Efficient Gaussian Process Classification Using Pólya-Gamma Data Augmentation

Florian Wenzel,[1,*] **Théo Galy-Fajou,**[2,*] **Christan Donner,**[2] **Marius Kloft,**[1,3] **Manfred Opper**[2]
[*]Contributed equally, [1]TU Kaiserslautern, Germany, [2]TU Berlin, Germany, [3]University of Southern California, USA
wenzelfl@hu-berlin.de, galy-fajou@tu-berlin.de, christian.donner@bccn-berlin.de,
kloft@cs.uni-kl.de, manfred.opper@tu-berlin.de

Abstract

We propose a scalable stochastic variational approach to GP classification building on Pólya-Gamma data augmentation and inducing points. Unlike former approaches, we obtain closed-form updates based on natural gradients that lead to efficient optimization. We evaluate the algorithm on real-world datasets containing up to 11 million data points and demonstrate that it is up to two orders of magnitude faster than the state-of-the-art while being competitive in terms of prediction performance.

1 Introduction

Gaussian processes (GPs) (Rasmussen and Williams 2005) provide a popular Bayesian non-linear non-parametric method for regression and classification. Because of their ability of accurately adapting to data and thus achieving high prediction accuracy while providing well calibrated uncertainty estimates, GPs are a standard method in several application areas, including geospatial predictive modeling (Stein 2012) and robotics (Dragiev, Toussaint, and Gienger 2011).

However, recent trends in data availability in the sciences and technology have made it necessary to develop algorithms capable of processing massive data (John Walker 2014). Currently, GP classification has limited applicability to big data. Naive inference typically scales cubic in the number of data points, and exact computation of posterior and marginal likelihood is intractable.

Nevertheless, the combination of so-called sparse Gaussian process techniques with approximate inference methods, such as expectation propagation (EP) or the variational approach, have enabled GP classification for datasets containing millions of data points (Hernández-Lobato and Hernández-Lobato 2016; Salimbeni, Eleftheriadis, and Hensman 2018).

While these results are already impressive, we will show in this paper that a speedup of up to two orders magnitudes can be achieved. Our approach is based on considering an augmented version of the original GP classification model and replacing the ordinary (stochastic) gradients for optimization by more efficient *natural gradients*, which is the standard Euclidean gradient multiplied by the inverse Fisher information matrix. Natural gradients recently have been successfully used in a variety of variational inference problems (Honkela et al. 2010; Wenzel et al. 2017; Jähnichen et al. 2018).

Unfortunately, an efficient computation of the natural gradient for the GP classification problem is not straight forward. The use of the probit link function in Dezfouli and Bonilla (2015); Hernández-Lobato and Hernández-Lobato (2016); Mandt et al. (2017); Salimbeni, Eleftheriadis, and Hensman (2018) leads to expectations in the variational objective functions that can only be computed by numerical quadrature, thus, preventing efficient optimization.

We derive a natural-gradient approach to variational inference in GP classification based on the *logit* link. We exploit that the corresponding likelihood has an auxiliary variable representation as a continuous mixture of Gaussians involving Pólya-Gamma random variables (Polson, Scott, and Windle 2013).

Unlike former approaches, our natural gradient updates can be computed in closed-form. Moreover, they have the advantage that they correspond to block-coordinate ascent updates and, therefore, learning rates close to one can be chosen. This leads to a fast and stable algorithm which is simple to implement. Our main contributions are as follows:

- We present a Gaussian process classification model using a logit link function that is based on Pólya-Gamma data augmentation and inducing points for Gaussian process inference.
- We derive an efficient inference algorithm based on stochastic variational inference and natural gradients. All natural gradient updates are given in closed-form and do not rely on numerical quadrature methods or sampling approaches. Natural gradients have the advantage that they provide effective second-order optimization updates.
- In our experiments, we demonstrate that our approach drastically improves speed up to two orders of magnitude while being competitive in terms of prediction performance. We apply our method to massive real-world datasets up to 11 million points and demonstrate superior scalability.

The paper is organized as follows. In section 2 we discuss related work. In section 3 we introduce our novel scalable

GP classification model and in section 4 we present an efficient variational inference algorithm. Section 5 concludes with experiments. Our code is available via Github[1].

2 Background and Related Work

Gaussian process classification Hensman and Matthews (2015) consider Gaussian process classification with a probit inverse link function and suggest a variational Gaussian model that builds on inducing points. By employing automatic differentiation, Salimbeni, Eleftheriadis, and Hensman (2018) generalize this approach to use natural gradients in non-conjugate GP models. Khan and Nielsen (2018) consider natural gradient updates in the setting of variational inference with exponential families. Unlike our approach, these methods do not benefit from closed-form updates and have to resort to numerical approximations. Moreover, our approach has the advantage that a higher learning rate close to one can be chosen leading to updates that can be interpreted as block-coordinate ascent updates.

Izmailov, Novikov, and Kropotov (2018) use tensor train decomposition to allow for the training of GP models with billions of inducing points. The updates are not computed in closed-form and they do not use natural gradients.

Dezfouli and Bonilla (2015) propose a general automated variational inference approach for sparse GP models with non-conjugate likelihood. Since they follow a black box approach and do not exploit model specific properties they do not employ efficient optimization techniques.

Hernández-Lobato and Hernández-Lobato (2016) follow an expectation propagation approach based on inducing points and have a similar computational cost as Hensman and Matthews (2015).

Pólya-Gamma data augmentation Polson, Scott, and Windle (2013) introduced the idea of data augmentation in logistic models using the class of Pólya-Gamma distributions. This allows for exact inference via Gibbs sampling or approximate variational inference schemes (Scott and Sun 2013).

Linderman, Johnson, and Adams (2015) extend this idea to multinomial models and discuss the application for Gaussian processes with multinomial observations but their approach does not scale to big datasets and they do not consider the concept of inducing points.

3 Model

The logit GP Classification model is defined as follows. Let $X = (\boldsymbol{x}_1, \dots, \boldsymbol{x}_n) \in \mathbb{R}^{d \times n}$ be the d-dimensional training points with labels $\boldsymbol{y} = (y_1, \dots, y_n) \in \{-1, 1\}^n$. The likelihood of the labels is

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

where $\sigma(z) = (1 + \exp(-z))^{-1}$ is the logit link function and f is the latent decision function. We place a GP prior over f and obtain the joint distribution of the labels and the latent GP

$$p(\boldsymbol{y}, \boldsymbol{f}|X) = p(\boldsymbol{y}|\boldsymbol{f}, X)p(\boldsymbol{f}|X), \tag{2}$$

where $p(\boldsymbol{f}|X) = \mathcal{N}(\boldsymbol{f}|\boldsymbol{0}, K_{nn})$ and K_{nn} denotes the kernel matrix evaluated at the training points X. For the sake of clarity we omit the conditioning on X in the following.

3.1 Pólya-Gamma data augmentation

Due to the analytically inconvenient form of the likelihood function, inference for logit GP classification is a challenging problem. We aim to remedy this issue by considering an augmented representation of the original model. Later we will see that the augmented model is indeed advantageous as it leads to efficient closed-form updates in our variational inference scheme.

Polson, Scott, and Windle (2013) introduced the class of Pólya-Gamma random variables and proposed a data augmentation strategy for inference in models with binomial likelihoods. The augmented model has the appealing property that the likelihood of the latent function $\boldsymbol{f}$ is proportional to a Gaussian density when conditioned on the augmented Pólya-Gamma variables. This allows for Gibbs sampling methods, where model parameters and Pólya-Gamma variables can be sampled alternately from the posterior (Polson, Scott, and Windle 2013). Alternatively, the augmentation scheme can be utilized to derive an efficient approximate inference algorithm in the variational inference framework, which will be pursued here.

The Pólya-Gamma distribution is defined as follows. The random variable $\omega \sim \mathrm{PG}(b, 0)$, $b > 0$ is defined by the moment generating function

$$\mathbb{E}_{\mathrm{PG}(\omega|\, b,0)}[\exp(-\omega t)] = \frac{1}{\cosh^b(\sqrt{t/2})}. \tag{3}$$

It can be shown that this is the Laplace transform of an infinite convolution of gamma distributions. The definition is related to our problem by the fact that the logit link can be written in a form that involves the cosh function, namely $\sigma(z_i) = \exp(\frac{1}{2}z_i)(2\cosh(\frac{z_i}{2}))^{-1}$. In the following we derive a representation of the logit link in terms of Pólya-Gamma variables.

First, we define the general $\mathrm{PG}(b, c)$ class which is derived by an exponential tilting of the $\mathrm{PG}(b, 0)$ density, it is given by

$$\mathrm{PG}(\omega|\, b, c) \propto \exp(-\frac{c^2}{2}\omega)\mathrm{PG}(\omega|\, b, 0).$$

From the moment generating function (3) the first moment can be directly computed

$$\mathbb{E}_{PG(\omega|b,c)}[\omega] = \frac{b}{2c}\tanh\left(\frac{c}{2}\right).$$

For the subsequently presented variational algorithm these properties suffice and the full representation of the Pólya-Gamma density $\mathrm{PG}(\omega|b, c)$ is not required.

We now adapt the data augmentation strategy based on Pólya-Gamma variables for the GP classification model. To

[1] https://github.com/theogf/AugmentedGaussianProcesses.jl

do this we write the non-conjugate logistic likelihood function (1) in terms of Pólya-Gamma variables

$$\sigma(z_i) = (1+\exp(-z_i))^{-1} = \frac{\exp(\frac{1}{2}z_i)}{2\cosh(\frac{z_i}{2})} = \frac{1}{2}\int \exp\left(\frac{z_i}{2} - \frac{z_i^2}{2}\omega_i\right) p(\omega_i)\mathrm{d}\omega_i, \quad (4)$$

where $p(\omega_i) = \mathrm{PG}(\omega_i|1,0)$ and by making use of (3). For more details see Polson, Scott, and Windle (2013). Using this identity and substituting $z_i = y_i f(x_i)$ we augment the joint density (2) with Pólya-Gamma variables

$$p(\boldsymbol{y},\boldsymbol{\omega},\boldsymbol{f}) \propto \exp\left(\frac{1}{2}\boldsymbol{y}^\top\boldsymbol{f} - \frac{1}{2}\boldsymbol{f}^\top\Omega\boldsymbol{f}\right) p(\boldsymbol{f})p(\boldsymbol{\omega}), \quad (5)$$

where $\Omega = \mathrm{diag}(\boldsymbol{\omega})$ is the diagonal matrix of the Pólya-Gamma variables $\{\omega_i\}$. In contrast to the original model (2) the augmented model is conditionally conjugate forming the basis for deriving closed-form updates in section 4.

Interestingly, employing a structured mean-field variational inference approach (cf. section 4) to the plain Pólya-Gamma augmented model (5) leads to the same bound for GP classification derived by Gibbs and MacKay (2000). This is an interesting new perspective on this bound since they do not employ a data augmentation approach. We provide a proof in appendix A.5. Our approach goes beyond Gibbs and MacKay (2000) by providing a fully Bayesian perspective, including a sparse GP prior (section 3.2) in the model and proposing a scalable inference algorithm based on natural gradients (section 4).

3.2 Sparse Gaussian process

Inference in GP models typically has the computational complexity $\mathcal{O}(n^3)$. We obtain a scalable approximation of our model and focus on inducing point methods (Snelson and Ghahramani 2006). We follow a similar approach as in Hensman and Matthews (2015) and reduce the complexity to $\mathcal{O}(m^3)$, where m is number of inducing points.

We augment the latent GP f with m additional input-output pairs $(Z_1,u_1),\ldots,(Z_m,u_m)$, termed as *inducing inputs* and *inducing variables*. The function values of the GP $\boldsymbol{f}$ and the inducing variables $\boldsymbol{u} = (u_1,\ldots,u_m)$ are connected via

$$\begin{aligned} p(\boldsymbol{f}|\boldsymbol{u}) &= \mathcal{N}\left(\boldsymbol{f}|K_{nm}K_{mm}^{-1}\boldsymbol{u}, \widetilde{K}\right) \\ p(\boldsymbol{u}) &= \mathcal{N}\left(\boldsymbol{u}|0, K_{mm}\right), \end{aligned} \quad (6)$$

where K_{mm} is the kernel matrix resulting from evaluating the kernel function between all inducing inputs, K_{nm} is the cross-kernel matrix between inducing inputs and training points and $\widetilde{K} = K_{nn} - K_{nm}K_{mm}^{-1}K_{mn}$. Including the inducing points in our model gives the augmented joint distribution

$$p(\boldsymbol{y},\boldsymbol{\omega},\boldsymbol{f},\boldsymbol{u}) = p(\boldsymbol{y}|\boldsymbol{\omega},\boldsymbol{f})p(\boldsymbol{\omega})p(\boldsymbol{f}|\boldsymbol{u})p(\boldsymbol{u}) \quad (7)$$

Note that the original model (2) can be recovered by marginalizing $\boldsymbol{\omega}$ and $\boldsymbol{u}$.

4 Inference

The goal of Bayesian inference is to compute the posterior of the latent model variables. Because this problem is intractable for the model at hand, we employ variational inference to map the inference problem to a feasible optimization problem. We first chose a family of tractable variational distributions and select the best candidate by minimizing the Kullback-Leibler divergence between the variational distribution and the posterior. This is equivalent to optimizing a lower bound on the marginal likelihood, known as evidence lower bound (ELBO) (Jordan et al. 1999; Wainwright and Jordan 2008).

In the following we develop a stochastic variational inference (SVI) algorithm that enables stochastic optimization based on natural gradient updates which are given in closed-form.

4.1 Why use natural gradients?

Using the natural gradient over the standard Euclidean gradient is favorable since natural gradients are invariant to reparameterization of the variational family (Amari and Nagaoka 2007; Martens 2017) and provide effective second-order optimization updates (Amari 1998; Hoffman et al. 2013).

The superiority of using natural gradients in our approach can be explained by the following. We reformulate the GP classification model as an augmented model which is conditionally conjugate. When using a learning rate of one, the natural gradient updates correspond to block-coordinate ascent updates, i.e. in each iteration each parameter is set to its optimal value given the remaining parameters (see appendix A.4 and Hoffman et al. (2013)). In practice, we employ stochastic variational inference, i.e. we only use mini-batches of the data to obtain a noisy version of the natural gradient. In this setting, learning rates slightly less than one have to be chosen.

This is in contrast to former natural gradient based approaches, e.g. (Salimbeni, Eleftheriadis, and Hensman 2018), that focus on the original non-conjugate GP classification model. Although they benefit from using natural gradients, they have the disadvantage that their updates do not correspond to coordinate-ascent updates. Thus, learning rates that are much smaller that one have to be used to assure convergence.

Therefore, in our approach, we can use much higher learning rates and optimization is faster and more stable which we demonstrate in the experiments.

4.2 Variational approximation

We aim to approximate the posterior of the inducing points $p(\boldsymbol{u}|\boldsymbol{y})$ and apply the methodology of variational inference to the marginal joint distribution $p(y,\omega,u) = p(\boldsymbol{y}|\boldsymbol{\omega},\boldsymbol{u})p(\boldsymbol{\omega})p(\boldsymbol{u})$. Following a similar approach as Hensman and Matthews (2015), we apply Jensen's inequality to obtain a tractable lower bound on the log-likelihood of the labels

$$\begin{aligned} \log p(\boldsymbol{y}|\boldsymbol{\omega},\boldsymbol{u}) &= \log \mathbb{E}_{p(f|u)}[p(\boldsymbol{y}|\boldsymbol{\omega},f)] \\ &\geq \mathbb{E}_{p(f|u)}[\log p(\boldsymbol{y}|\boldsymbol{\omega},f)]. \end{aligned} \quad (8)$$

By this inequality we construct a variational lower bound on the evidence

$$\begin{aligned}\log p(\boldsymbol{y}) &\geq \mathbb{E}_{q(\boldsymbol{u},\boldsymbol{\omega})}[\log p(\boldsymbol{y}|\boldsymbol{u},\boldsymbol{\omega})] - \mathrm{KL}\left(q(\boldsymbol{u},\boldsymbol{\omega})||p(\boldsymbol{u},\boldsymbol{\omega})\right) \\ &\geq \mathbb{E}_{p(\boldsymbol{f}|\boldsymbol{u})q(\boldsymbol{u})q(\boldsymbol{\omega})}[\log p(\boldsymbol{y}|\boldsymbol{\omega},\boldsymbol{f})] \\ &\quad - \mathrm{KL}\left(q(\boldsymbol{u},\boldsymbol{\omega})||p(\boldsymbol{u},\boldsymbol{\omega})\right) \\ &=: \mathcal{L},\end{aligned}$$

where the first inequality is the usual evidence lower bound (ELBO) in variational inference and the second inequality is due to (8).

We follow a structured mean-field approach (Wainwright and Jordan 2008) and assume independence between the inducing variables u and Pólya-Gamma variables ω, yielding a variational distribution of the form $q(u,\omega) = q(u)q(\omega)$. Setting the functional derivative of $\mathcal{L}$ w.r.t. $q(u)$ and $q(\omega)$ to zero, respectively, results in the following consistency condition for the maximum,

$$q(\boldsymbol{u},\boldsymbol{\omega}) = q(\boldsymbol{u})\prod_i q(\omega_i), \tag{9}$$

with $q(\omega_i) = \mathrm{PG}(\omega_i|1,c_i)$ and $q(\boldsymbol{u}) = \mathcal{N}(\boldsymbol{u}|\boldsymbol{\mu},\Sigma)$. Remarkably, we do not have to use the full Pólya-Gamma class $\mathrm{PG}(\omega_i|b_i,c_i)$, but instead consider the restricted class $b_i = 1$ since it already contains the optimal distribution.

We use (9) as variational family which is parameterized by the variational parameters $\{\boldsymbol{\mu},\Sigma,\boldsymbol{c}\}$ and obtain a closed-form expression of the variational bound

$$\begin{aligned}&\mathcal{L}(\boldsymbol{c},\boldsymbol{\mu},\Sigma) \\ &= \mathbb{E}_{p(\boldsymbol{f}|\boldsymbol{u})q(\boldsymbol{u})q(\boldsymbol{\omega})}[\log p(\boldsymbol{y}|\boldsymbol{\omega},\boldsymbol{f})] - \mathrm{KL}\left(q(\boldsymbol{u},\boldsymbol{\omega})||p(\boldsymbol{u},\boldsymbol{\omega})\right) \\ &\overset{\mathsf{C}}{=} \frac{1}{2}\Bigg(\log|\Sigma| - \log|K_{mm}|) - \mathrm{tr}(K_{mm}^{-1}\Sigma) - \boldsymbol{\mu}^\top K_{mm}^{-1}\boldsymbol{\mu} \\ &\quad + \sum_i \Big\{ y_i\boldsymbol{\kappa}_i\boldsymbol{\mu} - \theta_i\left(\widetilde{K}_{ii} - \boldsymbol{\kappa}_i\Sigma\boldsymbol{\kappa}_i^\top - \boldsymbol{\mu}^\top\boldsymbol{\kappa}_i^\top\boldsymbol{\kappa}_i\boldsymbol{\mu}\right) \\ &\quad + c_i^2\theta_i - 2\log\cosh\frac{c_i}{2}\Big\}\Bigg),\end{aligned} \tag{10}$$

where $\theta_i = \frac{1}{2c_i}\tanh\left(\frac{c_i}{2}\right)$ and $\boldsymbol{\kappa}_i = K_{im}K_{mm}^{-1}$. Remarkably, all intractable terms involving expectations of $\log\mathrm{PG}(\omega_i|1,0)$ cancel out. Details are provided in appendix A.2.

4.3 Stochastic variational inference

Our algorithm alternates between updates of the local variational parameters $\boldsymbol{c}$ and global parameters $\boldsymbol{\mu}$ and Σ. In each iteration we update the parameters based on a mini-batch of the data $\mathcal{S} \subset \{1,...,n\}$ of size $s = |\mathcal{S}|$.

We update the *local parameters* $\boldsymbol{c}_\mathcal{S}$ in the mini-batch $\mathcal{S}$ by employing coordinate ascent. To this end, we fix the global parameters and analytically compute the unique maximum of (10) w.r.t. the local parameters, leading to the updates

$$c_i = \sqrt{\widetilde{K}_{ii} + \boldsymbol{\kappa}_i\Sigma\boldsymbol{\kappa}_i^\top + \boldsymbol{\mu}^\top\boldsymbol{\kappa}_i^\top\boldsymbol{\kappa}_i\boldsymbol{\mu}} \tag{11}$$

for $i \in \mathcal{S}$.

We update the *global parameters* by employing stochastic optimization of the variational bound (10). The optimization is based on stochastic estimates of the natural gradients of the global parameters. We use the natural parameterization of the variational Gaussian distribution, i.e., the parameters $\boldsymbol{\eta}_1 := \Sigma^{-1}\boldsymbol{\mu}$ and $\eta_2 = -\frac{1}{2}\Sigma^{-1}$. Using the natural parameters results in simpler and more effective updates. The natural gradients based on the mini-batch $\mathcal{S}$ are given by

$$\begin{aligned}\widetilde{\nabla}_{\boldsymbol{\eta}_1}\mathcal{L}_\mathcal{S} &= \frac{n}{2s}\boldsymbol{\kappa}_\mathcal{S}^\top\boldsymbol{y}_\mathcal{S} - \boldsymbol{\eta}_1 \\ \widetilde{\nabla}_{\eta_2}\mathcal{L}_\mathcal{S} &= -\frac{1}{2}\left(K_{mm}^{-1} + \frac{n}{s}\boldsymbol{\kappa}_\mathcal{S}^\top\Theta_\mathcal{S}\boldsymbol{\kappa}_\mathcal{S}\right) - \eta_2,\end{aligned} \tag{12}$$

where $\Theta = \mathrm{diag}(\boldsymbol{\theta})$ and $\theta_i = \frac{1}{2c_i}\tanh\left(\frac{c_i}{2}\right)$. The factor $\frac{n}{s}$ is due to the rescaling of the mini-batches. The global parameters are updated according to a stochastic natural gradient ascent scheme. We employ the adaptive learning rate method described by Ranganath et al. (2013).

The natural gradient updates always lead to a positive definite covariance matrix[2] and in contrast to Hensman and Matthews (2015) our implementation does not require any assurance for positive-definiteness of the variational covariance matrix Σ. Details for the derivation of the updates can be found in appendix A.3. The complexity of each iteration in the inference scheme is $\mathcal{O}(m^3)$, due to the inversion of the matrix η_2.

On the quality of the approximation In other applications of variational inference to GP classification, one tries to approximate the posterior directly by a Gaussian $q^*(f)$ which minimizes the Kullback-Leibler divergence between the variational distribution and the true posterior (Hensman and Matthews 2015). On the other hand, in our paper, we apply variational inference to the augmented model, looking for the best distribution that factorizes in the Pólya-Gamma variables ω_i and the original function f. This approach also yields a Gaussian approximation $q(f)$ as a factor in the optimal density. Of course $q(f)$ will be different from the 'optimal' $q^*(f)$. We could however argue that asymptotically, in the limit of a large number of data, the predictions given by both densities may not be too different, as the posterior uncertainty for both densities should become small (Opper and Archambeau 2009).

It would be interesting to see how the ELBOs of the two variational approaches, which both give a lower bound on the likelihood of the data, differ. Unfortunately, such a computation would require the knowledge of the optimal $q^*(f)$. However, we can obtain some estimate of this difference when we assume that we use the *same* Gaussian density $q(f)$ for both bounds as an approximation. In this case, we obtain

$$\mathcal{L}_{\text{orig}} - \mathcal{L}_{\text{augmented}} = \mathbb{E}_{q(f)}[\mathrm{KL}\left(q(\omega)||p(\omega|f,y)\right)].$$

This lower bound on the gap is small if on average the variational approximation $q(\omega)$ is close to the posterior $p(\omega|f,y)$. For the sake of simplicity we consider here the non-sparse case, i.e. the inducing points equal the training points ($f = u$). However, it is straight-forward to extend the results also to the sparse case.

We empirically investigate the quality of our approximation in experiment 5.1.

[2]This follows directly since K_{mm} and Θ are positive definite.

Predictions The approximate posterior of the GP values and inducing variables is given by $q(\boldsymbol{f}, \boldsymbol{u}) = p(\boldsymbol{f}|\boldsymbol{u})q(\boldsymbol{u})$, where $q(\boldsymbol{u}) = \mathcal{N}(\boldsymbol{u}|\boldsymbol{\mu}, \Sigma)$ denotes the optimal variational distribution. To predict the latent function values f_* at a test point x_* we substitute our approximate posterior into the standard predictive distribution

$$\begin{aligned} p(f_*|y) &= \int p(f_*|\boldsymbol{f}, \boldsymbol{u})p(\boldsymbol{f}, \boldsymbol{u}|y)\mathrm{d}\boldsymbol{f}\mathrm{d}\boldsymbol{u} \\ &\approx \int p(f_*|\boldsymbol{f}, \boldsymbol{u})p(\boldsymbol{f}|\boldsymbol{u})q(\boldsymbol{u})\mathrm{d}\boldsymbol{f}\mathrm{d}\boldsymbol{u} \\ &= \int p(f_*|\boldsymbol{u})q(\boldsymbol{u})\mathrm{d}\boldsymbol{u} = \mathcal{N}\left(f^*|\mu_*,\ \sigma_*^2\right), \end{aligned} \tag{13}$$

where the prediction mean is $\mu_* = K_{*m}K_{mm}^{-1}\boldsymbol{\mu}$ and the variance $\sigma_*^2 = K_{**} + K_{*m}K_{mm}^{-1}(\Sigma K_{mm}^{-1} - I)K_{m*}$. The matrix K_{*m} denotes the kernel matrix between the test point and the inducing points and K_{**} the kernel value of the test point. The distribution of the test labels is easily computed by applying the logit link function to (13),

$$p(y_* = 1|y) = \int \sigma(f_*)p(f_*|y)\mathrm{d}f_*. \tag{14}$$

This integral is analytically intractable but can be computed numerically by quadrature methods. This is adequate and fast since the integral is only one-dimensional.

Computing the mean and the variance of the predictive distribution has complexity $\mathcal{O}(m)$ and $\mathcal{O}(m^2)$, respectively.

Optimization of the hyperparameters We select the optimal kernel hyperparameters by maximizing the marginal likelihood $p(y|h)$, where h denotes the set of hyperparameters (this approach is called empirical Bayes (Maritz and Lwin 1989)). We follow an approximate approach and optimize the fitted variational lower bound $\mathcal{L}(h)$ (10) as a function of h by alternating between optimization steps w.r.t. the variational parameters and the hyperparameters (Mandt, Hoffman, and Blei 2016).

5 Experiments

We compare our proposed method, efficient Gaussian process classification (X-GPC), with the state-of-the-art methods SVGPC (Salimbeni, Eleftheriadis, and Hensman 2018), provided in the package GPflow[3] (Matthews et al. 2017), which builds on TensorFlow and the EP approach EPGPC by Hernández-Lobato and Hernández-Lobato (2016), implemented in R. All methods are applied to real-world datasets containing up to 11 million data points.

In all experiments a squared exponential covariance function with a common length scale parameter for each dimension, an amplitude parameter and an additive noise parameter is used. The kernel hyperparameters are initialized to the same values and optimized using Adam (Kingma and Ba 2014), while inducing points location are initialized via k-means++ (Arthur and Vassilvitskii 2007) and kept fixed during training. The SVI based methods, X-GPC and SVGPC, use an adaptive learning rate. All algorithms are run on a single CPU. We experiment on 12 datasets from the OpenML website and the UCI repository ranging from 768 to 11 million data points. In the first experiment (section 5.1), we examine the quality of the approximation provided by X-GPC. In the next experiment, we evaluate the prediction performance and run time of X-GPC and SVGPC and EPGPC on several real-world datasets. Finally, in 5.3, we examine the sensitivity of all methods to the number of inducing points.

[3]We use GPflow version 1.2.0.

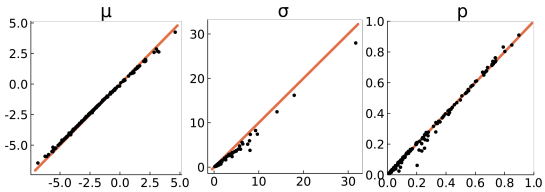

Figure 1: Posterior mean (μ), variance (σ) and predictive marginals (p) of the Diabetes dataset. Each plot shows the MCMC ground truth on the x-axis and the estimated value of our model on the y-axis. Our approximation is very close to the ground truth.

5.1 Quality of the approximation

We empirically examine the quality of the variational approximation provided by our method. In Fig. 1, we compare the approximations to the true posterior obtained by employing an asymptotically correct Gibbs sampler (Polson and Scott 2011; Linderman, Johnson, and Adams 2015). We compare the posterior mean and variance as well as the prediction probabilities with the ground truth. Since the Gibbs sampler does not scale to large datasets we experiment on the small Diabetes dataset. In Fig. 1 we plot the approximated values vs. the ground truth. We find that our approximation is very close to the true posterior.

5.2 Numerical comparison

We evaluate the prediction performance and run time of our method X-GPC and the competing methods SVGPC and EPGPC. We experiment on a variety of different datasets and report the resulting prediction error, negative test log-likelihood and run time for each method in table 1.

The experiments are conducted as follows. For each dataset we perform a 10-fold cross-validation and for datasets with more than 1 million points, we limit the test set to 100,000 points. We report the average prediction error, the negative test log-likelihood (14) and the run time along with one standard deviation. For all datasets, we use 100 inducing points and a mini-batch size of 100 points.

For X-GPC we find that the following simple convergence criterion on the global parameters leads to good results: a sliding window average being smaller than a threshold of 10^{-4} . Unfortunately, the original implementations of SVGPC and EPGPC do not include a convergence criterion. We find that the trajectories of the global parameters of SVGPC tend to be noisy, and using a convergence criterion on the global parameters often leads to poor results. To have

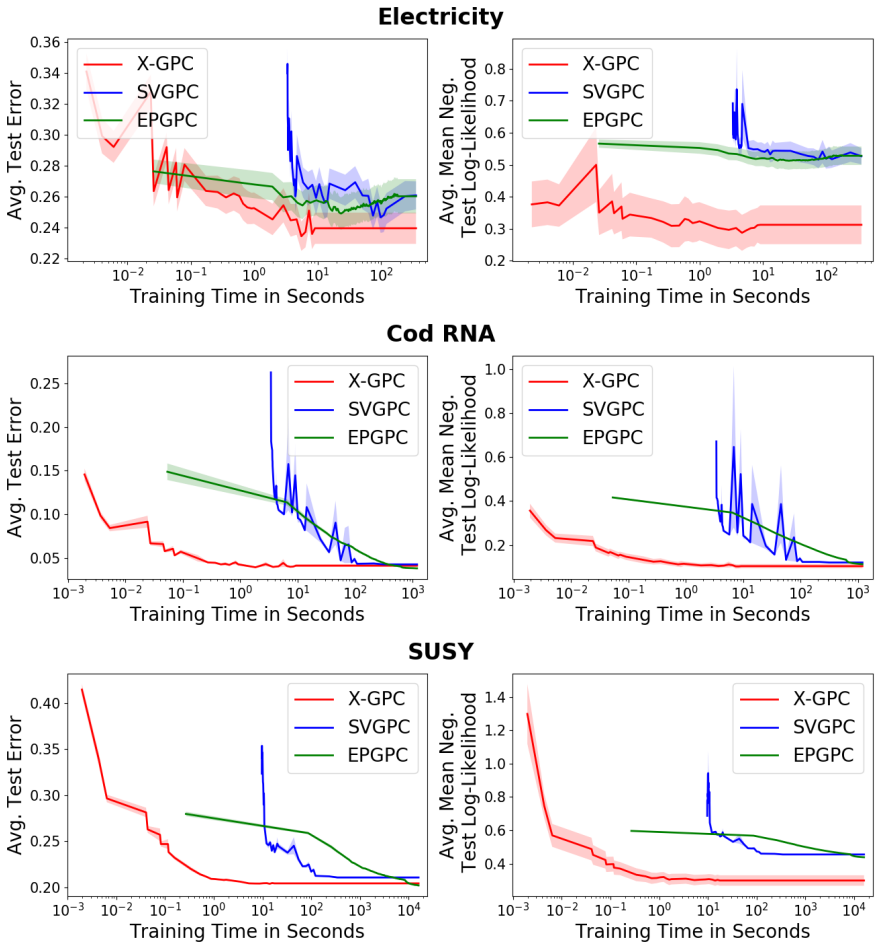

Figure 2: Average negative test log-likelihood and average test prediction error as a function of training time (seconds in a $\log_{10}$ scale) on the datasets Electricity (45,312 points), Cod RNA (343,564 points) and SUSY (5 million points). X-GPC (proposed) reaches values close to the optimum after only a few iterations, whereas SVGPC and EPGPC are one to two orders of magnitude slower.

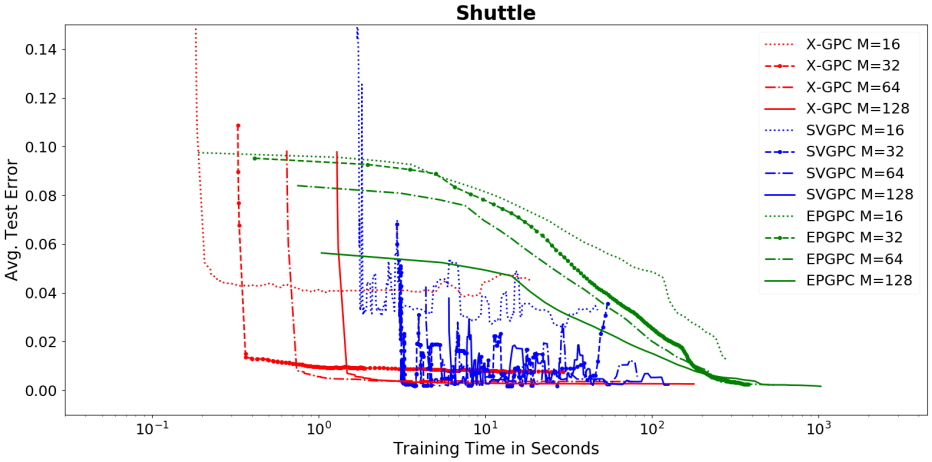

Figure 3: Prediction error as function of training time (on a $\log_{10}$ scale) for the Shuttle dataset. Different numbers of inducing points are considered, $M = 16, 32, 64, 128$. X-GPC (proposed) converges the fastest in all settings of different numbers of inducing points. Using only 32 inducing points is enought for obtaining allmost optimal prediction performance for all methods, but SVGPC becomes instable in settings of less than 128 inducing points.

Dataset		**X-GPC**	SVGPC	EPGPC
aXa	Error	$\mathbf{0.17 \pm 0.07}$	$\mathbf{0.17 \pm 0.07}$	$\mathbf{0.17 \pm 0.07}$
$n = 36,974$	NLL	$\mathbf{0.29 \pm 0.13}$	0.36 ± 0.13	0.34 ± 0.13
$d = 123$	Time	$\mathbf{47 \pm 2.2}$	451 ± 7.8	214 ± 4.8
Bank Market.	Error	0.14 ± 0.12	$\mathbf{0.12 \pm 0.12}$	$\mathbf{0.12 \pm 0.13}$
$n = 45,211$	NLL	$\mathbf{0.27 \pm 0.22}$	0.31 ± 0.26	0.33 ± 0.20
$d = 43$	Time	$\mathbf{9 \pm 1.5}$	205 ± 6.6	46 ± 3.5
Click Pred.	Error	$\mathbf{0.17 \pm 0.00}$	$\mathbf{0.17 \pm 0.00}$	$\mathbf{0.17 \pm 0.01}$
$n = 399,482$	NLL	$\mathbf{0.39 \pm 0.07}$	0.46 ± 0.00	0.46 ± 0.01
$d = 12$	Time	$\mathbf{4.5 \pm 1.3}$	102 ± 3.0	8.1 ± 0.45
Cod RNA	Error	$\mathbf{0.04 \pm 0.00}$	$\mathbf{0.04 \pm 0.00}$	$\mathbf{0.04 \pm 0.00}$
$n = 343,564$	NLL	$\mathbf{0.11 \pm 0.03}$	0.13 ± 0.00	0.12 ± 0.00
$d = 8$	Time	$\mathbf{3.7 \pm 0.13}$	115 ± 4.3	869 ± 5.2
Diabetes	Error	$\mathbf{0.23 \pm 0.07}$	$\mathbf{0.23 \pm 0.06}$	0.24 ± 0.06
$n = 768$	NLL	$\mathbf{0.47 \pm 0.11}$	0.47 ± 0.10	0.48 ± 0.09
$d = 8$	Time	8.8 ± 0.12	150 ± 5.1	$\mathbf{8 \pm 0.45}$
Electricity	Error	$\mathbf{0.24 \pm 0.06}$	0.26 ± 0.06	0.26 ± 0.06
$n = 45,312$	NLL	$\mathbf{0.31 \pm 0.17}$	0.53 ± 0.08	0.53 ± 0.06
$d = 8$	Time	$\mathbf{8.2 \pm 0.48}$	356 ± 6.9	13.5 ± 1.50
German	Error	$\mathbf{0.25 \pm 0.12}$	$\mathbf{0.25 \pm 0.11}$	0.26 ± 0.13
$n = 1,000$	NLL	$\mathbf{0.44 \pm 0.17}$	0.51 ± 0.15	0.53 ± 0.11
$d = 20$	Time	17 ± 0.42	374 ± 7.3	$\mathbf{5.2 \pm 0.03}$
Higgs	Error	$\mathbf{0.33 \pm 0.01}$	0.45 ± 0.01	0.38 ± 0.01
$n = 11,000,000$	NLL	$\mathbf{0.55 \pm 0.13}$	0.69 ± 0.00	0.66 ± 0.00
$d = 28$	Time	$\mathbf{23 \pm 0.88}$	294 ± 54	8732 ± 867
IJCNN	Error	0.03 ± 0.01	0.06 ± 0.01	$\mathbf{0.02 \pm 0.01}$
$n = 141,691$	NLL	0.10 ± 0.03	0.15 ± 0.07	$\mathbf{0.09 \pm 0.04}$
$d = 22$	Time	$\mathbf{17 \pm 0.44}$	1033 ± 45	756 ± 8.6
Mnist	Error	0.14 ± 0.01	0.44 ± 0.13	$\mathbf{0.12 \pm 0.01}$
$n = 70,000$	NLL	$\mathbf{0.24 \pm 0.10}$	0.66 ± 0.11	0.27 ± 0.01
$d = 780$	Time	$\mathbf{200 \pm 5.5}$	991 ± 23	806 ± 5.2
Shuttle	Error	$\mathbf{0.01 \pm 0.01}$	$\mathbf{0.01 \pm 0.00}$	$\mathbf{0.01 \pm 0.01}$
$n = 58,000$	NLL	$\mathbf{0.07 \pm 0.01}$	$\mathbf{0.07 \pm 0.00}$	$\mathbf{0.07 \pm 0.01}$
$d = 9$	Time	$\mathbf{0.01 \pm 0.00}$	7.5 ± 0.7	100 ± 0.63
SUSY	Error	$\mathbf{0.21 \pm 0.00}$	0.22 ± 0.00	0.22 ± 0.00
$n = 5,000,000$	NLL	$\mathbf{0.31 \pm 0.10}$	0.49 ± 0.01	0.50 ± 0.00
$d = 18$	Time	$\mathbf{14 \pm 0.29}$	$10,000$	$10,000$
wXa	Error	$\mathbf{0.03 \pm 0.01}$	0.04 ± 0.01	$\mathbf{0.03 \pm 0.01}$
$n = 34,780$	NLL	0.27 ± 0.07	0.25 ± 0.07	$\mathbf{0.19 \pm 0.06}$
$d = 300$	Time	66 ± 16	612 ± 11	$\mathbf{1.4 \pm 0.10}$

Table 1: Average test prediction error, negative test log-likelihood (NLL) and time in seconds along with one standard deviation. Best values are highlighted.

a fair comparison, we therefore monitor the convergence of the prediction performance on a hold-out set and use a sliding window average of size 5 and threshold 10^{-3} as convergence criterion for all methods.

We observe that X-GPC is about one to two orders of magnitude faster than SVGPC and EPGPC on most datasets. Only on the dataset wXa, EPGPC is slightly faster than X-GPC. The prediction error is similar for all methods but X-GPC outperforms the competitors in terms of the test log-likelihood on most datasets (aXa, Bank Marketing, Click Prediction, Cod RNA, Diabetes, Electricity, German, Higgs, Mnist, SUSY). This means that the confidence levels in the predictions are better calibrated for X-GPC, i.e. when predicting a wrong label SVGPC and EPGPC tend to be more confident than X-GPC.

Performance as a function of time Since all considered methods are based on an optimization schemes, there is a trade-off between the run time of the algorithm and the prediction performance. We make this trade-off transparent by plotting the prediction performance as function of time on each dataset. For each method we monitor on a 10-fold cross-validation the average negative test log-likelihood and prediction error on a hold-out test set as a function of time.

The results are displayed in Fig. 2 for three selected datasets, while the results for the remaining datasets are deferred to appendix A.1. For all datasets we observe that after a few iterations X-GPC is already close to the optimum due to its efficient closed form natural gradient updates. Both the prediction error and test log-likelihood converge around one to two orders of magnitude faster for X-GPC than for SVGPC and EPGPC. Moreover, the performance curves tend to be noisier for SVGPC than for X-GPC and EPGPC. For the datasets HIGGS and IJCNN, EPGPC lead to slightly better final prediction performance, but with the cost of a runtime being up to 4 orders of magnitude slower than X-GPC (approx. 28 hours vs. 9 and 435 seconds, respectively).

All three methods are implemented in different programming frameworks: X-GPC in Julia, SVGPC in TensorFlow and EPGPC in R leading to different efficient implementations. However, we find that the main speed-up of our method is due to the efficient natural gradient updates and only marginally related to the usage of a different programming language. To check this we implemented EPGPC also in Julia and obtained similar runtimes. Since SVGPC is part of the highly optimized GPflow package we only used the original implementation.

5.3 Inducing points

We examine the effect of different numbers of inducing points on the prediction performance and run time. For all methods we compare different numbers of inducing points: $M = 16, 32, 64, 128$. For each setting, we perform a 10-fold cross validation on the Shuttle dataset and plot the mean prediction error as function of time. The results are displayed in Fig. 3. We observe that the higher the number of inducing points, the better the prediction performance, but the longer the run time. Throughout all settings of inducing points our method is consistently faster of around one to two orders of magnitude than the competitors. On the Shuttle dataset using only $M = 32$ inducing points is enough and can only be marginally improved by using more inducing point for all methods. However, the performance curves of SVGPC are instable when using less than 128 inducing points.

6 Conclusions

We proposed an efficient Gaussian process classification method that builds on Pólya-Gamma data augmentation and inducing points. The experimental evaluations shows that our method is up to two orders of magnitude faster than the state-of-the-art approach while being competitive in terms of prediction performance. Speed improvements are due to the Pólya-Gamma data augmentation approach that enables efficient second order optimization.

The presented work shows how data augmentation can speed up variational approximation of GPs. Our analysis may pave the way for using data augmentation to derive efficient stochastic variational algorithms also for variational Bayesian models other than GPs. Furthermore, future work

may aim at extending the approach to multi-class and multi-label classification.

Acknowledgements We thank Stephan Mandt, James Hensman and Scott W. Linderman for fruitful discussions. This work was partly funded by the German Research Foundation (DFG) awards KL 2698/2-1 and GRK1589/2 and the by the Federal Ministry of Science and Education (BMBF) awards 031L0023A, 01IS18051A.

References

Amari, S., and Nagaoka, H. 2007. *Methods of Information Geometry*. American Mathematical Society.

Amari, S. 1998. Natural grad. works efficiently in learning. *Neural Computation*.

Arthur, D., and Vassilvitskii, S. 2007. k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, 1027–1035. Society for Industrial and Applied Mathematics.

Dezfouli, A., and Bonilla, E. V. 2015. Scalable inference for gaussian process models with black-box likelihoods. In *NIPS*, 1414–1422.

Dragiev, S.; Toussaint, M.; and Gienger, M. 2011. Gaussian process implicit surfaces for shape estimation and grasping. In *Robotics and Automation (ICRA)*, 2845–2850.

Gibbs, M. N., and MacKay, D. J. C. 2000. Variational Gaussian process classifiers. *IEEE Transactions on Neural Networks* 11(6):1458–1464.

Hensman, J., and Matthews, A. 2015. Scalable Variational Gaussian Process Classification. In *AISTATS*.

Hernández-Lobato, D., and Hernández-Lobato, J. M. 2016. Scalable gaussian process classification via expectation propagation. In *AISTATS*.

Hoffman, M. D.; Blei, D. M.; Wang, C.; and Paisley, J. 2013. Stochastic Variational Inference. *Journal of Machine Learning Research*.

Honkela, A.; Raiko, T.; Kuusela, M.; Tornio, M.; and Karhunen, J. 2010. Approximate riemannian conjugate gradient learning for fixed-form variational bayes. *Journal of Machine Learning Research* 11.

Izmailov, P.; Novikov, A.; and Kropotov, D. 2018. Scalable gaussian processes with billions of inducing inputs via tensor train decomposition. In *AISTATS*, 726–735.

Jähnichen, P.; Wenzel, F.; Kloft, M.; and Mandt, S. 2018. Scalable generalized dynamic topic models. In *AISTATS*.

John Walker, S. 2014. *Big data: A revolution that will transform how we live, work, and think*. Taylor & Francis.

Jordan, M. I.; Ghahramani, Z.; Jaakkola, T. S.; and Saul, L. K. 1999. An Introduction to Variational Methods for Graphical Models. *Machine Learning*.

Khan, M. E., and Nielsen, D. 2018. Fast yet simple natural-gradient descent for variational inference in complex models. *Arxiv Preprint*.

Kingma, D. P., and Ba, J. 2014. Adam: A method for stochastic optimization. *CoRR* abs/1412.6980.

Linderman, S. W.; Johnson, M. J.; and Adams, R. P. 2015. Dependent multinomial models made easy: Stick-breaking with the polya-gamma augmentation. In *NIPS*.

Mandt, S.; Wenzel, F.; Nakajima, S.; Cunningham, J. P.; Lippert, C.; and Kloft, M. 2017. Sparse Probit Linear Mixed Model. *Machine Learning Journal*.

Mandt, S.; Hoffman, M.; and Blei, D. 2016. A Variational Analysis of Stochastic Gradient Algorithms. *ICML*.

Maritz, J., and Lwin, T. 1989. Empirical Bayes Methods with Applications. *Monographs on Statistics and Applied Probability.*

Martens, J. 2017. New insights and perspectives on the natural gradient method. *Arxiv Preprint*.

Matthews, A. G. d. G.; van der Wilk, M.; Nickson, T.; Fujii, K.; Boukouvalas, A.; León-Villagrá, P.; Ghahramani, Z.; and Hensman, J. 2017. GPflow: A Gaussian process library using TensorFlow. *Journal of Machine Learning Research*.

Opper, M., and Archambeau, C. 2009. The variational gaussian approximation revisited. *Neural Comput.* 21(3):786–792.

Polson, N. G., and Scott, S. L. 2011. Data augmentation for support vector machines. *Bayesian Anal.*

Polson, N. G.; Scott, J. G.; and Windle, J. 2013. Bayesian inference for logistic models using pólya–gamma latent variables. *Journal of the American Statistical Association* 108(504):1339–1349.

Ranganath, R.; Wang, C.; Blei, D. M.; and Xing, E. P. 2013. An Adaptive Learning Rate for Stochastic Variational Inference. *ICML*.

Rasmussen, C. E., and Williams, C. K. I. 2005. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press.

Salimbeni, H.; Eleftheriadis, S.; and Hensman, J. 2018. Natural gradients in practice: Non-conjugate variational inference in gaussian process models. In *AISTATS*.

Scott, J. G., and Sun, L. 2013. Expectation-maximization for logistic regression. *arXiv preprint arXiv:1306.0040*.

Snelson, E., and Ghahramani, Z. 2006. Sparse GPs using Pseudo-inputs. *NIPS*.

Stein, M. L. 2012. *Interpolation of spatial data: some theory for kriging*. Springer Science & Business Media.

Wainwright, M. J., and Jordan, M. I. 2008. Graphical models, exponential families, and variational inference. *Found. Trends Mach. Learn.* 1–305.

Wenzel, F.; Galy-Fajou, T.; Deutsch, M.; and Kloft, M. 2017. Bayesian nonlinear support vector machines for big data. In *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*.