Variational Bayesian Monte Carlo and Noisy Likelihoods: A Review

Gagesh Madaan 20111406 | **Musale Krushna Pavan** 20111268 | **Pinaki Chakraborty** 20211265 | **Shruti Sharma** 20111061

Abstract

Probabilistic models that have expensive, black-box likelihoods need advanced techniques for Bayesian inference. Variational Bayesian Monte Carlo (VBMC) is a model-fitting framework that returns not just the full posterior over model parameters but also the model evidence than can be used for Bayesian model selection. This report is a review of the paper Variational Bayesian Monte Carlo by Luigi Acerbi [2] and of further extension [4] of VBMC that addresses the limitations of the former paper by including noisy likelihoods. VBMC is an approximate inference method that combines Variational Inference with Bayesian Quadrature. We perform experiments as done in the original paper, compare with the benchmarks and provide our insights regarding them.

1 Motivation

The motivation was to look beyond the content taught in the lectures, at some state-of-the-art work happening around this field. We had some idea about the problem of intractability around the systematic framework like Bayesian Inference and after reading some content, we became sure that approximation is the only way to escape from this. At the face of it, we were curious about how the Variational Bayes Monte Carlo approach, that sounds like a subtle combination of Variational Inference and MCMC (turns out it is not), will actually outperform its own ascendants in approximating the inference for likelihoods with realistic properties like multi-modality, medium dimension $(D \sim 10)$, heavy tails, computationally expensive likelihoods as claimed by the paper.

Outline

The outline of upcoming sections with some brief description.

S.No	Section	Brief Description
2	Background	Brief introduction to the background concepts used in the VBMC paper.
3	VBMC	A simple introduction to the VBMC algorithm for the reader to understand better before going into mathematical details. The actual details of the algorithm and the expressions.
4	Experiments/Evaluations	The experiments, the evaluations and the generated results/graphs.
5	Suggested Improvements	The improvements suggested and the intuition behind it so as to reason why they seem good improvements to us.
6	Learnings	Individual Learnings

2 Background

2.1 Variational Inference

In Variational Inference we propose the inference problem i.e calculating $p(\mathbf{x}|\mathcal{D})$ as an optimization problem. $p(\mathbf{x}|\mathcal{D})$ is approximated using the simple distribution $q_\phi(\mathbf{x})$ by reducing the Kullback-Leibler (KL) divergence between them. The log-likelihood,

$$\begin{aligned} \log p(\mathcal{D}) &= \mathcal{L}[q_\phi] + KL[q_\phi \| p(\mathbf{x}|\mathcal{D})] \\ \text{where, } ELB0 \;=\; \mathcal{L}[q_\phi] &= \mathbb{E}[\log \frac{p(\mathcal{D}|\mathbf{x})p(\mathbf{x})}{q_\phi(\mathbf{x})}] \\ &= \mathbb{E}[f(x)] + \mathcal{H}(q_\phi(x)) \\ &\mathcal{H} \text{ is entropy} \end{aligned}$$

Hence instead of minimizing the KL divergence we can maximize the evidence lower bound(ELBO). The ELBO can be calculated analytically or using coordinate ascent algorithm. This gives us both - the lower bound on marginal and the posterior.

2.2 Gaussian Process

Gaussian Process (GP) is defined as a distribution over functions which can be used to specify prior distribution over unknown functions $f : \mathcal{X} \subseteq \mathbb{R}^D \to \mathbb{R}$. GP is defined using the mean $\boldsymbol{\mu} : \mathcal{X} \to \mathbb{R}$ and a kernel function $\boldsymbol{\kappa} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. Each sample g from GP gives us a function $g(x)$. g value at any finite set of inputs is jointly Gaussian $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\kappa})$

2.3 Bayesian Quadrature [1]

Bayesian Quadrature is used to solve the intractable integral of the form $\langle f \rangle = \int f(\mathbf{x})\pi(\mathbf{x})d\mathbf{x}$ where f is our function and π is a known probability distribution. Idea here is to model f by a GP with a common choice of Gaussian kernel $\boldsymbol{\kappa}(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \mathcal{N}(\mathbf{x}; \mathbf{x}', \boldsymbol{\Sigma}_l)$ where $\Sigma_l = diag[l_1^2..l_D^2]$. When conditioned on the input data $X = \{\boldsymbol{x_1}, \boldsymbol{..x_n}\}$ the GP posterior [14] associated with $Y = f(X)$ is easily analytically computed with mean $\bar{f}_\Xi(\boldsymbol{x}) = f(\boldsymbol{x}; \boldsymbol{\Xi}, \boldsymbol{\psi})$ and covariance $C_{\boldsymbol{\Xi}}(\boldsymbol{x}, \boldsymbol{x'}) = C(\boldsymbol{x}, \boldsymbol{x'}; \boldsymbol{\Xi}, \boldsymbol{\psi})$ in closed form. $\boldsymbol{\Xi} = \{X, y\}; \boldsymbol{\psi}$ are the hyper-parameters.

2.4 Bayesian integration

The posterior mean and variance of the integral $\int f(\boldsymbol{x})\pi(\boldsymbol{x})d\boldsymbol{x}$ are [15]

$$\begin{aligned} \mathbb{E}_{f|\boldsymbol{\Xi}}[\langle f \rangle] &= \int \bar{f}_{\boldsymbol{\Xi}}(x)\pi(\boldsymbol{x})d\boldsymbol{x} \\ var_{f|\boldsymbol{\Xi}}[\langle f \rangle] &= \int\int C_{\boldsymbol{\Xi}}(\boldsymbol{x}, \boldsymbol{x'})\pi(\boldsymbol{x})d\boldsymbol{x}\pi(\boldsymbol{x'})d\boldsymbol{x'} \end{aligned}$$

When we use a Gaussian kernel GP and π as mixture of Gaussians we can compute the above quantities analytically.

2.5 Active Sampling

In the setting of Active learning, active sampling is a technique that tells us which data point to sample or take into consideration for calculation in such a way that our objective is achieved. Here we need samples such that our variational posterior is reduced. The acquisition function $a : \mathcal{X} \to \mathbb{R}$ is the way to find the usefulness of the datapoint and we can sample the next data point using $\boldsymbol{x}_{new} = \arg\max_{\boldsymbol{x}} a(\boldsymbol{x})$

[1] see [6]

3 VBMC

3.1 Overview

Without any doubt, Bayesian Inference is a clean and systematic way that helps us to account for model prediction and parameter uncertainty by computing various quantities like Marginal Likelihood, Posterior Distribution over parameters and Posterior Predictive Distribution over predictions. But Bayesian Inference is generally intractable analytically and MCMC and VI are used to approximate this intractability. Although used widely, a big drawback of these techniques are the prerequisites (access to the gradients or large number of model evaluations) that cannot be met by more realistic and practically used black-box probabilistic models with computationally expensive likelihoods.

A more recent alternative that has shown promising success in this scenario is building a probabilistic model-based approximation of the function of interest, for example via Gaussian Processes (GP). This approach has been extremely successful in Bayesian Optimization and in Bayesian Quadrature for the computation of intractable integrals, and also approximating Posteriors and Marginal Likelihoods but it hasn't been yet tried to simultaneously approximate both. Moreover, analysis of realistic requirements like medium dimensionality (up to 10), mild multi-modality, heavy tails, and parameters that exhibit strong correlations remain unexplored. Variational Bayesian Monte Carlo (VBMC)[2] [4] that we aim to study and evaluate combines variational inference and GP-based active-sampling Bayesian quadrature, and claims to give solutions to the above discussed issues showing indications of being a go-to tool for Bayesian Inference approximation.

3.2 Algorithm

In each iteration t,

1. **Exploration-Exploitation** : Actively sample new points that maximise the acquisition function $a(\theta)$ and evaluate log-joint $f = \log p(D|x_{new})p(x_{new})$ at each point.
2. Train GP surrogate model of the log-joint probability f ; Training set consists of the points evaluated so far.
3. Update variational posterior approximation $q_{\phi_t}(x)$ by optimizing surrogate ELBO calculated via Bayesian Quadrature.

Loop until termination criterion is met.

No sampling is performed in first iteration, so that variational posterior can adapt. In each next iteration, VBMC samples $n_{active} = 5$ points sequentially by optimising the acquisition function and performs fast rank-one updates of GP posterior.

Note we have missed some finer details which would be covered in the next section. Our aim here was to give you a far-sighted picture of what VBMC algorithm is during execution.

3.3 Variational Posterior ($q(\boldsymbol{x})$)

A mixture of K multivariate Gaussians,

$$q(\boldsymbol{x}) \equiv q_\phi(\boldsymbol{x}) = \sum_{k=1}^{K} w_k \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_k, \sigma_k^2 \boldsymbol{\Sigma})$$

where, $\boldsymbol{x} \in \mathcal{R}^D$. w_k, μ_k, σ_k are component-wise mixture weight, mean and scale respectively. Parameter $\phi \equiv (w_1, ..., w_K, \boldsymbol{\mu}_1, ..., \boldsymbol{\mu}_K, \sigma_1, ..., \sigma_K, \boldsymbol{\lambda})$. Total parameters are K(D+2) + D. $\boldsymbol{\Sigma} \equiv diag[(\lambda^{(1)})^2, ..., (\lambda^{(D)})^2]$ is shared diagonal covariance matrix common to all K elements of Gaussian mixture. K is set adaptively in each iteration. Initially, K=2.

3.4 Gaussian Process representation

$$f(\boldsymbol{x}) = \log p(D|\boldsymbol{x})p(\boldsymbol{x})$$

Approximate the expensive log joint posterior f with a GP [14] surrogate with squared exponential kernel, Gaussian likelihood with observation noise $\sigma_{obs} > 0$, and negative quadratic mean,

$$m_{NQ}(\boldsymbol{x}) = m_0 - \frac{1}{2}\sum_{i=1}^{D}\frac{(x^{(i)} - x_m^{(i)})^2}{(\omega^{(i)})^2}$$

GP hyperparameters are estimated via MCMC when there's large uncertainty about GP and then later via faster MAP estimation using gradient-based optimization when variability falls below a threshold for several iterations.

3.5 Variational Optimization

$$ELBO(\phi, f) = \int q_\phi(x) f(x) dx + \mathcal{H}[q_\phi(x)] = E_\phi[f(x)] + \mathcal{H}[q_\phi(x)]$$

Choice of $q_\phi(x)$ and other parameters of GP give analytical solution for expected log joint $\mathcal{G}[\phi|f] = E_\phi[f]$ and its gradients using Bayesian Quadrature. Entropy of variational posterior, $\mathcal{H}[q_\phi(x)]$ is estimated via Monte Carlo sampling, and its gradients via reparameterization trick [11] [12]. Using these expressions, optimize ELBO - the negative free energy via stochastic gradient descent[10].

3.6 Active Sampling

VBMC does active sampling to compute a sequence of integrals, $E_{\phi_1}[f], E_{\phi_2}[f], ..., E_{\phi_T}[f]$ in iterations 1,..., T. To evaluate $E_\phi[f]$, acquisition function "Vanilla" uncertainty sampling [3] is,

$$a_{us}(x) = s^2_\Xi(x) q_\phi(x)^2$$

where, $s^2_\Xi(x)$ is the posterior GP variance at x given current training set Ξ and q_ϕ is current variational posterior. a_{us} focuses on exploitation of regions with high probability mass and maximizes variance of integrand under current variational parameters.
To evaluate $E_{\phi_1}[f], E_{\phi_2}[f], ..., E_{\phi_T}[f]$, and focus on exploration of uncertain regions, acquisition function Prospective Uncertainty Sampling [3] is,

$$a_{pro}(x) = s^2_\Xi(x) q_\phi(x) \exp(\bar{f}_\Xi(x))$$

where $\bar{f}_\Xi(x)$ is GP posterior predictive mean. This function reduces uncertainty of variational objective both for current posterior and at prospective locations where it might go. It selects points from regions of high probability density.

3.7 Algorithmic Details

The probabilistic lower bound on ELBO that assesses the improvement in variational solution is,

$$ELCBO(\phi, f) = ELBO(\phi, f) - \beta_{LCB} SD[E_\phi[f]]$$

ELBO is estimated via Bayesian Quadrature and the second term is uncertainty in computation of expected log-joint multiplied by risk sensitivity parameter β_{LCB}.

1. Initialization and Warm-up : x_0 is the starting point from region of high posterior mass. Uniformly randomly sample $n_{init} = 10$ points in plausible box defined by PUB and PLB that are the vectors identifying region of high posterior mass in parameter space. Initialize variational posterior with K=2 components and $w_1 = w_2 = \frac{1}{2}$. Warm-up ends when ELCBO shows slow improvement for 1 to 3 consecutive iterations. Perform trimming.
2. Adaptive treatment of parameters : Add new components in each iteration (K++) if variational solution is improving i.e. ELCBO of last iteration > ELCBO of last $n_{recent} = 4$ iterations. Prune small components that don't affect ELCBO where $w_k < w_{min}$.
3. Termination : Assign reliability index $\rho(t)$ to current solution. Terminate when $\rho(t) \leq 1$ for $n_{stable} = 8$ iterations or when n_{max} function evaluations are reached.
4. Return : Estimate of mean and standard deviation of ELBO, and variational posterior.

3.8 Inference Space

VBMC works in an unconstrained space $\mathcal{R}^D$ so the constrained parameters have to be handled through non-linear remapping of input space via a shifted and rescaled logit transform, with Jacobian correction to log-joint probability density. The solution is then transformed to the original space via matched inverse transform.

VBMC, when originally proposed, didn't have the ability to deal with evaluations of noisy likelihoods. The surrogate-based models may fail in presence of even small noise in observations. Therefore, the following improvements were suggested.

3.9 Variational Whitening

The standard VBMC algorithm is axis-aligned which makes it unable to deal with highly correlated posteriors. Variational Whitening is the technique to deal with this issue and provide more accurate posterior approximation. Here, the inference space is linearly transformed $\boldsymbol{W}$ through rotation and rescaling such that the variational posterior $q(x)$ has unit diagonal covariance matrix $\boldsymbol{C}_\phi$. The whitening transform $\boldsymbol{W}$ can be found by doing SVD of $\boldsymbol{C}_\phi$. This is performed a few iterations after warm-up.

3.10 Acquisition Functions for Noisy Case

3.10.1 Noisy Prospective Uncertainty Sampling

$$a_{npro}(\theta) = \Delta s^2_\Xi(\theta) q_\phi(\theta) \exp(\bar{f}_\Xi(\theta))$$

where, $s^2_\Xi(\theta)$ is the GP posterior variance and $\bar{f}_\Xi(\theta)$ is the GP posterior latent mean at θ given current training set Ξ. a_{npro} has an additional multiplicative term that accounts for residual variance from noisy observation.

3.10.2 Expected Information Gain (EIG)

Sample points that maximize the EIG of integral $\mathcal{G}$ present in ELBO's equation above and choose the next location θ^* that maximizes mutual information $I[\mathcal{G}; y_*]$ [6].

$$a_{EIG}(\theta) = -\frac{1}{2}\log(1 - \rho^2(\theta))$$

where, $\rho(.)$ is the scalar correlation that has a closed-form solution.

3.10.3 Integrated Median / Variational Interquantile Range (IMIQR/ VIQR)

IQR [8] is the estimate of uncertainty of unnormalized posterior.

$$a_{IMIQR}(\theta) = -2\int_{\mathcal{X}} \exp(\bar{f}_\Xi(\theta')) \sinh(u s_{\Xi\cup\theta}(\theta')) d\theta'$$

This integral is intractable and so it needs to be approximated using non-surrogate based methods - MCMC and IS.

$$a_{VIQR}(\theta) = -2\int_{\mathcal{X}} q_\phi(\theta') \sinh(u s_{\Xi\cup\theta}(\theta')) d\theta'$$

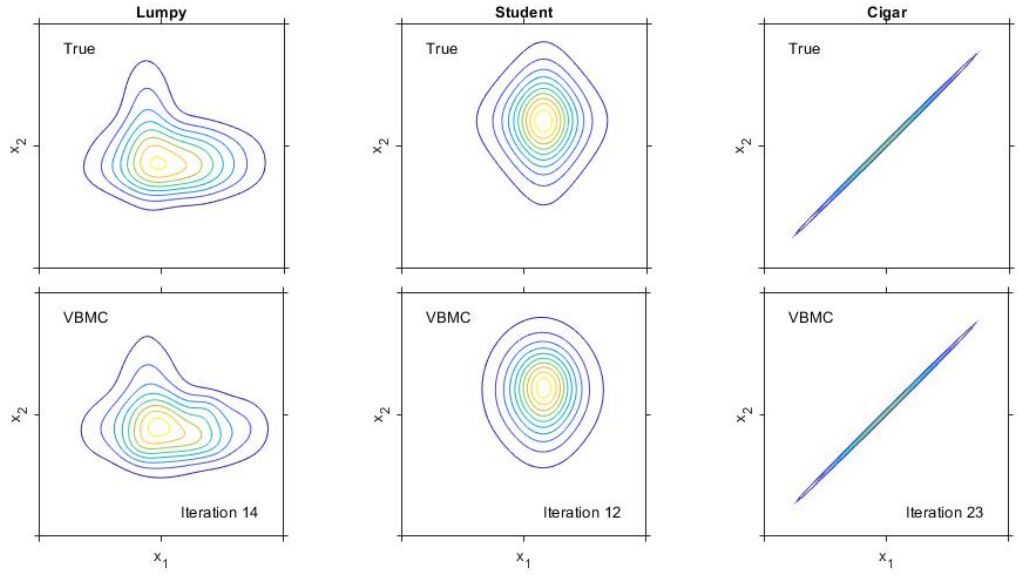
This integral can be approximated cheaply via simple Monte Carlo.

4 Experiments/Evaluations

Our aim was to reproduce the results as outlined in [4] and [2], and, if possible, use these methods on other data sets to verify and compare the results.

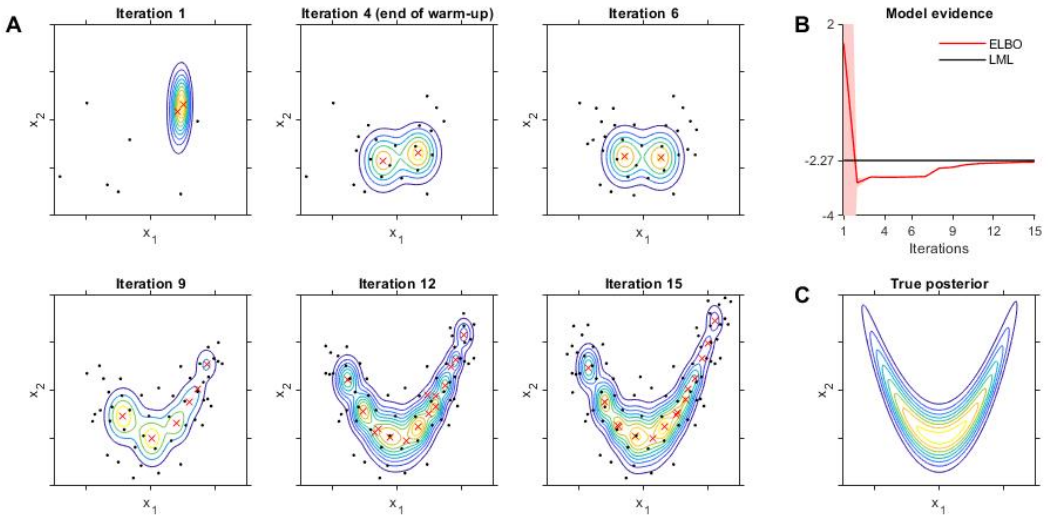
As a starting point, we verified the VBMC framework over three synthetic likelihoods, called Lumpy, Student and Cigar in [2]. We used the VBMC framework for our purpose.[2] The result for the 2-D case is presented in Figure 1. We found that this framework closely approximates the target densities after a finite (not very large) number of iterations. We further show another example

Figure 1: **Top:** Contour plots of 2D synthetic likelihoods **Bottom:** Contour plots of variational posteriors returned by VBMC



run of VBMC on another 2-D pdf known as the banana distribution, which is called as such owing to its shape, in Figure 2.

Figure 2: Example run of VBMC on 2-D Banana pdf: **A** Contour plots of the variational posterior at different iterations of the algorithm. **Red crosses indicate the centers of the variational mixture components, black dots are the training samples selected by active sampling**. **B** ELBO is plotted as a function of number of iterations. The black line is the true log marginal likelihood. **C** The true target pdf



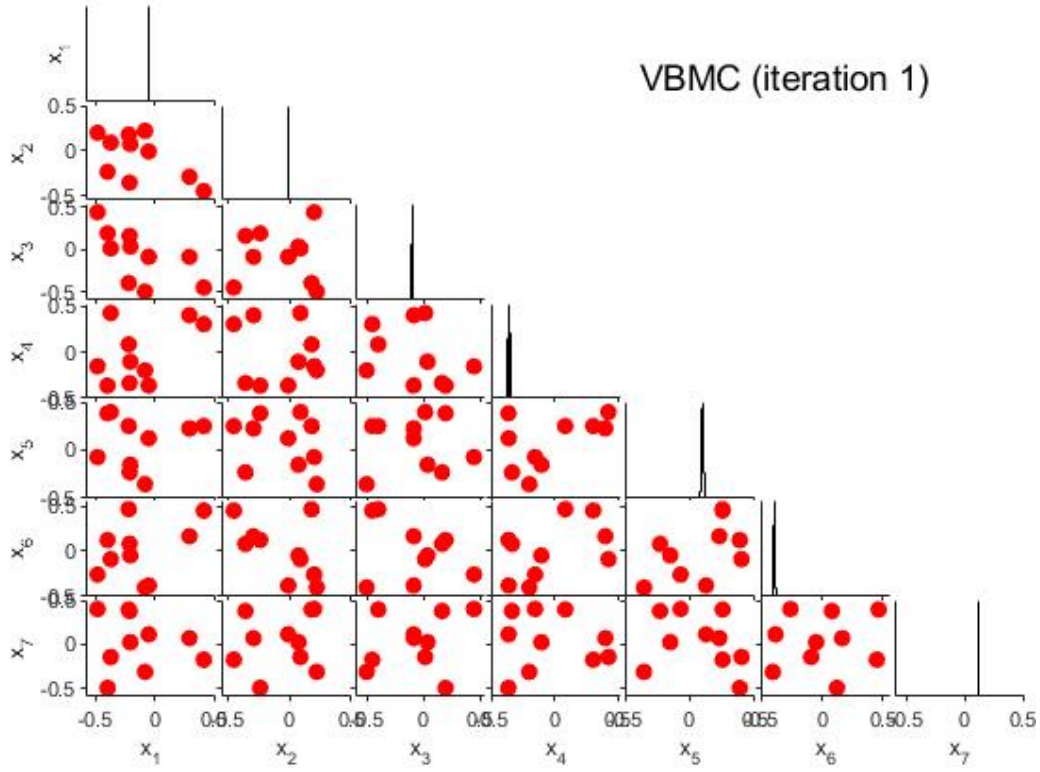
[2]Avaialble at `https://github.com/lacerbi/vbmc`

4.1 Results with Neuronal Data

Dataset for a computational model of neuronal orientation selectivity in visual cortex [7] has been considered. In [2], neuronal recordings of one V1 and another V2 cell with [7]'s neuronal model combining filtering, suppression and response non-linearity effects was fit. The author claimed that the model to be analytical albeit computationally expensive because of large data-sets. Our aim was to verify these claims by fitting this model on our own. For this goal, we tweaked the existing benchmark codes[3] made available by the author of [2] to produce pairwise 2-D contour plots.

For bench-marking purpose, some of the parameters of the original model was assumed to be fixed (the MAP values were considered) thereby reducing the dimensionality of the inference problem to only 7 free parameters. We found that VBMC, indeed approximates the, true posterior for this dataset after 90 iterations.[4] In these figures, red indicates the centers of the variational mixture components whereas black represents the chosen samples by active sampling method. In the first figure we show the initial set-up for the

Figure 3: Nueuronal Data Set: Initial Iteration



5 Suggested Improvements

The suggested improvements are inspired from the correlation that we observed between Policy Gradient based algorithms in Reinforcement Learning (RL) and VBMC. To enable the reader appreciate the same correlation, we will briefly explain the mathematical setup of Reinforcement Learning and the objective of Policy Gradient Algorithms in the following subsections. Later we will present line by line correlation that we observed between the two great algorithms. We will present the most basic "REINFORCE" algorithm. Also famous by the name "Vanilla Policy Gradient", it's a basis for many other advanced algorithms used in RL for training the agents.

[3]Refer `https://github.com/lacerbi/infbench`

[4]Refer to **??**Figure 3, Figure 4 and Figure 5

Figure 4: Nueuronal Data Set: Iteration 41

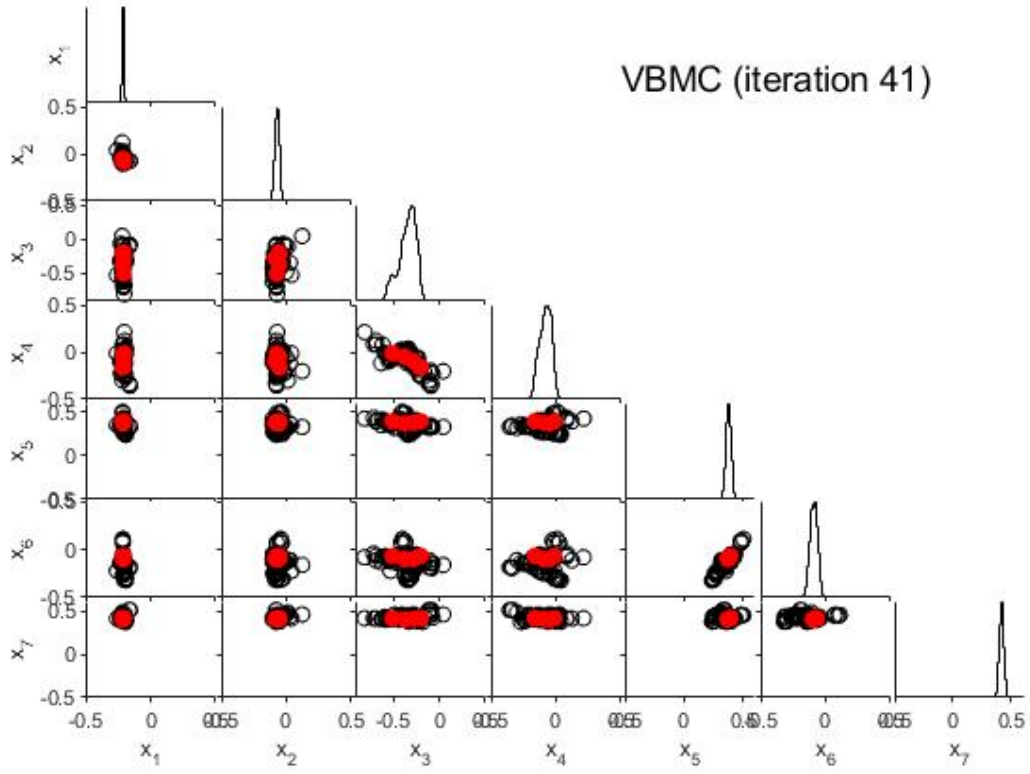
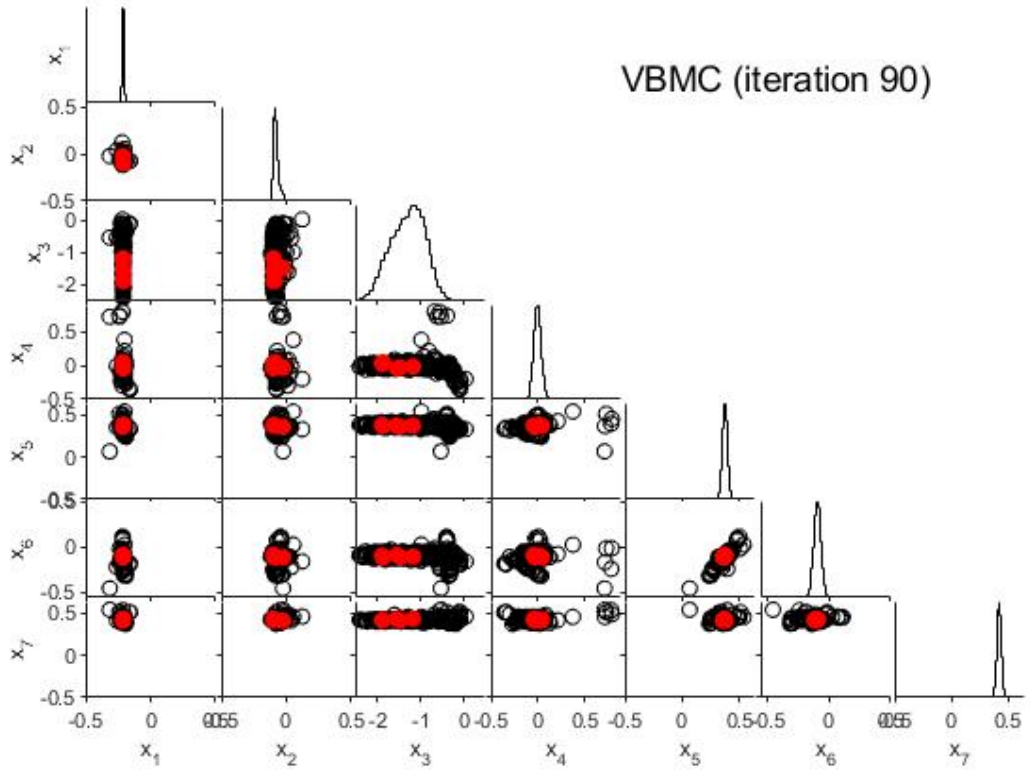


Figure 5: Nueuronal Data Set: Final iteration



5.1 Introduction to Reinforcement Learning Mathematical Setup

Consider an infinite-horizon discounted Markov Decision Process (MDP), defined by the tuple $(S, A, P, r, p_0, \gamma)$, where S is a finite/infinite set of states, A is a finite/infinite set of actions, P : S × A × S → [0,1] is the transition probability distribution, r : S x A → R is the reward function, p0 : S → [0,1] is the distribution of the initial state s0, and $\gamma \in (0, 1)$ is the discount factor. This defines the mathematical basis for achieving any objective. Let's consider episodic task/trajectory (can take $\gamma = 1$), the objective is to maximise the reward achieved in this trajectory.
Let π_θ : S × A → [0, 1] denote a stochastic policy parameterised by parameter set θ. This is the probability distribution that we want to approximate using function approximator so that the agent can earn maximum reward on an average. The above approximation problem becomes particularly hard when the state space is large enough to store. For example, in a self driving car, the state space, say, in the form of captured camera images will be very large. Now imagine approximating a policy for this agent i.e. in what situation/state, what action the agent should perform so that the average return is maximised.

The objective here is to maximise the average reward an agent gains in a trajectory:

$$\max_\theta U(\theta) = \mathbf{E}_{\tau \sim P(\tau;\theta)}[R(\tau)]$$

$$\text{where, Trajectory Reward; } R(\tau) = \sum_{t=0}^{T} r(s_t, a_t)$$

$$\text{Trajectory; } \tau = s_0, a_0, s_1 a_1, ...s_T, a_T$$

$$\text{Probability of a Trajectory; } P(\tau;\theta) = \prod_{t=0}^{T} \underbrace{P(s_{t+1}|s_t, a_t)}_{(Dynamics)} \underbrace{\pi_\theta(a_t|s_t)}_{(Policy)}$$

Note that the above objective is for an episodic task, similar kind of equation can be written for continuing task where there will be an explicit use of discount factor and N will approach to infinity. We will keep our discussion to this simple case. For more details, kindly refer to chapter 2 and 13 of [16] or [5]. The above objective's gradient with respect to θ can be further simplified and written as :

$$\nabla_\theta U(\theta) = \nabla \mathbf{E}_{\tau \sim P(\tau;\theta)}[R(\tau)] = \mathbf{E}_{P(\tau;\theta)}[\nabla log P(\tau;\theta) R(\tau)]$$

Since we don't have access to $P(\tau;\theta)$, we need to sample trajectories to compute the above expectation.

$$\nabla_\theta U(\theta) \approx 1/N * \sum_{n=1}^{N} \nabla log P(\tau^n;\theta) R(\tau^n)$$

This is further simplified and can be written in terms of π_θ.

$$\nabla_\theta U(\theta) \approx 1/N * \sum_{n=1}^{N} \sum_{t=1}^{T} \nabla_\theta log(\pi_\theta(a_t^n|s_t^n) R(\tau^n))$$

Note if we just remove the gradient with respect to theta from the above expression, our ultimate objective gets down to.

$$max_\theta\, U(\theta) \approx \mathbf{E}_{\tau \sim P(\tau;\theta)}[\sum_{t=1}^{T} log(\pi_\theta(a_t|s_t) R(\tau))]$$

We encourage the reader to correlate this objective with that of a VI objective. What do you find common between the two?. In both objectives, the final aim is to approximate a probability distribution (posterior there and policy here) with the help of some parameters. The quantity that we need to maximise is an expectation with respect to some function of the same unknown probability distribution (and eventually the parameters) and the inside expectation term is also a function of the parameters. In the next subsection, we try to show this correlation between the algorithms with the help of some colors.

5.2 Correlation between VBMC and Policy Gradient based Algorithm

1. **VBMC Objective**

$$max_{\phi} F(q(\phi)) = \mathbf{E}_{q_{\phi}(\mathbf{x})}[log(P(\mathbf{D}/\mathbf{x}) P(\mathbf{x}) - log(q_{\phi}(\mathbf{x})))]$$

2. **VBMC Rough Algorithm Sketch**

   (a) Initialise ϕ, the parameters of $q_{\phi}(x)$.

   (b) Exploration-Exploitation using some intuitive acquisition function so as to maximise the above discussed objective: actively sample from training examples.

   (c) Given these new actively sampled points, build the posterior of objective using the Bayesian Quadrature framework.

   (d) Use gradient methods to maximise this new posterior objective with respect to the parameters set ϕ

   (e) If not converged, go to step (b)

1. **Policy Gradient based RL objective**

$$max_{\theta}\ U(\theta) \approx \mathbf{E}_{\tau \sim P(\tau;\theta)}[\sum_{t=1}^{T} log(\pi_{\theta}(a_t|s_t)R(\tau))]$$

2. **Vanila Policy Gradient Algorithm Rough Sketch**

   (a) Initialise θ, the parameters of π_{θ}

   (b) Exploitation-Exploration : Sample trajectories $\{\tau_n = \{s_t^n, a_t^n\}_{t=1}^{T}\}_{n=1}^{N}$ using the current policy $\pi_{\theta}(a^t|s^t)$

   (c) Given these trajectories, build the objective U(θ) using Monte Carlo Averaging.

   (d) Use gradient methods to maximise this new objective with respect to the parameter set θ, $(\theta = \theta + \alpha * \nabla_{\theta} U(\theta))$

   (e) If not converged, go to step (b)

Now its little easy to appreciate the correlation above. Both algorithms are fulfilling their objectives by following similar kind of procedure .i.e. Exploration-Exploitation followed by updating of parameters with the help of value of the newly built objective.

5.3 Natural Gradient instead of Gradient

Referring to the correlation in the last subsection, we suggest taking a Natural Gradient instead gradient with respect to the parameters. The good use of Natural Gradient in the context of RL can be looked at from these references [9], [5] and [13].

Let's look at what Natural Gradient actually imply for VBMC. Note that the final aim of VBMC is to approximate the posterior distribution and gradient ascent step happens in the space of parameters. The parameters may change inadequately without caring much about the resulting posterior distribution. In a nutshell, gradient ascent in parameter space does not take into account the resulting distance between the old posterior distribution and the newly built one. It makes a lot more sense to take a gradient ascent step in this distribution's own space. Natural Gradient enables you to do exactly this by controlling the step size in the parameter space according to the user set maximum distance between the probability distributions of the two consecutive gradient descent updates. Kindly look at this awesome blog for a more thorough understanding of Natural Gradient Descent - [1]
Consider a parameterized distribution $q_{\phi}(\mathbf{x})$ and an objective $F(\phi)$ that depends on ϕ through $q_{\phi}(\mathbf{x})$. Here Gradient Descent update means the following:

$$\phi_{new} = \phi_{old} + d^*$$
$$\text{where}, d^* = \underset{d\ s.t.\ ||d|| \leq \epsilon}{ArgMax}\ F(\phi + d)$$

This above equation implies moving euclidean distance d in the steepest ascent direction of the above objective such that d remains in the ϵ-neighbourhood of the current value of parameter ϕ. Note that this d is taken in the parameter space.
On the other hand, the Natural Gradient Ascent interpretation is the following:

$$\begin{aligned} \phi_{new} &= \phi_{old} + d^* \\ \text{where, } d^* &= \underset{d \ s.t. \ KL(q_\phi(\mathbf{x})||q_{\phi+d}(\mathbf{x}))\leq\epsilon}{ArgMax} F(\phi + d) \end{aligned}$$

This time d distance is chosen taking into consideration the KL-divergence between the distributions before and after the update. This actually enables us to decide the step size such that $_{\phi_{old}}$ doesn't move beyond ϵ distance in the distribution space.

Luckily, the above condition can be satisfied by making a small but significant modification in the old procedure of doing an update to ϕ by finding the gradient of the maximisation objective. For detailed derivation, kindly refer to [1] and [5]. The new update equation for ϕ is as follows:

$$\begin{aligned} \phi_{new} &= \phi_{old} + \alpha_N * \mathbf{g}_N \\ \text{where, with} & \\ \mathbf{F}_I(\phi) &= \mathbf{E}_{x\sim q_{\phi_{old}}}[\nabla_\phi log q_\phi(\mathbf{x})|_{\phi_{old}} (\nabla_\phi log q_\phi(\mathbf{x})|_{\phi_{old}})^T] \\ \mathbf{g}_N &= \mathbf{F}_I^{-1}(\phi_{old}) \nabla_\phi F(\phi)|_{\phi_{old}} \\ \alpha_N &= \sqrt{\frac{2 * \epsilon}{\mathbf{g}_N^T \mathbf{F}_I^{-1} \mathbf{g}_N}} \end{aligned}$$

Note here the step size is determined by the parameter ϵ and this can be user defined.

6 Conclusion/Learnings

The VBMC inference framework to track intractable integrals considers inference as an optimization problem. It returns a non-parametric analytical approximation of the posterior distribution of the unobserved variables which can then be used to perform statistical inference and an approximate lower bound of the marginal likelihood of the observed data that can be used for model selection. Idea is that higher the marginal likelihood for a given model, better is the fit over the data by that model.

6.1 Individual Learnings:

Gagesh: Came to know about a new Bayesian Inference approximation approach. Got a glimpse of solving the open research question of finding an appropriate acquisition function in the real cutting-edge research of bayesian optimisation. Got to know about a benchmark consisting of variety of artificial likelihood models and real models(noisy and without noise). Saw real examples of how tricks like reparameterisation, jensen's inequality and logit transform are applied. Understood working ideas behind other competitive but similar approaches like WSABI and AGP. Learnt how to be patient while reading some mathematically involved papers like VBMC and WSABI. Finally enjoyed working in a group of these bright people like Shruti, Krushna and Pinaki.

Shruti: Learnt about a low-cost, sample-efficient bayesian inference framework VBMC. Choose an application and try comparing MCMC with VBMC in terms of efficiency, accuracy and computationally.

Krushna: Came to know about awesome way of finding the definite integral using the Bayesian quadrature method.
Also the key idea of interrelating the things (Variatinal Inference(VI), surrogate function(GP), Bayesian quadrature, Active sampling)

Pinaki: I learnt of Bayesian inference for learning from data and the role of approximation methods in this paradigm. As a concrete example, I learnt of the algorithmic concepts of the information-theoretic variational Bayes framework as applied to Monte-Carlo methodology and its role in approximating posterior distributions. Also learnt a bit of Bayesian optimisation and the role that

Gaussian Processes and acquisition function based active sampling methods play in it. From a practical perspective, I became familiar with the usage of MATLAB for the purpose of statistical inference. In summary, I learnt a little bit of the "big picture" behind these methodologies. Also I would like to express the joy of working together with highly enthusiastic people such as Gagesh, Krushna and Shruti.

Acknowledgments

We would like to thank our course instructor - Dr. Piyush Rai, CSE Dept. for providing us an opportunity to work on this project and learn new things. We also enjoyed working together as a team and tried to encourage and guide whenever one of us needed.

References

[1] Blog : https://wiseodd.github.io/techblog/2018/03/14/natural-gradient.

[2] Luigi Acerbi. Variational bayesian monte carlo, 2018.

[3] Luigi Acerbi. An exploration of acquisition and mean functions in variational bayesian monte carlo. volume 96 of *Proceedings of Machine Learning Research*. PMLR, 2019.

[4] Luigi Acerbi. Variational bayesian monte carlo with noisy likelihoods, 2020.

[5] Katerina Fragkiadaki. Lecture 15, 14, reinforce, actor-critic methods (cont.), natural pg.

[6] Alexandra Gessner, Javier Gonzalez, and Maren Mahsereci. Active multi-information source bayesian quadrature, 2021.

[7] Robbe L.T. Goris, Eero P. Simoncelli, and J. Anthony Movshon. Origin and function of tuning diversity in macaque visual cortex. *Neuron*, 88(4):819–831, 2015.

[8] Marko Järvenpää, Michael Gutmann, Aki Vehtari, and Pekka Marttinen. Parallel gaussian process surrogate bayesian inference with noisy likelihood evaluations, 2020.

[9] S. M. Kakade. A natural policy gradient, 2002.

[10] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[11] Diederik P. Kingma and M. Welling. Auto-encoding variational bayes. *CoRR*, 2014.

[12] Andrew C. Miller, Nicholas Foti, and Ryan P. Adams. Variational boosting: Iteratively refining posterior approximations, 2017.

[13] Jan Peters and Stefan Schaal. Reinforcement learning of motor skills with policy gradients. *Neural Networks*, 21(4):682–697, 2008. Robotics and Neuroscience.

[14] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[15] Carl Edward Rasmussen and Zoubin Ghahramani. Bayesian monte carlo. *Advances in neural information processing systems*, pages 505–512, 2003.

[16] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, second edition, 2018.