Gaussian Process-based Amortization of Variational Message Passing Update Rules

Hoang M.H. Nguyen, Semih Akbayrak, Magnus T. Koudahl, and Bert de Vries
Eindhoven University of Technology
Eindhoven, the Netherlands
{m.h.n.hoang, s.akbayrak, m.t.koudahl, bert.de.vries}@tue.nl

***Abstract*—Variational Message Passing facilitates automated variational inference in factorized probabilistic models where connected factors are conjugate pairs. Conjugate-computation Variational Inference (CVI) extends the applicability of VMP to models comprising both conjugate and non-conjugate factors. CVI makes use of a gradient that is estimated by Monte Carlo (MC) sampling, which potentially leads to substantial computational load. As a result, for models that feature a large number of non-conjugate pairs, CVI-based inference may not scale well to larger model sizes. In this paper, we propose a Gaussian Process-enhanced CVI approach, called GP-CVI, to amortize the computational costs caused by the MC sampling procedures in CVI. Specifically, we train a Gaussian process regression (GPR) model based on a set of incoming outgoing message pairs that were generated by CVI. In operation, we use the "cheaper" GPR model to produce outgoing messages and resort to the more accurate but expensive CVI message only if the variance of the outgoing message exceeds a threshold. By experimental validation, we show that GP-CVI gradually uses more fast memory-based update rule computations, and less sampling-based update rule computations. As a result, GP-CVI speeds up CVI with a controllable effect on the accuracy of the inference procedure.**

***Index Terms*—Conjugate-computation Variational Inference, Gaussian Process Regression, Variational Inference, Variational Message Passing**

I. Introduction

Bayesian inference has long been known as one of the pillars for the development of machine learning. In most practical problems, we rarely use exact Bayesian inference due to intractable computations over latent variables, but rather resort to variational inference [1], [2]. The variational method transforms an inference task to a Free Energy minimization problem. If a model is represented by a factor graph, then variational inference can be interpreted as a message passing procedure called Variational Message Passing (VMP) [3]. In principle, VMP applies to models comprising conjugate factors from the Exponential Family distributions, since in this case the variational posteriors can be cheaply computed in closed-form. For models with non-conjugate factor pairs, VMP does not yield closed-form update rules.

In order to extend the applicability of VMP to models consisting of non-conjugate factors, [4] introduced a novel stochastic approximation method called Conjugate-Computation Variational Inference (CVI). This method applies a stochastic mirror-descent method in mean-parameter space to the non-conjugate factors, so that each gradient step can be carried out by conjugate computations. As a result, CVI obtains closed-form variational posteriors as in VMP. However, the gradient in the gradient step of CVI is estimated by Monte Carlo sampling, which is computationally expensive and may result in long inference duration.

Inspired by [5], in this paper we amortize the computational cost of the Monte Carlo sampling phase in CVI by proposing a Gaussian Process-based enhancement to CVI, called GP-CVI. The idea is that we train a Gaussian process regression (GPR) model to learn the mapping from incoming messages to the corresponding outgoing messages, where the training data is supplied by usage of CVI. In operation, the GPR model is used to predict the outgoing message. If the uncertainty of the outgoing message is greater than a (user-selected) threshold value, then the sampling-based update rule of CVI is recruited to estimate the outgoing message. Moreover, the CVI message is used to update the GPR model. As a consequence, the GP-CVI method will over time use more fast memory-based update rules (by the GPR model) and fewer expensive sampling-based update rules (by CVI).

II. Background

In variational inference, we approximate the exact posterior by solving an optimization problem [2]. Specifically, given a generative model $p(\boldsymbol{y}, \boldsymbol{z})$, where $\boldsymbol{y}$ are the observed variables and $\boldsymbol{z} = (z_1, \dots, z_M)^T$ are the latent variables, the variational posterior $q^*(\boldsymbol{z}) \approx p(\boldsymbol{z}|\boldsymbol{y})$ is found by minimizing Free Energy (an upper bound on negative log-evidence) [6]

$$q^* = \arg\min_q \underbrace{\int_{\boldsymbol{z}} q(\boldsymbol{z}) \log \frac{q(\boldsymbol{z})}{p(\boldsymbol{y}, \boldsymbol{z})}\, d\boldsymbol{z}}_{\text{Free Energy } F[q]} . \tag{1}$$

A common constraint to facilitate the minimization process is the mean-field factorization $q(\boldsymbol{z}) = \prod_{k=1}^{M} q_k(z_k)$. This assumption leads to the following relationships [7]

$$\forall_k : \ q_k^*(z_k) \propto \exp\left(\mathbb{E}_{q^*_{\setminus k}}[\log p(\boldsymbol{y}, \boldsymbol{z})]\right), \tag{2}$$

where $\mathbb{E}_{q_{\setminus k}}[.]$ refers to the expectation with respect to all factors of $q(\boldsymbol{z})$ excluding $q_k(z_k)$. In a factor graph representation of $p(\boldsymbol{y}, \boldsymbol{z})$, (2) can be iteratively solved by Variational Message Passing (VMP) [3]. In this paper, we employ the Forney-style factor graph (FFG) representation for models [8] . An FFG is composed of nodes and edges that represent the factors and

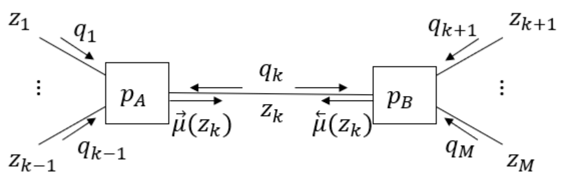


Fig. 1: An FFG representation of an edge z_k with two factors p_A and p_B in the model $p(\boldsymbol{y}, \boldsymbol{z})$. $\overrightarrow{\mu}(z_k)$ and $\overleftarrow{\mu}(z_k)$ are forward and backward VMP messages, respectively, and $q_k(z_k)$ is a marginal of the variational posterior $q(\boldsymbol{z})$. The VMP computations for $\overrightarrow{\mu}(z_k)$, $\overleftarrow{\mu}(z_k)$ and q_k are given by (3a), (3b) and (3c).

variables respectively. An edge connects to a node only if the variable of that edge is the argument of the factor of that node, see Fig. 1 as an example. For a more detailed description of FFGs, we refer to [8].

Consider Fig. 1 as a sub-graph of the FFG for $p(\boldsymbol{y}, \boldsymbol{z})$ at the edge z_k. The VMP update rules for forward and backward messages $\overrightarrow{\mu}(z_k)$, $\overleftarrow{\mu}(z_k)$ and marginal $q_k(z_k)$ were derived in [9] as

$$\overrightarrow{\mu}(z_k) \propto \exp\left(\mathbb{E}_{q_1,\ldots,q_{k-1}}\left[\log p_A(z_1, \ldots, z_k)\right]\right) \tag{3a}$$

$$\overleftarrow{\mu}(z_k) \propto \exp\left(\mathbb{E}_{q_{k+1},\ldots,q_M}\left[\log p_B(z_k, \ldots, z_M)\right]\right) \tag{3b}$$

$$q_k(z_k) \propto \overrightarrow{\mu}(z_k)\overleftarrow{\mu}(z_k)\,. \tag{3c}$$

Equation (3c) can be carried out with simple computations if the factors p_A and p_B are conjugate pairs drawn from the exponential family (EF). Recall that a distribution in EF has the form [7]

$$p(x; \boldsymbol{\eta}) = h(x)g(\boldsymbol{\eta})\exp\left(\boldsymbol{\eta}^T\boldsymbol{u}(x)\right)\,, \tag{4}$$

where $\boldsymbol{\eta}$ are the natural parameters, $\boldsymbol{u}(x)$ are the sufficient statistics, $h(x)$ is the base measure, and $g(\boldsymbol{\eta})$ can be interpreted as a normalizing factor. The superscript T denotes the transposition of matrices and vectors.

If p_A and p_B are a conjugate pair from EF, then the messages $\overrightarrow{\mu}(z_k)$ and $\overleftarrow{\mu}(z_k)$ also have the EF form with the same sufficient statistics, i.e.,

$$\overrightarrow{\mu}(z_k) \propto \exp\left(\boldsymbol{\eta}^T_{\overrightarrow{\mu}}\,\boldsymbol{u}(z_k)\right), \tag{5}$$

$$\overleftarrow{\mu}(z_k) \propto \exp\left(\boldsymbol{\eta}^T_{\overleftarrow{\mu}}\,\boldsymbol{u}(z_k)\right), \tag{6}$$

and the multiplication in (3c) results to the sum of the natural parameters in the exponent

$$q_k(z_k) \propto \exp\left[(\boldsymbol{\eta}_{\overrightarrow{\mu}} + \boldsymbol{\eta}_{\overleftarrow{\mu}})^T\boldsymbol{u}(z_k)\right]. \tag{7}$$

This type of computation is referred to as a conjugate computation and, by exploiting (7), VMP can efficiently update variational posteriors in closed-form. However, for models where $\overrightarrow{\mu}(z_k)$ and $\overleftarrow{\mu}(z_k)$ either come from non-conjugate factors or one of the messages is intractable, VMP loses the convenient solution (7).

To cope with these kind of models, [4] introduced Conjugate-computation Variational Inference (CVI), which employs a stochastic mirror descent method in the mean parameter space so that each gradient step can be carried out by the conjugate computation. CVI only applies stochastic computation to the non-conjugate factors and keeps using VMP for the conjugate part of the models. These features of CVI allows approximation of the non-conjugate (or intractable) messages by conjugate ones so the closed-form solution (7) can be obtained. More specifically, let us assume that $\overleftarrow{\mu}(z_k)$ is intractable while $\overrightarrow{\mu}(z_k)$ still keeps the form (5). Then $\overleftarrow{\mu}(z_k)$ can be approximated by a "CVI message" $\overleftarrow{m}(z_k)$, defined as

$$\overleftarrow{m}(z_k) \propto \exp\left(\tilde{\boldsymbol{\eta}}^T_{\overleftarrow{m}}\boldsymbol{u}(z_k)\right)\,. \tag{8}$$

Clearly, estimating $\overleftarrow{m}(z_k)$ amounts to estimating the natural parameter $\tilde{\boldsymbol{\eta}}_{\overleftarrow{m}}$. By employing Algorithm 2 in [4], $\tilde{\boldsymbol{\eta}}_{\overleftarrow{m}}$ can be estimated by

$$\begin{aligned}\tilde{\boldsymbol{\eta}}_{\overleftarrow{m},t+1} = (1-\rho_t)&\tilde{\boldsymbol{\eta}}_{\overleftarrow{m},t}\\ &+ \rho_t\tilde{\nabla}_{\boldsymbol{\mu}_k}\mathbb{E}_{q_k,\ldots,q_M}[\log p_B]|_{\boldsymbol{\mu}_k=\boldsymbol{\mu}_{k,t}}\,,\end{aligned} \tag{9}$$

where t is the iteration index, $\boldsymbol{\mu}_k = \mathbb{E}_{q_k}[\boldsymbol{u}(z_k)]$ is called the mean parameter, ρ is the step size, and $\tilde{\nabla}$ refers to the gradient, estimated by a Monte-Carlo method. After estimating $\tilde{\boldsymbol{\eta}}_{\overleftarrow{m},t+1}$, $q_k(z_k)$ can be updated in closed-form by

$$q_{k,t+1}(z_k) \propto \exp\left[(\boldsymbol{\eta}_{\overrightarrow{\mu}} + \tilde{\boldsymbol{\eta}}_{\overleftarrow{m},t+1})^T\boldsymbol{u}(z_k)\right]\,. \tag{10}$$

(9) and (10) then repeat until reaching a convergence criterion. As a notational convention, we will denote the final value of $\tilde{\boldsymbol{\eta}}_{\overleftarrow{m}}$ by $\tilde{\boldsymbol{\eta}}^*_{\overleftarrow{m}}$.

Usage of the Monte Carlo method in (9) may lead to a computational cost issue for CVI, especially when a large number of samples are required and we need many iterations to reach convergence $\tilde{\boldsymbol{\eta}}^*_{\overleftarrow{m}}$. This is the motivation for the Gaussian process-enhanced CVI approach in the next section.

III. Gaussian Process-CVI

In this section, we present the Gaussian Process-CVI (GP-CVI) approach in which we use Gaussian Process Regression (GPR) to estimate the CVI message $\tilde{\boldsymbol{\eta}}^*_{\overleftarrow{m}}$. More specifically, we assume $\tilde{\boldsymbol{\eta}}_{\overleftarrow{m}} \in \mathbb{R}^D$ follows a zero-mean multivariate Gaussian Process (GP)

$$\tilde{\boldsymbol{\eta}}_{\overleftarrow{m}} \sim \mathcal{GP}(\boldsymbol{0}, \boldsymbol{\mathcal{K}}(\boldsymbol{x}, \boldsymbol{x'})), \tag{11}$$

where $\boldsymbol{x}$ denotes the input of the GP, and $\boldsymbol{\mathcal{K}}(\boldsymbol{x}, \boldsymbol{x'}) \in \mathbb{R}^{D\times D}$ is the covariance matrix. From (9), we recognize that the input $\boldsymbol{x}$ comprises the information of $q(z_k), \ldots, q(z_M)$, namely their natural parameters $\boldsymbol{\eta}_k, \ldots, \boldsymbol{\eta}_M$. Thus, we define $\boldsymbol{x}$ to be a vector obtained by vertically stacking $\boldsymbol{\eta}_k, \ldots, \boldsymbol{\eta}_M$. In order to define the covariance matrix, we employ the intrinsic coregionalization model (ICM) [10], [11]

$$\boldsymbol{\mathcal{K}}(\boldsymbol{x}, \boldsymbol{x'}) = \boldsymbol{\mathcal{C}}\,k(\boldsymbol{x}, \boldsymbol{x'}), \tag{12}$$

where $\boldsymbol{\mathcal{C}} \in \mathbb{R}^{D\times D}$ is called the co-regionalization matrix, and $k(\boldsymbol{x}, \boldsymbol{x'}) \in \mathbb{R}$ is a kernel. For simplicity, we assign $\boldsymbol{\mathcal{C}}$ to an identity matrix and $k(\boldsymbol{x}, \boldsymbol{x'})$ to a squared-exponential kernel

$$k(\boldsymbol{x}, \boldsymbol{x'}) = \sigma_f^2\exp\left(-\frac{(\boldsymbol{x}-\boldsymbol{x'})^T\boldsymbol{\Lambda}^{-2}(\boldsymbol{x}-\boldsymbol{x'})}{2}\right), \tag{13}$$

where σ_f^2 is the function variance, $\boldsymbol{\Lambda} = \mathrm{diag}(\boldsymbol{l})$ for $\boldsymbol{l}$ being a vector of length scales. σ_f^2 and $\boldsymbol{l}$ are hyper-parameters of the GP, and we denote them collectively by $\boldsymbol{\theta}$.

Consider again the generative model $p(\boldsymbol{y}, \boldsymbol{z})$ with N observations $y_1, \dots, y_N$. First, we perform CVI on the first N_1 observations and collect all results $\tilde{\boldsymbol{\eta}}^*_{\overleftarrow{m}}$ to generate a training set $\mathcal{D} = \{\boldsymbol{x}_i, \tilde{\boldsymbol{\eta}}^*_{\overleftarrow{m},i}\}_{i=1}^{N_1}$. Then, we use $\mathcal{D}$ to optimize the hyper-parameters of the GP. Specifically, we minimize the negative log-evidence

$$\begin{aligned}\mathcal{L}(\mathcal{D}) &= -\log p(\mathcal{D}) \\ &= \frac{1}{2}\bar{\boldsymbol{\eta}}^T \mathbf{K}^{-1} \bar{\boldsymbol{\eta}} + \frac{1}{2}\log|\mathbf{K}| + \frac{N_1 D}{2}\log 2\pi \ , \qquad (14)\end{aligned}$$

where $\bar{\boldsymbol{\eta}} \in \mathbb{R}^{DN_1}$ is the vector in which vectors $\{\tilde{\boldsymbol{\eta}}^*_{\overleftarrow{m},i}\}_{i=1}^{N_1}$ are stacked vertically, and $\mathbf{K} \in \mathbb{R}^{DN_1 \times DN_1}$ is the covariance matrix corresponding to the input matrix $\mathbf{X} = (\boldsymbol{x}_1, \dots, \boldsymbol{x}_{N_1})$

$$\mathbf{K} = \mathbf{K}(\mathbf{X}, \mathbf{X}) = \boldsymbol{\mathcal{C}} \otimes k(\mathbf{X}, \mathbf{X}), \qquad (15)$$

where $k(\mathbf{X}, \mathbf{X}) \in \mathbb{R}^{N_1 \times N_1}$ is the matrix whose entries are calculated by (13), and $\otimes$ denotes the Kronecker product. Since the partial derivatives of (14) with respect to $\boldsymbol{\theta}$ are analytically computable [12], (14) can be minimized by a gradient-based method. For a fast training process, in this paper we use stochastic gradient descent [13], where $\mathcal{D}$ is split into mini-batches b and gradient descent is applied to each batch

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \rho \nabla_{\boldsymbol{\theta}} \mathcal{L}_b(\mathcal{D})|_{\boldsymbol{\theta}=\boldsymbol{\theta}_t}. \qquad (16)$$

After the initial training phase, the GPR model takes over to directly predict $\tilde{\boldsymbol{\eta}}^*_{\overleftarrow{m}}$ for the next observations (starting from y_{N_1+1}). Let $\boldsymbol{x}_+$ be a new input, then the natural parameters of the CVI message get computed by [11], [12]

$$\tilde{\boldsymbol{\eta}}^+_{\overleftarrow{m}} = \mathbf{K}_+^T \mathbf{K}(\mathbf{X}, \mathbf{X})^{-1} \bar{\boldsymbol{\eta}}\,, \qquad (17)$$

$$\boldsymbol{\Sigma}_{++} = \boldsymbol{\mathcal{K}}_{++} - \mathbf{K}_+^T \mathbf{K}(\mathbf{X}, \mathbf{X})^{-1} \mathbf{K}_+\,, \qquad (18)$$

where $\tilde{\boldsymbol{\eta}}^+_{\overleftarrow{m}}$ is the predictive value, $\boldsymbol{\Sigma}_{++} \in \mathbb{R}^{D \times D}$ is the predictive covariance matrix, $\mathbf{K}_+ = \mathbf{K}(\mathbf{X}, \boldsymbol{x}_+) \in \mathbb{R}^{DN_1 \times D}$ is defined as in (15), and $\boldsymbol{\mathcal{K}}_{++} = \boldsymbol{\mathcal{K}}(\boldsymbol{x}_+, \boldsymbol{x}_+) \in \mathbb{R}^{D \times D}$ is defined as in (12). The uncertainty of the prediction is on the main diagonal of $\boldsymbol{\Sigma}_{++}$, and for convenience we denote them collectively by σ^2_{++}. GP-CVI comes also with a user-selectable uncertainty threshold σ^2_{Thres}. If $\sigma^2_{++} \geq \sigma^2_{\text{Thres}}$, then we switch back to using CVI to estimate $\tilde{\boldsymbol{\eta}}^*_{\overleftarrow{m}}$ and we add the result to the training set $\mathcal{D}$ for the GPR. This process gets repeated until we have collected a certain number N_{new} of new data examples, after which we update the GPR models by stochastic gradient descent and the updated GPR models are used for the next observations. The GP-CVI method is summarized in pseudo-code in Algorithm 1.

IV. Experimental validation

In this section, we demonstrate how CVI-based inference can be executed faster by GP amortization. The experiment is based on inferring the hidden states of a Poisson Linear Dynamical System (PLDS) [14] that models a Covid data set.

Algorithm 1 Pseudo-code for GP-CVI.

```
Input:
    $\boldsymbol{x} \triangleq (\boldsymbol{\eta}_{\boldsymbol{k}}^T, \dots, \boldsymbol{\eta}_{\boldsymbol{M}}^T)^T$: natural parameters of incoming messages $q(z_k), \dots, q(z_M)$;
    $N$: observation length;
    $N_1$: number of observations for the GP training phase;
    $N_{\text{new}}$: number of new observations for the GP updating phase;
    $\sigma^2_{\text{Thres}}$: uncertainty threshold;
    $\boldsymbol{\theta}$: initial values of GP hyper-parameters.
Output:
    $\tilde{\boldsymbol{\eta}}^*_{\overleftarrow{m}}$: Natural parameter of the CVI message $\overleftarrow{m}(z_k)$.
Set $i = 1$
while $i \leq N_1$ do
    Compute $\tilde{\boldsymbol{\eta}}^*_{\overleftarrow{m}}$ at $y_i$ by (9);
end while
Create the set $\mathcal{D} = \{\boldsymbol{x}_i, \tilde{\boldsymbol{\eta}}^*_{\overleftarrow{m},i}\}_{i=1}^{N_1}$ ;
Optimize $\boldsymbol{\theta}$ from $\mathcal{D}$, using (16);
Set $j = N_1 + 1$ ;
while $j \leq N$ do
    Compute $\tilde{\boldsymbol{\eta}}^*_{\overleftarrow{m}}$ and $\boldsymbol{\Sigma}_{++}$ at $y_j$ by (17) and (18);
    Get $\sigma^2_{++}$ from $diag(\boldsymbol{\Sigma}_{++})$
    if $\sigma^2_{++} \geq \sigma^2_{\text{Thres.}}$ then
        Compute $\tilde{\boldsymbol{\eta}}^*_{\overleftarrow{m}}$ by (9);
        Add $\tilde{\boldsymbol{\eta}}^*_{\overleftarrow{m}}$ to $\mathcal{D}$ ;
        if added $N_{new}$ points then
            Update $\boldsymbol{\theta}$ by (16)
        end if
    end if
    $j = j + 1$ ;
end while
```

The Covid data set is the number of positive tests per day in the Noord-Brabant province in the Netherlands recorded from June 1st, 2020 to June 13th, 2021 by the National Institute for Public Health and the Environment (RIVM)[1]. There are 378 Covid data points in total, see Fig. 2.

The PLDS model is given by

$$p(y_{1:\mathcal{T}}, \boldsymbol{z}_{1:\mathcal{T}}) = p(\boldsymbol{z}_1) p(y_1|\boldsymbol{z}_1) \prod_{t=2}^{\mathcal{T}} p(y_t|\boldsymbol{z}_t) p(\boldsymbol{z}_t|\boldsymbol{z}_{t-1}) \qquad (19)$$

$$p(\boldsymbol{z}_1) = \mathcal{N}(\boldsymbol{z}_1; 0, \boldsymbol{\Sigma}) \quad \text{(initial state)}, \qquad (20)$$

$$p(y_t|\boldsymbol{z}_t) = \mathcal{P}ois(y_t; \exp(\mathbf{B}\boldsymbol{z}_t)) \quad \text{(likelihood)}, \qquad (21)$$

$$p(\boldsymbol{z}_t|\boldsymbol{z}_{t-1}) = \mathcal{N}(\boldsymbol{z}_t; \mathbf{A}\boldsymbol{z}_{t-1}, \boldsymbol{\Sigma}) \quad \text{(state transition)}, \qquad (22)$$

where $y_{1:\mathcal{T}} \triangleq (y_1, \dots, y_{\mathcal{T}})$, with $\mathcal{T} = 378$, denotes the observations (i.e. the number of positive tests per day), and $\boldsymbol{z}_t = [x_t, v_t]^T$ is a 2-dimensional hidden state. In (19), $\mathbf{B} = \begin{bmatrix}1 & 0\end{bmatrix}$, $\mathbf{A} = \begin{bmatrix}1 & 1 \\ 0 & 1\end{bmatrix}$, and $\boldsymbol{\Sigma} = \begin{bmatrix}1 & 0 \\ 0 & 1\end{bmatrix}$. The factor graph of (19) is shown in Fig. 3, in which p_A represents $p(\boldsymbol{z}_t|\boldsymbol{z}_{t-1})$, and p_B represents $p(y_t|\boldsymbol{z}_t)$.

[1] The data was downloaded from the RIVM website at https://data.rivm.nl/covid-19/COVID-19_uitgevoerde_testen.csv.

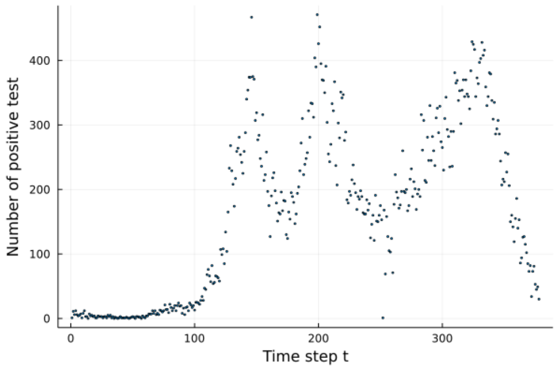


Fig. 2: Number of Covid-19 positive tests per day in the Noord-Brabant province, the Netherlands, recorded by RIVM in the period from June 1st, 2020 (represented by the index 0 on the x-axis) to June 13th, 2021 (represented by last index on the x-axis).

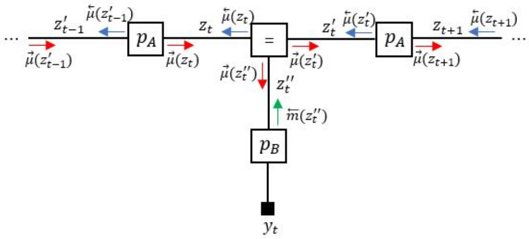


Fig. 3: The factor graph of the state space model (19) with message passing procedure. The red messages refer to the filtering stage, while the blue ones refer to the smoothing stage. The green message $\overleftarrow{m}(z''_t)$ is the CVI message.

In order to infer the hidden state $\boldsymbol{z}_t$, we approximate the posterior $p(\boldsymbol{z}_t|y_{1:\mathcal{T}})$ with $q(\boldsymbol{z}_t)$ by passing messages on the factor graph in Fig. 3. Since the likelihood node is not conjugate to the Gaussian distribution, we use GP-CVI[2] to approximate the message $\overleftarrow{\mu}(\boldsymbol{z}''_t)$ from the node p_B to the equality node by a message $\overleftarrow{m}(\boldsymbol{z}''_t)$ within Gaussian distribution family. The inputs to the Gaussian process in GP-CVI are the natural parameters of $\overrightarrow{\mu}(\boldsymbol{z}''_t)$ and y_t. In the inference stage, we employ a Rauch–Tung–Striebel [15] smoothing approach: we first perform filtering with a full forward pass (red messages) on the graph and run smoothing with a backward pass (blue messages) afterwards. The forward messages in the filtering stage are calculated by

$$\overrightarrow{\mu}(\boldsymbol{z}_t) = \int \overrightarrow{\mu}(\boldsymbol{z}'_{t-1}) p_A(\boldsymbol{z}_{t-1}, \boldsymbol{z}_t) d\boldsymbol{z}_{t-1}, \tag{23a}$$

$$\overrightarrow{\mu}(\boldsymbol{z}''_t) = \overrightarrow{\mu}(\boldsymbol{z}_t), \tag{23b}$$

$$\overrightarrow{\mu}(\boldsymbol{z}'_t) \propto \overrightarrow{\mu}(\boldsymbol{z}''_t) \overleftarrow{m}(\boldsymbol{z}''_t)\,. \tag{23c}$$

Once filtering has finished, we update the marginals by the following backward pass:

$$\overleftarrow{\mu}(\boldsymbol{z}'_t) = \int \overleftarrow{\mu}(\boldsymbol{z}_{t+1}) p_A(\boldsymbol{z}_t, \boldsymbol{z}_{t+1}) d\boldsymbol{z}_{t+1} \tag{24a}$$

$$q(z_t) \propto \overrightarrow{\mu}(\boldsymbol{z}'_t) \overleftarrow{\mu}(\boldsymbol{z}'_t) \tag{24b}$$

$$\overleftarrow{\mu}(\boldsymbol{z}_t) = q(z_t)\,. \tag{24c}$$

We employ the AdaMax optimizer [16] with a step size $\rho_t = 0.4$ in (9) for the gradient step in CVI. The setting of Gaussian Process Regression (GPR) is summarized in Table I.

Process	Parameter values
GPR training	$N_1 = 25$ $\sigma_f = 2$ $\boldsymbol{l} = (0.1, 0.2, 0.18, 0.26, 0.14, 0.09, 5)^T$ AdaMax: 1000 iterations, $\rho = 10^{-4}$, batch size = 5
GPR updating	$\sigma^2_{\text{Thres.}} = 0.8$ $N_{\text{new}} = 50$ AdaMax: 100 iterations, $\rho = 10^{-4}$, batch size = 5

TABLE I: GP configuration for the validation experiment.

To compare the performance of CVI and GP-CVI, we record the FE value and the inference time (in wall clock time) of both methods. The results[3] are given in Table II. In term of FE minimization, both CVI and GP-CVI yield similar FE values, implying that their results are nearly similar. To gain some intuition for this results, we can examine Fig. 4a, which shows the state x_t estimated by both methods from $t = 45$ to $t = 70$. The ribbon in the figure represents one standard deviation. Despite the similarity in FE, GP-CVI has a noticeably shorter inference time than CVI. To explain this difference, consider Fig. 4b that displays the uncertainty in the prediction of GPR in GP-CVI from $t = 26$ onward (since the first 25 time steps are used for training, no uncertainty is recorded for this period). By color coding, the figure also displays when GP-CVI uses GPR and CVI. At the beginning of the experiment there is little data for GPR so GP-CVI regularly uses CVI from $t = 100$ to $t = 150$. Thereafter, GP-CVI steadily switches to using Gaussian process regression (GPR) predominantly, leading to a large decrease in inference time compared to CVI. This decrease is also shown in Fig. 4b. We record the inference time of both CVI and GP-CVI in three intervals split by two fine-tuning points. We can see that GP-CVI gradually becomes faster than CVI while not sacrificing FE performance.

	CVI	GP-CVI
minimized FE	2277.86	2277.97
execution time	691.38 s	270.21 s

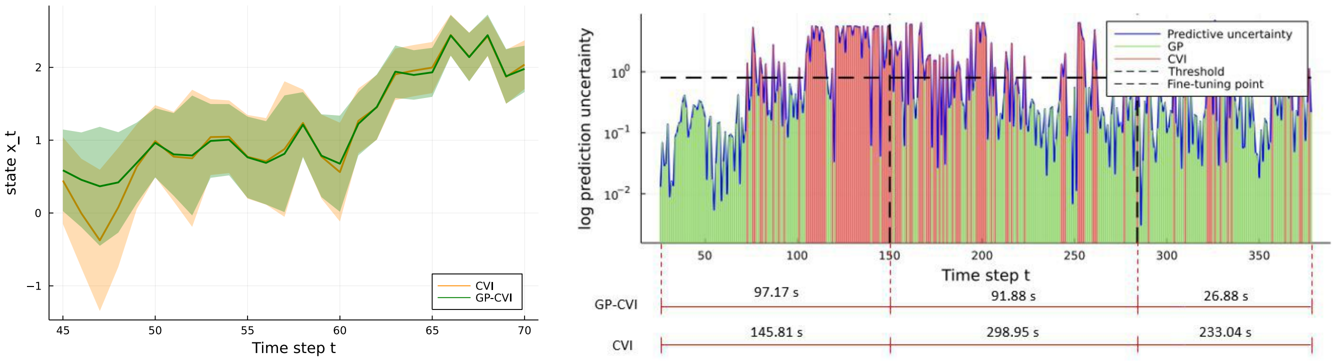
TABLE II: Minimized FE and execution time for CVI vs GP-CVI on the hidden state smoothing problem for model (19).

V. Conclusions and Future Work

In this paper, we have proposed a new approach, called GP-CVI, in which we extend CVI with Gaussian Process amor-

[2] The code to reproduce our experiments can be found at https://github.com/HoangMHNguyen/Gaussian-Process-based-CVI.

[3] We used the Julia programming language on an Intel Core i5 Processor 8250U (1.6GHz) with 8GB of RAM for all experiments.



(a) State x_t.

(b) Predictive uncertainty of GPR prediction.

Fig. 4: (a) The inferred state x_t by CVI and GP-CVI. The ribbon represents one standard deviation. (b) The uncertainty of GPR prediction in GP-CVI recorded from the time step 26-th. The red region refers to the time steps at which GP-CVI uses CVI to estimate x_t, whereas the green region refers to GPR prediction. The two vertical dashed lines are the time steps where we fine-tune GP component in the GP-CVI. The horizontal dashed line represents the threshold for the uncertainty. There are 3 intervals, and we record the inference time of both methods in each interval. We can observe that, over time, the green region dominates the red region, indicating that GP-CVI calculates messages mostly by amortization using GPR.

tization to shorten the inference time for models with non-conjugate factors. We train GPR models to predict intractable messages (or CVI messages), instead of always using Monte Carlo sampling as in CVI. Our experimental results illustrate that GP-CVI gradually speeds up CVI while achieving similar results.

There is room for future improvement in our work. As we have seen in Table I, GP-CVI has plenty of hyper-parameters to be defined, and it might require a lot of time to tune those hyper-parameters. A possible solution is applying transfer learning [17], i.e., using the same (optimized) hyper-parameters for similar models. Another drawback of the proposed method is that as GP-CVI progresses, the size of the training set of GP increases and this makes the computation of GP-CVI more complex. We can improve this feature by upgrading to more advanced GP models [18] that limit the dimension of the number of basis vectors. Extra work on these drawbacks can help improve the applicability of our GP-CVI method.

References

[1] Radford M. Neal and Geoffrey E. Hinton. A View of the Em Algorithm that Justifies Incremental, Sparse, and other Variants. In Michael I. Jordan, editor, *Learning in Graphical Models*, pages 355–368. Springer Netherlands, Dordrecht, 1998.

[2] David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. Variational Inference: A Review for Statisticians. *Journal of the American Statistical Association*, 112(518):859–877, April 2017. arXiv: 1601.00670.

[3] John Winn and Christopher M. Bishop. Variational Message Passing. *Journal of Machine Learning Research*, 6(23):661–694, 2005.

[4] Mohammad Emtiyaz Khan and Wu Lin. Conjugate-Computation Variational Inference : Converting Variational Inference in Non-Conjugate Models to Inferences in Conjugate Models. *arXiv:1703.04265 [cs]*, April 2017. arXiv: 1703.04265.

[5] Wittawat Jitkrittum, Arthur Gretton, Nicolas Heess, S. M. Ali Eslami, Balaji Lakshminarayanan, Dino Sejdinovic, and Zoltán Szabó. Kernel-Based Just-In-Time Learning for Passing Expectation Propagation Messages. *arXiv:1503.02551 [cs, stat]*, June 2015. arXiv: 1503.02551.

[6] Karl Friston, James Kilner, and Lee Harrison. A free energy principle for the brain. *Journal of Physiology-Paris*, 100(1-3):70–87, July 2006.

[7] Christopher M. Bishop. *Pattern recognition and machine learning*. Information science and statistics. Springer, New York, 2006.

[8] Hans-Andrea Loeliger, Justin Dauwels, Junli Hu, Sascha Korl, Li Ping, and Frank R. Kschischang. The Factor Graph Approach to Model-Based Signal Processing. *Proceedings of the IEEE*, 95(6):1295–1322, June 2007.

[9] Justin Dauwels. On Variational Message Passing on Factor Graphs. In *2007 IEEE International Symposium on Information Theory*, pages 2546–2550, Nice, June 2007. IEEE.

[10] Pierre Goovaerts. *Geostatistics for natural resources evaluation*. Oxford University Press, USA, 1997.

[11] Mauricio A. Alvarez, Lorenzo Rosasco, and Neil D. Lawrence. Kernels for Vector-Valued Functions: a Review. *arXiv:1106.6251 [cs, math, stat]*, April 2012. arXiv: 1106.6251.

[12] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian processes for machine learning*. Adaptive computation and machine learning. MIT Press, Cambridge, Mass, 2006. OCLC: ocm61285753.

[13] Nikhil Ketkar and Eder Santana. *Deep Learning with Python*, volume 1. Apress, Berkeley, CA, 2017.

[14] Jakob H. Macke, Lars Buesing, John P. Cunningham, Byron M. Yu, Krishna V. Shenoy, and Maneesh Sahani. Empirical models of spiking in neural populations. *Advances in Neural Information Processing Systems 24: 25th conference on Neural Information Processing Systems (NIPS 2011)*, pages 1350–1358, 2012.

[15] Herbert E. Rauch, F. Tung, and Charlotte T. Striebel. Maximum likelihood estimates of linear dynamic systems. *AIAA Journal*, 3(8):1445–1450, August 1965.

[16] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *arXiv:1412.6980 [cs]*, January 2017. arXiv: 1412.6980.

[17] Lisa Torrey and Jude Shavlik. Transfer learning. In *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*, pages 242–264. IGI global, 2010.

[18] Mauricio A. Alvarez and Neil D. Lawrence. Computationally efficient convolved multiple output Gaussian processes. *The Journal of Machine Learning Research*, 12:1459–1500, 2011.