Appendix

A Brief review of Gaussian processes

Gaussian Processes (GPs, Rasmussen & Williams, 2006), as a popular example of Bayesian nonparametrics, provides a principled probabilistic framework for non-parametric Bayesian inference over functions. This is achieved by imposing rich and flexible nonparametric priors over functions of interest. As flexible and interpretable function approximators, their Bayesian nature also enables GPs to provide valuable information of uncertainties regarding predictions for intelligence systems, all wrapped up in a single, *exact* closed form solution of posterior inference.

We briefly introduce GPs for regression. Assume that we have a set of observational data $\{(\mathbf{x}_n, y_n\}_{n=1}^N)$, where $\mathbf{x}_n$ is the D dimensional input of n th data point, and y_n is the corresponding scalar target of the regression problem. A Gaussian Process model assumes that y_n is generated according the following procedure: firstly a function $f(\cdot)$ is drawn from a Gaussian Process $\mathcal{GP}(m, k)$ (to be defined later). Then for each input data $\mathbf{x}_n$, the corresponding y_n is then drawn according to:

$$y_n = f(\mathbf{x}_n) + \epsilon_n, \quad \epsilon \sim \mathcal{N}(0, \sigma^2), \quad n = 1, \cdots, N$$

A Gaussian Process is a nonparametric distribution defined over the space of functions, such that:

Definition 2 (Gaussian Processes)**.** *A Gaussian process (GP) is a collection of random variables, any finite number of which have a joint Gaussian distributions. A Gaussian Process is fully specified by its mean function $m(\cdot) : \mathbb{R}^D \mapsto \mathbb{R}$ and covariance function $\mathcal{K}(\cdot, \cdot) : (\mathbb{R}^D, \mathbb{R}^D) \mapsto \mathbb{R}$, such that any finite collection of function values $\mathbf{f}$ are distributed as Gaussian distribution $\mathcal{N}(\mathbf{f}; \mathbf{m}, \mathbf{K_{ff}})$, where $(\mathbf{m})_n = m(\mathbf{x}_n)$, $(\mathbf{K_{ff}})_{n,n'} = \mathcal{K}(\mathbf{x}_n, \mathbf{x}_{n'})$.*

Now, given a set of observational data $\{(\mathbf{x}_n, y_n)\}_{n=1}^N$, we are able to perform probabilistic inference and assign posterior probabilities over all plausible functions that might have generated the data. Under the setting of regression, given a new test point input data $\mathbf{x}_*$, we are interested in posterior distributions over f_*. Fortunately, this posterior distribution of interest admits a closed form solution $f_* \sim \mathcal{N}(\mu_*, \Sigma_*)$:

$$\mu_* = \mathbf{m} + K_{\mathbf{x}_*\mathbf{f}}(\mathbf{K_{ff}} + \sigma^2\mathbf{I})^{-1}(\mathbf{y} - \mathbf{m}) \tag{A.1}$$

$$\Sigma_* = K_{\mathbf{x}_*\mathbf{x}_*} - K_{\mathbf{x}_*\mathbf{f}}(\mathbf{K_{ff}} + \sigma^2\mathbf{I})^{-1}K_{\mathbf{f}\mathbf{x}_*} \tag{A.2}$$

In our notation, $(\mathbf{y})_n = y_n$, $(K_{\mathbf{x}_*\mathbf{f}})_n = \mathcal{K}(\mathbf{x}_*, \mathbf{x}_n)$, and $K_{\mathbf{x}_*\mathbf{x}_*} = \mathcal{K}(\mathbf{x}_*, \mathbf{x}_*)$. Although the Gaussian Process regression framework is theoretically very elegant, in practice its computational burden is prohibitive for large datasets since the matrix inversion $(\mathbf{K_{ff}} + \sigma^2\mathbf{I})^{-1}$ takes $\mathcal{O}(N^3)$ time due to Cholesky decomposition. Once matrix inversion is done, predictions in test time can be made in $\mathcal{O}(N)$ for posterior mean μ_* and $\mathcal{O}(N^2)$ for posterior uncertainty Σ_*, respectively.

Despite the success and popularity of GPs (and other Bayesian non-parametric methods) in the past decades, their $\mathcal{O}(N^3)$ computation and $\mathcal{O}(N^2)$ storage complexities make it impractical to apply GPs to large-scale datasets. Therefore, people often resort to complicated approximate methods, e.g. see Seeger et al. (2003); Quiñonero-Candela & Rasmussen (2005); Snelson & Ghahramani (2006); Titsias (2009); Hensman et al. (2013); Bui et al. (2016b); Bui & Turner (2014); Saatçi (2012); Cunningham et al. (2008); Turner & Sahani (2010).

Another critical issue to be addressed is the representational power of GP kernels. It has been argued that local kernels commonly used for nonlinear regressions are not able to obtain hierarchical representations for high dimensional data (Bengio et al., 2005), which limits the usefulness of Bayesian non-parametric models for complicated tasks. A number of solutions were proposed, including deep GPs (Damianou & Lawrence, 2013; Cutajar et al., 2016; Bui et al., 2016a), the design of expressive kernels (van der Wilk et al., 2017; Duvenaud et al., 2013; Tobar et al., 2015), and the hybrid model with features from deep neural nets as the input of a GP (Hinton & Salakhutdinov, 2008; Wilson et al., 2016). However, the first two approaches still struggle to model complex high dimensional data such as texts and images; and in the third approach, the merits of fully Bayesian approach has been discarded.

B Brief review of variational inference, and the black-box α-energy

We give a brief review of modern variational techniques, including standard variational inference and black-box α-divergence minimization (BB-α), on which our methodology is heavily based. Considers the problem of finding the posterior distribution, $p(\theta|\mathcal{D}, \tau)$, $\mathcal{D} = \{\mathbf{x}_n\}_{n=1}^N$ under the model likelihood $p(\mathbf{x}|\theta, \tau)$ and a prior distribution $p_0(\theta)$:

$$p(\theta|\mathcal{D}, \tau) \propto \frac{1}{Z}p_0(\theta)\prod_n p(\mathbf{x}_n|\theta, \tau).$$

Here τ is the hyper-parameter of the model, which will be optimized by (approximate) maximum likelihood.

Variational inference (VI, Jordan et al., 1999) converts the above inference problem into an optimization problem, by first proposing a class of approximate posterior $q(\theta)$, and then minimize the KL-divergence from the approximate posterior to the true posterior $\mathcal{D}_{\text{KL}}[q||p]$. Equivalently, VI

optimizes the following variational free energy,

$$\mathcal{F}_{\text{VFE}} = \log p(\mathcal{D}|\tau) - \mathcal{D}_{\text{KL}}[q(\theta)||p(\theta|\mathcal{D},\tau)] = \mathbb{E}_{q(\theta)}\left[\log \frac{p(\mathcal{D},\theta|\tau)}{q(\theta)}\right].$$

Built upon the idea of VI, BB-α is a modern black-box variational inference framework that unifies and interpolates between VI (Jordan et al., 1999) and expectation propagation (EP)-like algorithms (Minka, 2001; Li et al., 2015). BB-α performs approximate inference by minimizing the following α-divergence (Zhu & Rohwer, 1995) $\mathrm{D}_\alpha[p||q]$:

$$\mathrm{D}_\alpha[p||q] = \frac{1}{\alpha(1-\alpha)}\left(1 - \int p(\theta)^\alpha q(\theta)^{1-\alpha} d\theta\right).$$

α-divergence is a generic class of divergences that includes the inclusive KL-divergence (α=1, corresponds to EP), Hellinger distance (α=0.5), and the exclusive KL-divergence (α = 0, corresponds to VI) as special cases.

Traditionally, power EP (Minka, 2004) optimizes an α-divergence locally with exponential family approximation $q(\theta) \propto \frac{1}{Z} p_0(\theta) \prod_n \tilde{f}_n(\theta)$,$\tilde{f}_n(\theta) \propto \exp\left[\lambda_n^T \phi(\theta)\right]$ via message passing. It converges to a fixed point of the so called *power EP energy*:

$$\mathcal{L}_{\text{PEP}}(\lambda_0, \{\lambda_n\}) = \log Z(\lambda_0) + (\frac{N}{\alpha} - 1)\log Z(\lambda_q) - \frac{1}{\alpha}\sum_{n=1}^{N} \log \int p(\mathbf{x}_n|\theta,\tau)^\alpha \exp\left[(\lambda_q - \alpha\lambda_n)^T \phi(\theta)\right] d\theta,$$

where $\lambda_q = \lambda_0 + \sum_{n=1}^N \lambda_n$ is the natural parameter of $q(\theta)$. On the contrary, BB-α directly optimizes $\mathcal{L}_{\text{PEP}}$ with tied factors $\tilde{f}_n = \tilde{f}$ to avoid prohibitive local factor updates and storage on the whole dataset. This means $\lambda_n = \lambda$ for all n and $\lambda_q = \lambda_0 + N\lambda$. Therefore instead of parameterizing each factors, one can directly parameterize $q(\theta)$ and replace all the local factors in the power-EP energy function by $\tilde{f}(\theta) \propto (q(\theta)/p_0(\theta))^{1/N}$. After re-arranging terms, this gives the BB-α energy:

$$\mathcal{L}_\alpha(q) = -\frac{1}{\alpha}\sum_n \log \mathbb{E}_q\left[\left(\frac{f_n(\theta)p_0(\theta)^{\frac{1}{N}}}{q(\theta)^{\frac{1}{N}}}\right)^\alpha\right].$$

which can be further approximated by the following if the dataset is large (Li & Gal, 2017):

$$\mathcal{L}_\alpha(q) = \mathcal{D}_{\text{KL}}[q||p_0] - \frac{1}{\alpha}\sum_n \log \mathbb{E}_q\left[p(\mathbf{x}_n|\theta,\tau)^\alpha\right].$$

The optimization of $\mathcal{L}_\alpha(q)$ could be performed in a black-box manner with reparameterization trick (Kingma & Welling, 2013), Monte Carlo (MC) approximation and mini-batch training. Empirically, it has been shown that BB-α with $\alpha \neq 0$ can return significantly better uncertainty estimation than VI, and has been applied successfully in different scenarios (Li & Gal, 2017; Depeweg et al., 2016). From hyper-parameter learning (i.e., τ in $p(\mathbf{x}_n|\theta,\tau)$), it is shown in Li & Turner (2016) that the BB-α energy $\mathcal{L}_\alpha(q)$ constitutes a better estimation of log marginal likelihood, $\log p(\mathcal{D}|\tau)$ when compared with the variational free energy. Therefore, for both inference and learning, BB-α energy is extensively used in this paper.

C Derivations

C.1 Proof of Proposition 1 (finite dimensional case)

Proposition 1. *If* $\mathbf{z}$ *is a finite dimensional random variable, then there exists a unique stochastic process, with finite marginals that are distributed exactly according to Definition 1.*

Proof Generally, consider the following noisy IP model:

$$f(\cdot) \sim \mathcal{IP}(g_\theta(\cdot,\cdot), p_{\mathbf{z}}),\ \ y_n = f(\mathbf{x}_n) + \epsilon_n,\ \epsilon_n \sim \mathcal{N}(0,\sigma^2).$$

For any finite collection of random variables $y_{1:n} = \{y_1, ..., y_n\}$, $\forall n$ we denote the induced distribution as $p_{1:n}(y_{1:n})$. Note that $p_{1:n}(y_{1:n})$ can be represented as $\mathbb{E}_{p(\mathbf{z})}[\prod_{i=1}^n \mathcal{N}(y_i; g(\mathbf{x}_i,\mathbf{z}),\sigma^2)]$. Therefore for any $m < n$, we have

$$\begin{aligned}
&\int p_{1:n}(y_{1:n}) dy_{m+1:n} \\
&= \int\int \prod_{i=1}^n \mathcal{N}(y_i; g(\mathbf{x}_i,\mathbf{z}),\sigma^2) p(\mathbf{z}) d\mathbf{z} dy_{m+1:n} \\
&= \int\int \prod_{i=1}^n \mathcal{N}(y_i; g(\mathbf{x}_i,\mathbf{z}),\sigma^2) p(\mathbf{z}) dy_{m+1:n} d\mathbf{z} \\
&= \int \prod_{i=1}^m \mathcal{N}(y_i; g(\mathbf{x}_i,\mathbf{z}),\sigma^2) p(\mathbf{z}) d\mathbf{z} = p_{1:m}(y_{1:m}).
\end{aligned}$$

Note that the swap of the order of integration relies on that the integral is finite, which is true when the prior $p(\mathbf{z})$ is proper. Therefore, the marginal consistency condition of Kolmogorov extension theorem is satisfied. Similarly, the permutation consistency condition of Kolmogorov extension theorem can be proved as follows: assume $\pi(1:n) = \{\pi(1), ..., \pi(n)\}$ is a permutation of the indices $1:n$, then

$$\begin{aligned}
&p_{\pi(1:n)}(y_{\pi(1:n)}) \\
&= \int \prod_{i=1}^n \mathcal{N}(y_{\pi(i)}; g(\mathbf{x}_{\pi(i)},\mathbf{z}),\sigma^2) p(\mathbf{z}) d\mathbf{z} \\
&= \int \prod_{i=1}^n \mathcal{N}(y_i; g(\mathbf{x}_i,\mathbf{z}),\sigma^2) p(\mathbf{z}) d\mathbf{z} = p_{1:n}(y_{1:n}).
\end{aligned}$$

Therefore, by Kolmogorov extension theorem, there exists a unique stochastic process, with finite marginals that are distributed exactly according to Definition 1.

□

C.2 Proof of Proposition 2 (infinite dimensional case)

Proposition 2. *Let $z(\cdot) \sim \mathcal{SP}(0, C)$ be a centered continuous stochastic process on $\mathcal{L}^2(\mathbb{R}^d)$ with covariance function $C(\cdot,\cdot)$. Then the operator $g(\mathbf{x}, z) = O(z)(\mathbf{x}) := h(\int \sum_{l=0}^{M} K_l(\mathbf{x}, \mathbf{x}')z(\mathbf{x}')d\mathbf{x}')$, $0 < M < +\infty$ defines a stochastic process if $K_l \in \mathcal{L}^2(\mathbb{R}^d \times \mathbb{R}^d)$, h is a Borel measurable, bijective function in $\mathbb{R}$ and there exist $0 \leq A < +\infty$ such that $|h(x)| \leq A|x|$ for $\forall x \in \mathbb{R}$.*

Proof Since $\mathcal{L}^2(\mathbb{R}^d)$ is closed under finite summation, without loss of generality, we consider the case of $M = 1$ where $O(z)(\mathbf{x}) = h(\int K(\mathbf{x}, \mathbf{x}')z(\mathbf{x}')d\mathbf{x}')$. According to Karhunen-Loeve expansion (K-L expansion) theorem (Loeve, 1977), the stochastic process z can be expanded as the *stochastic* infinite series,

$$z(\mathbf{x}) = \sum_i^{\infty} Z_i \phi_i(\mathbf{x}), \quad \sum_i^{\infty} \lambda_i < +\infty.$$

Where Z_i are zero-mean, uncorrelated random variables with variance λ_i. Here $\{\phi_i\}_{i=1}^{\infty}$ is an orthonormal basis of $\mathcal{L}^2(\mathbb{R}^d)$ that are also eigen functions of the operator $O_C(z)$ defined by $O_C(z)(\mathbf{x}) = \int C(\mathbf{x}, \mathbf{x}')z(\mathbf{x}')d\mathbf{x}'$. The variance λ_i of Z_i is the corresponding eigen value of $\phi_i(\mathbf{x})$.

Apply the linear operator

$$O_K(z)(\mathbf{x}) = \int K(\mathbf{x}, \mathbf{x}')z(\mathbf{x}')d\mathbf{x}'$$

on this K-L expansion of z, we have:

$$\begin{aligned} O_K(z)(\mathbf{x}) &= \int K(\mathbf{x}, \mathbf{x}')z(\mathbf{x}')d\mathbf{x}' \\ &= \int K(\mathbf{x}, \mathbf{x}') \sum_i^{\infty} Z_i \phi_i(\mathbf{x}')d\mathbf{x}' \\ &= \sum_i^{\infty} Z_i \int K(\mathbf{x}, \mathbf{x}')\phi_i(\mathbf{x}')d\mathbf{x}', \end{aligned} \tag{C.1}$$

where the exchange of summation and integral is guaranteed by Fubini's theorem. Therefore, the functions $\{\int_{\mathbf{x}} K(\mathbf{x}, \mathbf{x}')\phi_i(\mathbf{x}')d\mathbf{x}'\}_{i=1}^{\infty}$ forms a new basis of $\mathcal{L}^2(\mathbb{R}^d)$. To show that the stochastic series C.1 converge:

$$\begin{aligned} &|| \sum_i^{\infty} Z_i \int K(\mathbf{x}, \mathbf{x}')\phi_i(\mathbf{x}')d\mathbf{x}||_{\mathcal{L}^2}^2 \\ &\leq ||O_K||^2 || \sum_i^{\infty} Z_i \phi_i(\mathbf{x}')||_{\mathcal{L}^2}^2 \\ &= ||O_K||^2 \sum_i^{\infty} ||Z_i||_2^2, \end{aligned}$$

where the operator norm is defined by

$$||O_K|| := \inf\{c \geq 0 : ||O_k(f)||_{\mathcal{L}^2} \leq c||f||_{\mathcal{L}^2}, \forall f \in \mathcal{L}^2(\mathbb{R}^d)\}.$$

This is a well defined norm since O_K is a bounded operator ($K \in \mathcal{L}^2(\mathbb{R}^d \times \mathbb{R}^d)$). The last equality follows from the orthonormality of $\{\phi_i\}$. The condition $\sum_i^{\infty} \lambda_i < \infty$ further guarantees that $\sum_i^{\infty} ||Z_i||^2$ converges almost surely. Therefore, the random series (C.1) converges in $\mathcal{L}^2(\mathbb{R}^d)$ a.s..

Finally we consider the nonlinear mapping $h(\cdot)$. With $h(\cdot)$ a Borel measurable function satisfying the condition that there exist $0 \leq A < +\infty$ such that $|h(x)| \leq A|x|$ for $\forall x \in \mathbb{R}$, it follows that $h \circ O_K(z) \in \mathcal{L}^2(\mathbb{R}^d)$. In summary, $g = O_k(z) = h \circ O_K(z)$ defines a well-defined stochastic process on $\mathcal{L}^2(\mathbb{R}^d)$.

□

Despite of its simple form, the operator $g = h \circ O_K(z)$ is in fact the building blocks for many flexible transformations over functions (Guss, 2016; Williams, 1997; Stinchcombe, 1999; Le Roux & Bengio, 2007; Globerson & Livni, 2016) . Recently Guss (2016) proposed the so called Deep Function Machines (DFMs) that possess universal approximation ability to nonlinear operators:

Definition 3 (Deep Function Machines (Guss, 2016))**.** *A deep function machine $g = O_{DFM}(z, S)$ is a computational skeleton S indexed by I with the following properties:*

- *Every vertex in S is a Hilbert space $\mathbb{H}_l$ where $l \in I$.*
- *If nodes $l \in A \subset I$ feed into l' then the activation on l' is denoted $y^l \in \mathbb{H}_l$ and is defined as*

$$y^{l'} = h \circ (\sum_{l \in A} O_{K_l}(y^l))$$

Therefore, by Proposition 2, we have proved:
Corollary 2 *Let $z(\cdot) \sim \mathcal{SP}(0, C)$ be a centered continuous stochastic process on $\mathbb{H} = \mathcal{L}^2(\mathbb{R}^d)$. Then the Deep function machine operator $g = O_{DFM}(z, S)$ defines a well-defined stochastic process on $\mathbb{H}$.*

C.3 Inverse Wishart process as a prior for kernel functions

Definition 4 (Inverse Wishart processes (Shah et al., 2014))**.** *Let Σ be random function $\Sigma(\cdot,\cdot) : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. A stochastic process defined on such functions is called the inverse*

Wishart process on $\mathcal{X}$ with parameter ν and base function $\Psi : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, if for any finite collection of input data $\mathbf{X} = \{\mathbf{x}_s\}_{1 \le s \le N_s}$, the corresponding matrix-valued evaluation $\Sigma(\mathbf{X}, \mathbf{X}) \in \Pi(N_s)$ is distributed according to an inverse Wishart distribution $\Sigma(\mathbf{X}, \mathbf{X}) \sim IW_S(\nu, \Psi(\mathbf{X}, \mathbf{X}))$. We denote $\Sigma \sim \mathcal{IWP}(v, \Psi(\cdot, \cdot))$.

Consider the problem in Section 3.1 of minimizing the objective

$$\mathcal{U}(m, \mathcal{K}) = \mathcal{D}_{\text{KL}}[p(\mathbf{f}, \mathbf{y}|\mathbf{X}, \theta)||q_{\mathcal{GP}}(\mathbf{f}, \mathbf{y}|\mathbf{X}, m(\cdot), \mathcal{K}(\cdot, \cdot))]$$

Since we use $q(\mathbf{y}|\mathbf{f}) = p(\mathbf{y}|\mathbf{f})$, this reduces $\mathcal{U}(m, \mathcal{K})$ to $\mathrm{D}_{KL}[p(\mathbf{f}|\mathbf{X}, \theta)||q_{\mathcal{GP}}(\mathbf{f}|\mathbf{X}, m, \mathcal{K})]$. In order to obtain optimal solution wrt. $\mathcal{U}(m, \mathcal{K})$, it sufficies to draw S fantasy functions (each sample is a random function $f_s(\cdot)$) from the prior distribution $p(\mathbf{f}|\mathbf{X}, \theta)$, and perform moment matching, which gives exactly the MLE solution, i.e., empirical mean and covariance functions

$$m^{\star}_{\text{MLE}}(\mathbf{x}) = \sum_s \frac{1}{S} f_s(\mathbf{x}), \tag{C.2}$$

$$\mathcal{K}^{\star}_{\text{MLE}}(\mathbf{x}_1, \mathbf{x}_2) = \frac{1}{S} \sum_s \Delta_s(\mathbf{x}_1)\Delta_s(\mathbf{x}_2), \tag{C.3}$$

$$\Delta_s(\mathbf{x}) = f_s(\mathbf{x}) - m^{\star}_{\text{MLE}}(\mathbf{x}). \tag{C.4}$$

In practice, in order to gain computational advantage, the number of fantasy functions S is often small, therefore we further put an inverse wishart process prior over the GP covariance function, i.e. $\mathcal{K}(\cdot, \cdot) \sim \mathcal{IWP}(\nu, \Psi)$. By doing so, we are able to give MAP estimation instead of MLE estimation. Since inverse Wishart distribution is conjugate to multivariate Gaussian distribution, the maximum a posteriori (MAP) solution is given by

$$\mathcal{K}^{\star}_{\text{MAP}}(\mathbf{x}_1, \mathbf{x}_2) = \frac{1}{\nu + S + N + 1}\{\sum_s \Delta_s(\mathbf{x}_1)\Delta_s(\mathbf{x}_2) + \Psi(\mathbf{x}_1, \mathbf{x}_2)\}. \tag{C.5}$$

Where N is the number of data points in the training set $\mathbf{X}$ where $m(\cdot)$ and $\mathcal{K}(\cdot, \cdot)$ are evaluated. Alternatively, one could also use the posterior mean Estimator (PM) that minimizes posterior expected squared loss:

$$\mathcal{K}^{\star}_{\text{PM}}(\mathbf{x}_1, \mathbf{x}_2) = \frac{1}{\nu + S - N - 1}\{\sum_s \Delta_s(\mathbf{x}_1)\Delta_s(\mathbf{x}_2) + \Psi(\mathbf{x}_1, \mathbf{x}_2)\}. \tag{C.6}$$

In the implementation of this paper, we choose $\mathcal{K}_{\text{PM}}$ estimator with $\nu = N$ and $\Psi(\mathbf{x}_1, \mathbf{x}_2) = \psi\delta(\mathbf{x}_1, \mathbf{x}_2)$. The hyper parameter ψ is trained using fast grid search using the same procedure for the noise variance parameter, as detailed in Appendix F.

C.4 Derivation of the upper bound $\mathcal{U}(m, \mathcal{K})$or sleep phase

Applying the chaine rule of KL-divregence, we have

$$\begin{aligned}\mathcal{J}(m, \mathcal{K}) =& \mathcal{D}_{\text{KL}}[p(\mathbf{f}|\mathbf{X}, \mathbf{y}, \theta)||q_{\mathcal{GP}}(\mathbf{f}|\mathbf{X}, \mathbf{y}, m(\cdot), \mathcal{K}(\cdot, \cdot))] \\ =& \mathcal{D}_{\text{KL}}[p(\mathbf{f}, \mathbf{y}|\mathbf{X}, \theta)||q_{\mathcal{GP}}(\mathbf{f}, \mathbf{y}|\mathbf{X}, m(\cdot), \mathcal{K}(\cdot, \cdot))] \\ & - \mathcal{D}_{\text{KL}}[p(\mathbf{y}|\mathbf{X}, \theta)||q_{\mathcal{GP}}(\mathbf{y}|\mathbf{X}, m(\cdot), \mathcal{K}(\cdot, \cdot))] \\ =& \mathcal{U}(m, \mathcal{K}) - \mathcal{D}_{\text{KL}}[p(\mathbf{y}|\mathbf{X}, \theta)||q_{\mathcal{GP}}(\mathbf{y}|\mathbf{X}, m(\cdot), \mathcal{K}(\cdot, \cdot))].\end{aligned}$$

Therefore, by the non-negative property of KL divergence, we have $\mathcal{J}(m, \mathcal{K}) < \mathcal{U}(m, \mathcal{K})$. Since we select $q(\mathbf{y}|\mathbf{f}) = p(\mathbf{y}|\mathbf{f})$, the optimal solution of $\mathcal{U}(m, \mathcal{K})$ also minimizes $\mathcal{D}_{\text{KL}}(p(\mathbf{y}|\mathbf{X}, \theta)||q_{\mathcal{GP}}(\mathbf{y}|\mathbf{X}, m(\cdot), \mathcal{K}(\cdot, \cdot)))$. Therefore not only the upper bound $\mathcal{U}$ is optimized in sleep phase, the gap $-\mathcal{D}_{\text{KL}}(p(\mathbf{y}|\mathbf{X}, \theta)||q_{\mathcal{GP}}(\mathbf{y}|\mathbf{X}, m(\cdot), \mathcal{K}(\cdot, \cdot)))$ is also decreased when the mean and covariance functions are optimized.

C.5 Empirical Bayes approximation for VIP with a hierarchical prior on θ

The implicit processes (such as Bayesian neural networks and GPs) could be sensitive to the choice of the model parameters (that is, parameters θ of the prior). To make our variational implicit process more robust we further present an empirical Bayesian treatment, by introducing an extra hierarchical prior distribution $p(\theta)$ on the prior parameters θ, and fitting a variational approximation $q(\theta)$ to the posterior. Sleep phase updates remain the same when conditioned on a given configuration of θ. The α-energy term in wake phase learning becomes

$$\begin{aligned}&\log q_{\mathcal{GP}}(\mathbf{y}|\mathbf{X}) \\ &= \log \int_{\theta} q_{\mathcal{GP}}(\mathbf{y}|\mathbf{X}, \theta)p(\theta)d\theta \approx \mathcal{L}^{\alpha}_{\mathcal{GP}}(q(\mathbf{a}), q(\theta)), \\ &\mathcal{L}^{\alpha}_{\mathcal{GP}}(q(\mathbf{a}), q(\theta)) \\ &= \frac{1}{\alpha}\sum_n^N \log \mathbb{E}_{q(\mathbf{a})q(\theta)}\left[q^{\star}(y_n|\mathbf{x}_n, \mathbf{a}, \theta)^{\alpha}\right] \\ &- \mathrm{D}_{\text{KL}}[q(\mathbf{a})||p(\mathbf{a})] - \mathrm{D}_{\text{KL}}[q(\theta)||p(\theta)].\end{aligned} \tag{C.7}$$

Compared with the approximate MLE method, the only extra term needs to be estimated is $-\mathcal{D}_{\text{KL}}[q(\theta)||p(\theta)]$. Note that, introducing $q(\theta)$ will double the number of parameters. In the case of Bayesian NN as an IP, where θ contains means and variances for weight priors, then a simple Gaussian $q(\theta)$ will need two sets of means and variances variational parameters (i.e., posterior means of means, posterior variances of means,posterior means of variances, posterior variances of variances). Therefore, to make the representation compact, we choose $q(\theta)$ to be a Dirac-delta function $\delta(\theta_q)$, which results in an *empirical Bayesian solution.*

Another possible alternative approach is, instead of explicitly specifying the form and hyperparameters for $p(\theta)$,we

can notice that from standard variational lower bound

$$\log q_{\mathcal{GP}}(\mathbf{y}|\mathbf{X}) \approx \mathbb{E}_{q(\theta)}[\log q_{\mathcal{GP}}(\mathbf{y}|\mathbf{X},\theta)] - \mathcal{D}_{\mathrm{KL}}[q(\theta)||p(\theta)].$$

Then $\mathcal{D}_{\mathrm{KL}}[q(\theta)||p(\theta)]$ can be approximated by

$$\begin{aligned}-\mathcal{D}_{\mathrm{KL}}[q(\theta)||p(\theta)] &\approx -\mathbb{E}_{q(\theta)}[\log q_{\mathcal{GP}}(\mathbf{y}|\mathbf{X},\theta)] + \text{constant} \\ &= -\log q_{\mathcal{GP}}(\mathbf{y}|\mathbf{X},\theta_q) + \text{constant}\end{aligned}$$

Therefore, we can use $-\log q_{\mathcal{GP}}(\mathbf{y}|\mathbf{X},\theta_q)$ as the regularization term instead, which penalizes the parameter configurations that returns a *full* marginal log likelihood (as opposed to the diagonal likelihood in the original BB-α energy $\frac{1}{\alpha}\sum_n^N \log \mathbb{E}_{q(\mathbf{z})q(\theta)} q_{\mathcal{GP}}(y_n|\mathbf{x}_n,\mathbf{z},\theta)^\alpha)$ that is too high, especially the contribution from non-diagonal covariances. We refer this as *likelihood regularization*. In practice, $-\log q_{\mathcal{GP}}(\mathbf{y}|\mathbf{X},\theta_q)$ is estimated on each mini-batch.

D KL divergence on function space v.s. KL divergence on weight space

We briefly discuss KL divergence on function space in finite dimensional case. In the sleep phase of VIP, we have proposed minimizing the following KL divergence in function space:

$$\mathcal{U}(m,\mathcal{K}) = \mathrm{D}_{\mathrm{KL}}[p(\mathbf{y},\mathbf{f}|\mathbf{X},\theta)||q_{\mathcal{GP}}(\mathbf{y},\mathbf{f}|\mathbf{X},m,\mathcal{K})]. \tag{D.1}$$

This is an example of KL divergence in function space (i.e., the output $\mathbf{f}$). Generally speaking, we may assume that $p(\mathbf{f}) = \int_{\mathbf{W}} p(\mathbf{f}|\mathbf{W})p(\mathbf{W})d\mathbf{W}$, and $q(\mathbf{f}) = \int_{\mathbf{W}} p(\mathbf{f}|\mathbf{W})q(\mathbf{W})$, where $q(\mathbf{W})$ is weight-space variational approximation. That is to say, both stochastic processes p and q can be generated by finite dimensional weight space representation $\mathbf{W}$. This can be seen as a one-step Markov chain with preivious state $s_t = \mathbf{W}$, new state $s_{t+1} = \mathbf{f}$, and probability transition function $r(s_{t+1}|s_t) = p(\mathbf{f}|\mathbf{W})$. Then, by applying the second law of thermodynamics of Markov chains(Cover & Thomas (2012)), we have:

$$\mathrm{D}_{\mathrm{KL}}[p(\mathbf{f})||q(\mathbf{f})] \leq \mathrm{D}_{\mathrm{KL}}[p(\mathbf{W})||q(\mathbf{W})] \tag{D.2}$$

This shows that the KL divergence in function space forms a tighter bound than the KL divergence on weight space, which is one of the merits of function space inference.

E Further discussions on Bayesian neural networks

We provide a comparison between our kernel in equation (6), and the kernel proposed in Gal & Ghahramani (2016a). Notably, consider the following Gaussian process:

$$y(\cdot) \sim \mathcal{GP}(0, \mathcal{K}_{\mathrm{VDO}}(\cdot,\cdot)),$$

$$\mathcal{K}_{\mathrm{VDO}}(\mathbf{x}_1,\mathbf{x}_2) = \int p(\mathbf{w})p(b)\sigma(\mathbf{w}^\top\mathbf{x}_1 + b)\sigma(\mathbf{w}^\top\mathbf{x}_2 + b)d\mathbf{w}db. \tag{E.1}$$

Here $\sigma(\cdot)$ is a non-linear activation function, $\mathbf{w}$ is a vector of length D, b is the bias scaler, and $p(\mathbf{w})$, $p(b)$ the corresponding prior distributions. Gal & Ghahramani (2016a) considered approximating this GP with a one-hidden layer BNN $\hat{y}(\cdot) = \mathrm{BNN}(\cdot,\theta)$ with θ collecting the weights and bias vectors of the network. Denote the weight matrix of the first layer as $\boldsymbol{W} \in \mathbb{R}^{D\times K}$, i.e. the network has K hidden units, and the kth column of $\boldsymbol{W}$ as $\mathbf{w}_k$. Similarly the bias vector is $\boldsymbol{b} = (b_1, ..., b_K)$. We further assume the prior distributions of the first-layer parameters are $p(\boldsymbol{W}) = \prod_{k=1}^K p(\mathbf{w}_k)$ and $p(\boldsymbol{b}) = \prod_{k=1}^K p(b_k)$, and use mean-field Gaussian prior for the output layer. Then this BNN constructs an approximation to the GP kernel as:

$$\tilde{\mathcal{K}}_{\mathrm{VDO}}(\mathbf{x}_1,\mathbf{x}_2) = \frac{1}{K}\sum_k \sigma(\mathbf{w}_k^\top\mathbf{x}_1 + b_k)\sigma(\mathbf{w}_k^\top\mathbf{x}_2 + b_k),$$
$$\mathbf{w}_k \sim p(\mathbf{w}), \quad b_k \sim p(b).$$

This approximation is equivalent to the empirical estimation (6), if $S = K$ and the IP is defined by

$$g_\theta(\mathbf{x},\mathbf{z}) = \sigma(\mathbf{w}^\top\mathbf{x} + b), \mathbf{z} = \{\mathbf{w}, b\}, p(\mathbf{z}) = p(\mathbf{w})p(b),$$
$$p(\mathbf{z}),\ \sigma(\cdot) \text{ satisfy } \mathbb{E}_{p(\mathbf{z})}[\sigma(\mathbf{w}^\top\mathbf{x} + b)] = 0. \tag{E.2}$$

In such case, the output layer of that one-hidden layer BNN corresponds to the Bayesian linear regression "layer" in our final approximation. However, the two methods are motivated in different ways. Gal & Ghahramani (2016a) used this interpretation to approximate a GP with kernel (E.1) using a one-hidden layer BNN, while our goal is to approximate the IP E.2 by a GP (note that the IP is defined as the output of the *hidden* layer, not the output of the BNN). Also this coincidence only applies when the IP is defined by a Bayesian logistic regression model, and our approximation is applicable to BNN and beyond.

F Further experimental details

We provide further experimental details in this section. We opensource the code of VIP for UCI experiments at `https://github.com/LaurantChao/VIP`.

F.1 General settings for VIP

For small datasets we use the posterior GP equations for prediction, otherwise we use the $\mathcal{O}(S^3)$ approximation. We use $S = 20$ for VIP unless noted otherwise. When the VIP is equipped with a Bayesian NN/LSTM as prior over functions (Example 3-4), the prior parameters over each weight are untied, thus can be individually tuned. Empirical

Bayesian estimates of the prior parameters are used in 4.3 and 4.4.

F.2 Further experimental details of synthetic example

The compositional kernel for GP is the summation of RBF and Periodic kernels. In this toy experiment, both VDO and VIP use a BNN as the underlying model. Note that it appears that the GP slightly overfits. It is possible to hand-pick the kernel parameters for a smoother fit of GP. However, we have found that quantitatively this will result in a decrease in test predictive likelihood and an increase of RMSE. Therefore, we chose to optimize the kernel parameters by maximizing the marginal likelihood.

F.3 Further implementation details for multivariate regression experiments

- Variational Gaussian inference for BNN (VI-BNN): we implement VI for BNN using the Bayesian deep learning library, ZhuSuan (Shi et al., 2017). VI-BNN employs a mean-field Gaussian variational approximation but evaluates the variational free energy using the reparameterisation trick (Kingma & Welling, 2013). We use a diagonal Gaussian prior for the weights and fix the prior variance to 1. The noise variance of the Gaussian noise model is optimized together with the means and variances of the variational approximation using the variational free energy.

- Variational implicit process-Neural Sampler regressor (VIP-NS): we use neural sampler with two hidden layers of 10 hidden units. The input noise dimension is 10 or 50, which is determined using validation set.

- Variational dropout (VDO) for BNN: similar to Gal & Ghahramani (2016a), we fix the length scale parameter $0.5 * l^2 = 10e^{-6}$. Since the network size is relatively small, dropout probability is set as 0.005 or 0.0005. We use 2000 forward passes to evaluate posterior likelihood.

- α-dropout inference for BNN: suggested by Li & Gal (2017), we fix $\alpha = 0.5$ which often gives high quality uncertainty estimations, possibility due to it is able to achieve a balance between reducing training error and improving predictive likelihood. We use $K = 10$ for MC sampling.

- Variational sparse GPs and exact GPs: we implement the GP-related algorithms using GPflow (Matthews et al., 2017). variational sparse GPs uses 50 inducing points. Both GP models use the RBF kernel.

- About noise variance parameter grid search for VIPs (VIP-BNN and VIP-NS), VDOs and α-dropout: we start with random noise variance parameter, run optimization on the model parameters, and then perform a (thick) grid search over noise variance parameter on validation set. Then, we train the model on the entire training set using this noise variance parameter value. This coordinate ascent like procedure does not require training the model for multiple times as in Bayesian optimization, therefore can speed up the learning process. The same procedure is used to search for optimal hyperparameter ψ of the inverse-Wishart process of VIPs.

F.4 Additional implementation details for ABC experiment

Following the experimental setting of Papamakarios & Murray (2016), we set the ground truth L-V model parameter to be $\theta_1 = 0.01, \theta_2 = 0.5, \theta_3 = 1.0, \theta_4 = 0.01$. We simulate population data in the range of $[0, 30]$ with step size 0.05, which result in 600 gathered measurements. We use the first 500 measurements as training data, and the remaining as test set. For MCMC-ABC and SMC-ABC setup, we also follow the implementation of Papamakarios & Murray (2016).[3] MCMC-ABC is ran for 10000 samples with tolerance ϵ set to be 2.0 which is manually tuned to give the best performance. In MCMC-ABC, last 100 samples are taken as samples. Likewise SMC-ABC uses 100 particles. Model likelihood is calculated based on Gaussian fit. VIP ($\alpha = 0$) is trained for 10000 iterations with Adam optimizer using 0.001 learning rate.

F.5 Additional implementation details for predicting power conversion efficiency of organic photovoltaics molecules

For Bayesian LSTMs, we put Gaussian prior distributions over LSTM weights. The output prediction is defined as the final output at the last time step of the input sequence. We use $S = 10$ for VIP. All methods use Adam with a learning rate of 0.001 for stochastic optimization. Noise variance parameter are not optimized, but set to suggested value according to Hernández-Lobato et al. (2016).To match the run time of the fingerprint-based methods, all LSTM methods are trained for only 100 epochs with a batch size of 250. Among different models in the last few iterations of optimization, we choose the one with the best training likelihood for testing. Note that in the original paper of variational dropout and α-dropout inference, K sample paths ($K = 1$ for VDO and $K = 10$ for α-dropout) are created for *each* training data, which is too prohibitive for memory storage. Therefore, in our implementation, we enforce all training data to share K sample paths. This approximation

[3] https://github.com/gpapamak/epsilon_free_inference

is accurate since we use a small dropout rate, which is 0.005.

F.6 Additional Tables

Table 4: Interpolation performance on toy dataset.

Method	VIP	VDO	GP
Test NLL	**-0.60±0.01**	-0.07 ± 0.01	-0.27 ± 0.00
Test RMSE	**0.140±0.00**	0.161±0.00	0.152±0.00

Table 5: Interpolation performance on solar irradiance.

Method	VIP	VDO	SVGP	GP
Test NLL	**0.08±0.02**	0.21 ± 0.04	0.56 ± 0.23	0.832 ± 0.00
Test RMSE	**0.28±0.00**	0.29±0.01	0.55±0.08	0.650±0.0

Table 6: Performance on clean energy dataset

Metric	**VIP**	**VDO-LSTM**	**α-LSTM**	**BB-α**	**VI-BNN**	**FITC-GP**	**EP-DGP**
Test NLL	**0.65±0.01**	1.24±0.01	2.06±0.02	0.74±0.01	1.37±0.02	1.25±0.00	0.98±0.00
Test RMSE	**0.88±0.02**	0.93±0.01	1.38±0.02	1.08±0.01	1.07±0.01	1.35±0.00	1.17±0.00