


Sparse Information Filter for Fast Gaussian Process Regression Supplementary material

Lucas Kania¹^[0000-0001-5713-274X], Manuel Schürch^{1,2}^[0000-0003-2175-2511],
Dario Azzimonti²^[0000-0001-5080-3061], and Alessio
Benavoli³^[0000-0002-2522-7178]

¹ Università della Svizzera italiana (USI), Via Buffi 13, Lugano, Switzerland
`lucas.kania@usi.ch`

² Istituto Dalle Molle di Studi sull'Intelligenza Artificiale (IDSIA) ,
`{manuel.schuerch, dario.azzimonti}@idsia.ch`
Via la Santa 1, Lugano, Switzerland

³ School of Computer Science and Statistics, Trinity College, Dublin, Ireland
`alessio.benavoli@tcd.ie`

1 Derivation and efficient computation of \mathcal{L}_{IF}

Given the \mathcal{L}_{REC} lower-bound

$$\begin{aligned}\mathcal{L}_{\text{REC}} &= \sum_{k=1}^K \log \mathcal{N}(\mathbf{y}_k | \mathbf{H}_{X_k} \mu_{k-1}, S_k) - \frac{\text{Tr}(\mathbf{K}_{X_k X_k} - \mathbf{Q}_{X_k X_k})}{2 \sigma_n^2} \\ &= \sum_{k=1}^K \log \mathcal{N}(r_k | \mathbf{0}, S_k) - \frac{\text{Tr}(\mathbf{K}_{X_k X_k} - \mathbf{Q}_{X_k X_k})}{2 \sigma_n^2}\end{aligned}\quad (1)$$

where $r_k = \mathbf{y}_k - \mathbf{H}_{X_k} \mu_{k-1}$

$$\begin{aligned}\mathbf{H}_{X_k} &= \mathbf{K}_{X_k \text{R}} \mathbf{K}_{\text{RR}}^{-1} \\ S_k &= \mathbf{H}_{X_k} \Sigma_{k-1} \mathbf{H}_{X_k}^T + \sigma_n^2 \mathbb{I}\end{aligned}\quad (2)$$

We would like to rewrite it w.r.t. the rotated natural parameterization of q_k . For that, it's enough to transform the inverse of S_k and r_k . We start with the covariance S_k :

$$\begin{aligned}S_k^{-1} &= (\mathbf{H}_{X_k} \Sigma_{k-1} \mathbf{H}_{X_k}^T + \sigma_n^2 \mathbb{I})^{-1} \\ &= \frac{I}{\sigma_n^2} - \frac{1}{\sigma_n^4} \mathbf{H}_{X_k} \Lambda_k^{-1} \mathbf{H}_{X_k}^T \quad \text{Using the Woodbury formula} \\ &= \frac{I}{\sigma_n^2} - \frac{1}{\sigma_n^4} \mathbf{K}_{X_k \text{R}} (\Lambda_k^{\text{R}})^{-1} \mathbf{K}_{\text{R} X_k}\end{aligned}\quad (3)$$

Next, consider that we have the Cholesky factorization of the previous rotated precision matrix

$$\Lambda_{k-1}^R = C_{k-1} C_{k-1}^T \quad (4)$$

we can easily express the residual at each iteration

$$\begin{aligned} r_k &= y_k - H_{X_k} \mu_{k-1} \\ &= \mathbf{y}_k - K_{X_k R} \Lambda_{k-1}^R \eta_{k-1}^R \\ &= \mathbf{y}_k - (C_{k-1}^{-1} K_{R X_k})^T (C_{k-1}^{-1} \eta_{k-1}^R) \\ &= \mathbf{y}_k - G_k^T C_{k-1}^{-1} \eta_{k-1}^R \end{aligned} \quad (5)$$

where $G_k = C_{k-1}^{-1} K_{R X_k}$. Thus, by plugging eq.s (3) and (5) into \mathcal{L}_{REC} , we obtain \mathcal{L}_{IF} :

$$\mathcal{L}_{\text{IF}} = \sum_{k=1}^K \log \mathcal{N}^{-1}(r_k | 0, S_k^{-1}) - \frac{\text{Tr}(K_{X_k X_k} - Q_{X_k X_k})}{2 \sigma_n^2},$$

In the following, we detail the efficient computation of the lower bound. Notice, that \mathcal{L}_{IF} can be expanded into

$$\mathcal{L}_{\text{IF}} = \sum_{k=1}^K -\frac{B}{2} \log 2\pi + \frac{1}{2} \log |S_k^{-1}| - \frac{1}{2} r_k^T S_k^{-1} r_k - \frac{\text{Tr}(K_{X_k X_k} - Q_{X_k X_k})}{2 \sigma_n^2}, \quad (6)$$

Using eq. (4), we can express the current rotated precision matrix as

$$\Lambda_k^R = C_{k-1} O_k C_{k-1}^T$$

where

$$O_k = \mathbb{I} + C_{k-1}^{-1} A_k C_{k-1}^T = I + \frac{G_k G_k^T}{\sigma_n^2} \quad \text{with} \quad A_k = \frac{K_{R X_k} K_{X_k R}}{\sigma_n^2}$$

By computing the Cholesky of O_k , we obtain the Cholesky of Λ_k^R . That is, given $O_k = E_k E_k^T$, we have that

$$\Lambda_k^R = C_k C_k^T \quad \text{with} \quad C_k = C_{k-1} E_k \quad (7)$$

Since C_{k-1} and E_k are lower-triangular matrices, C_k is lower-triangular. Thus, it's the unique Cholesky factorization of Λ_k^R . Furthermore, note that O_k has bounded eigenvalues [1], which leads to stable decompositions during training. This relationship between the Cholesky decompositions of Λ_k^R and Λ_{k-1}^R , i.e. eq. (7), leads to a cheap computation of the log-determinant of S_k^{-1} . Specifically,

$$\begin{aligned}
\frac{1}{2} \log |S_k^{-1}| &= \frac{1}{2} \log \left| \frac{I}{\sigma_n^2} \right| \cdot |A_{k-1}^R| \cdot |(\Lambda_k^R)^{-1}| \quad \text{By the determinant lemma} \\
&= -\frac{B}{2} \log \sigma_n^2 + \log |C_{k-1}| - (\log |C_{k-1}| + \log |E_k|) \\
&= -\frac{B}{2} \log \sigma_n^2 + \log |E_k| \\
&= -\frac{B}{2} \log \sigma_n^2 + \text{Tr Diag } E_k,
\end{aligned} \tag{8}$$

where the last equality is due to E_k being a lower triangular matrix. Finally, we rewrite the quadratic term

$$\begin{aligned}
r_k^T S_k^{-1} r_k &= \frac{r_k^T r_k}{\sigma_n^2} - \frac{1}{\sigma_n^4} r_k^T K_{X_k R} (\Lambda_k^R)^{-1} K_{R X_k} r_k \\
&= \frac{r_k^T r_k}{\sigma_n^2} - p_k^T p_k
\end{aligned} \tag{9}$$

where $p_k = C_k^{-1} K_{R X_k} r_k$, and the regularizer, which can be simplified by using the Cholesky decomposition of the inducing points kernel matrix

$$K_{RR} = L_R L_R^T \tag{10}$$

as follows

$$\begin{aligned}
\text{Tr}(K_{X_k X_k} - Q_{X_k X_k}) &= \text{Tr } K_{X_k X_k} - \text{Tr}(Q_{X_k}^T Q_{X_k}) \\
&= \text{Tr}(K_{X_k X_k}) - \sum_{i,j} Q_{X_k i j}^2
\end{aligned} \tag{11}$$

Replacing eq.s (8),(9) and (11) into eq. (6), we get

$$\begin{aligned}
\mathcal{L}_{\text{IF}} &= \sum_{k=1}^K -\frac{B}{2} \log 2\pi \sigma_n^2 - \text{Tr Diag } E_k \\
&\quad - \frac{r_k^T r_k}{2 \sigma_n^2} + \frac{p_k^T p_k}{2} - \frac{\text{Tr}(K_{X_k X_k}) - \sum_{i,j} Q_{X_k i j}^2}{2 \sigma_n^2}
\end{aligned} \tag{12}$$

Algorithm 1 summarizes the computation of the bound. Furthermore, note that since C_k is a lower triangular matrix, whenever we must solve a system like $C_k a = b$, it's more efficient to solve the triangular system, denoted SOLVE in the algorithm, than to invert C_k , requiring $O(M^2)$ instead of $O(M^3)$.

Algorithm 1: \mathcal{L}_{IF} computation

```

1  $\mathcal{L}_{\text{REC}} = 0$  ;
2 for  $k = 1 \rightarrow K$  do
3    $A_k = \frac{\mathbf{K}_{\text{RX}_k} \mathbf{K}_{\text{X}_k \text{R}}}{\sigma_n^2}$  ;
4    $b_k = \frac{\mathbf{K}_{\text{RX}_k} y_k}{\sigma_n^2}$  ;
5    $\Lambda_k^{\text{R}} = \Lambda_{k-1}^{\text{R}} + A_k$  ;
6    $\eta_{r(k)} = \eta_{k-1}^{\text{R}} + b_k$  ;
7    $\mathbf{G}_k = \mathbf{C}_{k-1}^{-1} \mathbf{K}_{\text{RX}_k} = \text{SOLVE}(\mathbf{C}_{k-1}, \mathbf{K}_{\text{RX}_k})$  ;
8    $r_k = \mathbf{G}_k^T \eta_{k-1}^{\text{R}}$  ;
9    $\mathbf{O}_k = \mathbb{I} + \frac{\mathbf{G}_k^T \mathbf{G}_k}{\sigma_n^2}$  ;
10   $\mathbf{E}_k = \text{Cholesky } \mathbf{O}_k$  ;
11   $\mathbf{C}_k = \mathbf{C}_{k-1} \mathbf{E}_k$  ;
12   $p_k = \mathbf{C}_k^{-1} \mathbf{K}_{\text{RX}_k} r_k = \text{SOLVE}(\mathbf{C}_k, \mathbf{K}_{\text{RX}_k} r_k)$  ;
13   $d_k = -\frac{B}{2} \log 2\pi \sigma_n^2 - \text{Tr Diag } \mathbf{E}_k - \frac{r_k^T r_k}{2 \sigma_n^2} + \frac{p_k^T p_k}{2}$  ;
14   $\mathbf{L}_r = \text{Cholesky } \mathbf{K}_{\text{RR}}$  ;
15   $\mathbf{Q}_{\text{X}_k} = \mathbf{L}_r^{-1} \mathbf{K}_{\text{RX}_k} = \text{SOLVE}(\mathbf{L}_r, \mathbf{K}_{\text{RX}_k})$  ;
16   $a_k = \frac{\text{Tr } \mathbf{K}_{\text{X}_k \text{X}_k} - \sum_{i,j} \mathbf{Q}_{\text{X}_k i j}^2}{2 \sigma_n^2}$  ;
17   $l_k = d_k - a_k$  ;
18   $\mathcal{L}_{\text{REC}} = \mathcal{L}_{\text{REC}} + l_k$  ;
19 end
20  $\theta = \theta + \alpha \nabla_{\theta} \mathcal{L}_{\text{REC}}$  ;

```

2 Prediction

Given a new $\mathbf{X}_* \in \mathbb{R}^{A \times D}$, the predictive distribution of the SGP after seeing k mini-batches can be computed by

$$p(\mathbf{f}_* | \mathbf{y}_{1:k}) = \int p(\mathbf{f}_* | \mathbf{f}_{\text{R}}) p(\mathbf{f}_{\text{R}} | \mathbf{y}_{1:k}) d\mathbf{f}_{\text{R}} = \mathcal{N}(\mathbf{f}_* | \mu_*, \Sigma_*), \quad (13)$$

where $\mathbf{y}_{1:k} = \{\mathbf{y}_1, \dots, \mathbf{y}_k\}$. Taking into account the following distributions

$$p(\mathbf{f}_* | \mathbf{f}_{\text{R}}) = \mathcal{N}(\mathbf{f}_* | \mathbf{H}_{\text{X}*} \mathbf{f}_{\text{R}}, \mathbf{K}_{\text{X}* \text{X}*} - \mathbf{Q}_{\text{X}* \text{X}*}) \quad (14)$$

$$p(\mathbf{f}_{\text{R}} | \mathbf{y}_{1:k}) = \mathcal{N}(\mathbf{f}_{\text{R}} | \mu_k, \Sigma_k) \quad (15)$$

where $\mathbf{H}_{\text{X}*} = \mathbf{K}_{\text{X}* \text{R}} \mathbf{K}_{\text{RR}}^{-1}$ and $\mathbf{Q}_{\text{X}* \text{X}*} = \mathbf{K}_{\text{X}* \text{R}} \mathbf{K}_{\text{RR}}^{-1} \mathbf{K}_{\text{RX}*}$, and integrating out \mathbf{f}_{R} we obtain

$$p(\mathbf{f}_* | \mathbf{y}_{1:k}) = \mathcal{N}(\mathbf{f}_* | \mu_*, \Sigma_*) \quad (16)$$

where $\mu_* = \mathbf{H}_{\text{X}*} \mu_k$ and $\Sigma_* = \mathbf{K}_{\text{X}* \text{X}*} - \mathbf{Q}_{\text{X}* \text{X}*} + \mathbf{H}_{\text{X}*} \Sigma_k \mathbf{H}_{\text{X}*}^T$.

In the following, we rewrite this predictive distribution w.r.t. the rotated natural parameters Λ_k^{R} and η_k^{R} :

$$\begin{aligned}
\mu_* &= K_{X^*R} K_{RR}^{-1} \mu_k \\
&= K_{X^*R} K_{RR}^{-1} \Lambda_k^{-1} \eta_k && (\text{Since } \eta_k = \Lambda_k \mu_k) \\
&= K_{X^*R} (\Lambda_k^R)^{-1} \eta_k^R && (17)
\end{aligned}$$

and

$$\begin{aligned}
\Sigma_* &= K_{X^*X^*} - Q_{X^*X^*} + K_{X^*R} K_{RR}^{-1} \Sigma_k K_{RR}^{-1} K_{RX^*} \\
&= K_{X^*X^*} - Q_{X^*X^*} + K_{X^*R} K_{RR}^{-1} \Lambda_k^{-1} K_{RR}^{-1} K_{RX^*} \\
&= K_{X^*X^*} - Q_{X^*X^*} + K_{X^*R} (\Lambda_k^R)^{-1} K_{RX^*} && (18)
\end{aligned}$$

Plugging eq.s (17) and (18) into (16), we obtain the predictive distribution presented in the main paper. The predictions for \mathbf{y}_* are obtained by adding $\sigma_n^2 \mathbb{I}$ to Σ_* in eq. (16). That is,

$$\begin{aligned}
p(\mathbf{y}_* | \mathbf{y}_{1:k}) &= \mathcal{N}(\mathbf{y}_* | K_{X^*R} (\Lambda_k^R)^{-1} \eta_k^R, \\
&\quad K_{X^*X^*} - Q_{X^*X^*} + K_{X^*R} (\Lambda_k^R)^{-1} K_{RX^*} + \sigma_n^2 \mathbb{I})
\end{aligned}$$

Finally, the actual computation of the predictive distribution's parameters takes advantage of the computed Cholesky decompositions (4) and (10).

$$\mu_s = (C_k^{-1} K_{RX^*})^T (C_k^{-1} \eta_k^R) \quad (19)$$

$$\Sigma_* + \sigma_n^2 \mathbb{I} = K_{X^*X^*} - (L_R^{-1} K_{RX^*})^2 + (C_k^{-1} K_{RX^*})^2 + \sigma_n^2 \mathbb{I} \quad (20)$$

Naturally, we solve the triangular systems instead of inverting C_k and L_R .

3 Adaptive Switch

In the paper, we switched from optimizing \mathcal{L}_{IA} to doing the posterior propagation, i.e. $\mathcal{L}_{IA} \rightarrow PP$, for a fixed number of epochs at the end of the optimization. Consequently, there are potentially many *wasted* iterations since \mathcal{L}_{IA} has already converged in RMSE and log-likelihood but the switch has not been done. For instance, this behaviour can be observed in the **SONG** dataset in the figure 1 of the main paper.

Given that the posterior propagation only requires to run for one epoch in order to accumulate all the information in the dataset, a switch could be done earlier and therefore many iterations could be saved. A simple rule is to switch to the posterior propagation when all the parameters have not changed more than a threshold for a certain number of epochs. Figure 1 displays this method, denoted $\mathcal{L}_{IA} \rightarrow PP$ ADAPTIVE, for the 5D synthetic dataset presented in the main paper. The switch was done when the parameters did not change by more than 0.001 (i.e. the used learning rate) for at least 5 epochs.

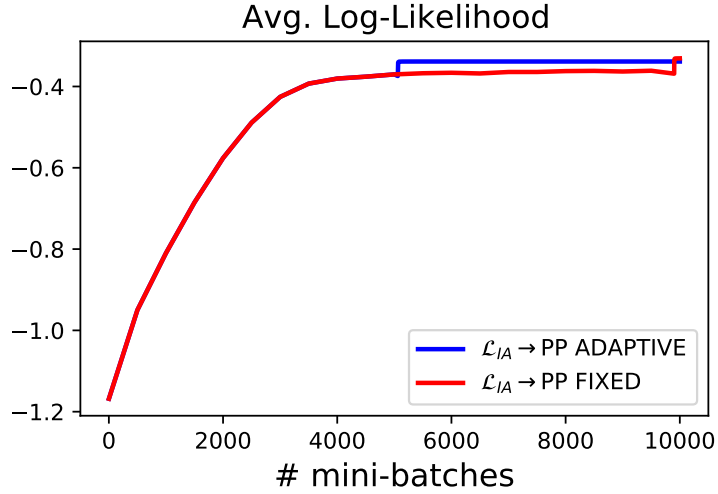


Figure 1: SGP trained twice with $\mathcal{L}_{IA} \rightarrow PP$ for the 5D synthetic dataset presented in the main paper. $\mathcal{L}_{IA} \rightarrow PP$ FIXED switches from optimizing \mathcal{L}_{IA} to doing the posterior propagation in the last 5 epochs, while $\mathcal{L}_{IA} \rightarrow PP$ ADAPTIVE switches when the parameters of the SGP have not changed more than a certain threshold (the learning rate, 0.001) for at least 5 epochs.

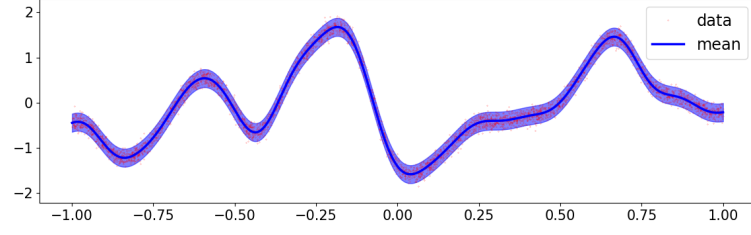
4 Fitted GP and SGPs for the 1D toy dataset

Figures 2a, 2b, 2c, 2d and 2e show the final fit after 20K iterations for GP and SGPs trained with \mathcal{L}_{VFE} , $\mathcal{L}_{IA} \rightarrow PP$, \mathcal{L}_{SVGP} using natural gradients and \mathcal{L}_{SVGP} not using natural gradients correspondingly. The dataset was the 1D toy dataset used in the main paper, with the same configuration for each method. Namely, the learning rate for the ADAM optimizer was 0.001, and 0.1 was used for the gamma parameter in the natural gradients.

The main remark is that using \mathcal{L}_{SVGP} without natural gradients usually leads to overconfident intervals in practice. Therefore, in this work we compare our method to SGPs trained with \mathcal{L}_{SVGP} and natural gradients for the variational parameters.

References

1. Rasmussen, C.E., Williams, C.K.I.: Gaussian processes for machine learning. MIT Press (2006)



(a) Full GP fit.

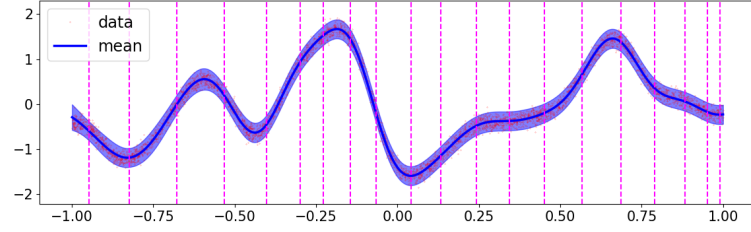
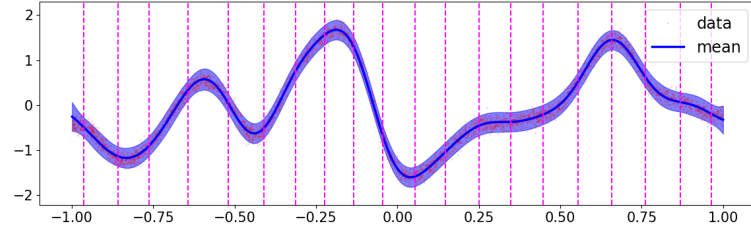
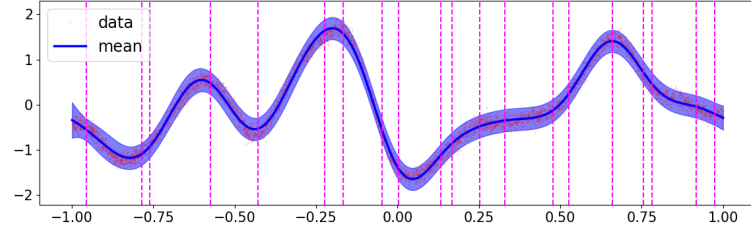
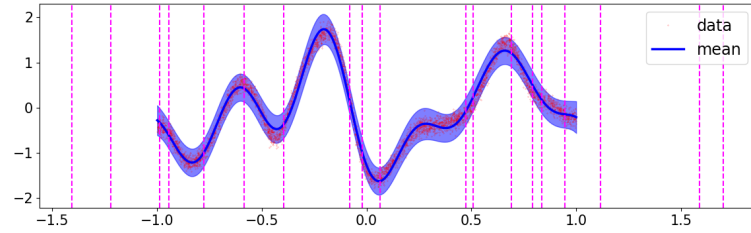
(b) SGP trained with \mathcal{L}_{VFE} .(c) SGP trained with $\mathcal{L}_{\text{IA}} \rightarrow \text{PP}$.(d) SGP trained with $\mathcal{L}_{\text{SVGP}}$ using natural gradients for the variational parameters.(e) SGP trained with $\mathcal{L}_{\text{SVGP}}$ using ADAM for all the parameters.

Figure 2: Comparison of different methods on the 1D toy dataset. The vertical bars indicate the inducing points positions.