Split-GPLVM

Junhao Xiong

August 2020

1 Overview

Let $\mathbf{Y} \in \mathbb{R}^{N \times D}$ be the data matrix, where N is the number of data points, D is the dimensionality of each data point (in the single cell setting, N is the number of cells and D is the number of genes, or some dimensionality-reduced representation of the expression profiles). These data points are associated with latent variables $\mathbf{X} \in \mathbb{R}^{N \times Q}$, where $Q << D$. In addition, each data point is associated with a *latent* category label: $\mathbf{c} = (c^{(1)}, c^{(2)}, ..., c^{(N)}), c^{(n)} \in \{1, ..., K\}$ where K is the number of categories.

The latent representations are related to the observed output through an unknown mapping function f where f follows a prior distribution defined as a Gaussian Process, i.e. for a particular output $\mathbf{y}$ and a particular latent input $(\mathbf{x}, c)$, we have the following:

$$\mathbf{y} = f((\mathbf{x}, c)) + \epsilon, f \in \mathcal{GP}(0, k) \tag{1}$$

where $\epsilon \in \mathcal{N}(0, \sigma^2 \mathbf{I}_N)$ is the observation noise and k is kernel function.

Given the observed $\mathbf{Y}$, the goal is to infer posterior estimates of both the latent locations $\mathbf{X}$ and the latent category $\mathbf{c}$, along with the unknown mapping function $f(\cdot)$.

We further separate the latent space into a *shared space* with dimensionality Q_s and a *private space* with dimensionality Q_p. Therefore, each latent representation can be denoted as $\mathbf{x} = [{\mathbf{x}_s}^T, {\mathbf{x}_p}^T]^T$, $\mathbf{x}_s \in \mathbb{R}^{Q_s}, \mathbf{x}_p \in \mathbb{R}^{Q_p}$, where $\mathbf{x}_s$ and $\mathbf{x}_p$ are the latent representation in the shared and private space respectively. We further decompose f into a component f_s for the shared space with kernel k_s and f_k with kernel k_k where $k \in \{1, ..., K\}$, each for a private space. Hence, our model can be equivalently written as:

$$\mathbf{y} = f_s(\mathbf{x}_s) + \sum_{k=1}^{K} \mathbf{1}(c = k) f_k(\mathbf{x}_p) + \epsilon \tag{2}$$

2 Model Assumption

Our model bears strong resemblance to the Structure Consolidated Latent Variable Model [Yousefi *et al.* 2016]. The main difference is that the category labels $\mathbf{c}$ are treated as *unobserved* rather than observed. In addition, our model has a different form in that we decompose the function f whereas they decompose the kernel k. The models are exactly equivalent, but lead to different approximations: writing the model in this form allows us to approximate f_s and the f_k's separately, which leads to simpler implementation and better interpretability.

Suppose the GPs are independent across the features, then our likelihood can be written as:

$$p(\mathbf{Y}|\mathbf{X},\mathbf{c}) = \prod_{d=1}^{D} p(\mathbf{y}_d|\mathbf{X},\mathbf{c}) = \prod_{d=1}^{D} \mathcal{N}(\mathbf{y}_d|\mathbf{0},\mathbf{K}_{ff}+\sigma^2\mathbf{I}_N) \tag{3}$$

where $\mathbf{K}_{ff}$ is $N \times N$ covariance matrix constructed by evaluating the kernel function on $\mathbf{X}$.

We put standard normal prior on $\mathbf{X}$, supposing $\mathbf{X}$ factorizes across samples and between the shared and private space:

$$p(\mathbf{X}) = \prod_{n=1}^{N} \mathcal{N}(\mathbf{x}|0,\mathbf{I}_Q) = \prod_{n=1}^{N} \mathcal{N}(\mathbf{x}_s^{(n)}|0,\mathbf{I}_{Q_s})\mathcal{N}(\mathbf{x}_p^{(n)}|0,\mathbf{I}_{Q_p}) \tag{4}$$

and uniform prior on $\mathbf{c}$:

$$p(\mathbf{c}) = \prod_{n=1}^{N} Cat(c^{(n)}|1/K) \tag{5}$$

Suppose the latent variables factorizes in the prior, i.e. $p(\mathbf{X},\mathbf{c}) = p(\mathbf{X})p(\mathbf{c})$, the marginal log likelihood is $\log p(\mathbf{Y}) = \int \log p(\mathbf{Y}|\mathbf{X},\mathbf{c})p(\mathbf{X})p(\mathbf{c})d\mathbf{X}d\mathbf{c}$. Since the marginal is intractable, we apply variational inference and derive a closed-formed lower bound (ELBO) to the log marginal likelihood by following a sparse GP approximation [Titsias & Lawrence 2010].

We explicitly specify our variational distribution as:

$$q(\mathbf{X},\mathbf{c}) = \prod_{n=1}^{N} q(\mathbf{x}_s^{(n)})q(\mathbf{x}_p^{(n)})q(c_{\mathbf{x}}^{(n)}) \tag{6}$$

$$q(\mathbf{x}_s^{(n)}) \sim \mathcal{N}(\mathbf{x}_s^{(n)}|\boldsymbol{\mu}_s^{(n)},\mathbf{S}_s^{(n)}) \tag{7}$$

$$q(\mathbf{x}_p^{(n)}) \sim \mathcal{N}(\mathbf{x}_p^{(n)}|\boldsymbol{\mu}_p^{(n)},\mathbf{S}_p^{(n)}) \tag{8}$$

$$q(c_{\mathbf{x}}^{(n)}) \sim Cat(\boldsymbol{\pi}_n) \tag{9}$$

where $\sum_{k=1}^{K} \pi_{nk} = 1$ for all n. For simplicity, the covariance matrices $\mathbf{S}_s^{(n)}, \mathbf{S}_p^{(n)}$ are assumed to be diagonal (so $\mathbf{x}$ also factorizes across the features). The variational parameters $\{\boldsymbol{\mu}_s^{(n)}, \mathbf{S}_s^{(n)}, \boldsymbol{\mu}_p^{(n)}, \mathbf{S}_p^{(n)}, \boldsymbol{\pi}_n\}_{n=1}^{N}$ plus model hyperparameters fully specify our model, and can be estimated by maximizing the ELBO over these parameters by applying gradient-based technique.

3 More Detailed Variational Inference

Our inference procedure begins similarly as Titsias & Lawrence 2010, with additional complexities in decomposing f into a term for the shared space and a category dependent term for the private space. Then we turn to Hensman *et al.* 2013 for a fully factorized lower bound which allows us to further factorize over the categories.

3.1 Introduction with Titsias & Lawrence 2010

The ELBO for $\log p(\mathbf{Y})$ takes the following form:

$$F(q) = \int q(\mathbf{X})q(\mathbf{c}) \frac{\log p(\mathbf{Y}|\mathbf{X}, \mathbf{c})p(\mathbf{X})p(\mathbf{c})}{q(\mathbf{X})q(\mathbf{c})} d\mathbf{X}d\mathbf{c} \tag{10}$$

$$= \int q(\mathbf{X})q(\mathbf{c}) \log p(\mathbf{Y}|\mathbf{X}, \mathbf{c})d\mathbf{X}d\mathbf{c} - D_{KL}(q(\mathbf{X}) \,||\, p(\mathbf{X})) - D_{KL}(q(\mathbf{c})) \,||\, p(\mathbf{c})) \tag{11}$$

$$= \hat{F(q)} - D_{KL}(q(\mathbf{X}) \,||\, p(\mathbf{X})) - D_{KL}(q(\mathbf{c})) \,||\, p(\mathbf{c})) \tag{12}$$

Both KL terms are tractable since in the first case both distributions are gaussian and in the second both are discrete. Hence, the challenge is in estimating the so-called "likelihood term" $\hat{F(q)}$. Since $\mathbf{c}$ is discrete, integrating over $q(\mathbf{c})$ is just a finite sum over possible values of $\mathbf{c}$. Since both the likelihood and variational distributions factorize over the samples, we can further factorize $\hat{F(q)}$:

$$\hat{F(q)} = \sum_{n=1}^{N} \int q(\mathbf{x}^{(n)})q(c^{(n)}) \log p(\mathbf{y}^{(n)}|\mathbf{x}^{(n)}, c^{(n)})d\mathbf{x}^{(n)}dc^{(n)} \tag{13}$$

$$= \sum_{n=1}^{N} \sum_{k=1}^{K} \pi_{nk} \int q(\mathbf{x}^{(n)}) \log p(\mathbf{y}^{(n)}|\mathbf{x}^{(n)}, c^{(n)} = k)d\mathbf{x}^{(n)} \tag{14}$$

$$= \sum_{n=1}^{N} \sum_{k=1}^{K} \pi_{nk} \int q(\mathbf{x}^{(n)}) \log \mathcal{N}(\mathbf{y}^{(n)}|\mathbf{f}_s(\mathbf{x}_s^{(n)}) + \mathbf{f}_k(\mathbf{x}_p^{(n)}), \sigma^2\mathbf{I}_D)d\mathbf{x}^{(n)} \tag{15}$$

$$\tag{16}$$

where $\mathbf{f}_s, \mathbf{f}_k \in \mathbb{R}^D$ are the latent function values associated with the noise corrupted observed values $\mathbf{y}^{(n)} \in \mathbb{R}^D$ of the nth sample for each output dimension. Note that we require the lower bound to factorize over the samples so we can write the conditional likelihood of y as dependent on f_s and f_k for a **specific** k, which will simplify computations later on.

3.2 A fully factorized bound for GPs

To compute $\hat{F(q)}$, we effectively need a version of the lower bound in Titsias & Lawrence 2010 that can be decomposed into a sum over N, where each term only depends on a particular input-output pair $(\mathbf{x}^{(n)}, \mathbf{y}^{(n)})$. This actually can't be done given the current form of the bound. The issue resides in the fact that marginalization over the inducing points $\mathbf{U}$ reintroduces dependency among the observations. This is noted in Hensman *et al.* 2013 as the reason why the original Bayesian GPLVM is not amenable for stochastic variational inference. To that end, Hensman *et al.* 2013 describes a bound in section 3.3 that is precisely what we need: a variational lower bound on $\log p(\mathbf{Y})$ that fully factorizes over the samples, but they did not explicitly specify the bound. We fill in the missing details here.

Also note that what we require is different from Gal *et al.* 2014 and Dai *et al.* 2014, which rewrite the original bound in Titsias & Lawrence 2010 into a form such that key operations are decomposable across the samples. However, as noted in Dai *et al.* 2014, these bounds still don't fully factorize. The bound in Hensman *et al.* 2013 is in fact less tight than the original bound, since they approximate $q(\mathbf{U})$ rather than eliminating it with its exact optimum as in Titsias & Lawrence 2010, but we choose it over the original bound because only it permits a fully factorized bound over the samples and allows us to decompose the bound over the categories.

We start with equation 4 in Hensman *et al.* 2013, which explicitly parametrizes the inducing variables: $q(\mathbf{u}) = \mathcal{N}(\mathbf{u}|\mathbf{m}, \mathbf{S})$. In Hensman *et al.* 2013 the bounds were written assuming the output y_i is one-dimensional. Since we have D-dimensional output, we make it explicit that the bound is for a specific dimension d by writing $q(\mathbf{u}_d) = \mathcal{N}(\mathbf{u}_d|\mathbf{m}_d, \mathbf{S}_d)$ with $\mathbf{u}_d \in \mathbb{R}^M, \mathbf{m}_d \in \mathbb{R}^M, \mathbf{S}_d \in \mathbb{R}^{M\times M}$, so we have the following based on Hensman *et al.* 2013:

$$\log p(\mathbf{y}_d|\mathbf{X}) \geq \mathbf{E}_{q(\mathbf{u}_d)}\left[\mathbf{E}_{p(\mathbf{f}_d|\mathbf{u}_d)}\left[\log p(\mathbf{y}_d|\mathbf{f}_d)\right]\right] - D_{KL}(q(\mathbf{u}_d) \,||\, p(\mathbf{u}_d)) \tag{17}$$

$$= \sum_{n=1}^{N}\left[\log \mathcal{N}(y_{nd}|\mathbf{k}_n^\top \mathbf{K}_{uu}^{-1}\mathbf{m}_d, \sigma^2) - \frac{1}{2\sigma^2}\tilde{k_{nn}} - \frac{1}{2\sigma^2}Tr[\mathbf{S}_d\mathbf{\Lambda}_n]\right] - D_{KL}(q(\mathbf{u}_d) \,||\, p(\mathbf{u}_d)) \tag{18}$$

$$= \mathcal{L}_3 \tag{19}$$

where $\mathbf{K}_{uu}$ is an $M \times M$ matrix constituted of evaluating the kernel function

on the inducing inputs $\mathbf{Z} \in \mathbb{R}^{M \times D}$. $\mathbf{k}_n \in \mathbb{R}^M$ is the nth column of $\mathbf{K}_{uf}$, with $\mathbf{K}_{uf} = \mathbf{K}_{fu}^T$ being the $M \times N$ cross-covariance matrix between $\mathbf{X}$ and $\mathbf{Z}$. $\tilde{k_{nn}}$ is the nth diagonal entry of $\tilde{\mathbf{K}} = \mathbf{K}_{ff} - \mathbf{K}_{fu}\mathbf{K}_{uu}^{-1}\mathbf{K}_{uf}$, with $\mathbf{K}_{ff}$ being the $N \times N$ matrix constituted of evaluating the kernel function on $\mathbf{X}$. $\mathbf{\Lambda}_n = \mathbf{K}_{uu}^{-1}\mathbf{k}_n\mathbf{k}_n^\top\mathbf{K}_{uu}^{-1}$.

Since $\log p(\mathbf{Y}|\mathbf{X}, \mathbf{c}) = \sum_{d=1}^{D} \log p(\mathbf{y}_d|\mathbf{X}, \mathbf{c})$, extending $\mathcal{L}_3$ to D-dimensional output simply involves summing over D. We denote each output as $\mathbf{y}^{(n)} \in \mathbb{R}^{1 \times D}$, $\mathbf{U} \in \mathbb{R}^{M \times D}$ is the whole set of inducing variables with $\mathbf{M} \in \mathbb{R}^{M \times D}$ as the means and $\mathbf{S} \in \mathbb{R}^{D \times M \times M}$ as the set of covariance matrices. We denote the bound for D-dimensional output as $\mathcal{L}^D$:

$$\mathcal{L}^D = \sum_{n=1}^{N}\sum_{d=1}^{D}\left[\log \mathcal{N}(y_{nd}|\mathbf{k}_n^\top\mathbf{K}_{uu}^{-1}\mathbf{m}_d, \sigma^2) - \frac{1}{2\sigma^2}\tilde{k_{nn}} - \frac{1}{2\sigma^2}Tr[\mathbf{S}_d\mathbf{\Lambda}_n]\right] - D_{KL}(q(\mathbf{u}_d) \,||\, p(\mathbf{u}_d)) \tag{20}$$

$$= \sum_{n=1}^{N}\left[\log \mathcal{N}(\mathbf{y}^{(n)}|\mathbf{k}_n^\top\mathbf{K}_{uu}^{-1}\mathbf{M}, \sigma^2\mathbf{I}_D) - \frac{D}{2\sigma^2}\tilde{k_{nn}} - \frac{1}{2\sigma^2}\sum_{d=1}^{D}Tr[\mathbf{S}_d\mathbf{\Lambda}_n]\right] - D_{KL}(q(\mathbf{U}) \,||\, p(\mathbf{U})) \tag{21}$$

$$\tag{22}$$

3.3 Computing the lower bound for GPLVM

To obtain the corresponding bound for GPLVM, we simply need to approximate $p(\mathbf{Y}|\mathbf{X})$ with $\mathcal{L}^D$. But it's important to note, since our model is a mixture model, each sample $\mathbf{y}$ is generated by the function $f_s + f_k$ **for a specific** k. Since the inducing variables $\mathbf{U}$ are supposed to approximate the latent function value $\mathbf{f}$, we actually need K sets of $\mathbf{U}$, each corresponding to a set of latent function values $\mathbf{f}_k$. We denote the inducing variables which approximate $\mathbf{f}_s + \mathbf{f}_k$ as $q(\mathbf{U}_k) = \mathcal{N}(\mathbf{U}_k|\mathbf{M}_k, \mathbf{S}_k)$. Starting with the lower bound in Equation 10, we

have:

$$\log p(\mathbf{Y}) \geq \int q(\mathbf{X})q(\mathbf{c}) \log p(\mathbf{Y}|\mathbf{X}, \mathbf{c}) d\mathbf{X} d\mathbf{c} - D_{KL}(q(\mathbf{X}) \,||\, p(\mathbf{X})) - D_{KL}(q(\mathbf{c})) \,||\, p(\mathbf{c}))$$

(23)

$$= \sum_{n=1}^{N} \sum_{k=1}^{K} \pi_{nk} \int q(\mathbf{x}^{(n)}) \log \mathcal{N}(\mathbf{y}^{(n)} | \mathbf{f}_s(\mathbf{x}_s^{(n)}) + \mathbf{f}_k(\mathbf{x}_p^{(n)}), \sigma^2 \mathbf{I}_D) d\mathbf{x}^{(n)} - D_{KL}(q(\mathbf{X}) \,||\, p(\mathbf{X})) - D_{KL}(q(\mathbf{c}))$$

(24)

$$\geq \sum_{n=1}^{N} \sum_{k=1}^{K} \pi_{nk} \int q(\mathbf{x}^{(n)}) \left[\log \mathcal{N}(\mathbf{y}^{(n)} | \mathbf{k}_n^\top \mathbf{K}_{uu}^{-1} \mathbf{M}_k, \sigma^2 \mathbf{I}_D) - \frac{D}{2\sigma^2} \tilde{k_{nn}} - \frac{1}{2\sigma^2} \sum_{d=1}^{D} Tr[\mathbf{S}_{kd} \mathbf{\Lambda}_n] \right] d\mathbf{x}^{(n)}$$

(25)

$$- D_{KL}(q(\mathbf{X}) \,||\, p(\mathbf{X})) - D_{KL}(q(\mathbf{c})) \,||\, p(\mathbf{c})) - \sum_{k=1}^{K} D_{KL}(q(\mathbf{U}_k) \,||\, p(\mathbf{U}_k))$$

(26)

$$= \sum_{n=1}^{N} \sum_{k=1}^{K} \pi_{nk} \mathcal{L}_{nk}^D - D_{KL}(q(\mathbf{X}) \,||\, p(\mathbf{X})) - D_{KL}(q(\mathbf{c})) \,||\, p(\mathbf{c})) - \sum_{k=1}^{K} D_{KL}(q(\mathbf{U}_k) \,||\, p(\mathbf{U}_k))$$

(27)

The factorization over samples is because $q(\mathbf{X}) = \prod_{n=1}^{N} q(\mathbf{x}^{(n)})$ and $\mathcal{L}^D$ also fully factorizes, which also allows us to decompose the bound over the mixtures.

Computing $\mathcal{L}_{nk}^D$ is reminiscent of computing the lower bound in Titsias & Lawrence 2010, where each kernel function involving $\mathbf{X}$ is replaced by an expectation under $q(\mathbf{X})$, which are given by the Psi statistics. We define:

$$\Psi_0^n = \mathbf{E}_{q(\mathbf{x}^{(n)})} \left[k(\mathbf{x}^{(n)}, \mathbf{x}^{(n)}) \right] \tag{28}$$

$$(\mathbf{\Psi}_1)_n = \mathbf{E}_{q(\mathbf{x}^{(n)})} \left[\mathbf{k}_n^\top \right] \tag{29}$$

$$\mathbf{\Psi}_2^n = \mathbf{E}_{q(\mathbf{x}^{(n)})} \left[\mathbf{k}_n \mathbf{k}_n^\top \right] \tag{30}$$

$\Psi_0^n \in \mathbb{R}, (\mathbf{\Psi}_1)_n \in \mathbb{R}^{1 \times M}, \mathbf{\Psi}_2^n \in \mathbb{R}^{M \times M}$. Explicit computations are available in Section 4.3.

Then, we can write $\mathcal{L}_{nk}^D$ as the following:

$$\mathcal{L}_{nk}^D = -\frac{D}{2} \log(2\pi\sigma^2) - \frac{1}{2\sigma^2} Tr \left[\mathbf{y}^{(n)\top} \mathbf{y}^{(n)} - 2\mathbf{y}^{(n)\top} (\mathbf{\Psi}_1)_n \mathbf{K}_{uu}^{-1} \mathbf{M}_k + \mathbf{M}_k^\top \mathbf{K}_{uu}^{-1} (\mathbf{\Psi}_2^n) \mathbf{K}_{uu}^{-1} \mathbf{M}_k \right]$$

(31)

$$- \frac{D}{2\sigma^2} \Psi_0^n + \frac{D}{2\sigma^2} Tr \left[\mathbf{K}_{uu}^{-1} (\mathbf{\Psi}_2^n) \right] - \frac{1}{2\sigma^2} \sum_{d=1}^{D} Tr \left[\mathbf{S}_{kd} \mathbf{K}_{uu}^{-1} (\mathbf{\Psi}_2^n) \mathbf{K}_{uu}^{-1} \right] \tag{32}$$

where $\mathbf{y}^{(n)}$ is the nth row vector of $\mathbf{Y}$. This bound has a somewhat simpler expression than the original bound for Bayesian GPLVM, since we explicitly

parametrize $q(\mathbf{U})$ with $\mathbf{M}, \mathbf{S}$ instead of having to sub in the optimal $q(\mathbf{U}) = \mathcal{N}(\mu_u, \Sigma_u)$ as in Titsias & Lawrence 2010. Substituting Equation 43 in Equation 27 gives our final ELBO, with the following important note made implicit in the notations.

Note that whenever a kernel function is involved (e.g. $\mathbf{K}_{uu}$ and the Psi statistics), the kernels involved are decomposed into k_s and k_k for a specific $k \in \{1, ...K\}$. So there need be a separate $\mathbf{K}_{uu}$ and a separate sets of Psi statistics for each k.

3.4 An alternative approximation

Instead of approximating each $\mathbf{f}_s + \mathbf{f}_k$ with $q(\mathbf{U}_k)$, we can introduce one additional set of variable $q(\mathbf{U}_s)$ to explicitly approximate $\mathbf{f}_s$, and have each $q(\mathbf{U}_k)$ approximating each $\mathbf{f}_k$. That way, we can examine the contribution of $\mathbf{f}_s$ and each $\mathbf{f}_k$ separately for each sample, leading to a potentially more interpretable model. That is, previously we have:

$$p(\mathbf{f}_s + \mathbf{f}_k|\mathbf{U}_k) = \mathcal{N}((\mathbf{K}^s_{fu} + \mathbf{K}^k_{fu})(\mathbf{K}^s_{uu} + \mathbf{K}^k_{uu})^{-1}\mathbf{U}_k, \tilde{K}) \tag{33}$$

where $\tilde{K} = (\mathbf{K}^s_{ff} + \mathbf{K}^k_{ff}) - (\mathbf{K}^s_{fu} + \mathbf{K}^k_{fu})(\mathbf{K}^s_{uu} + \mathbf{K}^k_{uu})^{-1}(\mathbf{K}^s_{uf} + \mathbf{K}^k_{uf})$

Now we have

$$p(\mathbf{f}_s + \mathbf{f}_k|\mathbf{U}_s, \mathbf{U}_k) = \mathcal{N}(\mathbf{K}^s_{fu}(\mathbf{K}^s_{uu})^{-1}\mathbf{U}_s + \mathbf{K}^k_{fu}(\mathbf{K}^k_{uu})^{-1}\mathbf{U}_k, \tilde{K_s} + \tilde{K_k}) \tag{34}$$

where $\tilde{K_s} = \mathbf{K}^s_{ff} - \mathbf{K}^s_{fu}(\mathbf{K}^s_{uu})^{-1}\mathbf{K}^s_{uf}, \tilde{K_k} = \mathbf{K}^k_{ff} - \mathbf{K}^k_{fu}(\mathbf{K}^k_{uu})^{-1}\mathbf{K}^k_{uf}$.

This is under the assumption that $p(\mathbf{f}_s|\mathbf{U}_s)$ and $p(\mathbf{f}_k|\mathbf{U}_k)$ are independent normals. To get the lower bound term inside the squared brackets in Equation 26, one simply takes $\mathbf{E}_{p(\mathbf{f}_s+\mathbf{f}_k|\mathbf{U}_s,\mathbf{U}_k)}[\log p(\mathbf{Y}|\mathbf{f}_s + \mathbf{f}_k)]$. Also, the predicted $\hat{\mathbf{Y}}$ based on the kth mixture component under the new approximation is $\mathbf{K}^s_{fu}(\mathbf{K}^s_{uu})^{-1}\mathbf{M}_s + \mathbf{K}^k_{fu}(\mathbf{K}^k_{uu})^{-1}\mathbf{M}_k$, in contrast to $(\mathbf{K}^s_{fu} + \mathbf{K}^k_{fu})(\mathbf{K}^s_{uu} + \mathbf{K}^k_{uu})^{-1}\mathbf{M}_k$ in the previous approximation. Both can be computed using MAP estimates of $\mathbf{X}$ (for evaluating the covariance matrices) and $\mathbf{U}_s, \mathbf{U}_k$ (for $\mathbf{M}_s, \mathbf{M}_k$).

This approximation requires a slightly different computation of $\mathcal{L}^D_{nk}$. First

we define the Psi statistics separately for the shared and private kernel:

$$(\Psi_0^s)_n = \mathbf{E}_{q(\mathbf{x}_s^{(n)})}\left[k_s(\mathbf{x}_s^{(n)}, \mathbf{x}_s^{(n)})\right] \tag{35}$$

$$(\Psi_0^k)_n = \mathbf{E}_{q(\mathbf{x}_p^{(n)})}\left[k_k(\mathbf{x}_p^{(n)}, \mathbf{x}_p^{(n)})\right] \tag{36}$$

$$(\mathbf{\Psi}_1^s)_n = \mathbf{E}_{q(\mathbf{x}_s^{(n)})}\left[(\mathbf{k}_n^s)^\top\right] \tag{37}$$

$$(\mathbf{\Psi}_1^k)_n = \mathbf{E}_{q(\mathbf{x}_p^{(n)})}\left[(\mathbf{k}_n^k)^\top\right] \tag{38}$$

$$(\mathbf{\Psi}_2^s)_n = \mathbf{E}_{q(\mathbf{x}_s^{(n)})}\left[\mathbf{k}_n^s(\mathbf{k}_n^s)^\top\right] \tag{39}$$

$$(\mathbf{\Psi}_2^k)_n = \mathbf{E}_{q(\mathbf{x}_p^{(n)})}\left[\mathbf{k}_n^k(\mathbf{k}_n^k)^\top\right] \tag{40}$$

$$(\mathbf{\Psi}_2^{sk})_n = \mathbf{E}_{q(\mathbf{x}_s^{(n)})q(\mathbf{x}_p^{(n)})}\left[\mathbf{k}_n^s(\mathbf{k}_n^k)^\top\right] \tag{41}$$

$$(\mathbf{\Psi}_2^{ks})_n = \mathbf{E}_{q(\mathbf{x}_s^{(n)})q(\mathbf{x}_p^{(n)})}\left[\mathbf{k}_n^k(\mathbf{k}_n^s)^\top\right] \tag{42}$$

where essentially, we did take each term in the sums of Section 4.3 and defines a separate statistic for each.

Then, $\mathcal{L}_{nk}^D$ is the following:

$$\mathcal{L}_{nk}^D = -\frac{D}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}Tr[\mathbf{y}^{(n)\top}\mathbf{y}^{(n)} - 2\mathbf{y}^{(n)\top}(\mathbf{\Psi}_1^s)_n(\mathbf{K}_{uu}^s)^{-1}\mathbf{M}_s - 2\mathbf{y}^{(n)\top}(\mathbf{\Psi}_1^k)_n(\mathbf{K}_{uu}^k)^{-1}\mathbf{M}_k \tag{43}$$

$$+ \mathbf{M}_s^\top(\mathbf{K}_{uu}^s)^{-1}(\mathbf{\Psi}_2^s)_n(\mathbf{K}_{uu}^s)^{-1}\mathbf{M}_s + \mathbf{M}_k^\top(\mathbf{K}_{uu}^k)^{-1}(\mathbf{\Psi}_2^k)_n(\mathbf{K}_{uu}^k)^{-1}\mathbf{M}_k \tag{44}$$

$$+ \mathbf{M}_s^\top(\mathbf{K}_{uu}^s)^{-1}(\mathbf{\Psi}_2^{sk})_n(\mathbf{K}_{uu}^k)^{-1}\mathbf{M}_k + \mathbf{M}_k^\top(\mathbf{K}_{uu}^k)^{-1}(\mathbf{\Psi}_2^{ks})_n(\mathbf{K}_{uu}^s)^{-1}\mathbf{M}_s] \tag{45}$$

$$- \frac{D}{2\sigma^2}((\Psi_0^s)_n + (\Psi_0^k)_n) + \frac{D}{2\sigma^2}(Tr\left[(\mathbf{K}_{uu}^s)^{-1}(\mathbf{\Psi}_2^s)_n\right] + Tr\left[(\mathbf{K}_{uu}^k)^{-1}(\mathbf{\Psi}_2^k)_n\right]) \tag{46}$$

$$- \frac{1}{2\sigma^2}\sum_{d=1}^{D} Tr\left[\mathbf{S}_{kd}(\mathbf{K}_{uu}^k)^{-1}(\mathbf{\Psi}_2^k)_n(\mathbf{K}_{uu}^k)^{-1}\right] + Tr\left[\mathbf{S}_{sd}(\mathbf{K}_{uu}^s)^{-1}(\mathbf{\Psi}_2^s)_n(\mathbf{K}_{uu}^s)^{-1}\right] \tag{47}$$

where $\mathbf{y}^{(n)}$ is the nth row vector of $\mathbf{Y}$.

3.5 Optimization of the lower bound

The optimal $q(\mathbf{u}_d|\mathbf{m}_d, \mathbf{S}_d)$ can be found by computing the gradient wrt $\mathbf{m}_d, \mathbf{S}_d$ and setting them to zero, as in equation 5 in Hensman *et al.* 2013. Also as noted in Bui & Turner 2015, the optimal $q(\mathbf{U})$ also factorizes: $q(\mathbf{U}) = \prod_{d=1}^D q(\mathbf{u}_d)$, since $p(\mathbf{U})$ factorizes; and the optimal $\mathbf{S}_d$ is the same for each d. The optimal $\mathbf{M}, \mathbf{S}$ are given as natural parameters in equation 7 of Bui & Turner 2015.

They are computed the same way as equation 5 in Hensman *et al.* 2013, except the covariance matrices involving the data are replaced by the Psi statistics. The optimal $q(\mathbf{X})$ can also be obtained in a similar way as $q(\mathbf{U})$, by setting the gradient of the lower bound wrt $\mathbf{X}$ to zero. Since the optimal $q(\mathbf{X}), q(\mathbf{U})$ depend on each other, we need to alternate between updating each.

Our strategy is similar to Bui & Turner 2015: for each iteration, take a minibatch of the data $\{\mathbf{y}^{(n)}\}_1^{N_b}$, update $q(\{\mathbf{x}^{(n)}\}_1^{N_b})$, $q(\{c^{(n)}\}_1^{N_b}$ and the model hyperparameters (kernel parameters and likelihood variance) by taking stochastic gradient step, while holding $q(\mathbf{U})$ fixed. Then, take stochastic *natural* gradient step in $q(\mathbf{U})$ using $q(\{\mathbf{x}^{(n)}\}_1^{N_b})$ (Hensman *et al.* 2013 mentions the natural gradient component seems to be important for the optimization to work). The two steps can be thought of an M step and a variational E step respectively.

There are further complication when optimizing our model compared to the original Bayesian GPLVM. For example, we need to set the inducing inputs $\mathbf{Z}$ in advanced instead of optimizing over them as in Titsias & Lawrence 2010, since now we are also optimizing $q(\mathbf{U})$, which is strongly correlated with $\mathbf{Z}$ (this is pointed out by Gal *et al.* 2014 and is what Hensman *et al.* 2013 did, but what what would happen if we also update $\mathbf{Z}$ in the M step?)

Also, Bui & Turner 2015 points out that this stochastic optimization scheme converges very slowly on even modest size data due to only updating the subset of $q(\mathbf{X})$ in the minibatch for each iteration, which they solve by sharing variational parameters across $q(\mathbf{X})$ using back constraints.

4 Implementation Details

4.1 Computing the ELBO

We first compute the Psi statistics separately using k_s and for each k_k, without any change in the kernel convolutions over $q(\mathbf{x}^{(n)})$ from Titsias & Lawrence 2010 (the computation is only tractable for particular kernels e.g. ARD squared exponential, which is already implemented in GPflow). Then, to compute each $\mathcal{L}_{nk}^D$, we use equation 43 with the Psi statistics for a specific n, k and $\mathbf{K}_{uu}$ for a specific k. Finally we can use equation 27 to arrive at the final ELBO.

To avoid having to explicitly loop over N, we use the `tf.vectorized map` function to parallelize the computation for each $\mathcal{L}_{nk}^D$ over N. Since K is usually small, we explicitly loop over K.

4.2 Computation involving $\mathbf{K}_{uu}^{-1}$

The computations involving $\mathbf{K}_{uu}^{-1}$ are expensive and can be computed more efficiently using the Cholesky decomposition $\mathbf{K}_{uu} = \mathbf{L}\mathbf{L}^\top$ and solving triangular linear systems. Notice we have expressions of two forms in equation 43: $Tr[A^{-1}C]$ and $Tr[A^{-1}BA^{-1}]$ where A, B are PSD (they are $\mathbf{K}_{uu}$ and $\mathbf{\Psi}_2^n$ respectively). Write $A = LL^\top$ and $B = L_B L_B^\top$, we can solve them in the following way:

$$Tr[A^{-1}C] = Tr[(L^\top)^{-1}L^{-1}C] \tag{48}$$

which involves solving two linear systems with triangular coefficient matrix, and

$$Tr[A^{-1}BA^{-1}] = Tr[(L^\top)^{-1}L^{-1}L_B L_B^\top (L^\top)^{-1}L^{-1}] \tag{49}$$

$$= Tr[DD^\top] \tag{50}$$

where $D = (L^\top)^{-1}L^{-1}L_B$, which again involves solving two linear systems with triangular coefficient matrix.

We also have a term $Tr[SA^{-1}BA^{-1}]$ where $S = L_s L_s^\top$ is also a covariance matrix. This can be absorbed easily into the previous computation:

$$Tr[SA^{-1}BA^{-1}] = Tr[SDD^\top] = Tr[L_s L_s^\top DD^\top] = Tr[L_s^\top DD^\top L_s] = Tr[(L_s^\top D)(L_s^\top D)^\top] \tag{51}$$

where we used the cyclic property of the trace. Also, in general, trace of the form $Tr[A^\top B]$ can be computed without matrix multiplication:

$$Tr[A^\top B] = \sum_{i=1}\sum_{j=1} A_{ij}B_{ij} \tag{52}$$

4.3 Psi statistics computations

Here we expand out the Psi statistics expression so each kernel function decomposes into k_s and k_k for a specific $k \in \{1, ...K\}$. This is due to the decomposition of the covariance matrices. The expressions below are very similar to equation 6-8 in Yousefi *et al.* 2016, except that we don't need to include the category

variables, since each Psi statistic is already conditioned on a specific k.

$$\Psi_0 = \sum_{n=1}^{N} \Psi_0^n = \int k(\mathbf{x}^{(n)}, \mathbf{x}^{(n)}) \mathcal{N}(\mathbf{x}^{(n)} | \boldsymbol{\mu}^{(n)}, \mathbf{S}^{(n)}) d\mathbf{x}^{(n)} \tag{53}$$

$$= \int k_s(\mathbf{x}_s^{(n)}, \mathbf{x}_s^{(n)}) \mathcal{N}(\mathbf{x}_s^{(n)} | \boldsymbol{\mu}_s^{(n)}, \mathbf{S}_s^{(n)}) d\mathbf{x}_s^{(n)} + \int k_k(\mathbf{x}_p^{(n)}, \mathbf{x}_p^{(n)}) \mathcal{N}(\mathbf{x}_p^{(n)} | \boldsymbol{\mu}_p^{(n)}, \mathbf{S}_p^{(n)}) d\mathbf{x}_p^{(n)} \tag{54}$$

$$(\boldsymbol{\Psi}_1)_{n,m} = \int k(\mathbf{x}^{(n)}, \mathbf{z}^{(m)}) \mathcal{N}(\mathbf{x}^{(n)} | \boldsymbol{\mu}^{(n)}, \mathbf{S}^{(n)}) d\mathbf{x}^{(n)} \tag{55}$$

$$= \int k_s(\mathbf{x}_s^{(n)}, \mathbf{z}_s^{(m)}) \mathcal{N}(\mathbf{x}_s^{(n)} | \boldsymbol{\mu}_s^{(n)}, \mathbf{S}_s^{(n)}) d\mathbf{x}_s^{(n)} + \int k_k(\mathbf{x}_p^{(n)}, \mathbf{z}_p^{(m)}) \mathcal{N}(\mathbf{x}_p^{(n)} | \boldsymbol{\mu}_p^{(n)}, \mathbf{S}_p^{(n)}) d\mathbf{x}_p^{(n)} \tag{56}$$

$$(\boldsymbol{\Psi}_2)_{m,m'} = \sum_{n=1}^{N} (\boldsymbol{\Psi}_2^n)_{m,m'} = \int k(\mathbf{x}^{(n)}, \mathbf{z}^{(m)}) k(\mathbf{z}^{(m')}, \mathbf{x}^{(n)}) \mathcal{N}(\mathbf{x}^{(n)} | \boldsymbol{\mu}^{(n)}, \mathbf{S}^{(n)}) d\mathbf{x}^{(n)} \tag{57}$$

$$= \int k_s(\mathbf{x}_s^{(n)}, \mathbf{z}_s^{(m)}) k_s(\mathbf{z}_s^{(m')}, \mathbf{x}_p^{(n)}) \mathcal{N}(\mathbf{x}_s^{(n)} | \boldsymbol{\mu}_s^{(n)}, \mathbf{S}_s^{(n)}) d\mathbf{x}_s^{(n)} \tag{58}$$

$$+ \int k_k(\mathbf{x}_p^{(n)}, \mathbf{z}_p^{(m)}) k_k(\mathbf{z}_p^{(m')}, \mathbf{x}_p^{(n)}) \mathcal{N}(\mathbf{x}_p^{(n)} | \boldsymbol{\mu}_p^{(n)}, \mathbf{S}_p^{(n)}) d\mathbf{x}_p^{(n)} \tag{59}$$

$$+ \int k_s(\mathbf{x}_s^{(n)}, \mathbf{z}_s^{(m)}) \mathcal{N}(\mathbf{x}_s^{(n)} | \boldsymbol{\mu}_s^{(n)}, \mathbf{S}_s^{(n)}) d\mathbf{x}_s^{(n)} \int k_k(\mathbf{z}_p^{(m')}, \mathbf{x}_p^{(n)}) \mathcal{N}(\mathbf{x}_p^{(n)} | \boldsymbol{\mu}_p^{(n)}, \mathbf{S}_p^{(n)}) d\mathbf{x}_p^{(n)} \tag{60}$$

$$+ \int k_k(\mathbf{x}_p^{(n)}, \mathbf{z}_p^{(m)}) \mathcal{N}(\mathbf{x}_p^{(n)} | \boldsymbol{\mu}_p^{(n)}, \mathbf{S}_p^{(n)}) d\mathbf{x}_p^{(n)} \int k_s(\mathbf{z}_s^{(m')}, \mathbf{x}_s^{(n)}) \mathcal{N}(\mathbf{x}_s^{(n)} | \boldsymbol{\mu}_s^{(n)}, \mathbf{S}_s^{(n)}) d\mathbf{x}_s^{(n)} \tag{61}$$

5 Further Ideas and Questions

5.1 Computing the posterior of f_s, f_k

Besides the approximations, here's what you can do theoretically: for each sample $\mathbf{x}^{(n)}$, we can compute the posterior latent function values $p(f_s(\mathbf{x}^{(n)})|y, f_k(\mathbf{x}^{(n)}))$ and $p(f_k(\mathbf{x}^{(n)})|y, f_s(\mathbf{x}^{(n)}))$, with $k = \arg\max_k \pi_{nk}$. Compare to the latent positions $q(\mathbf{x}_s^{(n)}), q(\mathbf{x}_p^{(n)})$, these estimates allow us to examine more explicitly the contributions of the shared space vs. the private space for each sample directly, which might lead to interesting interpretations.

The means of the two posteriors can be estimated in an iterative way:

$$\mathbf{E}\left[f_s(\mathbf{x}^{(n)})|y, f_k(\mathbf{x}^{(n)})\right] = \mathbf{K}^s_{\mathbf{x}n}(\sigma^2\mathbf{I} + \mathbf{K}^s_{ff})^{-1}(y - \mathbf{E}\left[f_k(\mathbf{x}^{(n)})|y, f_s(\mathbf{x}^{(n)})\right]) \tag{62}$$

$$\mathbf{E}\left[f_k(\mathbf{x}^{(n)})|y, f_s(\mathbf{x}^{(n)})\right] = \mathbf{K}^k_{\mathbf{x}n}(\sigma^2\mathbf{I} + \mathbf{K}^k_{ff})^{-1}(y - \mathbf{E}\left[f_s(\mathbf{x}^{(n)})|y, f_k(\mathbf{x}^{(n)})\right]) \tag{63}$$

where $\mathbf{K}^s_{\mathbf{x}n} = [k_s(\mathbf{x}^{(n)}, \mathbf{x}^{(1)}), ..., k_s(\mathbf{x}^{(n)}, \mathbf{x}^{(N)})]$ is an N-dimensional row vector by evaluating k_s between $\mathbf{x}^{(n)}$ and each sample of $\mathbf{X}$, $\mathbf{K}^s_{ff}$ is an $N \times N$ matrix composed of $\mathbf{K}^s_{\mathbf{x}n}$ as its rows. $\mathbf{K}^k_{\mathbf{x}n}, \mathbf{K}^k_{ff}$ are defined similarly but for kernel k_k.

The variances can be estimated separately:

$$\mathbf{V}\left[f_s(\mathbf{x}^{(n)})|y, f_k(\mathbf{x}^{(n)})\right] = k_s(\mathbf{x}^{(n)}, \mathbf{x}^{(n)}) - \mathbf{K}^s_{\mathbf{x}n}(\sigma^2\mathbf{I} + \mathbf{K}^s_{ff})^{-1}{\mathbf{K}^s_{\mathbf{x}n}}^\top \tag{64}$$

$$\mathbf{V}\left[f_k(\mathbf{x}^{(n)})|y, f_s(\mathbf{x}^{(n)})\right] = k_k(\mathbf{x}^{(n)}, \mathbf{x}^{(n)}) - \mathbf{K}^k_{\mathbf{x}n}(\sigma^2\mathbf{I} + \mathbf{K}^k_{ff})^{-1}{\mathbf{K}^k_{\mathbf{x}n}}^\top \tag{65}$$

5.2 Comparison to the approximation in Yousefi *et al.* 2016

If we take the model formulation in Yousefi *et al.* 2016, treat $\mathbf{c}$ as unobserved, and derive the ELBO, we will arrive at equation 5 from that work with the only difference being (1) each term involving a convolution of the private space kernel when computing the psi statistics (equation 6, 7, 8) requires an additional expectation over $q(\mathbf{c})$, which is a summation over the values of $\mathbf{c}$; (2) there is an additional KL term $D_{KL}(q(\mathbf{c}) \,||\, p(\mathbf{c}))$.

We suspect the ELBO will be different from ours, since the mixing is done within the kernel convolution, it will directly change those computations and lead to different expectations of the covariance matrices under the variational distribution, whereas in our case, the mixing is done after the kernel expectations are computed.

I haven't thought through how exactly these two approximations differ. Which bound is tighter? Does it involve different assumptions regarding how the kernel hyperparameters factorize?

References

1. Bui, T. D. & Turner, R. E. *Stochastic variational inference for Gaussian process latent variable models using back constraints* in *Black Box Learning and Inference NIPS workshop* (2015).

2. Dai, Z., Damianou, A., Hensman, J. & Lawrence, N. Gaussian process models with parallelization and GPU acceleration. *arXiv preprint arXiv:1410.4984* (2014).

3. Gal, Y., Van Der Wilk, M. & Rasmussen, C. E. *Distributed variational inference in sparse Gaussian process regression and latent variable models* in *Advances in neural information processing systems* (2014), 3257–3265.

4. Hensman, J., Fusi, N. & Lawrence, N. D. Gaussian processes for big data. *arXiv preprint arXiv:1309.6835* (2013).

5. Titsias, M. & Lawrence, N. D. *Bayesian Gaussian process latent variable model* in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics* (2010), 844–851.

6. Yousefi, F., Dai, Z., Ek, C. H. & Lawrence, N. Unsupervised Learning with Imbalanced Data via Structure Consolidation Latent Variable Model. *arXiv preprint arXiv:1607.00067* (2016).