так называемых вспомогательных точек (inducing inputs). Изначально такие методы использовались для задачи регрессии, но в недавней работе Хенсмэна с коллегами (2015 г.) подобный метод был разработан для задач классификации. В этом методе используется глобальная нижняя оценка на правдоподобие, которая максимизируется по параметрам гауссовского процесса и по дополнительным вариационным параметрам с помощью стохастической оптимизации. Вычислительная сложность данного метода составляет $\mathcal{O}(nm^2)$, где $m$ — число вспомогательных точек, которое обычно существенно меньше, чем $n$. Однако число переменных в оптимизации составляет $\mathcal{O}(m^2)$, что делает задачу поиска оптимальных параметров весьма сложной при больших значениях $m$. Предлагаются две новые оценки на маргинальное правдоподобие в модели гауссовских процессов со вспомогательными точками для задач классификации, а также несколько методов для их оптимизации. В новых оценках количество численно оптимизируемых переменных не зависит от числа вспомогательных точек $m$. В результате новые процедуры обучения становятся эффективными для широкого диапазона параметров $n$ и $m$. Кроме того, в отличие от стохастического метода из статьи Хенсмэна с коллегами (2015 г.), новые процедуры не требуют настройки параметров пользователем. Это значительно облегчает использование новых методов на практике. Проведенные эксперименты показывают, что новые методы демонстрируют сравнимое или лучшее качество по сравнению с методом из работы Хенсмэна с коллегами (2015 г.).

**Ключевые слова**: *гауссовский процесс; классификация; большие данные; байесовский вариационный вывод; оптимизация*

**DOI:** 10.21469/22233792.3.1.02

## Литература

[1] *Rasmussen C. E., Williams C. K. I.* Gaussian processes for machine learning. — Cambridge, MA, USA: The MIT Press, 2006. 266 p.

[2] *Smola A., Bartlett P.* Sparse greedy Gaussian process regression // Advances in neural information processing systems / Eds. T. K. Lean, T. G. Dietterich, V. Tresp. — The MIT Press, 2001. Vol. 13. P. 619–625.

[3] *Csato L., Opper M.* Sparse online Gaussian processes // Neural Comput., 2002. Vol. 14. P. 641–668.

[4] *Quinonero-Candela J., Rasmussen C.* A unifying view of sparse approximate Gaussian process regression // J. Mach. Learn. Res., 2005. Vol. 6. P. 1939–1959.

[5] *Titsias M.* Variational learning of inducing variables in sparse Gaussian processes // Proc. Machine Learning Res., 2009. Vol. 5. P. 567–574.

[6] *Hensman J., Fusi N., Lawrence N. D.* Gaussian processes for Big data // 29th Conference on Uncertainty in Artificial Intelligence Proceedings / Eds. A. Nicholson, P. Smyth. — Bellevue, WA, USA, 2013. P. 282–290.

[7] *Hensman J., Matthews A., Ghahramani Z.* Scalable variational Gaussian process classification // Proc. Machine Learning Res., 2015. Vol. 38. P. 351–360.

[8] *Jaakkola T.* Tutorial on variational approximation methods // Advanced mean field methods: Theory and practice / Eds. M. Opper, D. Saad. — Neural information processing ser. — Cambridge, MA – London: The MIT Press, 2001. P. 129–159.

[9] *Jaakkola T., Jordan M.* A variational approach to Bayesian logistic regression models and their extensions // Conference on Artificial Intelligence and Statistics Proceedings. — Fort Lauderdale, FL, USA, 1997.

[10] Climin. `http://github.com/BRML/climin`.

*Поступила в редакцию 17.11.2016*