Supplementary Material: Sparse Algorithms for Markovian Gaussian Processes

A Statistical Properties of Linear SDEs

A.1 Marginals

A linear time invariant (LTI) stochastic differential equation (SDE) can be expressed as follows:

$$\dot{\mathbf{s}}(x) = \mathbf{F}\mathbf{s}(x) + \mathbf{L}\boldsymbol{\varepsilon}(x)\,, \qquad f(x) = \mathbf{H}\mathbf{s}(x)\,, \tag{24}$$

where $\boldsymbol{\varepsilon}(x)$ is a white noise process, $\mathbf{F}$ is the feedback matrix, $\mathbf{L}$ is the noise effect matrix, and $\mathbf{H}$ is the measurement matrix.

The marginal distribution of the solution to this LTI-SDE evaluated at any ordered set $\mathbf{x} = [x_1, \ldots, x_N] \in \mathbb{R}^N$ follows a discrete-time linear system:

$$\begin{aligned} \mathbf{s}(x_{n+1}) &= \mathbf{A}_{n,n+1}\mathbf{s}(x_n) + \mathbf{q}_n, & \mathbf{q}_n &\sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_{n,n+1}), \\ \mathbf{s}(x_0) &\sim \mathcal{N}(\mathbf{0}, \mathbf{P}_0), & f_n &= \mathbf{H}\mathbf{s}(x_n), \end{aligned} \tag{25}$$

where the state transition matrices, $\mathbf{A}_{n,n+1} \in \mathbb{R}^{d\times d}$, noise covariance matrices, $\mathbf{Q}_{n,n+1} \in \mathbb{R}^{d\times d}$, and stationary state covariance matrix $\mathbf{P}_0 \in \mathbb{R}^{d\times d}$ can be computed analytically. Denoting the matrix exponential as $\boldsymbol{\Phi}$ and with step size $\Delta_n = x_{n+1} - x_n$, we have

$$\begin{aligned} \mathbf{A}_{n,n+1} &= \boldsymbol{\Phi}(\mathbf{F}\Delta_n)\,, \\ \mathbf{Q}_{n,n+1} &= \textstyle\int_0^{\Delta_n} \boldsymbol{\Phi}(\Delta_n - \tau)\mathbf{L}\mathbf{Q}_c\mathbf{L}^\top\boldsymbol{\Phi}(\Delta_n - \tau)^\top \,\mathrm{d}\tau\,. \end{aligned} \tag{26}$$

A.2 Conditionals

This section is adapted from Appendix A.1 of Adam et al. (2020). We consider a stationary Markovian GP with state dimension d and denote by $(\mathbf{u}_m, \mathbf{s}, \mathbf{u}_{m+1})$ its evaluation on the triplet (z_m, x, z_{m+1}). We here detail the derivation of $p(\mathbf{s} \,|\, \mathbf{v} = [\mathbf{u}_m, \mathbf{u}_{m+1}])$.

Derivation from the joint precision

$$\begin{aligned} p(\mathbf{s} \,|\, \mathbf{u}_m, \mathbf{u}_{m+1}) &\propto p(\mathbf{s} \,|\, \mathbf{u}_m) p(\mathbf{u}_{m+1} \,|\, \mathbf{s}) \\ &\propto \mathcal{N}(\mathbf{s}; \mathbf{A}_{m,x}\mathbf{u}_m, \mathbf{Q}_{m,x})\mathcal{N}(\mathbf{u}_{m+1}; \mathbf{A}_{x,m+1}\mathbf{s}, \mathbf{Q}_{x,m+1}) \\ &\propto \exp -\frac{1}{2}\left[\|\mathbf{s} - \mathbf{A}_{m,x}\mathbf{u}_m\|^2_{\mathbf{Q}^{-1}_{m,x}} + \|\mathbf{u}_{m+1} - \mathbf{A}_{x,m+1}\mathbf{s}\|^2_{\mathbf{Q}^{-1}_{x,m+1}}\right] \\ &\propto \exp -\frac{1}{2}\big[\mathbf{s}^\top \underbrace{(\mathbf{Q}^{-1}_{m,x} + (\mathbf{A}_{x,m+1})^\top \mathbf{Q}^{-1}_{x,m+1}\mathbf{A}_{x,m+1})}_{\mathbf{T}^{-1}}\mathbf{s} - 2\mathbf{s}^\top \underbrace{\big[\mathbf{Q}^{-1}_{m,x}\mathbf{A}_{m,x}, \quad \mathbf{A}^\top_{x,m+1}\mathbf{Q}^{-1}_{x,m+1}\big]}_{\mathbf{M}=[\mathbf{M}_1,\mathbf{M}_2]}\mathbf{v}\big] \\ &\propto \exp -\frac{1}{2}\left[\mathbf{s}^\top\mathbf{T}^{-1}\mathbf{s} - 2\mathbf{s}^\top\mathbf{M}\mathbf{v}\right] = \mathcal{N}(\mathbf{s}; \mathbf{R}\mathbf{v}, \mathbf{T}) \end{aligned} \tag{27}$$

with

$$\begin{aligned} \mathbf{T} &= (\mathbf{Q}^{-1}_{m,x} + \mathbf{A}^\top_{x,m+1}\mathbf{Q}^{-1}_{x,m+1}\mathbf{A}_{x,m+1})^{-1} \text{ (Woodbury identity)} \\ &= \mathbf{Q}_{m,x} - \mathbf{Q}_{m,x}\mathbf{A}^\top_{x,m+1}(\mathbf{Q}_{x,m+1} + \mathbf{A}_{x,m+1}\mathbf{Q}_{m,x}\mathbf{A}^\top_{x,m+1})^{-1}\mathbf{A}_{x,m+1}\mathbf{Q}_{m,x} \\ &= \mathbf{Q}_{m,x} - \mathbf{Q}_{m,x}\mathbf{A}^\top_{x,m+1}\mathbf{Q}^{-1}_{m,m+1}\mathbf{A}_{x,m+1}\mathbf{Q}_{m,x} \end{aligned} \tag{28}$$

and $\mathbf{R} = [\mathbf{R}_1, \mathbf{R}_2] = \mathbf{TM} = [\mathbf{TM}_1, \mathbf{TM}_2]$ given by

$$\begin{aligned}
\mathbf{R}_1 &= (\mathbf{Q}_{m,x} - \mathbf{Q}_{m,x}\mathbf{A}_{x,m+1}^\top \mathbf{Q}_{m,m+1}^{-1}\mathbf{A}_{x,m+1}\mathbf{Q}_{m,x})\mathbf{Q}_{m,x}^{-1}\mathbf{A}_{m,x} \\
&= \mathbf{A}_{m,x} - \mathbf{Q}_{m,x}\mathbf{A}_{x,m+1}^\top \mathbf{Q}_{m,m+1}^{-1}\mathbf{A}_{m,m+1}, \qquad (29) \\
\mathbf{R}_2 &= (\mathbf{Q}_{m,x} - \mathbf{Q}_{m,x}\mathbf{A}_{x,m+1}^\top \mathbf{Q}_{m,m+1}^{-1}\mathbf{A}_{x,m+1}\mathbf{Q}_{m,x})\mathbf{A}_{x,m+1}^\top \mathbf{Q}_{x,m+1}^{-1} \\
&= \mathbf{Q}_{m,x}\mathbf{A}_{x,m+1}^\top \mathbf{Q}_{x,m+1}^{-1} - \mathbf{Q}_{m,x}\mathbf{A}_{x,m+1}^\top \mathbf{Q}_{m,m+1}^{-1}(\mathbf{Q}_{m,m+1} - \mathbf{Q}_{x,m+1})\mathbf{Q}_{x,m+1}^{-1} \text{ (Woodbury identity)} \\
&= \mathbf{Q}_{m,x}\mathbf{A}_{x,m+1}^\top \mathbf{Q}_{m,m+1}^{-1}. \qquad (30)
\end{aligned}$$

The conditional function evaluation $f(x) = \mathbf{Hs}$ is thus:

$$p(f(x) \,|\, \mathbf{u}_m, \mathbf{u}_{m+1}) = \mathcal{N}(f(x); \mathbf{HRv}, \mathbf{HTH}^\top) = \mathcal{N}(f(x); \mathbf{Wv}, \nu). \qquad (31)$$

B Inference in Site-based Sparse Markovian GP Models

The site based algorithms build an approximation to the posterior of the form:

$$q(\mathbf{s}(\cdot)) \propto p(\mathbf{u})\, p(\mathbf{s}(\cdot) \,|\, \mathbf{u}) \textstyle\prod_m t_m(\mathbf{v}_m). \qquad (32)$$

The factors t_m are called *sites* and are parameterized as unnormalized Gaussian distributions in the natural parameterization: $t_m(\mathbf{v}_m) = z_m \exp(\mathbf{v}_m^\top \mathbf{T}_{1,m} - {}^1\!/_2\, \mathbf{v}_m^\top \mathbf{T}_{2,m}\mathbf{v}_m) = \tilde{\mathcal{N}}(\mathbf{v}_m; z_m, \mathbf{T}_{1,m}, \mathbf{T}_{2,m})$.

B.1 Filtering and Smoothing

It is possible to compute the posterior marginals over the individual inducing states $q(\mathbf{u}_m)$ and pairwise consecutive inducing states $q(\mathbf{v}_m = [\mathbf{u}_m, \mathbf{u}_{m+1}])$ by introducing the forward (f) and backward (b) filters:

$$\begin{aligned}
q^f(\mathbf{u}_m) &\propto \textstyle\int p(\mathbf{u}_{\le m}) \prod_{m'<m} t_{m'}(\mathbf{v}_{m'})\, \mathrm{d}\mathbf{u}_{<m}, \\
q^b(\mathbf{u}_m) &\propto \textstyle\int p(\mathbf{u}_{\ge m}) \prod_{m'\ge m} t_{m'}(\mathbf{v}_{m'})\, \mathrm{d}\mathbf{u}_{>m}.
\end{aligned} \qquad (33)$$

These can be evaluated using the following recursions:

$$\begin{aligned}
q^f(\mathbf{u}_{m+1}) &= \textstyle\int p(\mathbf{u}_{\le m+1}) \prod_{m'<m+1} t_{m'}(\mathbf{v}_{m'})\, \mathrm{d}\mathbf{u}_{<m+1} \\
&= \textstyle\int p(\mathbf{u}_{\le m}) \prod_{m'<m} t_{m'}(\mathbf{v}_{m'}) \int p(\mathbf{u}_{m+1} \,|\, \mathbf{u}_m) t_m(\mathbf{v}_m)\, \mathrm{d}\mathbf{u}_{<m+1} \\
&= \textstyle\int \big[\int p(\mathbf{u}_{\le m}) \prod_{m'<m} t_{m'}(\mathbf{v}_{m'})\, \mathrm{d}\mathbf{u}_{<m}\big] p(\mathbf{u}_{m+1} \,|\, \mathbf{u}_m) t_m(\mathbf{v}_m)\, \mathrm{d}\mathbf{u}_m \\
&= \textstyle\int q^f(\mathbf{u}_m)\, p(\mathbf{u}_{m+1} \,|\, \mathbf{u}_m)\, t_m(\mathbf{v}_m)\, \mathrm{d}\mathbf{u}_m, \\
q^b(\mathbf{u}_m) &= \textstyle\int p(\mathbf{u}_{\ge m}) \prod_{m'\ge m} t_{m'}(\mathbf{v}_{m'})\, \mathrm{d}\mathbf{u}_{>m}. \\
&= \textstyle\int \big[\int p(\mathbf{u}_{\ge m+1}) \prod_{m'\ge m+1} t_{m'}(\mathbf{v}_{m'})\, \mathrm{d}\mathbf{u}_{>m+1}\big] p(\mathbf{u}_m \,|\, \mathbf{u}_{m+1}) t_m(\mathbf{v}_m)\, \mathrm{d}\mathbf{u}_{m+1}. \\
&= \textstyle\int q^b(\mathbf{u}_{m+1})\, p(\mathbf{u}_m \,|\, \mathbf{u}_{m+1})\, t_m(\mathbf{v}_m)\, \mathrm{d}\mathbf{u}_{m+1} \\
&= \textstyle\int q^b(\mathbf{u}_{m+1})\, p(\mathbf{u}_{m+1} \,|\, \mathbf{u}_m)\, p(\mathbf{u}_m) \,/\, p(\mathbf{u}_{m+1})\, t_m(\mathbf{v}_m)\, \mathrm{d}\mathbf{u}_{m+1}.
\end{aligned} \qquad (34)$$

The desired marginals are then obtained as the product of the forward and backward filtering distributions, divided by the prior:

$$\begin{aligned}
q^s(\mathbf{u}_m) &= \textstyle\int q(\mathbf{u})\, \mathrm{d}\mathbf{u}_{\neq m} \\
&= \textstyle\int p(\mathbf{u}) \prod_{m'} t_{m'}(\mathbf{v}_{m'})\, \mathrm{d}\mathbf{u}_{\neq m} \\
&= \textstyle\int \big[p(\mathbf{u}_{\le m}) \prod_{m'<m} t_{m'}(\mathbf{v}_{m'})\big] \big[p(\mathbf{u}_{>m} \,|\, \mathbf{u}_m) \prod_{m'\ge m} t_{m'}(\mathbf{v}_{m'})\big]\, \mathrm{d}\mathbf{u}_{\neq m} \\
&= \big[p(\mathbf{u}_{\le m}) \textstyle\int \prod_{m'<m} t_{m'}(\mathbf{v}_{m'})\, \mathrm{d}\mathbf{u}_{<m}\big] \big[\textstyle\int p(\mathbf{u}_{\ge m}) \prod_{m'\ge m} t_{m'}(\mathbf{v}_{m'})\, \mathrm{d}\mathbf{u}_{>m}\big] / p(\mathbf{u}_m) \\
&= q^f(\mathbf{u}_m)\, q^b(\mathbf{u}_m) \,/\, p(\mathbf{u}_m), \\
q^s(\mathbf{v}_m) &= \textstyle\int q(\mathbf{u})\, \mathrm{d}\mathbf{u}_{\neq(m,m+1)} \\
&= \textstyle\int p(\mathbf{u}) \prod_{m'} t_{m'}(\mathbf{v}_{m'})\, \mathrm{d}\mathbf{u}_{\neq(m,m+1)} \\
&= \textstyle\int \big[p(\mathbf{u}_{\le m}) \prod_{m'<m} t_{m'}(\mathbf{v}_{m'})\big] t_m(\mathbf{v}_m) p(\mathbf{u}_{m+1} \,|\, \mathbf{u}_m) \big[p(\mathbf{u}_{>m+1} \,|\, \mathbf{u}_{m+1}) \prod_{m'\ge m+1} t_{m'}(\mathbf{v}_{m'})\big]\, \mathrm{d}\mathbf{u}_{\neq(m,m+1)} \\
&= \big[\textstyle\int p(\mathbf{u}_{\le m}) \prod_{m'<m} t_{m'}(\mathbf{v}_{m'})\, \mathrm{d}\mathbf{u}_{<m}\big] t_m(\mathbf{v}_m) p(\mathbf{u}_{m+1} \,|\, \mathbf{u}_m) / p(\mathbf{u}_{m+1}) \big[\textstyle\int p(\mathbf{u}_{\ge m+1}) \prod_{m'\ge m+1} t_{m'}(\mathbf{v}_{m'})\, \mathrm{d}\mathbf{u}_{>m+1}\big] \\
&= q^f(\mathbf{u}_m)\, p(\mathbf{u}_{m+1} \,|\, \mathbf{u}_m)\, t_m(\mathbf{v}_m) \,/\, p(\mathbf{u}_{m+1})\, q^b(\mathbf{u}_{m+1}).
\end{aligned} \qquad (35)$$

Kalman recursions The above product of forward and backward filters is known as *two-filter smoothing* (Särkkä, 2013). An alternative way to implement this is via the more standard Kalman filter (f) and Rauch-Tung-Striebel (RTS) smoother (s). Letting $\mathbf{u}_0 = \mathbf{s}(-\infty)$ and $\mathbf{u}_{M+1} = \mathbf{s}(\infty)$,

$$\begin{aligned}
q^f(\mathbf{u}_0) &= \mathcal{N}(\mathbf{u}_0 \,|\, \mathbf{0}, \mathbf{P}_0), && \textit{initialise Kalman filter} \\
q^f(\mathbf{v}_m) &= q^f(\mathbf{u}_m)\, p(\mathbf{u}_{m+1} \,|\, \mathbf{u}_m)\, t_m(\mathbf{v}_m), && \textit{compute joint, include site} \\
q^f(\mathbf{u}_{m+1}) &= \textstyle\int q^f(\mathbf{v}_m)\, \mathrm{d}\mathbf{u}_m, && \textit{marginalise (filtering dist.)} \\
q^s(\mathbf{u}_M) &= \textstyle\int q^f(\mathbf{v}_M)\, \mathrm{d}\mathbf{u}_{M+1}, && \textit{initialise RTS smoother} \\
q^p(\mathbf{u}_{m+1}) &= \textstyle\int q^f(\mathbf{u}_m) p(\mathbf{u}_{m+1} \,|\, \mathbf{u}_m)\, \mathrm{d}\mathbf{u}_m, && \textit{forward prediction} \\
q^s(\mathbf{u}_m) &= q^f(\mathbf{u}_m) \int \frac{p(\mathbf{u}_{m+1} \,|\, \mathbf{u}_m)\, q^s(\mathbf{u}_{m+1})}{q^p(\mathbf{u}_{m+1})}\, \mathrm{d}\mathbf{u}_{m+1}, && \textit{smoothing dist.}
\end{aligned} \tag{36}$$

where $q^p(\cdot)$ is the forward filter prediction and $q^s(\cdot)$ is the desired smoothing distribution, *i.e.*, the marginal posterior.

To derive the last line in Eq. (36) we let $\tilde{\mathbf{y}}$ represent *pseudo data* implied by the sites, $p(\tilde{\mathbf{y}}_m \,|\, \mathbf{v}_m) = t_m(\mathbf{v}_m)$. With this notation the forward filter is given by $q^f(\mathbf{u}_m) = p(\mathbf{u}_m \,|\, \tilde{\mathbf{y}}_{1:m}) \approx p(\mathbf{u}_m \,|\, \mathbf{y}_{1:n(m)})$, where $n(m)$ is the number of data points to the left of z_m, and the smoother by $q^s(\mathbf{u}_m) = p(\mathbf{u}_m \,|\, \tilde{\mathbf{y}}_{1:M}) \approx p(\mathbf{u}_m \,|\, \mathbf{y}_{1:N})$, so we can write,

$$\begin{aligned}
q^s(\mathbf{u}_m) &= p(\mathbf{u}_m \,|\, \tilde{\mathbf{y}}_{1:M}) \\
&= \int p(\mathbf{u}_m, \mathbf{u}_{m+1} \,|\, \tilde{\mathbf{y}}_{1:M})\, \mathrm{d}\mathbf{u}_{m+1} \\
&= \int p(\mathbf{u}_m \,|\, \mathbf{u}_{m+1}, \tilde{\mathbf{y}}_{1:M})\, p(\mathbf{u}_{m+1} \,|\, \tilde{\mathbf{y}}_{1:M})\, \mathrm{d}\mathbf{u}_{m+1} \\
&= \int p(\mathbf{u}_m \,|\, \mathbf{u}_{m+1}, \tilde{\mathbf{y}}_{1:m})\, q^s(\mathbf{u}_{m+1})\, \mathrm{d}\mathbf{u}_{m+1} \\
&= \int \frac{p(\mathbf{u}_m, \mathbf{u}_{m+1} \,|\, \tilde{\mathbf{y}}_{1:m})}{p(\mathbf{u}_{m+1} \,|\, \tilde{\mathbf{y}}_{1:m})}\, q^s(\mathbf{u}_{m+1})\, \mathrm{d}\mathbf{u}_{m+1} \\
&= q^f(\mathbf{u}_m) \int \frac{p(\mathbf{u}_{m+1} \,|\, \mathbf{u}_m)\, q^s(\mathbf{u}_{m+1})}{q^p(\mathbf{u}_{m+1})}\, \mathrm{d}\mathbf{u}_{m+1}.
\end{aligned} \tag{37}$$

B.2 Normaliser

We are interested in the normalizer of $q(\mathbf{s})$. A dense formulation can be obtained as follows:

$$\begin{aligned}
\log \textstyle\int q(\mathbf{s}(.))\, \mathrm{d}\mathbf{s} &= \log \textstyle\int p(\mathbf{s}(.) \,|\, \mathbf{u})\, p(\mathbf{u}) \prod_m t(\mathbf{v}_m)\, \mathrm{d}\mathbf{s}\, \mathrm{d}\mathbf{u} \\
&= \log \textstyle\int p(\mathbf{u}) \prod_m t(\mathbf{v}_m)\, \mathrm{d}\mathbf{u} \\
&= \log \textstyle\int \frac{e^{G(p(\mathbf{u}))} \prod_m z_m}{e^{G(q(\mathbf{u}))}} q(\mathbf{u})\, \mathrm{d}\mathbf{u} \\
&= G(q(\mathbf{u})) - G(p(\mathbf{u})) + \textstyle\sum_m \log\, z_m,
\end{aligned} \tag{38}$$

where we have defined the log-normaliser as the functional $G(\tilde{\mathcal{N}}(\mathbf{u}; z, \mathbf{T}_1, \mathbf{T}_2)) = \log \int \tilde{\mathcal{N}}(\mathbf{u}; z, \mathbf{T}_1, \mathbf{T}_2)\, \mathrm{d}\mathbf{u}$.

A more efficient formulation dedicated to Markovian GPs is obtained using the filtering recursions of the previous section:

$$\begin{aligned}
\textstyle\int q(\mathbf{u})\, \mathrm{d}\mathbf{u} &= \textstyle\int p(\mathbf{u}) \prod_m t_m(\mathbf{v}_m)\, \mathrm{d}\mathbf{u} \\
&= \textstyle\int p(\mathbf{u}_{m>1} \,|\, \mathbf{u}_1) \prod_{m>0} t_m(\mathbf{v}_m) \underbrace{\left[\int p(\mathbf{u}_1 \,|\, \mathbf{u}_0)\, t_0(\mathbf{v}_0)\, p(\mathbf{v}_0)\, \mathrm{d}\mathbf{u}_0\right]}_{c_1^f\, q^f(\mathbf{u}_1)} \mathrm{d}\mathbf{u}_{>0} \\
&= \textstyle\int p(\mathbf{u}_{m>2} \,|\, \mathbf{u}_2) \prod_{m>1} t_m(\mathbf{v}_m) c_1^f \underbrace{\left[\int p(\mathbf{u}_2 \,|\, \mathbf{u}_1)\, t_1(\mathbf{v}_1)\, q^f(\mathbf{u}_1)\, \mathrm{d}\mathbf{u}_1\right]}_{c_2^f\, q^f(\mathbf{u}_2)} \mathrm{d}\mathbf{u}_{>1} \\
&= \cdots = c_1^f \ldots c_{M-1}^f \textstyle\int q^f(\mathbf{u}_M)\, \mathrm{d}\mathbf{u}_M = c_1^f \ldots c_M^f.
\end{aligned} \tag{39}$$

The terms c_m^f are the normalisers computed during the forward filtering recursions described in the previous section. The normaliser can equivalently be computed using the backward filter:

$$\begin{aligned}\int q(\mathbf{u})\,\mathrm{d}\mathbf{u} &= \int p(\mathbf{u})\,\textstyle\prod_m t_m(\mathbf{v}_m)\,\mathrm{d}\mathbf{u} \\ &= \int q^b(\mathbf{u}_0)\,\mathrm{d}\mathbf{u}_0\, c_0^b \dots c_M^b = c_0^b \dots c_M^b. \end{aligned} \tag{40}$$

Finally, the normaliser can also be computed using both the forward and backward filters, meeting at site indexed m:

$$\begin{aligned}\int q(\mathbf{u})\,\mathrm{d}\mathbf{u} &= \int p(\mathbf{u})\,\textstyle\prod_m t_m(\mathbf{v}_m)\,\mathrm{d}\mathbf{u} \\ &= c_1^f \dots c_m^f \int q^f(\mathbf{u}_m)\, p(\mathbf{u}_{m+1} \,|\, \mathbf{u}_m)\;\frac{q^b(\mathbf{u}_{m+1})}{p(\mathbf{u}_{m+1})} t_m(\mathbf{v}_m)\,\mathrm{d}\mathbf{v}_m\, c_{m+1}^b \dots c_M^b. \end{aligned} \tag{41}$$

This latter expression is useful when one needs to compute the normaliser of a site-based approximation after a single site update, as is the case in EP.

C Algorithms

C.1 S^{2}VGP Algorithm

The approximate posterior process is parametrized as

$$\begin{aligned} q(\mathbf{s}(\cdot)) &= p(\mathbf{s}(\cdot) \,|\, \mathbf{u})\, q(\mathbf{u}) \\ &\propto p(\mathbf{s}(\cdot) \,|\, \mathbf{u}) q(\mathbf{u}_0) \textstyle\prod_{m=1}^M q_m(\mathbf{v}_{m+1} \,|\, \mathbf{u}_m). \end{aligned} \tag{42}$$

The variational lower bound to the marginal evidence is:

$$\mathcal{L}(q) = \mathbb{E}_q \log p(\mathbf{y} \,|\, f) - \mathrm{KL}[q(\mathbf{u}) \,\|\, p(\mathbf{u})]. \tag{43}$$

The KL divergence is between two linear Gaussian state space models and thus decomposes as:

$$\mathrm{KL}[q(\mathbf{u}) \,\|\, p(\mathbf{u})] = \mathrm{KL}[q(\mathbf{u}_1) \,\|\, p(\mathbf{u}_1)] + \textstyle\sum_{m=1}^M \mathrm{KL}\left[q(\mathbf{u}_{m+1} \,|\, \mathbf{u}_m) \,\|\, p(\mathbf{u}_{m+1} \,|\, \mathbf{u}_m)\right]. \tag{44}$$

Due to the locality of the conditional $f \,|\, \mathbf{u}$, the variational expectation for a data point at x such that $z_m \le x < z_{m+1}$ is:

$$\begin{aligned} q(f(x)) &= \int p(f(x) \,|\, \mathbf{v}_m = [\mathbf{u}_m, \mathbf{u}_{m+1}])\, q(\mathbf{v}_m)\,\mathrm{d}\mathbf{v}_m \\ &= \int \mathcal{N}(f(x) \,|\, \mathbf{W}\mathbf{v}, \nu)\, \mathcal{N}(\mathbf{v}_m \,|\, \boldsymbol{\mu}_{\mathbf{v}_m}, \boldsymbol{\Sigma}_{\mathbf{v}_m})\,\mathrm{d}\mathbf{v}_m \\ &= \mathcal{N}(f(x) \,|\, \mathbf{W}\boldsymbol{\mu}_{\mathbf{v}_m}, \mathbf{W}\boldsymbol{\Sigma}_{\mathbf{v}_m}\mathbf{W}^\top + \nu), \end{aligned} \tag{45}$$

where $q(\mathbf{v}_m)$ is a pairwise posterior marginal over the consecutive inducing states $[\mathbf{u}_m, \mathbf{u}_{m+1}]$, which can be evaluated with linear time complexity in M, using classic Kalman smoothing algorithms (see App. B.1).

C.2 S^{2}CVI Algorithm

We follow the derivation of Khan and Lin (2017). The approximate posterior process is parametrized using shared sites:

$$\begin{aligned} q(\mathbf{s}(\cdot)) &= p(\mathbf{s}(\cdot) \,|\, \mathbf{u}) q(\mathbf{u}) \\ &\propto p(\mathbf{s}(\cdot) \,|\, \mathbf{u}) p(\mathbf{u}) \prod_n t_m(\mathbf{v}_m). \end{aligned} \tag{46}$$

This is the same structure as for the S^{2}PEP algorithm, but here, since we approximate the posterior as a Gaussian, the normaliser of the sites are irrelevant.

The approximate posterior is optimized to get close to the true posterior in the sense of the KL divergence $\mathrm{KL}[q(\mathbf{s}(\cdot)) \,\|\, p(\mathbf{s}(\cdot) \,|\, \mathbf{y})]$, or equivalently by maximizing the variational objective:

$$\mathcal{L}(q) = \mathbb{E}_q \log p(\mathbf{y} \,|\, f) - \mathrm{KL}[q(\mathbf{u}) \,\|\, p(\mathbf{u})]. \tag{47}$$

The joint model is split into a conjugate and a non-conjugate part:

$$p(\mathbf{f}, \mathbf{u}, \mathbf{y}) = \underbrace{p(\mathbf{u})}_{p_c(\mathbf{u})} \underbrace{p(\mathbf{f} \,|\, \mathbf{u})\, p(\mathbf{y} \,|\, \mathbf{f})}_{p_{nc}(\mathbf{f}, \mathbf{u})}. \tag{48}$$

The conjugate part has *sparse* minimal sufficient statistics $\phi(\mathbf{u}) = [(\mathbf{u}_k, \mathbf{u}_k \mathbf{u}_k^\top)_{k=1}^M, (\mathbf{u}_{k+1} \mathbf{u}_k^\top)_{k=1}^{M-1}]$, with the bilinear terms corresponding to the block-tridiagonal entries of matrix $\mathbf{u}\mathbf{u}^\top$ which we note $\text{btd}[\mathbf{u}\mathbf{u}^T]$. We denote by $\mathbf{\Lambda}$ the natural parameters of the prior $p(\mathbf{u})$ associated to sufficient statistics $\phi(\mathbf{u})$.

CVI approximates the non-conjugate part using Gaussian sites with the same sufficient statistics as the conjugate part: $\tilde{p}_{nc}(\mathbf{u}) \approx p(\mathbf{f} \,|\, \mathbf{u}) t(\mathbf{u})$, where $t(\mathbf{u}) = \prod_{m=1}^M t_m(\mathbf{v}_m)$. Each site t_m has natural parameter $\boldsymbol{\lambda}^{(m)}$ associated to local minimal sufficient statistics $\phi_m(\mathbf{u}) = [\mathbf{v}_m, \mathbf{v}_m \mathbf{v}_m^\top] \subset \phi(\mathbf{u})$. We denote by $\mathcal{P}_m$ the linear operator projecting these minimal natural parameter into natural parameter with 'full' sufficient statistics $\phi(\mathbf{u})$ and setting the rest of the natural parameters to 0. We denote by $\boldsymbol{\lambda}$ the projected natural parameters of the sites $\prod_m t(\mathbf{v}_m)$, *i.e.*, $\boldsymbol{\lambda} = \sum_m \mathcal{P}_m(\boldsymbol{\lambda}^{(m)})$. The natural parameters of the posterior over $\mathbf{u}$ are thus $\mathbf{\Lambda} + \boldsymbol{\lambda}$.

One can show that a natural gradient step on the variational parameters $\boldsymbol{\lambda}^{(m)}$ boils down to (Khan and Lin, 2017):

$$\begin{aligned} \mathbf{g}^{(m)} &= \nabla_{\boldsymbol{\mu}^{(m)}} \mathbb{E}_{q(\mathbf{f}^{(m)})} \log p(\mathbf{y}^{(m)} \,|\, \mathbf{f}^{(m)}) \\ \boldsymbol{\lambda}_{k+1}^{(m)} &= (1 - \rho)\, \boldsymbol{\lambda}_k^{(m)} + \rho\, \mathbf{g}^{(m)}, \end{aligned} \tag{49}$$

where $\mathbf{f}^m$ and $\mathbf{y}^{(m)}$ are here the subset of the data where the input x falls in $[z_m, z_{m+1}]$, and $\boldsymbol{\mu}^{(m)}$ are the expectation parameters of the posterior $q(\mathbf{u}^{(m)})$.

These updates to the parameters can be written in terms of the derivatives of the variational expectations with respect to the mean and variance of the posterior marginal via the chain rule,

$$\begin{aligned} \mathbf{g}_2^{(m)} &= \sum_{n \in \mathcal{M}} \mathbf{W}_n^\top \frac{\partial \mathcal{L}_n}{\partial \Sigma_n} \mathbf{W}_n, \\ \mathbf{g}_1^{(m)} &= \sum_{n \in \mathcal{M}} \mathbf{W}_n^\top \frac{\partial \mathcal{L}_n}{\partial \mu_n} - 2 \mathbf{W}_n^\top \frac{\partial \mathcal{L}_n}{\partial \Sigma_n} \mathbf{W}_n \boldsymbol{\mu}_{m(n)}, \end{aligned} \tag{50}$$

where $\mathcal{L}_n = \mathbb{E}_{q(\mathbf{f}^{(m)})} \log p(\mathbf{y}^{(m)} \,|\, \mathbf{f}^{(m)})$ and $\mu_n = \mathbf{W}_n \boldsymbol{\mu}_{m(n)}$, $\Sigma_n = \mathbf{W}_n \mathbf{\Sigma}_{m(n)} \mathbf{W}_n^\top + \nu_n$, and where $\mathcal{M}$ represents the indices to the data points whose inputs fall in $[z_m, z_{m+1}]$.

C.2.1 S^{2}CVI ELBO

Although the CVI method sidesteps direct computation of the ELBO for the variational parameter updates, it can still be used for hyperparameter learning. As in Adam et al. (2020), the ELBO is given by:

$$\mathcal{L} = \mathbb{E}_{q(\mathbf{s})} \log p(\mathbf{y} \,|\, \mathbf{s}) - \text{KL}\left[q(\mathbf{u}) \,\|\, p(\mathbf{u})\right] \tag{51}$$

In S^{2}CVI, we are interested in the normalized posterior, *i.e.*, $q(\mathbf{u}) = \mathcal{Z}^{-1} p(\mathbf{u}) \prod_m t_m(\mathbf{v}_m)$, where $\mathcal{Z} = \int p(\mathbf{u}) \prod_m t_m(\mathbf{v}_m)\, \mathrm{d}\mathbf{u}$ is the normalizer (*i.e.*, the marginal likelihood of the approximate conjugate model) and can be computed as shown in App. B.2. The KL term in the ELBO is:

$$\begin{aligned} \text{KL}\left[q(\mathbf{u}) \,\|\, p(\mathbf{u})\right] &= \text{KL}\left[\mathcal{Z}^{-1} p(\mathbf{u}) \textstyle\prod_m t_m(\mathbf{v}_m) \,\|\, p(\mathbf{u})\right] \\ &= -\log \mathcal{Z} + \textstyle\sum_m \mathbb{E}_{q(\mathbf{v}_m)} \log t_m(\mathbf{v}_m). \end{aligned} \tag{52}$$

So the ELBO is:

$$\mathcal{L} = \mathbb{E}_{q(\mathbf{s})} \log p(\mathbf{y} \,|\, \mathbf{s}) + \log \mathcal{Z} - \sum_{m=1}^M \mathbb{E}_{q(\mathbf{v}_m)} \log t_m(\mathbf{v}_m). \tag{53}$$

C.3 S^{2}PEP Algorithm

We follow the notation of Bui et al. (2017) in their derivation of the sparse PEP algorithm. There are two differences in our derivation: the latent process is an SDE, and the sites are inherently local due to the Markovian property of the model. The starting point is a joint model of the data $\mathbf{y}$ and the process prior $\mathbf{s}(\cdot)$:

$$p(\mathbf{s}(\cdot), \mathbf{y} \mid \theta) = p(\mathbf{s}(\cdot)) \prod_{n=1}^{N} p(y_n \mid f_n, \theta). \tag{54}$$

In this setting, sparse EP consists of singling out a set of inducing inputs $\mathbf{z} = (z_1, \ldots, z_M) \in \mathbb{R}^M$ and using the associated inducing states $\mathbf{u} = \mathbf{s}(\mathbf{z}) \in \mathbb{R}^{M \times d}$ to parametrize an approximation to this joint distribution of the form:

$$p(\mathbf{s}(\cdot), \mathbf{y} \mid \theta) \approx p(\mathbf{s}(\cdot) \mid \mathbf{u})\, p(\mathbf{u}) \prod_{n=1}^{N} t_n(\mathbf{u}) = q(\mathbf{s}(\cdot)), \tag{55}$$

where we denote $q(\mathbf{s}(\cdot))$ to be the approximate *joint*, which differs from the other algorithms we present. The factors t_n are called *sites* and are parameterized as unnormalized Gaussian distributions in the natural parameterization: $t_n(\mathbf{u}) = z_n \exp(\mathbf{u}^\top \mathbf{T}_{1,n} - {}^1\!/_2\, \mathbf{u}^\top \mathbf{T}_{2,n} \mathbf{u}) = \tilde{\mathcal{N}}(\mathbf{u}; z_n, \mathbf{T}_{1,n}, \mathbf{T}_{2,n})$.

When there is one site per data point, the optimal form of the site is rank one: $t_n(\mathbf{u}) = \tilde{\mathcal{N}}(\mathbf{W}_n \mathbf{u}; z_n, T_{1,n}, T_{2,n})$, where $\mathbf{W}_n$ is the projection is the prior conditional mean $\mathbb{E}_p[f_n \mid \mathbf{u}] = \mathbf{W}_n \mathbf{u}$, and $z_n, T_{1,n}, T_{2,n}$ are scalars.

When working with Markovian GPs, the optimal site for data point n can be shown to depend on the subset of inducing variables consisting of the two nearest inducing states $\mathbf{v}_m(n) = [\mathbf{u}_{m(n)}, \mathbf{u}_{m(n)+1}]$, where $m(n)$ is such that $z_{m(n)} \le x_n < z_{m(n)+1}$. So the final parameterization is $t_n(\mathbf{v}_{m(n)}) = \tilde{\mathcal{N}}(\mathbf{W}_n \mathbf{v}_{m(n)}; z_n, T_{1,n}, T_{2,n})$, where $\mathbf{W}_n$ is the *sparse* projection is the prior conditional mean $\mathbb{E}_p[f_n \mid \mathbf{u}] = \mathbb{E}_p[f_n \mid \mathbf{v}_{m(n)}] = \mathbf{W}_n \mathbf{v}_{m(n)}$.

Noting that all the N_m data points whose input falls in $[z_m, z_{m+1}]$ have sites over $\mathbf{v}_m$ makes those sites natural candidates to be *locally tied* together: for each segment $[z_m, z_{m+1}]$, we replace each of the rank one sites $\{t_n(\mathbf{v}_m); x_n \in [z_m, z_{m+1}]\}$ by a fraction of a full rank site $t_m(\mathbf{v}_m)^{1/N_m}$. Our approximation to the join thus becomes:

$$q(\mathbf{s}(\cdot)) = p(\mathbf{s}(\cdot) \mid \mathbf{u})\, p(\mathbf{u}) \prod_{n=1}^{N} t_{m(n)}(\mathbf{v}_{m(n)})^{1/N_{m(n)}} = p(\mathbf{s}(\cdot) \mid \mathbf{u})\, p(\mathbf{u}) \prod_{m=0}^{M+1} t_m(\mathbf{v}_m). \tag{56}$$

Given the above parametrisation, the S^{2}PEP algorithm involves three main steps: the cavity computation ('deletion'), moment matching ('projection'), and finally the update to the site parameters.

C.3.1 Updates

The three steps of the algorithm to update the sites are:

1. **Deletion**: for a data point n, compute a cavity (which is an unnormalized Gaussian) by removing a fraction $k = {}^\alpha\!/_{N_{m(n)}}$ of a factor from the approximate joint $q(\mathbf{s}(\cdot))$:

   $$q^{\setminus n}(\mathbf{s}(\cdot)) = \frac{q(s(\cdot))}{t^k_{m(n)}(\mathbf{v}_{m(n)})}. \tag{57}$$

   This fraction k can be understood as first picking the fraction of the shared site attributed to a data point ($1/N_{m(n)}$ where $N_{m(n)}$ is the number of sites tied together locally), then updating only a fraction α of this fraction.

2. **Projection**: The new site is computed in the context of the other sites through the cavity, by minimizing the unnormalized KL divergence between the *tilted* distribution $q^{\setminus n}(\mathbf{s}(\cdot))\, p^\alpha(y_n \mid f_n)$ and the full approximate joint. Minimising the KL directly gives the new approximate joint $q^*(\mathbf{s}(\cdot))$ as,

   $$q^*(\mathbf{s}(\cdot)) \;\leftarrow\; \operatorname*{arg\,min}_{q(\mathbf{s}(\cdot)) \in \mathcal{Q}} \overline{\mathrm{KL}} \left[q^{\setminus n}(\mathbf{s}(\cdot))\, p^\alpha(y_n \mid f_n) \,\middle\|\, q(\mathbf{s}(\cdot)) \right]. \tag{58}$$

   Here, $\mathcal{Q}$ is the set of acceptable distributions and corresponds to $\{q^{\setminus n}(\mathbf{s}(\cdot))\, t^k(\mathbf{v}_{m(n)}); \forall t\}$, in other words, the optimization only changes the site that has been removed to build the cavity. One can show that

$q^*(\mathbf{v}_{m(n)}) = \mathcal{N}(\mathbf{v}_{m(n)} \,|\, \boldsymbol{\mu}^*_{m(n)}, \boldsymbol{\Sigma}^*_{m(n)})$, where

$$\begin{aligned}
\log Z_n &= \log \mathbb{E}_{q^{\setminus n}}[p^\alpha(y_n \,|\, f_n)], \\
\boldsymbol{\mu}^*_{m(n)} &= \boldsymbol{\mu}_{m(n)} + \mathbf{W}_{m(n)} \frac{\mathrm{d} \log Z_n}{\mathrm{d}\mu_n}, \\
\boldsymbol{\Sigma}^*_{m(n)} &= \boldsymbol{\Sigma}_{m(n)} + \mathbf{W}_{m(n)} \frac{\mathrm{d}^2 \log Z_n}{\mathrm{d}\mu_n^2} \mathbf{W}^\top_{m(n)}.
\end{aligned} \tag{59}$$

3. **Update**: Compute a new fraction of the approximate factor by dividing the new approximate joint by the cavity $t^k_{m(n),new}(\mathbf{v}_{m(n)}) = q^*(\mathbf{v}_{m(n)})/q^{\setminus n}(\mathbf{v}_{m(n)})$ which is a rank one site. This fraction is then incorporated back to obtain the new site: $t^*_{m(n)}(\mathbf{v}_{m(n)}) = t^{1-k}_{m(n),old}(\mathbf{v}_{m(n)}) t^k_{m(n),new}(\mathbf{v}_{m(n)})$.
   The normaliser is then updated by matching the integral of the two terms in the KL divergence:

$$\begin{aligned}
\log \textstyle\int q^{\setminus n}(\mathbf{s}(\cdot)) p^\alpha(y_n \,|\, f_n)\, \mathrm{d}\mathbf{s}(\cdot) &= \log \textstyle\int q^{\setminus n}(\mathbf{s}(\cdot)) t^k_{m(n),new}(\mathbf{v}_{m(n)})\, \mathrm{d}\mathbf{s}(\cdot) \\
\log Z_n &= \log \textstyle\int q^{\setminus n}(\mathbf{u}) t^k_{m(n),new}(\mathbf{v}_{m(n)})\, \mathrm{d}\mathbf{u} \\
&= G(q^*(\mathbf{u})) - G(q^{\setminus n}(\mathbf{u})) + k \log z_{m(n),new}, \\
\log z_{m(n),new} &= \frac{1}{k} \left(\log Z_n - G(q^*(\mathbf{u})) + G(q^{\setminus n}(\mathbf{u})) \right).
\end{aligned}$$

   So the new site normaliser is $\log z^*_{m(n)} = (1-k) \log z_{m(n),old} + k \log z_{m(n),new}$. The normalizer can be computed efficiently using the recursions described in App. B.2

C.3.2 S^{2}PEP Energy

Following the approach of Bui et al. (2017), the PEP energy is defined as the marginal likelihood of the approximate joint:

$$\begin{aligned}
\log \mathcal{Z}_{\mathrm{PEP}} &= \log \textstyle\int q(\mathbf{s}(.))\, \mathrm{d}\mathbf{s} \\
&= \log \textstyle\int p(\mathbf{u})\, p(\mathbf{s}(.) \,|\, \mathbf{u}) \prod_m t(\mathbf{v}_m)\, \mathrm{d}\mathbf{s}\, \mathrm{d}\mathbf{u} \\
&= \log \textstyle\int p(\mathbf{u}) \prod_m t(\mathbf{v}_m)\, \mathrm{d}\mathbf{u} \\
&= \log \int \frac{e^{G(p(\mathbf{u}))} \prod_m z_m}{e^{G(q(\mathbf{u}))}} q(\mathbf{u})\, \mathrm{d}\mathbf{u} \\
&= G(q(\mathbf{u})) - G(p(\mathbf{u})) + \textstyle\sum_m \log\, z_m,
\end{aligned} \tag{60}$$

This normalizer can be implemented efficiently as described in App. B.2. It depends on the sites normalizer z_m which themselves depend on the model hyper-parameters through the site update equations. The energy function thus provides an objective to perform parameter optimization, as a proxy to the marginal likelihood $p(\mathbf{y})$.

We provide an alternative derivation of the same energy which is arguably easier to implement, and highlights the connection to the S^{2}CVI ELBO. Recall that $t(\mathbf{v}_m) = \tilde{\mathcal{N}}(\mathbf{u}; z_m, \mathbf{T}_{1,m}, \mathbf{T}_{2,m}) = z_m \mathcal{N}(\mathbf{u} \,|\, \mathbf{T}_{1,m}, \mathbf{T}_{2,m})$, where $\mathbf{T}_{1,m}$, $\mathbf{T}_{2,m}$ are the natural parameters, then

$$\begin{aligned}
\log \mathcal{Z}_{\mathrm{PEP}} &= \log \textstyle\int p(\mathbf{u}) \prod_m t(\mathbf{v}_m)\, \mathrm{d}\mathbf{u} \\
&= \log \int p(\mathbf{u}) \textstyle\prod_m z_m \mathcal{N}(\mathbf{v}_m \,|\, \mathbf{T}_{1,m}, \mathbf{T}_{2,m})\ \mathrm{d}\mathbf{u} \\
&= \log \textstyle\prod_m z_m \int p(\mathbf{u}) \prod_m \mathcal{N}(\mathbf{v}_m \,|\, \mathbf{T}_{1,m}, \mathbf{T}_{2,m})\ \mathrm{d}\mathbf{u} \\
&= \log \textstyle\prod_m z_m + \log \int p(\mathbf{u}) \prod_m \mathcal{N}(\mathbf{v}_m \,|\, \mathbf{T}_{1,m}, \mathbf{T}_{2,m})\ \mathrm{d}\mathbf{u} \\
&= \textstyle\sum_m \log z_m + \log \mathcal{Z},
\end{aligned} \tag{61}$$

where $\log \mathcal{Z} = \log \int p(\mathbf{u}) \prod_m \mathcal{N}(\mathbf{v}_m \,|\, \mathbf{T}_{1,m}, \mathbf{T}_{2,m})\ \mathrm{d}\mathbf{u}$ is the normaliser of the approximate model, and can be computed in closed form via the Kalman filter as shown in App. B.2, or using the method in App. C.4, replacing the true likelihood with $\mathcal{N}(\mathbf{v}_m \,|\, \mathbf{T}_{1,m}, \mathbf{T}_{2,m})$.

To compute z_m, the idea is to reuse the cavity computation, and to match the zero-th moment of the tilted distribution in the same way as we do for the first and second moments during inference. Let $\mathcal{Z}_{\mathrm{lik},m} = \mathbb{E}_{q_{\mathrm{cav}}(\mathbf{v}_m)}[\prod_{n \in \mathcal{M}} \mathbb{E}_{p(f_n \,|\, \mathbf{v}_m)}[p^\alpha(y_n \,|\, f_n)]] = \prod_{n \in \mathcal{M}} \mathbb{E}_{q_{\mathrm{cav}}(f_n)}[p^\alpha(y_n \,|\, f_n)]$, and $\mathcal{Z}_{\mathrm{site},m} = \mathbb{E}_{q_{\mathrm{cav}}(\mathbf{v}_m)}[\mathcal{N}^\alpha(\mathbf{v}_m \,|\, \mathbf{T}_{1,m}, \mathbf{T}_{2,m})]$ be the cavity normalisers of the true likelihoods and the site approximations. We

require the site constant factor, z_m, to be such that

$$\begin{aligned} z_m^\alpha \mathcal{Z}_{\text{site},m} &= \mathcal{Z}_{\text{lik},m} \\ \implies z_m^\alpha &= \mathcal{Z}_{\text{lik},m} / \mathcal{Z}_{\text{site},m} \\ \implies \log z_m &= \frac{1}{\alpha}(\log \mathcal{Z}_{\text{lik},m} - \log \mathcal{Z}_{\text{site},m}), \end{aligned} \tag{62}$$

so the full S^{2}PEP energy can be written,

$$\begin{aligned} \log \mathcal{Z}_{\text{PEP}} &= \frac{1}{\alpha} \textstyle\sum_m (\log \mathcal{Z}_{\text{lik},m} - \log \mathcal{Z}_{\text{site},m}) + \log \mathcal{Z} \\ &= \frac{1}{\alpha} \textstyle\sum_n \log \mathbb{E}_{q_{\text{cav}}(f_n)}[p^\alpha(y_n \,|\, f_n)] - \frac{1}{\alpha} \textstyle\sum_m \log \mathbb{E}_{q_{\text{cav}}(\mathbf{v}_m)}[\mathcal{N}^\alpha(\mathbf{v}_m \,|\, \mathbf{T}_{1,m}, \mathbf{T}_{2,m})] + \log \mathcal{Z}. \end{aligned} \tag{63}$$

C.4 Approximate Marginal Likelihood via Approximate Filtering

The marginal likelihood can be expressed as,

$$p(\mathbf{y}) = p(y_1) \textstyle\prod_{n=2}^N p(y_n \,|\, \mathbf{y}_{1:n-1}). \tag{64}$$

Further, each conditional term can be written (letting $\mathbf{s}(x_n) = \mathbf{s}_n$),

$$p(y_n \,|\, \mathbf{y}_{1:n-1}) = \textstyle\int p(y_n \,|\, f_n = \mathbf{H}\mathbf{s}_n)\, p(\mathbf{s}_n \,|\, \mathbf{y}_{1:n-1})\, \mathrm{d}\mathbf{s}(x_n), \tag{65}$$

where $p(\mathbf{s}_n \,|\, \mathbf{y}_{1:n-1})$ is the intractable forward filtering distribution:

$$p(\mathbf{s}_n \,|\, \mathbf{y}_{1:n-1}) = \textstyle\int p(\mathbf{s}_n \,|\, \mathbf{s}_{n-1})\, p(\mathbf{s}_{n-1} \,|\, \mathbf{y}_{1:n-2})\, \mathrm{d}\mathbf{s}_{n-1}. \tag{66}$$

Our approximation consists of running the approximate forward filter described in Eq. (36) to obtain $q^f(\mathbf{u}_m)$ for $m = 1, \ldots, M$. We then approximate a single term $p(y_n \,|\, \mathbf{y}_{1:n-1})$ as,

$$p(y_n \,|\, \mathbf{y}_{1:n-1}) \approx \textstyle\int p(y_n \,|\, f_n)\, p(f_n \,|\, \mathbf{u}_{m(n)})\, q^f(\mathbf{u}_{m(n)})\, t^{k_n}(\mathbf{u}_{m(n)})\, \mathrm{d}\mathbf{u}_{m(n)}, \tag{67}$$

where $t(\mathbf{u}_{m(n)}) = \int t_{m(n)}(\mathbf{v}_{m(n)})\, \mathrm{d}\mathbf{u}_{m(n)+1}$ is the contribution of the site in the forward direction and $k_n = N_n^{left}/N_{m(n)}$, with $N_{m(n)}$ being the number of data points whose inputs lie in $[z_{m(n)}, z_{m(n)+1}]$ and N_n^{left} being the number of data points whose inputs lie in $[z_{m(n)}, x_n)$. Intuitively, this means the fraction of the site corresponding to the data points to the *left* of x_n are included. Here $f_n \,|\, \mathbf{u}_{m(n)} \sim \mathcal{N}(f_n \,|\, \mathbf{A}_{m(n),n}\boldsymbol{\mu}_{m(n)}, \mathbf{A}_{m(n),n}\boldsymbol{\Sigma}_{m(n)}\mathbf{A}_{m(n),n}^\top + \mathbf{Q}_{m(n),n})$.

C.5 Posterior Linearisation (S^{2}PL)

In the general non-Gaussian likelihood case, when performing posterior linearisation we typically use the approximation $p(y_n \,|\, f_n) \approx \mathcal{N}(\mathbb{E}[y_n \,|\, f_n], \mathrm{Cov}[y_n \,|\, f_n])$, allowing us to use the additive noise statistical linear regression (SLR) equations (Särkkä, 2013) in order to linearise the expected likelihood:

$$\begin{aligned} \omega_n &= \int \mathbb{E}[y_n \,|\, f_n] q(f_n)\, \mathrm{d}f_n, \\ B_n &= \int \left[(\mathbb{E}[y_n \,|\, f_n] - \omega_n)(\mathbb{E}[y_n \,|\, f_n] - \omega_n)^\top + \mathrm{Cov}[y_n \,|\, f_n]\right] q(f_n)\, \mathrm{d}f_n, \\ C_n &= \int (f_n - \mu_n)(\mathbb{E}[y_n \,|\, f_n] - \omega_n)^\top q(f_n)\, \mathrm{d}f_n, \end{aligned} \tag{68}$$

where μ_n is the mean of the approximate marginal posterior $q(f_n)$.

As in S^{2}CVI, the site updates for our extension to PL, S^{2}PL, require only the posterior marginals, $q(f_n)$, whose moments are $\mu_n = \mathbf{W}_n\boldsymbol{\mu}_{m(n)}$ and $\Sigma_n = \mathbf{W}_n\boldsymbol{\Sigma}_{m(n)}^{-1}\mathbf{W}_n^\top + \nu_n^2$. The site update rule then proceeds as in Wilkinson et al. (2020), but now including the projection back from f_n to $\mathbf{v}_{m(n)}$ through the conditional $f_n \,|\, \mathbf{v}_{m(n)}$,

$$\begin{aligned} \boldsymbol{\lambda}_{2,n} &= -\frac{1}{2}\mathbf{W}_n^\top \Omega_n^\top \tilde{\Sigma}_n^{-1} \Omega_n \mathbf{W}_n, \\ \boldsymbol{\lambda}_{1,n} &= -2\boldsymbol{\lambda}_{2,n}\boldsymbol{\mu}_{m(n)} + \mathbf{W}_n^\top \Omega_n^\top \tilde{\Sigma}_n^{-1} r_n. \end{aligned} \tag{69}$$

where we have introduced

$$
\begin{aligned}
r_n &= y_n - \omega_n, \\
\tilde{\Sigma}_n &= B_n - C_n^\top \Sigma_n^{-1} C_n, \\
\Omega_n &= \frac{\partial \omega_n}{\partial \mu_n} = \mathbb{E}_{q(f_n)}\left[\mathbb{E}[y_n \,|\, f_n] \Sigma_n^{-1}(f_n - \mu_n)\right].
\end{aligned} \tag{70}
$$

Extended Kalman Smoother (S^{2}EKS) If the statistical linear regression equations are replaced by a first-order Taylor expansion, then PL reduces to the EKS. Hence we can also obtain a doubly sparse EKS (S^{2}EKS) algorithm by similarly substituting a Taylor expansion into the above. In practice, this amounts to setting $\tilde{\Sigma}_n = \mathrm{Cov}[y_n \,|\, f_n]$ and $\Omega_n = \frac{\partial \mathbb{E}[y_n \,|\, f_n]}{\partial f_n}|_{f_n = \mu_n}$. Whilst the EKS is not a common choice for modern day machine learning tasks, it does provide a useful trade off between efficiency, stability and performance. In particular, inference in S^{2}EKS avoids numerical integration, making it applicable in some scenarios where other methods are impractical.

PL Marginal Likelihood Approximation When defining the PL marginal likelihood, García-Fernández et al. (2019) assume a restrictive form for the sites, and discard a term in the marginal likelihood. However, the resulting approximation can be seen as a simplified form of the EP energy given in App. C.3.2. Therefore, to enable fair comparison, we use the EP energy for both S^{2}PL and S^{2}EKS in all our experiments.

D Experimental Details

The following descriptions of our experimental tasks are adapted from Wilkinson et al. (2020).

Motorcycle (heteroscedastic noise) The motorcycle crash data set (Silverman, 1985) contains 131 non-uniformly spaced measurements from an accelerometer placed on a motorcycle helmet during impact, over a period of 60 ms. It is a challenging benchmark (Tolvanen et al., 2014), due to the heteroscedastic noise variance. We model both the process itself and the measurement noise scale with independent GP priors with Matérn-3/2 kernels: $y_n \,|\, f_n^{(1)}, f_n^{(2)} \sim \mathcal{N}(y_n \,|\, f^{(1)}(x_n), [\phi(f^{(2)}(x_n))]^2)$, with softplus link function $\phi(f) = \log(1 + e^f)$ to ensure positive noise scale.

Coal (log-Gaussian Cox process) The coal mining disaster data set (Vanhatalo et al., 2013) contains 191 explosions that killed ten or more men in Britain between 1851–1962. We use a log-Gaussian Cox process, *i.e.* an inhomogeneous Poisson process (approximated with a Poisson likelihood for $N = 333$ equal time interval bins). We use a Matérn-5/2 GP prior with likelihood $p(\mathbf{y} \,|\, \mathbf{f}) \approx \prod_{n=1}^N \mathrm{Poisson}(y_n \,|\, \exp(f(\hat{x}_n)))$, where $\hat{x}_n$ is the bin coordinate and y_n the number of disasters in the bin. This model reaches posterior consistency in the limit of bin width going to zero (Tokdar and Ghosh, 2007). For the linearisation-based inference methods (S^{2}PL, S^{2}EKS) we utilise the fact that the first two moments are equal to the intensity, $\mathbb{E}[y_n \,|\, f_n] = \mathrm{Cov}[y_n \,|\, f_n] = \lambda(x_n) = \exp(f(x_n))$.

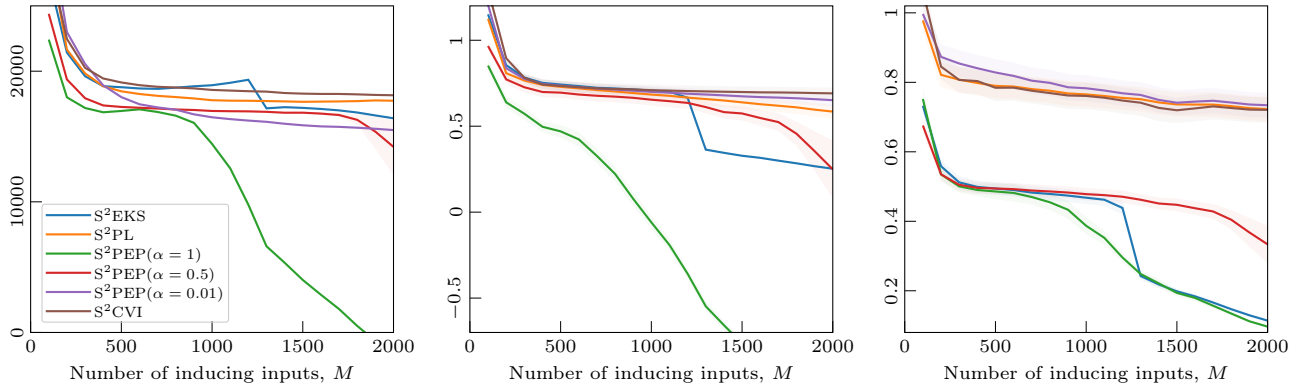
Airline (log-Gaussian Cox process) The airline accidents data (Nickisch et al., 2018) consists of 1210 dates of commercial airline accidents between 1919–2017. We use a log-Gaussian Cox process with bin width of one day, leading to $N = 35{,}959$ observations. The prior has multiple components, $\kappa(x, x') = \kappa(x, x')^{\nu=5/2}_{\text{Mat.}} + \kappa(x, x')^{1\,\text{year}}_{\text{per.}} \kappa(x, x')^{\nu=1/2}_{\text{Mat.}} + \kappa(x, x')^{1\,\text{week}}_{\text{per.}} \kappa(x, x')^{\nu=1/2}_{\text{Mat.}}$, capturing a long-term trend, time-of-year variation (with decay), and day-of-week variation (with decay). The state dimension is $d = 59$.

Binary (1D classification) As a 1D classification task, we create a long binary time series, $N = 10{,}000$, using the generating function $y(x) = \mathrm{sign}\{\frac{12\sin(4\pi x)}{0.25\pi x + 1} + \sigma_x\}$, with $\sigma_x \sim \mathcal{N}(0, 0.01^2)$. Our GP prior has a Matérn-7/2 kernel, $d = 4$, and the sigmoid function $\psi(f) = (1 + e^{-f})^{-1}$ maps $\mathbb{R} \mapsto [0, 1]$ (logit classification).

Audio (product of GPs) We apply a simplified version of the Gaussian Time-Frequency model from Wilkinson et al. (2019) to half a second of human speech, sampled at 44.1 kHz, $N = 22{,}050$. The prior consists of 3 quasi-periodic ($\kappa_{\exp}(x, x')\kappa_{\cos}(x, x')$) 'subband' GPs, and 3 smooth ($\kappa_{\text{Mat-}5/2}(x, x')$) 'amplitude' GPs. The likelihood consists of a sum of the product of these processes with additive noise and a softplus mapping $\phi(\cdot)$ for the positive amplitudes: $y_n \,|\, \mathbf{f}_n \sim \mathcal{N}(\sum_{i=1}^3 f_{i,n}^{\text{sub.}} \phi(f_{i,n}^{\text{amp.}}), \sigma_n^2)$. The nonlinear interaction of 6 GPs ($d = 15$) in the likelihood makes this a challenging task.

In Fig. 4 we analyse the effect of increasing the number of inducing inputs in the Audio task. We observe that the training marginal likelihood (NLML) and the test predictive density (NLPD) improve as M increases, as

expected for all methods. S^{2}PEP significantly outperforms the other methods, requiring fewer than 1000 inducing inputs to provide good results.



(a) NLML (b) NLPD (c) RMSE

Figure 5: Analysis of the Audio task with varying number of inducing inputs. S^{2}PEP ($\alpha = 1$) performs best in terms of test predictive density (NLPD) and RMSE, and requires many fewer inducing points. Whilst we expect S^{2}PEP ($\alpha = 0.01$) and S^{2}CVI to give similar results, the numerical integration error when using 3-dimensional cubature causes the results to differ in practice.

Banana (2D classification) The banana data set, $N = 5300$, is a common 2D classification benchmark (Hensman et al., 2015). We use the logit likelihood with a separable space-time kernel: $\kappa(r, x; r', x') = \kappa(x, x')^{\nu=5/2}_{\text{Mat.}} \kappa(r, r')^{\nu=5/2}_{\text{Mat.}}$. The vertical dimension is treated as space, r, and the horizontal as the sequential ('temporal') dimension, x. We use $M = 15$ inducing points in r, as well as $M = 15$ inducing points in x. The state dimension is $d = 3M = 45$. For the SVGP baseline, we use $M = 15^2 = 225$ inducing points placed on a 2D grid.

Electricity (large scale regression) We analyse the electricity consumption of one household (Hébrail and Bérard, 2012; Solin et al., 2018) recorded every minute (in log kW) over 1,442 days (2,075,259 total data points, with 25,979 missing observations). We assign the model a GP prior with a covariance function accounting for slow variation (Matérn-$^3/_2$) and daily periodicity with decay (quasi-periodic Matérn-$^1/_2$). We fit a GP to one 6 month's worth of data, which amounts to $N = 262{,}080$ points.