1st Symposium on Advances in Approximate Bayesian Inference, 2018 1–12

Scalable Multi-Class Gaussian Process Classification via Data Augmentation

Théo Galy-Fajou[1], **Florian Wenzel**[2], **Christian Donner**[1] **and Manfred Opper**[1]
[1] *TU Berlin,* [2] *TU Kaiserslautern*
Contact: galy-fajou@tu-berlin.de

Abstract

This paper proposes a new scalable multi–class Gaussian process classification approach building on a novel modified softmax likelihood function. This form of likelihood allows for a latent variable augmentation that leads to a conditionally conjugate model and enables efficient variational inference via block coordinate ascent updates. Our experiments show that our method outperforms state-of-the-art Gaussian process based methods in terms of speed while achieving competitive prediction performance.

Keywords: Gaussian Process, Multi-Class Classification, Data Augmentation, Pólya-Gamma, Variational Inference

1. Introduction

Gaussian processes (GPs) provide a Bayesian non-linear and non-parametric approach to classification (Williams and Barber, 1998; Rasmussen and Williams, 2005). In the easier setting of *binary* classification, GPs can be easily applied to big datasets using variational inference methods (Hensman and Matthews, 2015; Wenzel et al., 2018).

Aiming to this scalability for *multi-class* classification is however more complicated since it involves not only one latent GP but, one GP for each class. In most multi-class likelihoods, the GPs are coupled, forbidding a direct application of variational inference techniques. There have been several works addressing the problem (Williams and Barber, 1998; Kim and Ghahramani, 2006; Riihimäki et al., 2013), but do not scale well with the size of the data points, while more recent scalable methods (Villacampa-Calvo and Hernández-Lobato, 2017; Hensman and Matthews, 2015; Buchholz et al., 2018) rely on approximations of the likelihood.

We propose a new multi-class GP classification model using a modification of the softmax likelihood building on binary logistic functions. By applying various variable augmentations, we obtain a conditionally conjugate model. This allows us to derive an efficient inference algorithm based on stochastic variational inference and closed-form block coordinate-ascent updates.

2. Conjugate Multi-Class Gaussian Process Classification

The logistic-softmax GP model

We consider a dataset of N data points $\boldsymbol{X} = (\boldsymbol{x}_1, \dots, \boldsymbol{x}_N)$ with labels $\boldsymbol{y} = (y_1, \dots, y_N)$, where $y_i \in \{1, \dots, C\}$ and C is the total number of classes. The multi-class GP classification

model consists of a latent GP prior for each class $\boldsymbol{f} = (f^1, \ldots, f^C)$, where $f^c \sim \mathrm{GP}(0, k^c)$ and k^c is the corresponding kernel function. The labels are modeled by a categorical likelihood $p(y_i = k|\boldsymbol{x}_i, \boldsymbol{f}_i) = g^k(\boldsymbol{f}(\boldsymbol{x}_i))$, where $g^k(f)$ is a function that maps the real vector of the GP values to a probability vector.

The most common way to form a categorical likelihood is through the softmax transformation $p(y_i = k|\boldsymbol{f}_i) = \exp\left(f_i^k\right) / \sum_{c=1}^C \exp\left(f_i^c\right)$, where we use the shorthand $f_i^c = f^c(x_i)$ and for the sake of clarity we omit the conditioning on x_i.

In this work, we propose the *logistic-softmax :*

$$p(y_i = k|\boldsymbol{f}_i) = \frac{\sigma\left(f_i^k\right)}{\sum_{c=1}^C \sigma\left(f_i^c\right)}, \tag{1}$$

where $\sigma(z) = (1+\exp(-z))^{-1}$ is the logistic function. It can be interpreted as the standard softmax applied to a non-linearly transformed GP, i.e. $p(y_i|\boldsymbol{f}_i) = \mathrm{softmax}(\log \sigma(\boldsymbol{f}_i))$. Note that the likelihood reduces to the binary logistic likelihood for $C = 2$, by setting the symmetry $f^2 = -f^1$.

We now follow a data augmentation strategy (Linderman et al., 2015; Wenzel et al., 2017) and expand our likelihood (1) by three augmentations to obtain a *conditionally conjugate model*.

Augmentation 1: Gamma augmentation.
To remedy the intractable normalizer term we make use of the integral identity $\frac{1}{z} = \int_0^\infty \exp(-\lambda z) d\lambda$ and express the likelihood (1) as

$$p(y_i = k|\boldsymbol{f}_i) = \frac{\sigma\left(f_i^k\right)}{\sum_{c=1}^C \sigma\left(f_i^c\right))} = \sigma\left(f_i^k\right) \int_0^\infty \exp\left(-\lambda_i \sum_{c=1}^C \sigma\left(f_i^c\right)\right) d\lambda_i.$$

This augmentation is well known in the Gibbs sampling community to deal with intractable normalization constants (see e.g. (Walker, 2011)) but is not often used in the setting of variational inference. We write the likelihood as

$$p(y_i = k|\boldsymbol{f}_i, \lambda_i) = \sigma(f_i^k) \prod_{c=1}^C \exp\left(-\lambda_i \sigma(f_i^c)\right), \tag{2}$$

and impose the improper prior $p(\lambda_i) = \mathbb{1}_{[0,\infty)}(\lambda_i)$.

Augmentation 2: Poisson augmentation.
The moment generating function of a Poisson distribution $\mathrm{Po}(\cdot|\lambda)$ with mean parameter λ is $\exp(\lambda(z-1)) = \sum_{n=1}^\infty z^n \mathrm{Po}(z|\lambda)$.
Using $z = \sigma(-f)$ and the fact that $\sigma(f) = 1 - \sigma(-f)$ we rewrite the exponential factors in (2) as $\exp\left(-\lambda_i \sigma(f_i^c)\right) = \exp\left(\lambda_i(\sigma(-f_i^c) - 1)\right) = \sum_{n_i^c=0}^\infty (\sigma(-f_i^c))^{n_i^c} \mathrm{Po}(n_i^c|\lambda_i)$, which leads to the augmented likelihood

$$p(y_i = k|\boldsymbol{f}_i, \lambda_i, \boldsymbol{n}_i) = \sigma(f_i^k) \prod_{c=1}^C (\sigma(-f_i^c))^{n_i^c}, \tag{3}$$

where $\boldsymbol{n}_i = (n_i^1, \ldots, n_i^C)$ and the augmented Poisson variables are distributed as $p(n_i^c|\lambda_i) = \mathrm{Po}(n_i^c|\lambda_i)$.

Augmentation 3: Pólya-Gamma augmentation.
In the last augmentation step, we aim for a Gaussian representation of the sigmoid function.

The Pólya-Gamma representation (Polson et al., 2013) allows for rewritting the sigmoid function as a scale mixture of Gaussians: $\sigma(z)^n = \int_0^\infty 2^{-n} \exp\left(\frac{nz}{2} - \frac{z^2}{2}\omega\right) \mathrm{PG}(\omega|n,0)$, where $\mathrm{PG}(\omega|n,b)$ is a Pólya-Gamma distribution. Applying this augmentation to (3) we obtain

$$p(y_i = k|\boldsymbol{f}_i, \lambda_i, \boldsymbol{n}_i, \boldsymbol{\omega}_i, \tilde{\omega}_i^k) = \frac{1}{2}\exp\left(\frac{f_i^k}{2} - \frac{(f_i^k)^2}{2}\tilde{\omega}_i^k\right)\prod_{c=1}^{C} 2^{-n_i^c}\exp\left(-\frac{n_i^c f_i^c}{2} - \frac{(f_i^c)^2}{2}\omega_i^c\right), \quad (4)$$

where $\boldsymbol{\omega}_i = (\omega_i^1, \ldots, \omega_i^C)$. The priors over the Pólya-Gamma variables $\boldsymbol{\omega}_i$ and $\tilde{\omega}_i^k$ are $p(\boldsymbol{\omega}_i|\boldsymbol{n}_i) = \prod_{c=1}^{C}\mathrm{PG}(\omega_i^c|\boldsymbol{n}_i^c, 0)$, $p(\tilde{\omega}_i^k) = \mathrm{PG}(\tilde{\omega}_i^k|1,0)$, respectively.

The full conditional distributions.
Now, the effort of the augmentations finally pays off as the final augmented model is tractable and the complete conditional distributions are given in closed-form. The exact forms are given in the appendix A.1.

3. Scalable Inference

We aim to find a variational approximation of the augmented model posterior. In the following we augment our model with inducing points and develop a stochastic variational inference (SVI) algorithm that enables stochastic optimization based on block coordinate ascent updates which are given in an analytic closed-form.

Sparse Gaussian process
To scale our model to big datasets we follow a similar approach as Hensman and Matthews (2015) and approximate the latent GPs $\boldsymbol{f}^c$ by *sparse GPs* building on *inducing points*. This reduces the complexity to $\mathcal{O}(M^3)$, where M is the number of inducing points.

For each GP $\boldsymbol{f}^c$, we introduce M additional input-output pairs $\{(Z_1^c, u_1^c), \ldots, (Z_M^c, u_M^c)\}$, termed as *inducing inputs* and *inducing variables*. The GP values and the inducing variables are connected via $p(\boldsymbol{f}^c|\boldsymbol{u}^c) = \mathcal{N}\left(\boldsymbol{f}^c|K_{nm}^c\left(K_{mm}^c\right)^{-1}\boldsymbol{u}^c, \widetilde{K}^c\right)$, $p(\boldsymbol{u}^c) = \mathcal{N}\left(\boldsymbol{u}|\boldsymbol{0}, K_{mm}^c\right)$ where K_{mm}^c is the kernel matrix evaluated at the inducing variables locations $\{Z_1^c, \ldots, Z_M^c\}$, K_{nm}^c is the cross-kernel matrix between training points and inducing points and $\widetilde{K}^c = K_{nn}^c - K_{nm}^c\left(K_{mm}^c\right)^{-1}K_{mn}^c$.

Variational approximation
We aim to approximate the joint posterior distribution of the latent GPs $p(\boldsymbol{f}^c|\boldsymbol{y})$ for each class $c = 1, \ldots, C$ and apply the methodology of variational inference to the sparse augmented variational distribution $p(\boldsymbol{u}, \boldsymbol{\lambda}, \boldsymbol{n}, \boldsymbol{\omega}, \tilde{\boldsymbol{\omega}}|\boldsymbol{y})$.

We assume the the following structure of the variational distribution
$q(\boldsymbol{u}, \boldsymbol{\lambda}, \boldsymbol{n}, \boldsymbol{\omega}, \tilde{\boldsymbol{\omega}}) = q(\boldsymbol{u}, \boldsymbol{\lambda})q(\boldsymbol{n}, \boldsymbol{\omega}, \tilde{\boldsymbol{\omega}})$. Since our model is conditionally conjugate, the family of the optimal variational distribution can be easily determined by averaging the complete conditionals in log-space (Blei et al., 2017). The optimal variational posterior has a factorizing form $q(\boldsymbol{u}, \boldsymbol{\lambda}, \boldsymbol{n}, \boldsymbol{\omega}, \tilde{\boldsymbol{\omega}}) = q(\boldsymbol{u})q(\boldsymbol{\lambda})q(\boldsymbol{\omega}|\boldsymbol{n})q(\boldsymbol{n})q(\tilde{\boldsymbol{\omega}})$ and the factors are
$q(\boldsymbol{u}) = \prod_c^C \mathcal{N}(\boldsymbol{u}^c|\boldsymbol{\mu}^c, \Sigma^c)$, $q(\boldsymbol{\lambda}) = \prod_i \mathrm{Ga}(\lambda_i|\alpha_i, \beta_i)$, $q(\boldsymbol{\omega}|\boldsymbol{n}) = \prod_{i,c}\mathrm{PG}(\omega_i^c|n_i^c, b_i^c)$,
$q(\tilde{\boldsymbol{\omega}}) = \prod_{i,c}\mathrm{PG}(\omega_i^c|1, d_i^c)$ and $q(\boldsymbol{n}) = \prod_{i,c}\mathrm{Po}(n_i^c|\gamma_i^c)$, where $\boldsymbol{\mu}^c$, Σ^c, α_i, β_i, b_i^c, d_i^c, γ_i^c, for all $i \in \{1, \ldots, N\}$ and $c \in \{1, \ldots, C\}$ are the *variational parameters*. Details of the variational updates are found in appendix A.3.

Stochastic variational inference

We implement the classic SVI algorithm described by Hoffman et al. (2013), with block coordinate ascent (CAVI) updates given in closed-form and be computed by averaging the parameters of each complete conditional in log space (Blei et al., 2017). Details on the variational updates are deferred to the appendix A.3. The inference algorithm is summarized in Alg. 1 in appendix A.2, its complexity is of $\mathcal{O}(K(M^2 + M^3))$.

4. Experiments

In all experiments we use a squared exponential covariance function with automatic relevance determination (ARD): $k(\boldsymbol{x}, \boldsymbol{x'}) = \eta \exp(-\sum_{d=1}^{D} (x_d - x'_d)^2 / (2l_d^2))$. The hyperparameters are optimized using Adam (Kingma and Ba, 2014). We use a collection of datasets from the LIBSVM repository[1], every dataset has been normalized to mean 0 and variance 1. For each method, we use 200 inducing points which locations are determined through the kmeans++ algorithm (Arthur and Vassilvitskii, 2007) and fixed during training, mini-batches of size 100 are used.

We evaluate the prediction performance and convergence speed of our sparse conjugate method (*SC-MGPC*) against *SVI-MGPC* proposed by Hensman and Matthews (2015) using the natural gradient method of Salimbeni et al. (2018) and *SEP-MGPC* proposed by Villacampa-Calvo and Hernández-Lobato (2017).

We employ a 10-fold cross validation and report the average negative test log-likelihood ($p(y = y_\star | \boldsymbol{x}_\star, \boldsymbol{X})$) and the average predictive test error (number of missclassified test points divided by the total number of test points) as function of time with one standard deviation on figure along with one standard deviation in table 1 in appendix B.1. Aiming on speed, we report the prediction performance of each methods after a fixed time budget of 100 seconds. We display the full optimization curves of each method with unlimited time budget on figure 1 for the MNIST dataset. Due to a need of concision plots, more datasets are deferred to the appendix B.2.

Our method is particularly fitted to reach convergence quickly, especially on datasets with very large number of points. Due to the closed form updates it is also the most stable.

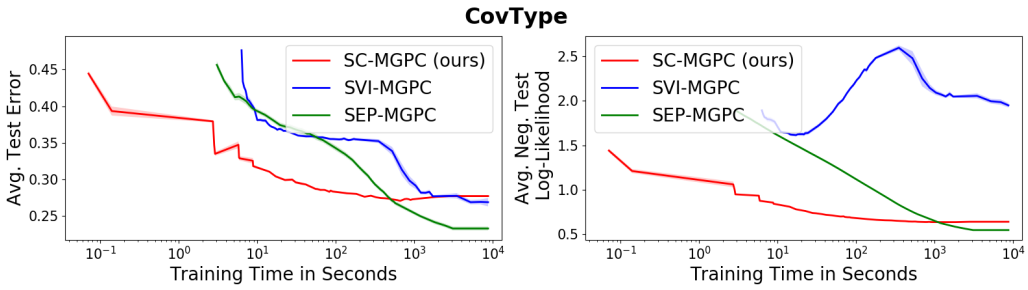

Figure 1: Test prediction error and negative test log-likelihood as a function of training time (seconds in $\log_{10}$ scale) on CovType (851K points, 54 features, 7 classes).

1. https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/multiclass.html

References

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

David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 2017. URL http://arxiv.org/abs/1601.00670.

Alexander Buchholz, Florian Wenzel, and Stephan Mandt. Quasi-monte carlo variational inference. *ICML*, 2018.

James Hensman and Alexander Matthews. Scalable Variational Gaussian Process Classification. *AISTATS*, 2015. ISSN 15337928. URL https://papers.nips.cc/paper/3351-the-generalized-fitc-approximation.pdf.

Matthew D. Hoffman, David M. Blei, Chong Wang, and John Paisley. Stochastic Variational Inference. *JMLR*, 2013. URL http://jmlr.org/papers/v14/hoffman13a.html.

Hyun-Chul Kim and Zoubin Ghahramani. Bayesian gaussian process classification with the em-ep algorithm. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(12):1948–1959, December 2006. ISSN 0162-8828. doi: 10.1109/TPAMI.2006.238. URL http://dx.doi.org/10.1109/TPAMI.2006.238.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Scott W. Linderman, Matthew J. Johnson, and Ryan P. Adams. Dependent multinomial models made easy: Stick-breaking with the polya-gamma augmentation. *NIPS*, 2015. URL https://arxiv.org/abs/1506.05843.

Nicholas G. Polson, James G. Scott, and Jesse Windle. Bayesian inference for logistic models using pólya–gamma latent variables. *Journal of the American Statistical Association*, 108 (504):1339–1349, 2013. doi: 10.1080/01621459.2013.829001. URL http://dx.doi.org/10.1080/01621459.2013.829001.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005. ISBN 026218253X.

Jaakko Riihimäki, Pasi Jylänki, and Aki Vehtari. Nested expectation propagation for gaussian process classification. *J. Mach. Learn. Res.*, 14(1):75–109, January 2013. ISSN 1532-4435. URL http://dl.acm.org/citation.cfm?id=2502581.2502584.

Otto Rudolf Rocktäschel. *Methoden zur berechnung der gammafunktion für komplexes argument*. PhD thesis, 1922.

Hugh Salimbeni, Stefanos Eleftheriadis, and James Hensman. Natural gradients in practice: Non-conjugate variational inference in gaussian process models. *AISTATS*, 2018.

Carlos Villacampa-Calvo and Daniel Hernández-Lobato. Scalable multi-class gaussian process classification using expectation propagation. *ICML*, 2017.

Stephen G Walker. Posterior sampling when the normalizing constant is unknown. *Communications in Statistics–Simulation and Computation®*, 40(5):784–792, 2011.

Florian Wenzel, Matthäus Deutsch, Theo Galy-Fajou, and Marius Kloft. Bayesian nonlinear support vector machines for big data. *ECML PKDD*, 2017.

Florian Wenzel, Théo Galy-Fajou, Christian Donner, Marius Kloft, and Manfred Opper. Efficient gaussian process classification using polya-gamma data augmentation. *arXiv*, 2018.

Christopher K. I. Williams and David Barber. Bayesian classification with gaussian processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20:1342–1351, 1998.

Appendix A. Algorithm details

A.1. The full conditional distributions.

From the likelihood we get the full conditionals of the GPs $\boldsymbol{f}^c$ are

$$p(\boldsymbol{f}^c \mid \boldsymbol{y}, \tilde{\boldsymbol{\omega}}^c, \boldsymbol{\omega}^c, \boldsymbol{n}^c) = \mathcal{N}\left(\boldsymbol{f}^c \mid \frac{1}{2}A^c(\boldsymbol{y}'^c - \boldsymbol{n}^c), A^c\right),$$

$\boldsymbol{y}'^c$ is a N dimensional one-hot encoding of the labels : $y_i'^c$ is 1 if $y_i = c$, 0 otherwise. The conditional covariance matrix is given by $A^c = \left(\mathrm{diag}(\boldsymbol{y}'^c \circ \tilde{\boldsymbol{\omega}}^c + \boldsymbol{\omega}^c) + K_c^{-1}\right)^{-1}$ where K_c is the kernel matrix of the GP $\boldsymbol{f}^c$, and $\circ$ is the element-wise product.
The full conditionals of $\boldsymbol{\lambda}, \boldsymbol{n}, \boldsymbol{\omega}, \tilde{\boldsymbol{\omega}}$ are:

$$p(\lambda_i \mid \boldsymbol{n}_i) = \mathrm{Ga}\left(\lambda_i \mid 1 + \sum_{c=1}^{C} n_i^c, C\right), \quad p(n_i^c \mid f_i^c, \lambda_i) = \mathrm{Po}\left(n_i^c \mid \lambda_i F_i^c\right),$$

$$p(\omega_i^c \mid n_i^c, f_i^c) = \mathrm{PG}\left(\omega_i^c \mid n_i^c, f_i^c\right), \quad p(\tilde{\omega}_i^c \mid f_i^c) = \mathrm{PG}\left(\tilde{\omega}_i^c \mid 1, f_i^c\right),$$

where $\mathrm{Ga}(\cdot|a, b)$ denotes a gamma distribution with shape parameter a and rate parameters b, $F_i^c = \sigma(f_i^c)$.

A.2. Algorithm for Conjugate Multi–Class Gaussian Process Classification

Algorithm 1: Conjugate Multi-Class Gaussian Process Classification

input : data $\boldsymbol{X}$,$\boldsymbol{y}$, minibatch size $|\mathcal{S}|$
output: variational posterior GPs $p(u^c|\mu^c, \Sigma^c)$
Set the learning rate schedules ρ_t, ρ_t^h appropriately;
Initialize all variational parameters and hyperparameters;
Select M inducing points locations (e.g. kMeans);
for *iteration* $t = 1, 2, \ldots$ **do**
  `# Sample minibatch:`
  Sample a minibatch of the data $\mathcal{S} \subset \{1, \ldots, N\}$;
  `# Local variational updates`
  for $i \in \mathcal{S}$ **do**
    Update $(\alpha_i, \boldsymbol{\gamma}_i)$ (Eq. 6,7);
    for *each class* c **do**
      Update b_i^c, d_i^c (Eq. 8,9)
    end
  end
  `# Global variational GP updates`
  for *each class* c **do**
    $\mu^c \leftarrow (1-\rho_t)\mu^c + \rho_t \hat{\mu}^c$ (Eq. 10) ;
    $\Sigma^c \leftarrow (1-\rho_t)\Sigma^c + \rho_t \hat{\Sigma}^c$ (Eq. 11)
  end
  `# Hyperparameter updates`
  Gradient step $h \leftarrow h + \rho_t^h \nabla_h \mathcal{L}$
end

Algorithm 2: Conjugate Multi-Class Gaussian Process Classification with class subsampling

input : data $\boldsymbol{X}$,$\boldsymbol{y}$, minibatch size $|\mathcal{S}|$ and $|\mathcal{B}|$
output: variational posterior GPs $p(u^c|\mu^c, \Sigma^c)$
Set the learning rate schedules ρ_t, ρ_t^h appropriately
Initialize all variational parameters and hyperparameters
Select M inducing points locations (e.g. kMeans)
for *iteration* $t = 1, 2, \ldots$ **do**
  `# Sample minibatch:`
  Sample a minibatch of the data $\mathcal{S} \subset \{1, \ldots, N\}$;
  Sample a set of labels $\mathcal{K} \subset \{1, \ldots, C\}$;
  `# Local variational updates`
  for $i \in \mathcal{S}$ **do**
    Update $(\alpha_i, \gamma_i^c)_{c \in \mathcal{K}}$ (Eq. 6,12);
    for $c \in \mathcal{K}$ **do**
      Update b_i^c, d_i^c (Eq. 8,9) ;
    end
  end
  `# Global variational GP updates`
  for $c \in \mathcal{K}$ **do**
    $\mu^c \leftarrow (1-\rho_t)\mu^c + \rho_t \hat{\mu}^c$ (Eq. 10) ;
    $\Sigma^c \leftarrow (1-\rho_t)\Sigma^c + \rho_t \hat{\Sigma}^c$ (Eq. 11) ;
  end
  `# Hyperparameter updates`
  Gradient step $h \leftarrow h + \rho_t^h \nabla_h \mathcal{L}$;
end

A.3. Block coordinate ascent updates

Given the variational distribution $q(\boldsymbol{u}, \boldsymbol{\lambda}, \boldsymbol{n}, \boldsymbol{\omega}, \tilde{\boldsymbol{\omega}}) = q(\boldsymbol{u})q(\boldsymbol{\lambda})q(\boldsymbol{\omega}|\boldsymbol{n})q(\boldsymbol{n})q(\tilde{\boldsymbol{\omega}})$ and the factors are

$$
\begin{aligned}
q(\boldsymbol{u}) &= \prod_{c}^{C} \mathcal{N}(\boldsymbol{u}^c|\boldsymbol{\mu}^c, \Sigma^c), \quad q(\boldsymbol{\lambda}) = \prod_{i} \mathrm{Ga}(\lambda_i|\alpha_i, \beta_i), \\
q(\boldsymbol{\omega}|\boldsymbol{n}) &= \prod_{i,c} \mathrm{PG}(\omega_i^c|n_i^c, b_i^c), \quad q(\tilde{\boldsymbol{\omega}}) = \prod_{i,c} \mathrm{PG}(\omega_i^c|1, d_i^c), \\
q(\boldsymbol{n}) &= \prod_{i,c} \mathrm{Po}(n_i^c|\gamma_i^c),
\end{aligned}
$$

The CAVI updates are derived from the optimal variational distribution

$$
q^*(\theta) \propto \exp\left(\mathbb{E}_{q(\lambda)}\left[\log\left(p(\theta|\lambda)\right)\right]\right)
$$

$$
\overline{f_i^c} = \sqrt{\mathbb{E}_{q(f^c)}\left[(f_i^c)^2\right]} = \sqrt{\widetilde{K}_{ii}^c + \kappa_i^c \Sigma^c \kappa_i^{c\top} + (\kappa_i^c \boldsymbol{\mu}^c)^\top \kappa_i^c \boldsymbol{\mu}^c} \tag{5}
$$

$$
\gamma_i^c = \frac{\beta_i}{2\Gamma(\alpha_i)\cosh\left(\frac{\overline{f_i^c}}{2}\right)} \exp\left(-\alpha_i - (1-\alpha_i)\psi(\alpha_i) - \frac{\kappa_i^c \boldsymbol{\mu}^c}{2}\right) \tag{6}
$$

$$
\alpha_i = 1 + \sum_{c=1}^{C} \gamma_i^c, \quad \beta_i = C \tag{7}
$$

$$
b_i^c = \overline{f_i^c}, \qquad \theta_i^c = \mathbb{E}_{q(\omega_i^c, n_i^c)}\left[\omega_i^c\right] = \frac{\gamma_i^c}{b_i^c} \tanh\frac{d_i^c}{2} \tag{8}
$$

$$
d_i^c = \overline{f_i^c}, \qquad \tilde{\theta}_i^c = \mathbb{E}_{q(\tilde{\omega}_i^c)}\left[\tilde{\omega}_i^c\right] = \frac{1}{d_i^c} \tanh\frac{d_i^c}{2} \tag{9}
$$

$$
\boldsymbol{\mu}^c = \frac{1}{2} \Sigma^{c-1} \kappa^{c\top} \left(\boldsymbol{y}^c - \boldsymbol{\gamma}^c\right) \tag{10}
$$

$$
\Sigma^c = \left(\kappa^{c\top} \mathrm{diag}\left(\boldsymbol{y'}^c \circ \tilde{\boldsymbol{\theta}}^c + \boldsymbol{\theta}^c\right) \kappa^c + K_{mm}^{-1\ c}\right)^{-1} \tag{11}
$$

Where $\Gamma()$ is the gamma function, $\psi()$ is the digamma function and $\circ$ is the elementwise product.

For large value of α (high number of classes), equation 6 easily overflows. However one can use the approximation $\log\Gamma(x) \approx (x - \frac{1}{2})\log x - x + \frac{1}{2}\log(2\pi)$ (Rocktäschel, 1922) for large x. From equation 6 and 7, one can see that α_i and γ_i^c are directly depending on each other. It does not make a difference for a full-batch algorithm but when using stochastic updates, an inner loop of a few iterations is needed so γ_i^c is not updated on random values.

Moreover if a subsampling in classes used like in algorithm 2, α_i is approximated by :

$$
\alpha_i = 1 + \frac{C}{|\mathcal{K}|} \sum_{c \in \mathcal{K}} \gamma_i^c \tag{12}
$$

A.4. Predictive distribution approximation

Looking at the predictive distribution for one class among K for a point $x^\star$

$$p(y=k|x^\star,D)=\int_{\mathbb{R}^C}\frac{\sigma(f_\star^k)}{\sum_{c=1}^C\sigma(f_\star^c)}\prod_{c=1}^C\mathcal{N}(f_\star^c|\mu_\star^c,\sigma_\star^{2^c})df_\star^c$$

Let $g^k(\boldsymbol{f})=p(y=k|\boldsymbol{f})=\frac{\sigma(f_\star^k)}{\sum_{c=1}^C\sigma(f_\star^c)}$ be the function representing the logistic softmax likelihood for the test label k given the latent functions $\boldsymbol{f}$. The Taylor expansion of the second order of the mean and variance are defined as

$$\begin{aligned}
&\mathbb{E}_{p(\boldsymbol{f}_\star)}\left[g^k(\boldsymbol{f}_\star)\right]=\mathbb{E}_{p(\boldsymbol{f}_\star)}\left[g^k(\boldsymbol{\mu}_\star+\boldsymbol{\delta f}_\star)\right]\\
\approx&g^k(\boldsymbol{\mu}_\star)+\frac{1}{2}\mathbb{E}_{p(\boldsymbol{f}_\star)}\left[\sum_i\sum_j\frac{d^2g^k(\boldsymbol{\mu}_\star)}{df_\star^idf_\star^j}(\delta f_\star^i)(\delta f_\star^j)\right]\\
=&g^k(\boldsymbol{\mu}_\star)+\frac{1}{2}\sum_i\frac{d^2g^k(\boldsymbol{\mu}_\star)}{d^2f_\star^i}\mathrm{Var}(f_\star^i)+\frac{1}{2}\sum_{i,j;i\neq j}\frac{d^2g^k(\boldsymbol{\mu}_\star)}{df_\star^idf_\star^j}\underbrace{\mathrm{Cov}(f_\star^i,f_\star^j)}_{=0}\\
\approx&g^k(\boldsymbol{\mu}_\star)+\frac{1}{2}\sum_i\frac{d^2g^k(\boldsymbol{\mu}_\star)}{d^2f_\star^i}\mathrm{Var}(f_\star^i)
\end{aligned}$$

$$\mathrm{Var}_{p(\boldsymbol{f}_\star)}\left[g^k(\boldsymbol{f}_\star)\right]\approx\sum_i\left(\frac{dg^k(\boldsymbol{\mu}_\star)}{df_\star^i}\right)^2\mathrm{Var}(f_\star^i)-\frac{1}{4}\sum_i\left(\frac{d^2g^k(\boldsymbol{\mu}_\star)}{d^2f_\star^i}\right)^2\mathrm{Var}^2(f_\star^i)$$

Where we used the fact that the latent GP functions are independent from each other. For the 0th order, $\mathbb{E}_{p(\boldsymbol{f}_\star)}\left[g^k(\boldsymbol{f}_\star)\right]$ simplify as $g^k(\boldsymbol{\mu}_\star)$
The derivatives of g_k are simply defined using the notation $\sigma_i=\sigma(f^i)$:

$$\frac{dg^k(\boldsymbol{\mu})}{df^c}=\begin{cases}-\frac{\sigma_k(\sigma_c-\sigma_c^2)}{\left(\sum_j\sigma_j\right)^2} & \text{for } c\neq k\\ \frac{\left[\sum_{j\neq1}\sigma_j\right](\sigma_k-\sigma_k^2)}{\left(\sum_j\sigma_j\right)^2} & \text{for } c=k\end{cases}$$

$$\frac{d^2g^k(\boldsymbol{\mu})}{d^2f_c}=\begin{cases}-\frac{\left(\sigma_k(\sigma_c-\sigma_c^2)\right)\left[(1-2\sigma_c)\left(\sum_j\sigma_j\right)^2-2\left(\sigma_c-\sigma_c^2\right)\sum_j\sigma_j\right]}{2\left(\sum_j\sigma_j\right)^3} & \text{for } c\neq k\\ \frac{\left(\sum_{j\neq k}\sigma_j\right)(\sigma_k-\sigma_k^2)\left[(1-2\sigma_k)\left(\sum_j\sigma_j\right)^2-2\left(\sigma_k-\sigma_k^2\right)\sum_j\sigma_j\right]}{2\left(\sum_j\sigma_j\right)^3} & \text{for } c=k\end{cases}$$

Appendix B. Experiments

B.1. Performance table

Dataset			**SC-MGPC** (ours)	SVI-MGPC	SEP-MGPC
Acoustic	$C = 3$	Error	$\mathbf{0.29 \pm 0.01}$	0.43 ± 0.01	0.35 ± 0.01
$N = 98528$	$D = 50$	NLL	$\mathbf{0.68 \pm 0.01}$	1.59 ± 0.08	0.83 ± 0.01
Combined	$C = 3$	Error	$\mathbf{0.20 \pm 0.00}$	0.34 ± 0.01	0.32 ± 0.01
$N = 98528$	$D = 50$	NLL	$\mathbf{0.53 \pm 0.01}$	1.18 ± 0.02	1.02 ± 0.00
CovType	$C = 7$	Error	$\mathbf{0.28 \pm 0.00}$	0.36 ± 0.01	0.34 ± 0.01
$N = 851012$	$D = 54$	NLL	$\mathbf{0.68 \pm 0.00}$	2.21 ± 0.06	1.09 ± 0.04
DNA	$C = 3$	Error	$\mathbf{0.03 \pm 0.01}$	0.06 ± 0.02	0.55 ± 0.01
$N = 3386$	$D = 180$	NLL	0.29 ± 0.01	$\mathbf{0.19 \pm 0.06}$	1.10 ± 0.00
MNIST	$C = 10$	Error	$\mathbf{0.11 \pm 0.00}$	0.17 ± 0.14	0.16 ± 0.01
$N = 70000$	$D = 784$	NLL	$\mathbf{0.70 \pm 0.01}$	1.23 ± 1.36	2.30 ± 0.00
SatImage	$C = 6$	Error	0.12 ± 0.01	0.13 ± 0.02	$\mathbf{0.10 \pm 0.01}$
$N = 6430$	$D = 36$	NLL	$\mathbf{0.33 \pm 0.02}$	0.59 ± 0.14	0.40 ± 0.03
Segment	$C = 7$	Error	0.06 ± 0.02	$\mathbf{0.04 \pm 0.02}$	0.05 ± 0.02
$N = 2310$	$D = 19$	NLL	$\mathbf{0.17 \pm 0.06}$	0.19 ± 0.11	0.28 ± 0.04
Seismic	$C = 3$	Error	$\mathbf{0.30 \pm 0.00}$	0.38 ± 0.01	0.37 ± 0.01
$N = 98528$	$D = 50$	NLL	$\mathbf{0.67 \pm 0.01}$	2.12 ± 0.16	1.01 ± 0.01
Sensorless	$C = 11$	Error	$\mathbf{0.11 \pm 0.01}$	0.91 ± 0.00	0.22 ± 0.01
$N = 58509$	$D = 48$	NLL	$\mathbf{0.62 \pm 0.01}$	2.40 ± 0.00	1.73 ± 0.01
Shuttle	$C = 7$	Error	$\mathbf{0.00 \pm 0.00}$	0.01 ± 0.00	0.00 ± 0.00
$N = 58000$	$D = 9$	NLL	0.06 ± 0.00	0.03 ± 0.02	$\mathbf{0.02 \pm 0.00}$
Vehicle	$C = 4$	Error	0.27 ± 0.07	$\mathbf{0.19 \pm 0.05}$	0.25 ± 0.06
$N = 846$	$D = 18$	NLL	$\mathbf{0.55 \pm 0.05}$	0.71 ± 0.37	0.61 ± 0.06

Table 1: Average test prediction error and average negative test log-likelihood (NLL) along with one standard deviation for a time budget of 100 seconds. Best values are highlighted in bold.

B.2. Additional convergence figures

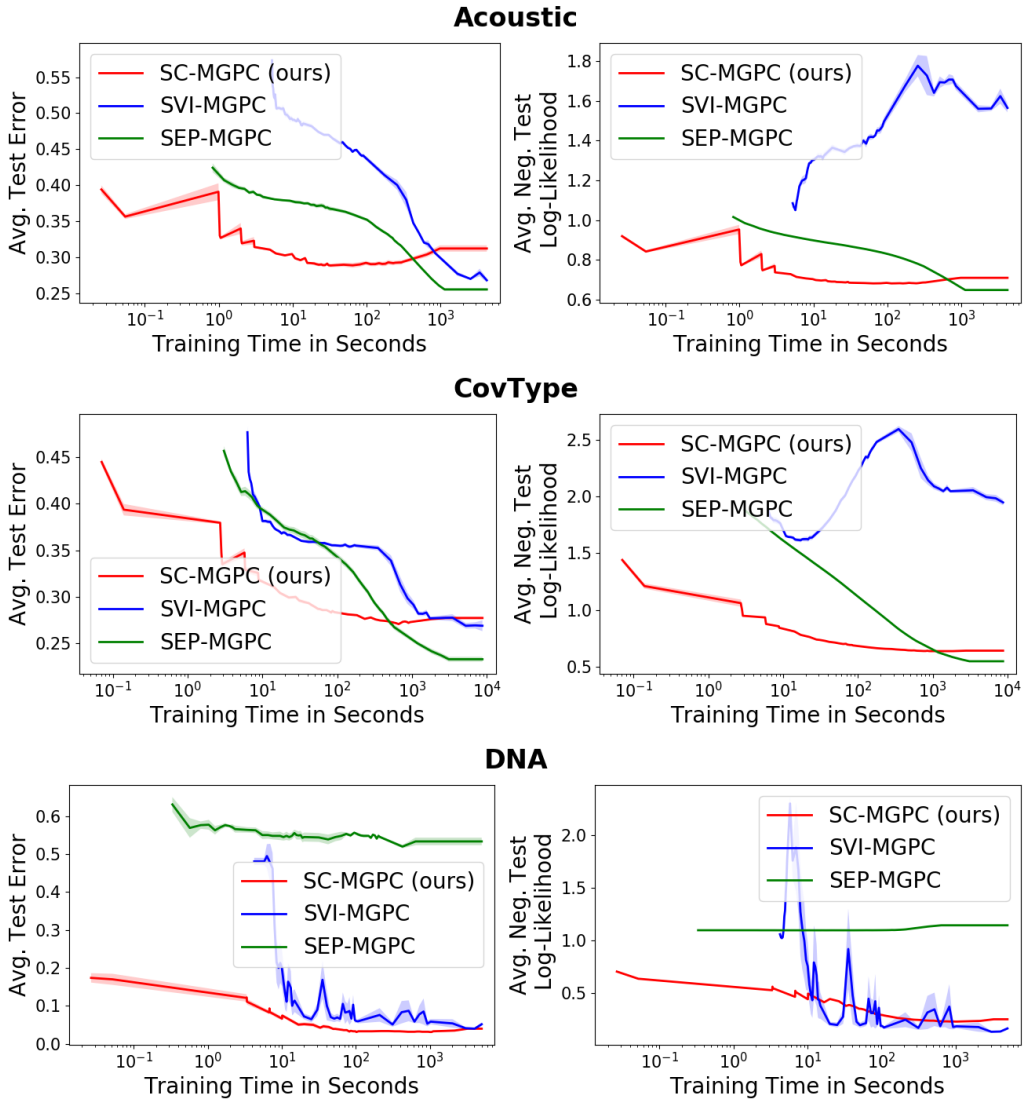

Figure 2: Average test prediction error and average negative test log likelihood as a function of training time (seconds in a $\log_{10}$ scale) on the datasets Acoustic (100k points, 50 features, 3 Classes), CovType (581K points, 54 features, 7 classes), DNA(3386 points, 180 features, 3 classes)

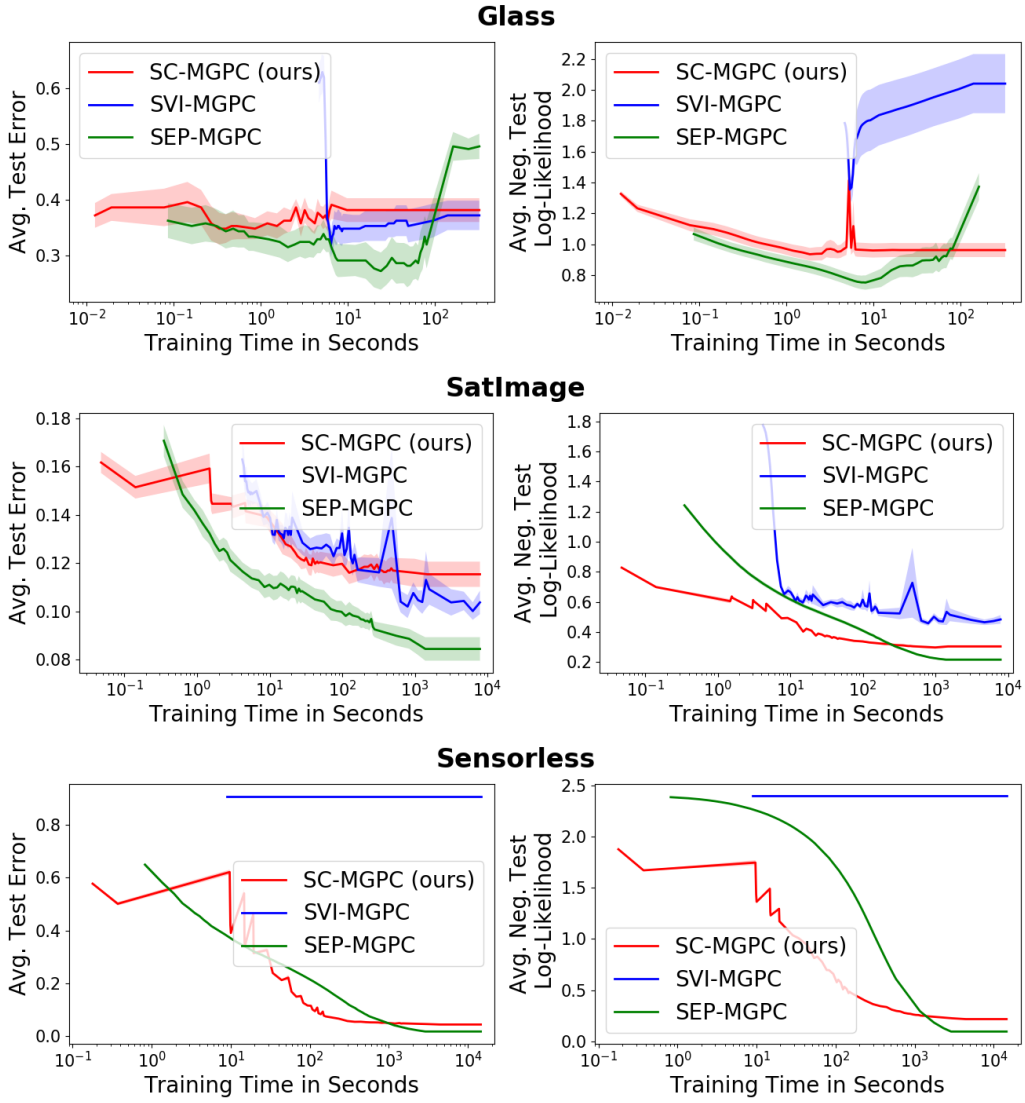

Figure 3: Average test prediction error and average negative test log likelihood as a function of training time (seconds in a $\log_{10}$ scale) on the datasets Glass (214 points, 9 features, 6 Classes), SatImage (6430 points, 36 features, 6 classes), Sensorless (58509 points, 48 features, 11 classes)

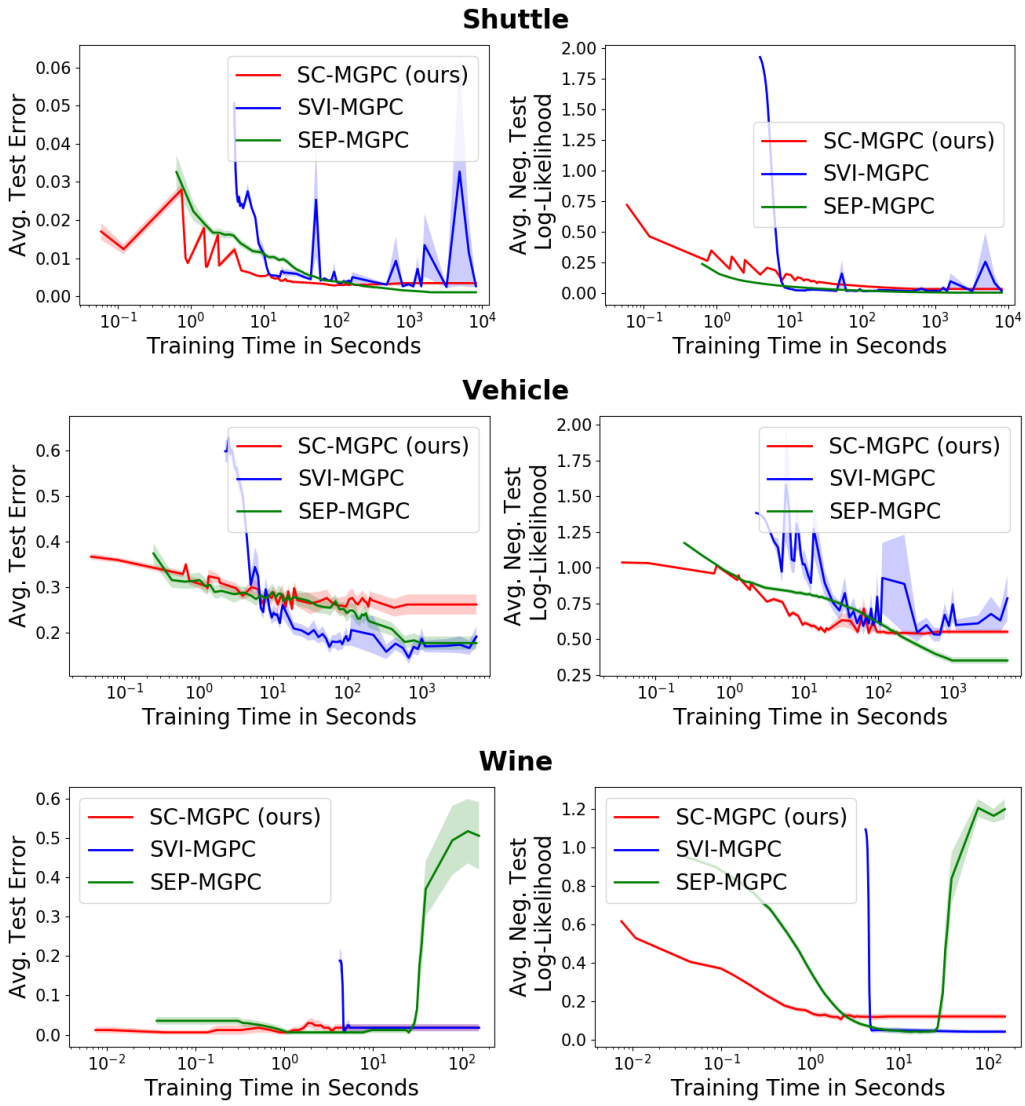

Figure 4: Average test prediction error and average negative test log likelihood as a function of training time (seconds in a $\log_{10}$ scale) on the datasets Shuttle (58000 points, 9 features, 7 Classes), Vehicle (846 points, 18 features, 4 classes), Wine (178 points, 13 features, 3 classes)