

Lomonosov Moscow State University  
Faculty of Computer Science

Review of materials on  
**Gaussian Processes for Machine Learning**

Pavel Izmailov

Moscow, 2016

# 1 Theory

In this section an introduction to Gaussian process theory is provided.

## 1.1 Gaussian Process

Consider the following definition

**Definition 1.** *A Gaussian process is a collection of random variables, any finite number of which have a joint Gaussian distribution.*

A Gaussian process is completely specified by its mean function and covariance function. These functions are defined as follows

**Definition 2.** *Let  $f(x)$  be a real-valued Gaussian process. Then the functions*

$$m(x) = \mathbb{E}[f(x)],$$

$$k(x, x') = \mathbb{E}[(f(x) - m(x))(f(x') - m(x'))],$$

*are the mean function and the covariance function of the process  $f$  respectively.*

We will write the Gaussian process as  $f(x) \sim \mathcal{GP}(m(x), k(x, x'))$ .

## 1.2 GP-regression

Consider the following task. We have a dataset  $\{(x_i, f_i) | i = 1, \dots, n\}$ , generated from a Gaussian process  $f \sim \mathcal{GP}(m(x), k(x, x'))$ , let  $x \in \mathbb{R}^d$ . We will denote the matrix comprised of points  $x_1, \dots, x_n$  by  $X \in \mathbb{R}^{n \times d}$  and the vector of corresponding values  $f_1, \dots, f_n$  by  $f \in \mathbb{R}^n$ . We want to predict the values  $f_* \in \mathbb{R}^m$  of this random process at a set of other  $m$  points  $X_* \in \mathbb{R}^{m \times d}$ . The joint distribution of  $f$  and  $f_*$  is given by

$$\begin{bmatrix} f \\ f_* \end{bmatrix} \sim \mathcal{N} \left( 0, \begin{bmatrix} K(X, X) & K(X, X_*) \\ K(X_*, X) & K(X_*, X_*) \end{bmatrix} \right),$$

where  $K(X, X) \in \mathbb{R}^{n \times n}$ ,  $K(X, X_*) = K(X_*, X)^T \in \mathbb{R}^{n \times m}$ ,  $K(X_*, X_*) \in \mathbb{R}^{m \times m}$  are the matrices comprised of pairwise values of the covariance function  $k$  for the given sets.

The conditional distribution

$$f_* | X_*, X, f \sim \mathcal{N}(\hat{m}, \hat{K}),$$

where

$$\begin{aligned} \mathbb{E}[f_* | f] &= \hat{m} = K(X_*, X)K(X, X)^{-1}f, \\ \text{cov}(f_* | f) &= \hat{K} = K(X_*, X_*) - K(X_*, X)K(X, X)^{-1}K(X, X_*). \end{aligned}$$

Thus, predicting the values of the Gaussian process at a new data point requires solving a linear system with a matrix of size  $n \times n$  and thus scales as  $O(n^3)$ .

### 1.2.1 Noisy case

Consider the following model. We now have a dataset  $\{(x_i, y_i) | i = 1, \dots, n\}$ , where  $y_i = f(x_i) + \varepsilon$ ,  $\varepsilon \sim \mathcal{N}(0, \sigma_n)$ . This means that we only have access to the noisy observations and not the true values of the process at data points. With the notation and logics similar to the one we used in the previous section we can find the conditional distribution for the values  $f_*$  of the process at new points  $X_*$  in this case:

$$f_*|y \sim \mathcal{N}(\hat{m}, \hat{K}),$$

$$\mathbb{E}[f_*|y] = \hat{m} = K(X_*, X)(K(X, X) + \sigma_n^2 I)^{-1}y,$$

$$\text{cov}(f_*|y) = \hat{K} = K(X_*, X_*) - K(X_*, X)(K(X, X) + \sigma_n^2 I)^{-1}K(X, X_*).$$

## 1.3 GP-classification

To be written.

## 1.4 Kernel functions

To be written.

## 1.5 Hyper-parameter estimation

Bayesian paradigm provides a way of estimating the kernel hyper-parameters of the GP-model through the maximization of the marginal likelihood of the model. Marginal likelihood is given by

$$p(y|X) = \int p(y|f, X)p(f|X)df,$$

which is the likelihood, marginalized over the hidden values  $f$  of the underlying process.

## 1.6 Theoretical perspectives

To be written.

## 2 Review of existing methods

It follows from the discussion above, that full Gaussian process regression scales as  $O(n^3)$  and thus cannot be applied to big datasets. In this section we will review several approximate methods, that make Gaussian processes practical.

### 2.1 Methods, based on inducing inputs

Most of the existing methods are based on introducing a set of  $m$  function points that are called inducing inputs. Using these inputs one can make approximate predictions of the values of the hidden process at test points with a complexity of  $O(nm^3)$  instead of  $O(n^3)$ .

Consider the following situation. We have a dataset of  $n$  examples  $x_i$  with corresponding values  $y_i$ . We will denote the matrix of pairwise values of the covariance function by  $K_{nn}$ . Now we introduce a set of  $m$  inducing inputs. We will denote the corresponding covariance matrix by  $K_{mm}$  and the matrices of covariances between the inducing points and training points by  $K_{nm}$  and  $K_{mn}$ . We will denote the vectors, comprised of noisy and true function values  $y_i$  and  $f_i$  at training points by  $y$  and  $f$  respectively. We will also introduce a distribution  $q(u)$  over the hidden function values  $u$  at the inducing inputs.

It's easy to see, that

$$\begin{aligned} p(y|f) &= \mathcal{N}(y|f, \sigma_n I), \\ p(f|u) &= \mathcal{N}(f|K_{nm}K_{mm}^{-1}u, \tilde{K}), \\ p(u) &= \mathcal{N}(u|0, K_{mm}), \end{aligned}$$

where  $\tilde{K} = K_{nn} - K_{nm}K_{mm}^{-1}K_{mn}$ .

#### 2.1.1 Variational learning of inducing points

The method discussed here was introduced in [1]. This method provides a way to find the optimal positions of the inducing points, as well as the optimal distribution of the process value at these points.

Let  $z$  denote a vector comprised of the process values at some new points. We can calculate the predictive distribution at these points as follows

$$p(z|y) = \int p(z|f)p(f|y)df.$$

Let's fix the inducing point positions  $x_1, \dots, x_m$ . As above,  $u$  is the vector comprised of the process values at these points. We can rewrite the above equation

$$p(z|y) = \iint p(z|u, f)p(f|u, y)p(u|y)dfdu, \tag{1}$$

as  $p(z|u, f, y) = p(z|u, f)$ .

Suppose for a moment, that  $u$  is a sufficient statistics for the parameter  $f$  in the scence that  $z$  and  $f$  are conditionally independent given  $u$ . Then we have

$$\begin{aligned} p(z|f, u) &= \frac{p(z, f|u)}{p(f|u)} = \frac{p(z|u)p(f|u)}{p(f|u)} = p(z|u), \\ p(z|y, u) &= \frac{p(z, y, u)}{p(y, u)} = \frac{\int p(y|f)p(f, z, u)du}{\iint p(y|f)p(f, z, u)dfdz} = \frac{\int p(y|f)p(z|u)p(u|f)p(f)df}{\iint p(y|f)p(z|u)p(u|f)p(f)dfdz} = \\ &= \frac{\int p(y|f)p(f)p(u|f)df \cdot p(z|u)}{\int p(y|f)p(f)p(u|f)df \cdot \int p(z|u)dz} = \frac{\int p(y, f)p(u|f)df}{\int p(y, f)p(u|f)df} p(z|u) = p(z|u). \end{aligned}$$

So,  $p(z|y, u) = p(z|u)$ . If we set the points, corrwspoding to the process values  $z$ , to the traing points, we will have  $z = f$ , and thus  $p(f|y, u) = p(f|u)$ .

Substituting these formulas into (1) we achieve

$$\begin{aligned} q(z) = p(z|y) &= \iint p(z|u)p(f|u)p(u|y)dfdu = \iint p(z|u)p(u|y)du = \\ &= \int p(z|u)\varphi(u)du = \int q(z, u)du, \end{aligned} \tag{2}$$

where  $\varphi(u) = p(u|y)$ ,  $q(z, u) = p(z|u)\varphi(u)$ .

In practice however it's difficult to guarantee that  $u$  is a sufficient statistics. Thus we can only expect  $q(z)$  to be an approximation to  $p(z|y)$ . In such case we can choose  $\varphi(u)$  to be a variational distribution, where in general  $\varphi(u) \neq p(u|y)$ . We will consider  $\varphi(u)$  to be Gaussian with a mean vector  $\mu$  and covariance matrix  $\Sigma$ .

By using the eq. (2) we can calculate the approximate posterior GP mean at point  $x$  and covariance at points  $x, x'$

$$\begin{aligned} \mathbb{E}[z(x)] &= K_{xm}K_{mm}^{-1}\mu, \\ \text{cov}(z(x), z(x')) &= k(x, x') - K_{xm}K_{mm}^{-1}K_{mx'} + K_{xm}AK_{mx'}, \end{aligned}$$

where  $A = K_{mm}^{-1}\Sigma K_{mm}^{-1}$ .

Now we have to specify a way to find the variational distribution parameters  $\mu$  and  $\Sigma$ , and the inducing input positions  $X_m$  and a way to optimize the kernel hyper-parameters. In order to do so, we will form the variational distribution  $q(f, u)$  and the exact posterior  $p(f, u|y)$  on the training function values and the values at the inducing points, and then minimize the KL-divergence between these two distributions. This minimization is equivalently expressed as the maximization of the following lower bound of the true marginal likelihood:

$$F_V(X_m, \varphi) = \iint p(f|u)\varphi(u) \log \frac{p(y|f)p(u)}{\varphi(u)} dfdu.$$

This bound can be optimized analytically with respect to  $\phi$ . The optimal distribution  $\varphi(u) \sim \mathcal{N}(u|\hat{u}, \Lambda^{-1})$ , where

$$\Lambda = \frac{1}{\sigma_n} K_{mm}^{-1} K_{mn} K_{nm} K_{mm}^{-1} + K_{mm}^{-1},$$

$$\hat{u} = \frac{1}{\sigma_n} \Lambda^{-1} K_{mm}^{-1} K_{mn} y.$$

Substituting the optimal values of variational parameters into the  $F_V$  we obtain the following bound

$$F_V(X_m) = \log \mathcal{N}(y|0, \sigma_n^2 I + K_{nm} K_{mm}^{-1} K_{mn}) - \frac{1}{2\sigma_n^2} \text{tr}(\tilde{K}).$$

Another derivation of this lower bound is provided in section (2.1.2).

The bound  $F_V(X_m)$  is computed in  $o(nm^2)$  time. Now we will calculate it's gradient in order to be able to maximize it with respect to  $X_m$  and kernel hyper-parameters. We will denote  $B = \sigma_n^2 I + K_{nm} K_{mm}^{-1} K_{mn}$ . Then

$$\begin{aligned} F_V(X_m, \theta, \sigma_n) &= -\frac{1}{2} \left( n \log 2\pi + \log |B| + y^T B^{-1} y + \frac{1}{\sigma_n^2} \text{tr}(\tilde{K}) \right), \\ \frac{\partial F_V}{\partial \theta} &= \frac{1}{2} \left( -\text{tr} \left( B^{-1} \frac{\partial B}{\partial \theta} \right) + y^T B^{-1} \frac{\partial B}{\partial \theta} B^{-1} y - \right. \\ &\quad \left. - \frac{1}{\sigma_n^2} \text{tr} \left( \frac{\partial K_{nn}}{\partial \theta} - \left( \frac{\partial K_{nm}}{\partial \theta} K_{mm}^{-1} - K_{nm} K_{mm}^{-1} \frac{\partial K_{mm}}{\partial \theta} K_{mm}^{-1} \right) K_{mn} - K_{nm} K_{mm}^{-1} \frac{\partial K_{mn}}{\partial \theta} \right) \right), \end{aligned}$$

where

$$\frac{\partial B}{\partial \theta} = \left( \frac{\partial K_{nm}}{\partial \theta} K_{mm}^{-1} - K_{nm} K_{mm}^{-1} \frac{\partial K_{mm}}{\partial \theta} K_{mm}^{-1} \right) K_{mn} + K_{nm} K_{mm}^{-1} \frac{\partial K_{mn}}{\partial \theta}.$$

We can rewrite

$$\frac{\partial F_V}{\partial \theta} = \frac{1}{2} \left( -\text{tr} \left( B^{-1} \frac{\partial B}{\partial \theta} \right) + y^T B^{-1} \frac{\partial B}{\partial \theta} B^{-1} y - \frac{1}{\sigma_n^2} \text{tr} \left( \frac{\partial K_{nn}}{\partial \theta} - \frac{\partial B}{\partial \theta} \right) \right).$$

Now we can optimize  $F_V$  with respect to kernel hyper-parameters. Similarly, we can take derivatives with respect to  $X_m$  and  $\sigma_n$  and optimize  $F_V$  with respect to them as well.

However, if we compute  $F_v$  and it's derivatives as they are, it takes  $O(n^3)$  time which is not faster, than recovering the full Gaussian process. So, we have to rewrite these values in a form that allows for faster computation.

First of all, let's deal with  $\log |B|$  and  $B^{-1}$ . Using the matrix determinant lemma we obtain

$$|B| = |\sigma_n^2 I + K_{nm} K_{mm}^{-1} K_{mn}| = \frac{\left| K_{mm} + \frac{K_{mn} K_{nm}}{\sigma_n^2} \right| \sigma_n^2}{|K_{mm}|}.$$

So, denoting  $A = K_{mm} + \frac{K_{mn} K_{nm}}{\sigma_n^2}$ , we obtain

$$\log |B| = \log |A| + 2 \log \sigma_n - \log |K_{mm}|.$$

Note tha this is computed in  $O(nm^2)$  instead of  $O(n^3)$ .

Using the Woodbury identity, we obtain

$$B^{-1} = (\sigma_n^2 I + K_{nm} K_{mm}^{-1} K_{mn})^{-1} = \frac{I}{\sigma_n^2} - \frac{K_{nm} A^{-1} K_{mn}}{\sigma^4},$$

which allows for computing  $y^T B^{-1} y$  in  $O(nm)$ .

Similarly, we can compute the gradient in  $O(nm^2)$ . In order to do so, we need to rewrite every trace  $\text{tr}(M_{nm} M_{mm} M_{mn})$ , where  $M_{kl} \in \mathbb{R}^{k \times l}$ , in the form  $\text{tr}(M_{mm} M_{mn} M_{nm})$ , which is computed in  $O(nm^2)$ , and use the derived formulas for  $B^{-1}$ .

Now let's try to compute the second order derivatives.

$$\begin{aligned} \frac{\partial^2 F_V}{\partial \theta_j \partial \theta_i} &= \frac{\partial}{\partial \theta_j} \left( \frac{\partial F_V}{\partial \theta_i} \right) = \frac{1}{2} \frac{\partial}{\partial \theta_j} \left( -\text{tr} \left( B^{-1} \frac{\partial B}{\partial \theta_i} \right) + y^T B^{-1} \frac{\partial B}{\partial \theta_i} B^{-1} y - \frac{1}{\sigma_n^2} \text{tr} \left( \frac{\partial K_{nn}}{\partial \theta_i} - \frac{\partial B}{\partial \theta_i} \right) \right) = \\ &= \frac{1}{2} \left( \text{tr} \left( B^{-1} \frac{\partial B}{\partial \theta_j} B^{-1} \frac{\partial B}{\partial \theta_i} - B^{-1} \frac{\partial^2 B}{\partial \theta_j \partial \theta_i} \right) + y^T \left( B^{-1} \frac{\partial^2 B}{\partial \theta_j \partial \theta_i} B^{-1} - 2 B^{-1} \frac{\partial B}{\partial \theta_j} B^{-1} \frac{\partial B}{\partial \theta_i} B^{-1} \right) y - \right. \\ &\quad \left. - \frac{1}{\sigma_n^2} \text{tr} \left( \frac{\partial^2 K_{nn}}{\partial \theta_j \partial \theta_i} - \frac{\partial^2 B}{\partial \theta_j \partial \theta_i} \right) \right), \end{aligned}$$

where

$$\begin{aligned} \frac{\partial^2 B}{\partial \theta_j \partial \theta_i} &= \frac{\partial}{\partial \theta_j} \left( \frac{\partial K_{nm}}{\partial \theta_i} K_{mm}^{-1} K_{mn} - K_{nm} K_{mm}^{-1} \frac{\partial K_{mm}}{\partial \theta_i} K_{mm}^{-1} K_{mn} + K_{nm} K_{mm}^{-1} \frac{\partial K_{mn}}{\partial \theta_i} \right) = \\ &= \frac{\partial^2 K_{nm}}{\partial \theta_j \partial \theta_i} K_{mm}^{-1} K_{mn} + K_{nm} K_{mm}^{-1} \frac{\partial^2 K_{mn}}{\partial \theta_j \partial \theta_i} - \frac{\partial K_{nm}}{\partial \theta_i} K_{mm}^{-1} \frac{\partial K_{mm}}{\partial \theta_j} K_{mm}^{-1} K_{mn} - \\ &\quad - K_{nm} K_{mm}^{-1} \frac{\partial K_{mm}}{\partial \theta_j} K_{mm}^{-1} \frac{\partial K_{mn}}{\partial \theta_i} + \frac{\partial K_{nm}}{\partial \theta_j} K_{mm}^{-1} \frac{\partial K_{mn}}{\partial \theta_i} + \frac{\partial K_{nm}}{\partial \theta_i} K_{mm}^{-1} \frac{\partial K_{mn}}{\partial \theta_j} \\ &\quad - \frac{\partial K_{nm}}{\partial \theta_j} K_{mm}^{-1} \frac{\partial K_{mm}}{\partial \theta_i} K_{mm}^{-1} K_{mn} + K_{nm} K_{mm}^{-1} \frac{\partial K_{mm}}{\partial \theta_j} K_{mm}^{-1} \frac{\partial K_{mm}}{\partial \theta_i} K_{mm}^{-1} K_{mn} \\ &\quad - K_{nm} K_{mm}^{-1} \frac{\partial^2 K_{mm}}{\partial \theta_j \partial \theta_i} K_{mm}^{-1} K_{mn} + K_{nm} K_{mm}^{-1} \frac{\partial K_{mm}}{\partial \theta_i} K_{mm}^{-1} \frac{\partial K_{mm}}{\partial \theta_j} K_{mm}^{-1} K_{mn} - \\ &\quad - K_{nm} K_{mm}^{-1} \frac{\partial K_{mm}}{\partial \theta_i} K_{mm}^{-1} \frac{\partial K_{mn}}{\partial \theta_j}. \end{aligned}$$

### 2.1.2 Stochastic variational inference

The method discussed here was proposed in [2]. The method doesn't provide a way to choose the positions of inducing points. It provides a way to find the predictive distribution and optimize hyper-parameters for large datasets.

For using stochastic variational inference, we have to provide a lower bound for the marginal likelihood, that factorizes over the training examples. To obtain such an ELBO (evidence lower bound) two ancillary lower bounds are found.

By applying the Jensen inequality we obtain

$$\log p(y|u) = \log \left( \int p(y|f)p(f|u)du \right) \geq \int \log(p(y|f))p(f|u)du = L_1.$$

As  $p(y|f)$  factorizes over examples we obtain

$$\exp(L_1) = \prod_{i=1}^n \mathcal{N}(y_i|\mu_i, \sigma_n^2) \exp \left( -\frac{1}{2\sigma_n^2} \tilde{K}_{ii} \right).$$

Note that

$$\log p(y|u) - L_1 = \text{KL} (p(f|u) || p(f|u, y)).$$

Using the lower bound  $L_1$  we obtain a lower bound for the marginal likelihood

$$\log p(y) = \log \left( \int p(y|u)p(u)du \right) \geq \log \left( \int \exp(L_1)p(u)du \right) = L_2.$$

With some algebraic manipulations we obtain the following expression for  $L_2$

$$L_2 = \log \mathcal{N}(y|0, K_{nm}K_{mm}^{-1}K_{mn} + \sigma_n^2 I) - \frac{1}{2\sigma_n^2} \text{tr}(\tilde{K}).$$

This is exactly the expression for the lower bound, used in the method, described in the section 2.1.1 for the optimal approximating distribution  $q(u) = \mathcal{N}(u|\hat{u}, \Lambda^{-1})$ , where

$$\Lambda = \frac{1}{\sigma_n^2} K_{mm}^{-1} K_{mn} K_{nm} K_{mm}^{-1} + K_{mm}^{-1},$$

$$\hat{u} = \frac{1}{\sigma_n^2} \Lambda^{-1} K_{mm}^{-1} K_{mn} y.$$

In the method, described in section 2.1.1, this lower bound is being maximized over the kernel hyper-parameters and the optimal distribution  $q(u)$  is used for making predictions at unseen points  $x$  as follows

$$\mathbb{E}f(x) = K_{xm}K_{mm}^{-1}\hat{u},$$

$$\text{cov}(f(x), f(x')) = k(x, x') - K_{xm}K_{mm}^{-1}K_{mx'} + K_{xm}K_{mm}^{-1}\Lambda^{-1}K_{mm}^{-1}K_{mx'}.$$



Unfortunately, evaluating  $\Lambda$  takes  $O(nm^2)$  operations and thus this method cannot be applied to big datasets. To overcome this limitation, we will use stochastic optimization to find the approximate optimal distribution  $q(u)$  and to optimize for hyper-parameters.

Let the variational distribution  $q$  be normal with mean  $\mu$  and covariance matrix  $\Sigma$ . The final ELBO is derived as follows

$$\log p(y) \geq \int (L_1 + \log p(u) - \log q(u)) q(u) du = L_3.$$

This lower bound factorizes over the examples

$$\begin{aligned} L_3 &= \sum_{i=1}^n \left( \log \mathcal{N}(y_i | k_i^T K_{mm}^{-1} \mu, \sigma_n^2) - \frac{1}{2\sigma_n^2} \tilde{K}_{ii} - \frac{1}{2} \text{tr} \left( \frac{1}{\sigma_n^2} \Sigma K_{mm}^{-1} k_i k_i^T K_{mm}^{-1} \right) \right) - \text{KL}(q(u) \parallel p(u)) = \\ &= \sum_{i=1}^n \left( \log \mathcal{N}(y_i | k_i^T K_{mm}^{-1} \mu, \sigma_n^2) - \frac{1}{2\sigma_n^2} \tilde{K}_{ii} - \frac{1}{2} \text{tr}(\Sigma \Lambda_i) \right) - \\ &\quad - \frac{1}{2} \left( \log \frac{|K_{mm}|}{|\Sigma|} - m + \text{tr}(K_{mm}^{-1} \Sigma) + \mu^T K_{mm}^{-1} \mu \right), \end{aligned}$$

where  $\Lambda_i = \frac{1}{\sigma_n^2} K_{mm}^{-1} k_i k_i^T K_{mm}^{-1}$ , and  $k_i$  is the  $i$ -th column of the matrix  $K_{mn}$ .

In stochastic variational inference natural gradients are used to maximize the ELBO. The canonical parameters for the normal distribution  $q(u)$  are

$$\eta_1 = \Sigma^{-1} \mu, \quad \eta_2 = -\frac{1}{2} \Sigma^{-1}.$$

The expectation parameters are

$$\beta_1 = \mu, \quad \beta_2 = \mu \mu^T + \Sigma.$$

In the exponential family the natural gradients are equal to the gradients with respect to expectation parameters. To find these gradients we first reparametrise the ELBO

$$\begin{aligned} L_3(\beta_1, \beta_2) &= \sum_{i=1}^n \left( \log \mathcal{N}(y_i | k_i^T K_{mm}^{-1} \beta_1, \sigma_n^2) - \frac{1}{2\sigma_n^2} \tilde{K}_{ii} - \frac{1}{2} \text{tr}((\beta_2 - \beta_1 \beta_1^T) \Lambda_i) \right) - \\ &\quad - \frac{1}{2} (\log |K_{mm}| - \log |\beta_2 - \beta_1 \beta_1^T| - m + \text{tr}(K_{mm}^{-1} (\beta_2 - \beta_1 \beta_1^T)) + \beta_1^T K_{mm}^{-1} \beta_1). \end{aligned}$$

Differentiating with respect to expectation parameters we obtain

$$\frac{\partial L_3}{\partial \beta_1} = -\frac{1}{\sigma_n^2} \sum_{i=1}^n (K_{mm}^{-1} k_i y_i) + \Sigma^{-1} \mu, \quad (3)$$

$$\frac{\partial L_3}{\partial \beta_2} = \frac{1}{2} \left( -\sum_{i=1}^n (\Lambda_i) + \Sigma^{-1} - K_{mm}^{-1} \right). \quad (4)$$

The natural gradient descent updates of these parameters are

$$\begin{aligned}\eta_{1(t+1)} &= \Sigma_{(t+1)}^{-1} \mu_{(t+1)} = \Sigma_{(t)}^{-1} \mu_{(t)} + \ell \left( \frac{1}{\sigma_n^2} K_{mm}^{-1} K_{mn} y - \Sigma_{(t)}^{-1} \mu_{(t)} \right), \\ \eta_{2(t+1)} &= -\frac{1}{2} \Sigma_{(t+1)}^{-1} = -\frac{1}{2} \Sigma_{(t)}^{-1} + \ell \left( -\frac{1}{2} \Lambda + \frac{1}{2} \Sigma_{(t+1)}^{-1} \right),\end{aligned}$$

where  $\ell$  is the step length. It's easy to see, that if  $\ell = 1$  the method converges to the optimal distribution  $q(u)$  in one iteration. Unfortunately, we can not directly compute the updates described above, because the computational complexity of computing the matrix  $\Lambda$  is  $O(nm^2)$ . We will use approximations to the natural gradients, obtained by considering the data points individually or in batches. The formulas for these approximations can be obtained from equalities 3, 4.

Finally, we need to find the derivatives of the ELBO with respect to kernel hyper-parameters  $\theta$  apart from  $\sigma_n^2$

$$\begin{aligned}\frac{\partial L_3}{\partial \theta} &= \sum_{i=1}^n \left[ \frac{1}{\sigma_n^2} (y_i - k_i^T K_{mm}^{-1} \mu) \left( \frac{\partial k_i^T}{\partial \theta} K_{mm}^{-1} - k_i^T K_{mm}^{-1} \frac{\partial K_{mm}}{\partial \theta} K_{mm}^{-1} \right) \mu + \right. \\ &+ \frac{1}{2\sigma_n^2} \left( -\frac{\partial K_{nn}}{\partial \theta} + \frac{\partial K_{nm}}{\partial \theta} K_{mm}^{-1} K_{mn} + K_{nm} K_{mm}^{-1} \frac{\partial K_{mm}}{\partial \theta} K_{mm}^{-1} K_{mn} + K_{nm} K_{mm}^{-1} \frac{\partial K_{mn}}{\partial \theta} \right)_{ii} \\ &+ \left. \frac{1}{\sigma_n^2} \text{tr} \left( \Sigma \left( K_{mm}^{-1} \frac{\partial K_{mm}}{\partial \theta} K_{mm}^{-1} k_i k_i^T K_{mm}^{-1} - K_{mm}^{-1} \frac{\partial k_i^T}{\partial \theta} k_i^T K_{mm}^{-1} \right) \right) \right] - \\ &- \frac{1}{2} \text{tr} \left( K_{mm}^{-1} \frac{\partial K_{mm}}{\partial \theta} \right) + \frac{1}{2} \text{tr} \left( \Sigma K_{mm}^{-1} \frac{\partial K_{mm}}{\partial \theta} K_{mm}^{-1} \right) + \frac{1}{2} \mu^T K_{mm}^{-1} \frac{\partial K_{mm}}{\partial \theta} K_{mm}^{-1} \mu,\end{aligned}$$

and for  $\sigma_n$  we have the same formula plus the following correction

$$\sum_{i=1}^n \left( -\frac{1}{\sigma_n} + \frac{1}{\sigma_n^3} (k_i^T K_{mm}^{-1} \mu - y_i)^2 + \frac{1}{\sigma_n^3} \tilde{K}_{ii} + \frac{\text{tr}(\Sigma \Lambda_i)}{\sigma_n} \right).$$

Now, we can optimize the kernel hyper-parameters and the noise variance alongside the variational parameters.

We can also maximize the  $L_3$  with procedures, other than stochastic gradient descent. However, in most of the effective optimization methods we can't use natural gradients, because they are not necessarily a descending direction. Thus, we have to use the usual gradients. However, there is a problem with this approach as well. The steps in the direction of the antigradient does not guarantee that the updated covariance  $\Sigma$  is positive definite.

To solve this problems, we use Choletsky decomposition  $L_\Sigma$  of  $\Sigma$  and optimize  $L_3$  with respect to it.

$$L_3(L_\Sigma, \mu) = \sum_{i=1}^n \left( \log \mathcal{N}(y_i | k_i^T K_{mm}^{-1} \mu, \sigma_n^2) - \frac{1}{2\sigma_n^2} \tilde{K}_{ii} - \frac{1}{2} \text{tr}(L_\Sigma L_\Sigma^T \Lambda_i) \right) -$$

$$\begin{aligned}
& -\frac{1}{2} \left( \log \frac{|K_{mm}|}{|L_\Sigma L_\Sigma^T|} - m + \text{tr}(K_{mm}^{-1} L_\Sigma L_\Sigma^T) + \mu^T K_{mm}^{-1} \mu \right) = \\
& = \sum_{i=1}^n \left( \log \mathcal{N}(y_i | k_i^T K_{mm}^{-1} \mu, \sigma_n^2) - \frac{1}{2\sigma_n^2} \tilde{K}_{ii} - \frac{1}{2} \text{tr}(L_\Sigma^T \Lambda_i L_\Sigma) \right) - \\
& - \frac{1}{2} \left( \log |K_{mm}| - 2 \sum_{j=1}^m \log(L_\Sigma)_{jj} - m + \text{tr}(L_\Sigma^T K_{mm}^{-1} L_\Sigma) + \mu^T K_{mm}^{-1} \mu \right)
\end{aligned}$$

The gradients with respect to  $\mu$  and  $L_\sigma$  are given by

$$\begin{aligned}
\frac{\partial L_3}{\partial \mu} &= \sum_{i=1}^n \left( \Lambda_i \mu - \frac{y_i}{\sigma_n^2} K_{mm}^{-1} k_i \right) + K_{mm}^{-1} \mu, \\
\frac{\partial L_3}{\partial L_\Sigma} &= - \sum_{i=1}^n \Lambda_i L_\Sigma + \begin{pmatrix} \frac{1}{(L_\Sigma)_{11}} & 0 & \dots & 0 \\ 0 & \frac{1}{(L_\Sigma)_{22}} & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & \frac{1}{(L_\Sigma)_{mm}} \end{pmatrix} - K_{mm}^{-1} L_\Sigma.
\end{aligned}$$

### 3 Experiments

In this section the results of the numerical experiments are provided.

#### 3.1 Variations of the stochastic variational inference method

In this section we compare several variations of the stochastic variational inference method.

The first variation is denoted by `svi-natural`. It is the method described in [2]. It uses stochastic gradient descent with natural gradients for minimizing the ELBO with respect to the variational parameters, and usual gradients with respect to kernel hyperparameters.

The methods `svi-L-BFGS-B` and `svi-FG` use the full (non-stochastic) ELBO from the same article [2] and minimize it with L-BFGS-B and gradient descent respectively. These methods use Cholesky factorization (see 2.1.2) for the variational parameters.

Finally, the `svi-SAG` method to minimize the ELBO. This method also uses Cholesky factorization.

We will compare the methods on datasets, generated from some gaussian process and on real data.

##### 3.1.1 Small data

In this section we compare the methods performance on small datasets.

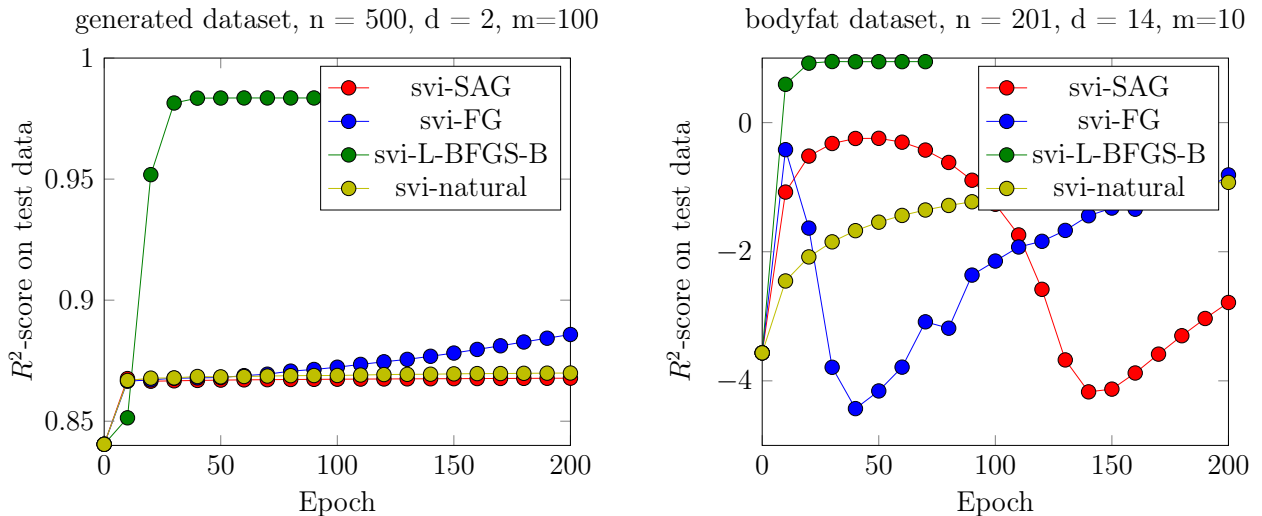


Рис. 1: Method's performance on small datasets

##### 3.1.2 Medium data

In this section we compare the methods performance on medium datasets.

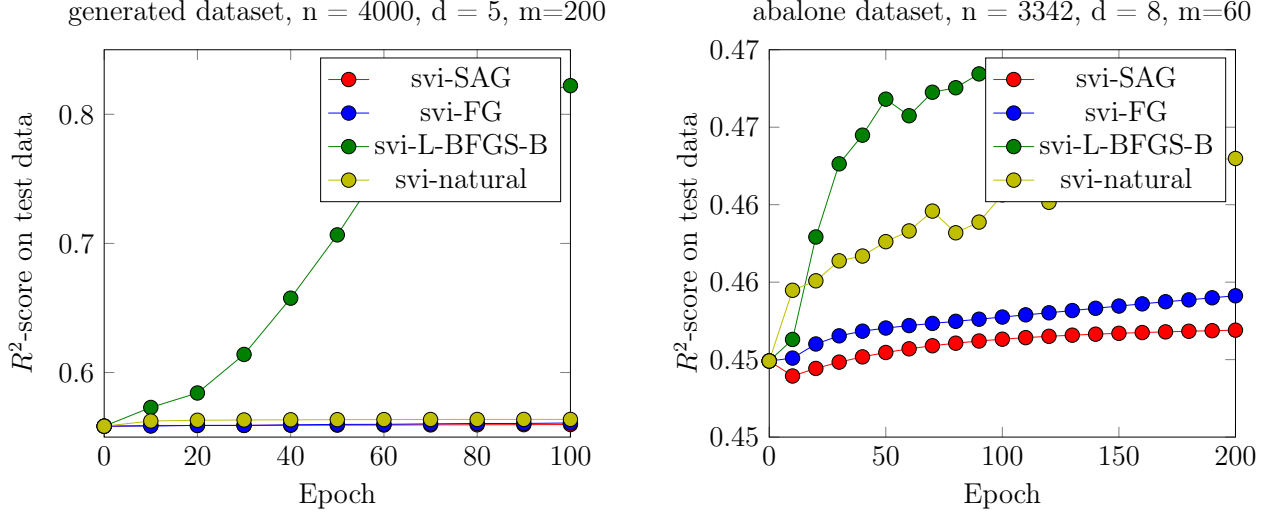


FIG. 2: Method's performance on medium datasets

### 3.2 Comparison of stochastic and non-stochastic variational inference methods

In this section we compare the **vi-means** method with **svi-L-BFGS-B**. The **vi-means** method is a variation of the method, described in section 2.1.1. It does not optimize for the inducing point positions and does uses L-BFGS-B to maximize the ELBO.

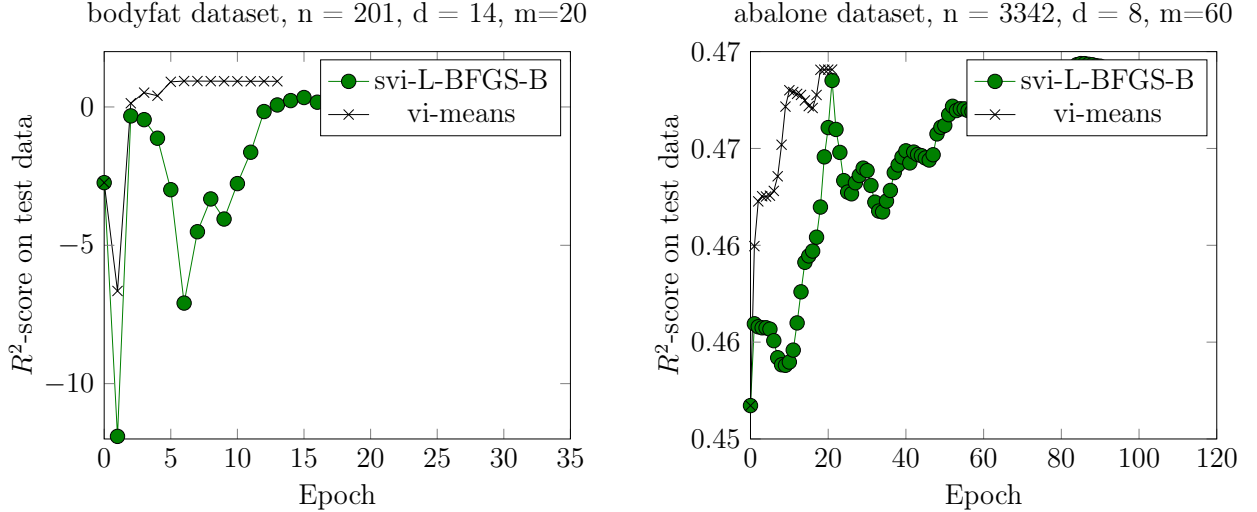


FIG. 3: Method's performance on small and medium datasets

We can see, that **vi-means** beats it's oponent in all the experiments. One could expect these results, because **vi-means** optimizes the exact same functional as it's oponent, but

cadata dataset,  $n = 10000$ ,  $d = 8$ ,  $m=200$

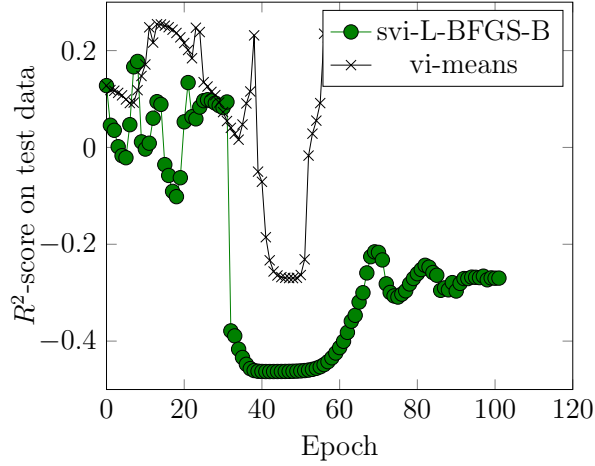


Рис. 4: Method's performance on a big dataset

it uses exact optimal values for some of the parameters. Thus, on moderate problems the `vi-means` method beats all the discussed svi variations.

Finally, we will compare `vi-means` with stochastic `svi-natural` and `svi-SAG` on a big dataset. The results can be found in fig. 5.

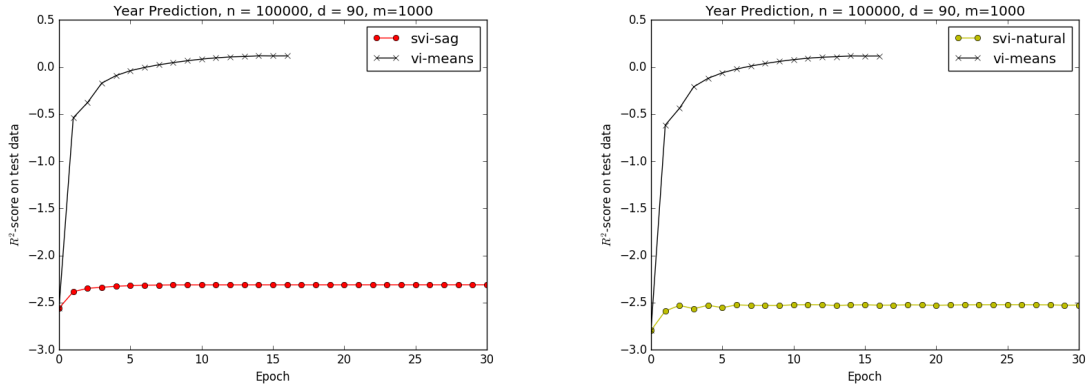


Рис. 5: vi and svi methods comparison on a big dataset

### 3.3 Variations of variational inference method

In this section we compare several variations of the stochastic variational inference method. The method itself is described in section 2.1.1. We compare two different optimization methods for minimizing the Titsias's ELBO.

The first variation is denoted by `means-PN`. It uses Projected-Newton method for minimizing

the ELBO. The second variation is denoted by **means-L-BFGS-B** and uses L-BFGS-B optimization method.

The **means-PN** uses finite-difference approximation of the hessian. It also makes hessian-correction in order to make it symmetric positive-definite.

We compare the methods on several different datasets. The results on a small and medium datasets can be found in fig. ?? . The results on a bigger dataset can be found in fig. ??

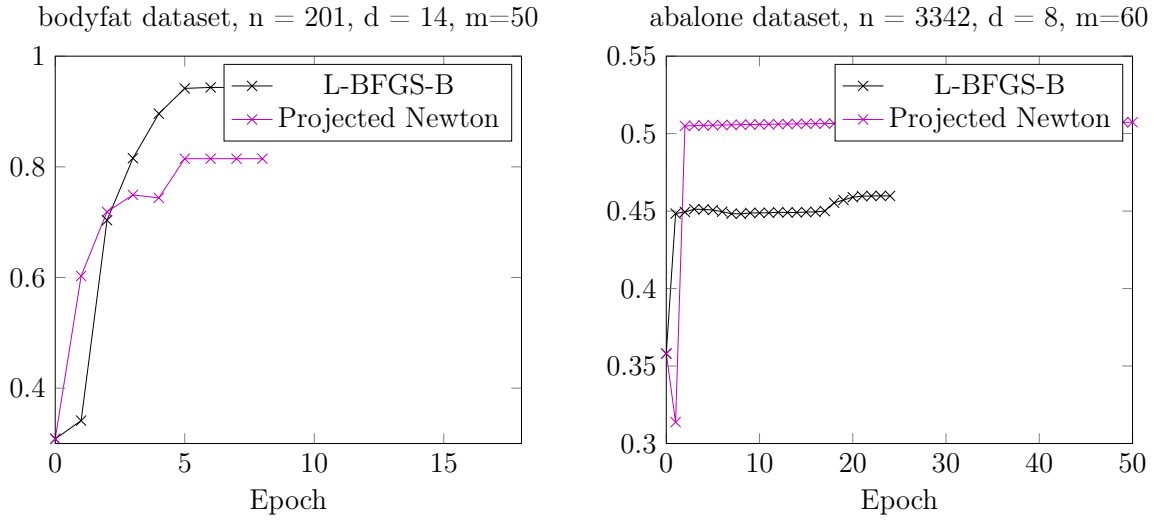


Рис. 6: Method's performance on small and medium datasets

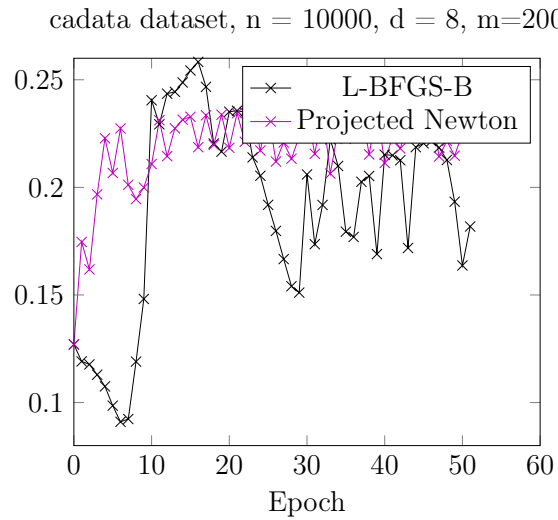


Рис. 7: Method's performance on a bigger dataset

## Literature

- [1] Titsias M. K. (2009). Variational Learning of Inducing Variables in Sparse Gaussian Processes. In: *International Conference on Artificial Intelligence and Statistics*, pp. 567–574.
- [2] Hensman J., Fusi N., Lawrence D. (2013). Gaussian Processes for Big Data. In: *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence*.