A Supplementary Material

A.1 Collapsed Heteroscedastic

Lazaro-Gredilla and Titsias [2011] form a bound by 'collapsing out' the $q(\mathbf{f})$ distribution, such that it need not take a Gaussian form. As a brief review their bound can be derived as follows.

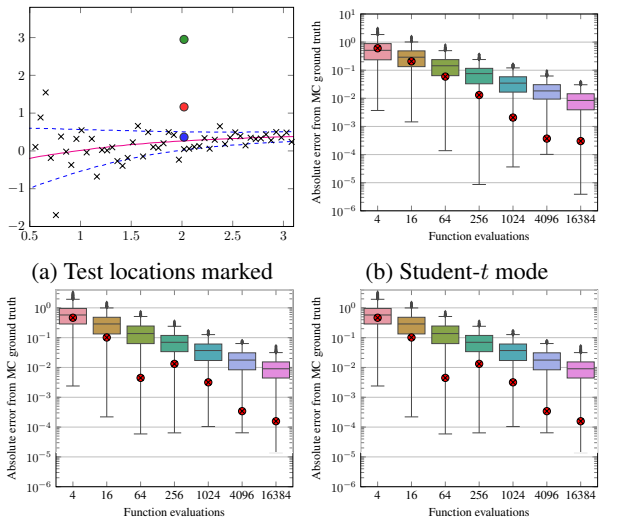
$$\log p(\mathbf{y}|\mathbf{f}) \geq \mathbb{E}_{q(\mathbf{f})}\left[\log p(\mathbf{y}|\mathbf{f},\mathbf{g})\right] + \log \mathbb{E}_{q(\mathbf{g})}\left[\frac{p(\mathbf{g})}{q(\mathbf{g})}\right]$$
$$= \log \mathcal{N}\left(\mathbf{y}|\mathbf{f}, e^{\boldsymbol{m}_{f_i} - \frac{\boldsymbol{v}_{f_i}}{2}}\right) - \frac{1}{4}\sum_{i=1}^{n} \boldsymbol{v}_{f_i} = L'$$

$$\begin{aligned} L &= \int p(\mathbf{y}|\mathbf{f})p(\mathbf{f})d\mathbf{f} \\ &\geq \int e^{L'} p(\mathbf{f})d\mathbf{f} \\ &= \int \mathcal{N}\left(\mathbf{y}|\mathbf{f}, e^{\boldsymbol{m}_{f_i} - \frac{\boldsymbol{v}_{f_i}}{2}}\right)\mathcal{N}\left(\mathbf{f}|0,\mathbf{K_{ff}}\right) - \frac{1}{4}\sum_{i=1}^{n}\boldsymbol{v}_{f_i} \\ &\quad - \mathrm{KL}\left(q(\mathbf{g}) \,\|\, p(\mathbf{g})\right) \\ &= \mathcal{N}\left(\mathbf{y}|0, \mathbf{K_{ff}} + e^{\boldsymbol{m}_{f_i} - \frac{\boldsymbol{v}_{f_i}}{2}}\right) - \frac{1}{4}\sum_{i=1}^{n}\boldsymbol{v}_{f_i} \\ &\quad - \mathrm{KL}\left(q(\mathbf{g}) \,\|\, p(\mathbf{g})\right) \end{aligned}$$

The bound that we assumes a sparse approximation, however it also constrains $q(\mathbf{f})$ to be Gaussian. This leads to an additional KL divergence since the optimal is not chosen, and additional penalty term arising from the mismatch of the constrained form of $q(\mathbf{f})$.

A.2 Quadrature and Monte Carlo

Computing the expected likelihood requires many low-dimensional integrals. Recently, there has been progress in using stochastic methods to obtain unbiased estimates in this area using centered representations [Kingma and Welling, 2014, Rezende et al., 2014]. In this section, we re-examine the effectiveness of Gauss-Hermite quadrature in this setting. Gauss-Hermite quadrature approximates Gaussian integrals in one dimension using a pre-defined grid. For expectations of polynomial functions, the method is exact when the grid size meets the degree of the polynomial; for non-polynomial functions as we will encounter in general, we must accept a small amount of bias. To integrate higher dimensional functions, we must nest the quadrature, doing an integral across one dimension for each quadrature point in the other. Our experiments suggest that even in this case, the amount of bias is negligible, as Figure 7 investigates, examining the accuracy of nested quadrature as compared to Monte Carlo estimates using the centered parameterization [Kingma and Welling, 2014]. Inspired by an examination of quadrature for expectation propagation [Jylänki et al., 2011], we examine the effectiveness for a several positions of the integral of a Student-t.



(a) Test locations marked (b) Student-t mode

(c) Student-t shoulder (d) Student-t tail

Figure 7: Two dimensional Gauss-Hermite quadrature vs Monte Carlo. Each plot shows the log absolute error in estimating the two dimension integral required by our Heteroscedastic Student-t model (see section 4.2). In each case, the bias introduced by quadrature (circles) is small: a long way into the tail of the variance from the MC approximation. In fact, for small numbers of quadrature points, we often do better than the expected value using many more MC samples. Boxplots shows the absolute error on 1000 separate reruns of MC, whereas quadrature is deterministic. The error was evaluated at various points in the tail of the distribution as shown in a).

Gauss-Hermite quadrature is appropriate for our integral as the Gaussian posteriors $q(\mathbf{f}_i)q(\mathbf{g}_i)$ are convolved with a function $p(\mathbf{y}_i|\mathbf{g}_i, \mathbf{f}_i)$. Monte Carlo integration is exact in the limit of infinite samples, however in practice a subset of samples must be used. Gauss-Hermite requires ph^b evaluations per point in the mini-batch, where h is the number of Gauss-Hermite points used, p is the number of output dimensions, and b is the number of latent functions. Since Monte Carlo is unbiased, using a stochastic optimizer with the stochastic estimates of the integral and its gradients will work effectively [Nguyen and Bonilla, 2014, Kingma and Welling, 2014], though we find the bias introduced by the quadrature approach to be negligible. For higher number of latent functions it may be more efficient to make use of low variance Monte Carlo estimates for the integrals. Gradients for the model can be computed in a similar way with the Gaussian idenities used by Opper and Archambeau [2009].

A.3 Gradients and Optimization

Gradients can be computed similarly to [Hensman et al., 2015] using the equalities,

$$\frac{\partial}{\partial \mu}\mathbb{E}_{\mathcal{N}(x|\mu,\sigma^2)}\Big[f(x)\Big] = \mathbb{E}_{\mathcal{N}(x|\mu,\sigma^2)}\Big[\frac{\partial}{\partial x}f(x)\Big] \quad (11)$$

$$\frac{\partial}{\partial \sigma^2}\mathbb{E}_{\mathcal{N}(x|\mu,\sigma^2)}\Big[f(x)\Big] = \frac{1}{2}\mathbb{E}_{\mathcal{N}(x|\mu,\sigma^2)}\Big[\frac{\partial}{\partial x^2}f(x)\Big] \quad (12)$$

and the chain rule.

Since our posterior assumes factorization between $q(\mathbf{f})$ and $q(\mathbf{g})$ we simply do the gradients independently. That is calculate

$$\frac{\partial}{\partial \boldsymbol{\mu}_f}\mathbb{E}_{\mathcal{N}(\mathbf{x}_i|\boldsymbol{m}_f,\boldsymbol{v}_f)}\Big[\log p(\mathbf{y}|\mathbf{f},\mathbf{g})\Big]$$
$$\frac{\partial}{\partial \boldsymbol{\mu}_g}\mathbb{E}_{\mathcal{N}(\mathbf{x}_i|\boldsymbol{m}_g,\boldsymbol{v}_g)}\Big[\log p(\mathbf{y}|\mathbf{f},\mathbf{g})\Big]$$
$$\frac{\partial}{\partial \boldsymbol{v}_f}\mathbb{E}_{\mathcal{N}(\mathbf{x}_i|\boldsymbol{m}_f,\boldsymbol{v}_f)}\Big[\log p(\mathbf{y}|\mathbf{f},\mathbf{g})\Big]$$
$$\frac{\partial}{\partial \boldsymbol{v}_g}\mathbb{E}_{\mathcal{N}(\mathbf{x}_i|\boldsymbol{m}_g,\boldsymbol{v}_g)}\Big[\log p(\mathbf{y}|\mathbf{f},\mathbf{g})\Big],$$

independently using (11) and (12). The expectations can then be done using quadrature, or Monte Carlo sampling. As before

$$\boldsymbol{m}_f = \mathbf{K}_{\mathbf{f}\mathbf{u}_f}\mathbf{K}_{\mathbf{u}_f\mathbf{u}_f}^{-1}\boldsymbol{\mu}_f$$
$$\boldsymbol{v}_f = \mathbf{K}_{\mathbf{f}\mathbf{f}} + \mathbf{K}_{\mathbf{f}\mathbf{u}_f}\mathbf{K}_{\mathbf{u}_f\mathbf{u}_f}^{-1}(\boldsymbol{S}_f - \mathbf{K}_{\mathbf{u}_f\mathbf{u}_f})\mathbf{K}_{\mathbf{u}_f\mathbf{u}_f}^{-1}\mathbf{K}_{\mathbf{u}_f\mathbf{f}}$$
$$\boldsymbol{m}_g = \mathbf{K}_{\mathbf{g}\mathbf{u}_g}\mathbf{K}_{\mathbf{u}_g\mathbf{u}_g}^{-1}\boldsymbol{\mu}_g$$
$$\boldsymbol{v}_g = \mathbf{K}_{\mathbf{g}\mathbf{g}} + \mathbf{K}_{\mathbf{g}\mathbf{u}_g}\mathbf{K}_{\mathbf{u}_g\mathbf{u}_g}^{-1}(\boldsymbol{S}_g - \mathbf{K}_{\mathbf{u}_g\mathbf{u}_g})\mathbf{K}_{\mathbf{u}_g\mathbf{u}_g}^{-1}\mathbf{K}_{\mathbf{u}_g\mathbf{g}}.$$

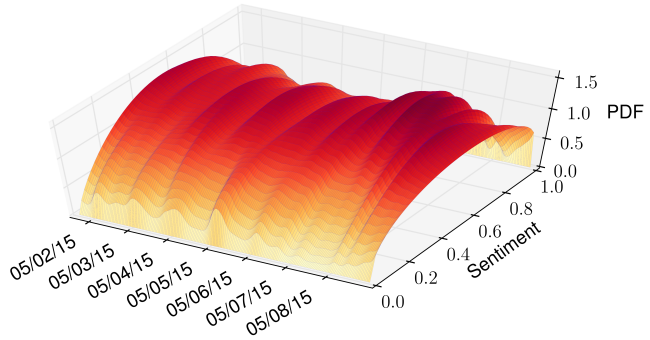
We then can chain using $\frac{\partial}{\partial \boldsymbol{m}_f}\mathbb{E}_{\mathcal{N}(\mathbf{x}_i|\boldsymbol{m}_f,\boldsymbol{v}_f)}\Big[\log p(\mathbf{y}|\mathbf{f},\mathbf{g})\Big]\frac{\partial \boldsymbol{m}_f}{\partial \mathbf{K}_{\mathbf{f}\mathbf{u}_f}}\frac{\partial \mathbf{K}_{\mathbf{f}\mathbf{u}_f}}{\partial \theta}$, where θ is a hyper parameter of the kernel k_f. Similar chain rules can be written for the other derivatives.

The model contains variational parameters corresponding to $q(\mathbf{u}_f) = \mathcal{N}(\mathbf{u}_f|\boldsymbol{\mu}_f, \boldsymbol{S}_f)$ and $q(\mathbf{u}_g) = \mathcal{N}(\mathbf{u}_g|\boldsymbol{\mu}_g, \boldsymbol{S}_g)$ and the latent input locations, $\mathbf{Z}$. As such the parameters do not scale with n. Naively the number of parameters is $\mathcal{O}(b(m^2 + m) + m)$ however we can reduce this to $\mathcal{O}(b(\frac{m^2}{2} + m))$ by parameterizing the Choleksy of the covariance matrices, $\boldsymbol{S}_f = L_{\mathbf{f}}L_{\mathbf{f}}^\top$ and $\boldsymbol{S}_g = L_{\mathbf{g}}L_{\mathbf{g}}^\top$. This has the added benefit of enforcing that $\boldsymbol{S}_f$ and $\boldsymbol{S}_g$ are symmetrical and positive definite.

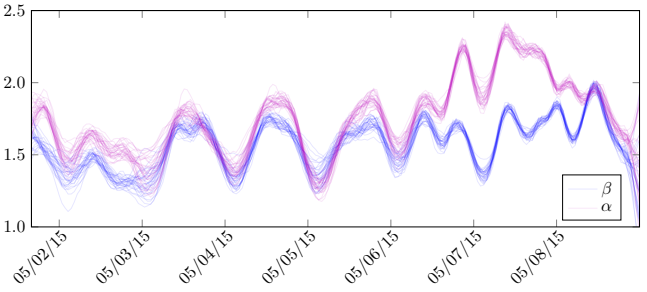
We initialize the model with random or informed length-scales within the right region, $\boldsymbol{\mu}_f$ and $\boldsymbol{\mu}_g$ are assigned small random values, $\boldsymbol{S}_g$ and $\boldsymbol{S}_f$ are given an identity form. In practice during optimization we find it helpful to initially fix all the kernel hyperparameters and $\mathbf{Z}$ at their initial locations, optimize for a small number of steps, then allow the optimization to run freely. This allows the latent means $\boldsymbol{\mu}_f$ and $\boldsymbol{\mu}_g$ to move to sensible locations before the model is allowed to completely change the form of the function through the modification of the kernel hyperparameters. True convergence can be difficult to achieve due to the potentially number of strongly dependent parameters and the non-convex optimization problem, and in practice we find it helpful to monitor convergence. It is important to note however that the number of parameters to be optimized is *fixed* with respect to n.

A.4 Further Twitter experiment details

The model used to model the twitter data has some interesting properties, such as the ability to model a transition from a unimodal distribution to a bimodal distribution. The following plot shows how the distribution changes throughout time for the Labour dataset.



Posterior GP for latent functions of Beta(α, β)



The latent functions α and β which are modelled in Section 4.2.2 can be plotted themselves. If both latent functions went below 1.0 then the distribution at that time would turn into a bathtub shape. If both are larger than one but one is larger than the other, we have a skewed distribution. If one is below zero and the over above, it appears exponential or negative exponential.

A.5 Survival details

To generate the synthetic survival dataset we first define latent functions that we wish to infer. These are a complex function of an input, $\mathbf{x}$, with two dimensions,

$$\alpha = \exp\left(2\exp(-30(\mathbf{x}_{:,0} - \frac{1}{4})^2) + \sin(\pi\mathbf{x}_{:,1}^2) - 2\right)$$
$$\beta = \exp\left(\sin(2\pi\mathbf{x}_{:,0}) + \cos(2\pi\mathbf{x}_{:,1})\right).$$

We then make 1000 synthetic individuals, with covariates sampled uniformly from $\mathbf{x}_{i,0} \sim \text{Uniform}(0, 1)$ and $\mathbf{x}_{i,1} \sim \text{Uniform}(0, 1)$.

Using these two latent functions, α_i and β_i, computed using covariates $\mathbf{x}_i$ for individual i, we sample a simulated failure time from a log-logistic distribution,

$$\mathbf{y} \sim LL(\alpha, \beta) = \frac{\left(\frac{\beta}{\alpha}\right)\left(\frac{\mathbf{y}}{\alpha}\right)^{\beta-1}}{\left(1 + \frac{\mathbf{y}}{\alpha}^{\beta}\right)^2}.$$

These are then the true failure times of individuals with covariates $\mathbf{x}_i$. 20% of the data is chosen to be censored censor. A time is uniformly drawn, $\mathbf{t}_i \in [0, \mathbf{y}_i]$, and the observed time is truncated to this time, $\mathbf{y}_i = \mathbf{t}_i$. Otherwise $\mathbf{t}_i = \mathbf{y}_i$. Additionally a indicator $\delta_i = 1$ is provided to the model if censoring occurs, and $\delta_i = 0$ if the real failure time was observed. This mimics patients dropping out of a trial, with the assumption that the time at which they drop out is independent of the failure time and covariates. For these censored times, we only know that $\mathbf{T}_i > \mathbf{t}_i$, and for the uncensored individuals it is known that $\mathbf{T}_i = \mathbf{t}_i$.

As such the likelihood is decomposed into $P(t_i \leq \mathbf{y}_i < \mathbf{t}_i + \delta t|\alpha_i, \beta_i, \delta_i = 0)$ and $P(\mathbf{y}_i|\alpha_i, \beta_i, \delta_i = 1) = 1 - P(\mathbf{y}_i > \mathbf{t}_i|\alpha_i, \beta_i, \delta_i = 1)$

$$p(\mathbf{y}|\alpha, \beta, \delta) = \prod_i^{K:\delta\neq 1} \frac{\left(\frac{\beta_i}{\alpha_i}\right)\left(\frac{\mathbf{y}_i}{\alpha_i}\right)^{\beta_i - 1}}{\left(1 + \frac{\mathbf{y}_i}{\alpha_i}^{\beta_i}\right)^2} \prod_j^{M:\delta=1} \frac{1}{1 + \left(\frac{\mathbf{y}_j}{\alpha_j}\right)^{\beta_j}}$$

The task is then to infer α and β, such that we know how the failure time distribution varies in response to covariate information.