State Space Gaussian Processes with Non-Gaussian Likelihood

Hannes Nickisch [1] **Arno Solin** [2] **Alexander Grigorievskiy** [2 3]

Abstract

We provide a comprehensive overview and tooling for GP modeling with non-Gaussian likelihoods using state space methods. The state space formulation allows to solve one-dimensional GP models in $\mathcal{O}(n)$ time and memory complexity. While existing literature has focused on the connection between GP regression and state space methods, the computational primitives allowing for inference using general likelihoods in combination with the Laplace approximation (LA), variational Bayes (VB), and assumed density filtering (ADF, a.k.a. single-sweep expectation propagation, EP) schemes has been largely overlooked. We present means of combining the efficient $\mathcal{O}(n)$ state space methodology with existing inference methods. We extend existing methods, and provide unifying code implementing all approaches.

1. Introduction

Gaussian processes (GPs) (Rasmussen & Williams, 2006) form a versatile class of probabilistic machine learning models with applications in regression, classification as well as robust and ordinal regression. In practice, there are computational challenges arising from *(i)* non-conjugate (non-Gaussian) likelihoods and *(ii)* large datasets.

The former *(i)* can be addressed by approximating the non-Gaussian posterior by an effective Gaussian giving rise to a number of algorithms such as the Laplace approximation (LA, Williams & Barber, 1998), variational Bayes (VB, Gibbs & MacKay, 2000), direct Kullback–Leibler (KL) divergence minimization (Opper & Archambeau, 2009) and expectation propagation (EP, Minka, 2001) with different tradeoffs in terms of accuracy and required computations (Kuss & Rasmussen, 2005; Nickisch & Rasmussen, 2008; Naish-Guzman & Holden, 2008). The latter *(ii)* can be addressed by approximate covariance computations using sparse inducing point methods (Quiñonero-Candela & Rasmussen, 2005) based on variational free energy (VFE, Titsias, 2009), fully independent training conditionals (FITC, Snelson & Ghahramani, 2006), hybrids (Bui et al., 2017), or stochastic approximations (Hensman et al., 2013; Krauth et al., 2017) applicable to any data dimension D. A second class of covariance interpolation methods, KISS-GP (Wilson & Nickisch, 2015; Wilson et al., 2015), are based on grids of inducing points. For $1 < D < 5$, product covariance, and rectilinear grids, the covariance matrix has Kronecker structure. For $D = 1$, stationary covariance, and a regular grid, the covariance matrix has Toeplitz structure (a special case of block-Toeplitz with Toeplitz blocks (BTTB) obtained for $1 < D < 5$), which can be exploited for fast matrix-vector multiplications (MVMs). A third covariance approximation methodology is based on basis function expansions such as sparse spectrum GPs (Lázaro-Gredilla et al., 2010), variational Fourier features (Hensman et al., 2018), or Hilbert space GPs (Solin & Särkkä, 2014b) for stationary covariance functions. Higher input dimensions $D > 4$ either tend to get computationally heavy or prone to overfitting.

In time-series data, with $D = 1$, the data sets tend to become long (or unbounded) when observations accumulate over time. For these time-series models, leveraging sequential *state space* methods from signal processing makes it possible to solve GP inference problems in *linear time complexity* $\mathcal{O}(n)$ if the underlying GP has Markovian structure (Reece & Roberts, 2010; Hartikainen & Särkkä, 2010). This reformulation is *exact* for Markovian covariance functions (see, e.g., Solin, 2016) such as the exponential, half-integer Matérn, noise, constant, linear, polynomial, Wiener, etc. (and their sums and products). Covariance functions such as the squared exponential (Hartikainen & Särkkä, 2010), rational quadratic (Solin & Särkkä, 2014a), and periodic (Solin & Särkkä, 2014) can be approximated by their Markovian counterparts. Grigorievskiy & Karhunen (2016); Grigorievskiy et al. (2017) bridge the state space connection further by leveraging sparse matrices (SpInGP) in connection with the Markovian state space models. Another issue is that if time gaps between data points are very uneven then the computational power is spend on computing required matrix exponentials. This still makes the method slow for the large datasets with uneven sampling despite the linear

Proceedings of the 35^{th} International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018. [1]Digital Imaging, Philips Research, Hamburg, Germany [2]Department of Computer Science, Aalto University, Espoo, Finland [3]Silo.AI, Helsinki, Finland. Correspondence to: Hannes Nickisch <hannes@nickisch.org>, Arno Solin <arno.solin@aalto.fi>.

computational complexity of inference. This shows as a large cost per time step (the 'hidden' constant in the big-O notation) due to evaluating matrix exponentials.

The previous literature has focused on rewriting the GP in terms of a state space model (focusing on challenge *(i)*). Addressing challenge *(ii)*, non-Gaussian likelihoods have been touched upon by Solin & Särkkä (2014a) (inner-loop Laplace approximation) and Hartikainen et al. (2011) in a spatio-temporal log Gaussian Cox process (using EP combined with local extended Kalman filtering updates). However, deriving approximate inference schemes is in the state space regime is complicated and requires hand-crafting for each likelihood.

Related work also includes Kalman filtering for optimization in parametric models (Aravkin et al., 2013; 2014), and non-linear GP priors in system identification models (a.k.a. 'GP state space' models, see, e.g., Frigola et al., 2014).

This paper advances the state-of-the-art in two ways:

- We present a unifying framework for solving computational primitives for non-Gaussian inference schemes in the state space setting, thus directly enabling inference to be done through LA, VB, KL, and ADF/EP.
- We present a novel way for solving the continuous-time state space model through interpolation of the matrix exponential, which further speeds up the linear time-complexity by addressing the large-constant problem.

Code for the paper is available as part of the GPML toolbox version 4.2 (Rasmussen & Nickisch, 2010).

2. Methods

We introduce the GP framework in Sec. 2.1, then name four computational primitives that can be used to operate approximate inference schemes beyond the exact Gaussian case in Sec. 2.2. The state space representation of GPs is introduced in 2.3 along with the Kalman filtering and smoothing algorithms, Algs. 2+3. Then, we will show how these primitives including prediction can be implemented for GPs using the state space representation in Sec. 2.5. Further, we detail how they can be used to operate inference for Laplace approximation (LA) in Sec. 2.6, variational Bayes (VB) in Sec. 2.7, assumed density filtering (ADF) a.k.a. single sweep expectation propagation (EP) in Sec. 2.9 and Kullback–Leibler (KL) minimization in Sec. 2.8. For the first three algorithms, we are also able to perform full-fledged gradient-based hyperparameter learning.

2.1. Gaussian process training and prediction

The models we are interested, in take the following standard form of having a latent Gaussian process prior and a measurement (likelihood) model:

$$f(t) \sim \mathrm{GP}(m(t), k(t,t')), \quad \mathbf{y}|\mathbf{f} \sim \prod_{i=1}^{n} \mathbb{P}(y_i|f(t_i)),$$

where the likelihood factorizes over the observations. This family of models covers many types of modeling problems including (robust or ordinal) regression and classification.

We denote the data as a set of scalar input–output pairs $\mathcal{D} = \{(t_i, y_i)\}_{i=1}^n$. We are interested in models following Rasmussen & Nickisch (2010) that – starting from the Gaussian prior $\mathbf{f} = \mathrm{N}(\mathbf{f}|\mathbf{m}, \mathbf{K})$ given by the GP – admit an approximate posterior of the form

$$\mathbb{Q}(\mathbf{f}|\mathcal{D}) = \mathcal{N}\left(\mathbf{f}|\mathbf{m} + \mathbf{K}\boldsymbol{\alpha}, (\mathbf{K}^{-1} + \mathbf{W})^{-1}\right), \quad (1)$$

where $m_i = m(t_i)$ and $K_{i,j} = k(t_i, t_j)$ are the prior mean and covariance. The vector $\boldsymbol{\alpha}$ and the (likelihood precision) matrix $\mathbf{W} = \mathrm{diag}(\mathbf{w})$ form the set of $2n$ parameters. Elements of $\mathbf{w}$ are non negative for log-concave likelihoods. Equivalently, we can use the natural parameters $(\mathbf{b}, \mathbf{W})$ of the effective likelihood, where $\mathbf{b} = \mathbf{W}\mathbf{K}\boldsymbol{\alpha} + \boldsymbol{\alpha}$ in general and for Gaussian likelihood $\mathbf{b} = \mathbf{W}(\mathbf{y} - \mathbf{m})$ in particular.

Given these parameters, the predictive distribution for an unseen test input t_* is obtained by integrating the Gaussian latent marginal distribution $\mathcal{N}(f_*|\mu_{f,*}, \sigma^2_{f,*})$

$$\mu_{f,*} = \mathbf{m}_* + \mathbf{k}_*^\top \boldsymbol{\alpha};\ \sigma^2_{f,*} = k_{**} - \mathbf{k}_*^\top \left(\mathbf{K} + \mathbf{W}^{-1}\right)^{-1} \mathbf{k}_* \quad (2)$$

against the likelihood $\mathbb{P}(y_*|f_*)$ to obtain

$$\mathbb{P}(y_*) = \int \mathbb{P}(y_*|f_*)\, \mathcal{N}(f_*|\mu_{f,*}, \sigma^2_{f,*})\, \mathrm{d}f_* \quad (3)$$

the predictive distribution whose first two moments can be used to make a statement about the unknown y_*.

The model may have hyperparameters $\boldsymbol{\theta} = [a, d, \sigma_f, \ell, \sigma_n]$ of the mean e.g. $m(t) = at + d$, the covariance e.g. $k(t,t') = \sigma_f^2 \exp(-(t-t')^2/(2\ell^2))$ and the likelihood e.g. $\mathbb{P}(y_i|f_i) = \mathcal{N}(f_i|y_i, \sigma_n^2)$ which can be fit by maximizing the (log) marginal likelihood of the model

$$\log Z(\boldsymbol{\theta}) = \log \int \mathcal{N}\left(\mathbf{f}|\mathbf{m}, \mathbf{K}\right) \prod_i \mathbb{P}(y_i|f_i)\, \mathrm{d}\mathbf{f}, \quad (4)$$

which is an intractable integral in the non-Gaussian case but can be approximated or bounded in various ways.

A prominent instance of this setting is plain GP regression (see Alg. 1), where the computation is dominated by the $\mathcal{O}(n^3)$ log-determinant computation and the linear system for $\boldsymbol{\alpha}$. To overcome the challenges arising from non-conjugacy and large dataset size n, we define a set of generic computations and replace their dense matrix implementation (see Alg. 1) with state space algorithms.

Algorithm 1 Predictions and log marginal likelihood $\log Z$ for Gaussian process regression (Alg. 2.1 in Rasmussen & Williams (2006)). Complexity is $\mathcal{O}(n^3)$ for the Cholesky decomposition, and $\mathcal{O}(n^2)$ for solving triangular systems.

Input: $\{t_i\},\ \{y_i\}$ # training inputs and targets
$k,\ \sigma_n^2,\ t_*$ # covariance, noise variance, test input
$\mathbf{L} \leftarrow \text{Cholesky}(\mathbf{K} + \sigma_n^2 \mathbf{I});\ \boldsymbol{\alpha} \leftarrow \mathbf{L}^{-\top}(\mathbf{L}^{-1}(\mathbf{y} - \mathbf{m}))$
$\log Z \leftarrow -\frac{1}{2}(\mathbf{y} - \mathbf{m})^\top \boldsymbol{\alpha} - \sum_i \log L_{i,i} - \frac{n}{2} \log 2\pi$
$\mu_{f,*} \leftarrow \mathbf{m}_* + \mathbf{k}_*^\top \boldsymbol{\alpha};\ \sigma^2_{f,*} \leftarrow k_{**} - \|\mathbf{L} \backslash \mathbf{k}_*\|_2^2$
Return: $\mu_{f,*},\ \sigma^2_{f,*},\ \log Z$ # mean, variance, evidence

2.2. Gaussian process computational primitives

The following computational primitives allow to cast the covariance approximation in more generic terms:

1. Linear system with "regularized" covariance: $\text{solve}_{\mathbf{K}}(\mathbf{W}, \mathbf{r}) := (\mathbf{K} + \mathbf{W}^{-1})^{-1}\mathbf{r}$.
2. Matrix-vector multiplications: $\text{mvm}_{\mathbf{K}}(\mathbf{r}) := \mathbf{K}\mathbf{r}$. For learning we also need $\frac{\text{mvm}_{\mathbf{K}}(\mathbf{r})}{\partial \boldsymbol{\theta}}$.
3. Log-determinants: $\text{ld}_{\mathbf{K}}(\mathbf{W}) := \log|\mathbf{B}|$ with symmetric and well-conditioned $\mathbf{B} = \mathbf{I} + \mathbf{W}^{\frac{1}{2}}\mathbf{K}\mathbf{W}^{\frac{1}{2}}$. For learning, we need derivatives: $\frac{\partial \text{ld}_{\mathbf{K}}(\mathbf{W})}{\partial \boldsymbol{\theta}}$, $\frac{\partial \text{ld}_{\mathbf{K}}(\mathbf{W})}{\partial \mathbf{W}}$.
4. Predictions need latent mean $\mathbb{E}[f_*]$ and variance $\mathbb{V}[f_*]$.

Using these primitives, GP regression can be compactly written as $\mathbf{W} = \mathbf{I}/\sigma_n^2$, $\boldsymbol{\alpha} = \text{solve}_{\mathbf{K}}(\mathbf{W}, \mathbf{y} - \mathbf{m})$, and

$$\log Z_{\text{GPR}} = -\frac{1}{2}\left[\boldsymbol{\alpha}^\top(\mathbf{y} - \mathbf{m}) + \text{ld}_{\mathbf{K}}(\mathbf{W}) + n\log(2\pi\sigma_n^2)\right]. \quad (5)$$

Approximate inference (LA, VB, KL, ADF/EP) – in case of non-Gaussian likelihoods – requires these primitives as necessary building blocks. Depending on the covariance approximation method e.g. exact, sparse, grid-based, or state space, the four primitives differ in their implementation and computational complexity.

2.3. State space form of Gaussian processes

GP models with covariance functions with a Markovian structure can be transformed into equivalent state space models. The following exposition is based on Solin (2016, Ch. 3), which also covers how to derive the equivalent exact models for sum, product, linear, noise, constant, Matérn (half-integer), Ornstein–Uhlenbeck, and Wiener covariance functions. Other common covariance functions can be approximated by their Markovian counterparts, including squared exponential, rational quadratic, and periodic covariance functions.

Algorithm 2 Kalman (forward) filtering. For ADF, $(\mathbf{W}, \mathbf{b})$ are not required as inputs. Note, $\mathbf{b} = \mathbf{W}\mathbf{r}$.

Input: $\{t_i\}, \mathbf{y}$ # training inputs and targets
$\{\mathbf{A}_i\}, \{\mathbf{Q}_i\}, \mathbf{H}, \mathbf{P}_0$ # state space model
$\mathbf{W}, \mathbf{b}$ # likelihood eff. precision and location
for $i = 1$ **to** n **do**
  if $i == 1$ **then**
    $\mathbf{m}_i \leftarrow \mathbf{0};\ \mathbf{P}_i \leftarrow \mathbf{P}_0$ # init
  else
    $\mathbf{m}_i \leftarrow \mathbf{A}_i\mathbf{m}_{i-1};\ \mathbf{P}_i \leftarrow \mathbf{A}_i\mathbf{P}_{i-1}\mathbf{A}_i^\top + \mathbf{Q}_i$ # predict
  end if
  if has label y_i **then**
    $\mu_f \leftarrow \mathbf{H}\mathbf{m}_i;\ \mathbf{u} \leftarrow \mathbf{P}_i\mathbf{H}^\top;\ \sigma_f^2 \leftarrow \mathbf{H}\mathbf{u}$ # latent
    if ADF (assumed density filtering) **then**
      set (b_i, W_{ii}) to match moments of $\mathbb{P}(y_i|f_i)$ and $\exp(b_i f_i - W_{ii} f_i^2/2)$ w.r.t. latent $\mathcal{N}(f_i|\mu_f, \sigma_f^2)$
    end if
    $z_i \leftarrow W_{ii}\sigma_f^2 + 1;\ c_i \leftarrow W_{ii}\mu_f - b_i$
    $\mathbf{k}_i \leftarrow W_{ii}\mathbf{u}/z_i;\ \mathbf{P}_i \leftarrow \mathbf{P}_i - \mathbf{k}_i\mathbf{u}^\top$ # variance
    $\gamma_i \leftarrow -c_i/z_i;\ \mathbf{m}_i \leftarrow \mathbf{m}_i + \gamma_i\mathbf{u}$ # mean
  end if
end for
$\text{ld}_{\mathbf{K}}(\mathbf{W}) \leftarrow \sum_i \log z_i$

Algorithm 3 Rauch–Tung–Striebel (backward) smoothing.

Input: $\{\mathbf{m}_i\}, \{\mathbf{P}_i\}$ # Kalman filter output
$\{\mathbf{A}_i\}, \{\mathbf{Q}_i\}$ # state space model
for $i = n$ **down to** 2 **do**
  $\mathbf{m} \leftarrow \mathbf{A}_i\mathbf{m}_{i-1};\ \mathbf{P} \leftarrow \mathbf{A}_i\mathbf{P}_{i-1}\mathbf{A}_i^\top + \mathbf{Q}_i$ # predict
  $\mathbf{G}_i \leftarrow \mathbf{P}_{i-1}\mathbf{A}_i^\top\mathbf{P}^{-1};\ \Delta\mathbf{m}_{i-1} \leftarrow \mathbf{G}_i(\mathbf{m}_i - \mathbf{m})$
  $\mathbf{P}_{i-1} \leftarrow \mathbf{P}_{i-1} + \mathbf{G}_i(\mathbf{P}_i - \mathbf{P})\mathbf{G}_i^\top$ # variance
  $\mathbf{m}_{i-1} \leftarrow \mathbf{m}_{i-1} + \Delta\mathbf{m}_{i-1}$ # mean
  $\rho_{i-1} \leftarrow \mathbf{H}\Delta\mathbf{m}_{i-1}$ # posterior
end for
$\text{solve}_{\mathbf{K}}(\mathbf{W}, \mathbf{r}) = \boldsymbol{\alpha} \leftarrow \boldsymbol{\gamma} - \mathbf{W}\boldsymbol{\rho}$ # posterior

A state space model describes the evolution of a dynamical system at different time instances $t_i,\ i = 1, 2, \ldots$ by

$$\mathbf{f}_i \sim \mathbb{P}(\mathbf{f}_i|\mathbf{f}_{i-1}), \quad y_i \sim \mathbb{P}(y_i|\mathbf{f}_i), \quad (6)$$

where $\mathbf{f}_i := \mathbf{f}(t_i) \in \mathbb{R}^d$ and $\mathbf{f}_0 \sim \mathbb{P}(\mathbf{f}_0)$ with $\mathbf{f}_i$ being the latent (hidden/unobserved) variable and y_i being the observed variable. In continuous time, a simple dynamical system able to represent many covariance functions is given by the following linear time-invariant stochastic differential equation:

$$\dot{\mathbf{f}}(t) = \mathbf{F}\,\mathbf{f}(t) + \mathbf{L}\,\mathbf{w}(t), \quad y_i = \mathbf{H}\,\mathbf{f}(t_i) + \epsilon_i, \quad (7)$$

where $\mathbf{w}(t)$ is an s-dimensional white noise process, the measurement noise $\epsilon_i \sim \mathcal{N}(0, \sigma_n^2)$ is Gaussian, and $\mathbf{F} \in \mathbb{R}^{d\times d}$, $\mathbf{L} \in \mathbb{R}^{d\times s}$, $\mathbf{H} \in \mathbb{R}^{1\times d}$ are the feedback, noise effect,

and measurement matrices, respectively. The initial state is distributed according to $\mathbf{f}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{P}_0)$.

The latent GP is recovered by $f(t) = \mathbf{H}\mathbf{f}(t)$ and $\mathbf{w}(t) \in \mathbb{R}^s$ is a multivariate white noise process with spectral density matrix $\mathbf{Q}_c \in \mathbb{R}^{s\times s}$. For discrete values, this translates into

$$\mathbf{f}_i \sim \mathcal{N}(\mathbf{A}_{i-1}\mathbf{f}_{i-1}, \mathbf{Q}_{i-1}), \quad y_i \sim \mathbb{P}(y_i|\mathbf{H}\,\mathbf{f}_i), \tag{8}$$

with $\mathbf{f}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{P}_0)$. The discrete-time matrices are

$$\mathbf{A}_i = \mathbf{A}[\Delta t_i] = e^{\Delta t_i \mathbf{F}}, \tag{9}$$

$$\mathbf{Q}_i = \int_0^{\Delta t_i} e^{(\Delta t_k - \tau)\mathbf{F}} \mathbf{L}\,\mathbf{Q}_c\,\mathbf{L}^\top e^{(\Delta t_i - \tau)\mathbf{F}^\top} \mathrm{d}\tau, \tag{10}$$

where $\Delta t_i = t_{i+1} - t_i \geq 0$.

For stationary covariances $k(t, t') = k(t - t')$, the stationary state is distributed by $\mathbf{f}_\infty \sim \mathcal{N}(\mathbf{0}, \mathbf{P}_\infty)$ and the stationary covariance can be found by solving the Lyapunov equation

$$\dot{\mathbf{P}}_\infty = \mathbf{F}\,\mathbf{P}_\infty + \mathbf{P}_\infty\,\mathbf{F}^\top + \mathbf{L}\,\mathbf{Q}_c\,\mathbf{L}^\top = \mathbf{0}, \tag{11}$$

which leads to the identity $\mathbf{Q}_i = \mathbf{P}_\infty - \mathbf{A}_i\,\mathbf{P}_\infty\,\mathbf{A}_i^\top$.

2.4. Fast computation of $\mathbf{A}_i$ and $\mathbf{Q}_i$ by interpolation

In practice, the evaluation of the n discrete-time transition matrices $\mathbf{A}_i = e^{\Delta t_i \mathbf{F}}$ and the noise covariance matrices $\mathbf{Q}_i = \mathbf{P}_\infty - \mathbf{A}_i\mathbf{P}_\infty\mathbf{A}_i^\top$ (in the stationary case) for different values of Δt_i is a computational challenge. When the distribution of Δt_i in the dataset is narrow then computed matrices can be reused. However, when the distribution is wide, then computing $\mathbf{A}_i$ and $\mathbf{Q}_i$ consumes roughly 50% of the time on average if done naïvely.

Since the matrix exponential $\psi : s \mapsto e^{s\mathbf{X}}$ is smooth, its evaluation can be accurately approximated by convolution interpolation (Keys, 1981) as done for the covariance functions in the KISS-GP framework (Wilson & Nickisch, 2015; Wilson et al., 2015). The idea is to evaluate the function on a set of equispaced discrete locations $s_1, s_2, .., s_K$, where $s_j = s_0 + j \cdot \Delta s$ and interpolate $\mathbf{A} = e^{s\mathbf{X}}$ from the closest precomputed $\mathbf{A}_j = e^{s_j\mathbf{X}}$ i.e. use the 4 point approximation $\mathbf{A} \approx c_1\mathbf{A}_{j-1} + c_2\mathbf{A}_j + c_3\mathbf{A}_{j+1} + c_4\mathbf{A}_{j+2}$. The grid resolution Δs governs approximation accuracy.

The same interpolation can be done for the noise covariance matrices $\mathbf{Q}_i$. Finally, the number of matrix exponential evaluations can be reduced from n to K, which – for large datasets – is practically negligible. The accuracy of the interpolation depends on the underlying grid spacing Δs. In practice, we use an equispaced grid covering range $[\min_i \Delta t_i, \max_i \Delta t_i]$, but hybrid strategies, where the bulk of the mass of the Δt_i is covered by the grid and outliers are evaluated exactly, are – of course – possible. Very diverse sets of Δt_i with vastly different values, could benefit from a clustering with an individual grid per cluster.

2.5. State space computational primitives

In the following, we will detail how the SpInGP viewpoint of Grigorievskiy et al. (2017) can be used to implement the computational primitives of Sec. 2.2 with linear complexity in the number of inputs n. The covariance matrix of the latent GP $f(t)$ evaluated at the training inputs $t_1, \ldots, t_n$ is denoted $\mathbf{K} \in \mathbb{R}^{n\times n}$ and the (joint) covariance of the dynamical system state vectors $[\mathbf{F}_0; \mathbf{F}_1; ..; \mathbf{F}_n]$ is denoted by $\mathcal{K} \in \mathbb{R}^{(n+1)d\times(n+1)d}$. Defining the sparse matrix $\mathbf{G}^{n\times(n+1)d} = [\mathbf{0}_{n\times d}, \mathbf{I}_n \otimes \mathbf{H}]$, we obtain $\mathbf{K} = \mathbf{G}\mathcal{K}\mathbf{G}^\top$. Further, define the symmetric block diagonal matrix

$$\mathbf{Q} = \begin{bmatrix} \mathbf{P}_0 & \mathbf{0} & \ldots & \mathbf{0} \\ \mathbf{0} & \mathbf{Q}_1 & \ldots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \ldots & \mathbf{Q}_n \end{bmatrix} \in \mathbb{R}^{(n+1)d\times(n+1)d}$$

and $(n+1)d \times (n+1)d$ matrix $\mathbf{T} = \mathbf{A}^{-1} =$

$$\begin{bmatrix} \mathbf{I} & \mathbf{0} & \mathbf{0} & \ldots & \mathbf{0} \\ -\mathbf{A}[\Delta t_1] & \mathbf{I} & \mathbf{0} & \ldots & \mathbf{0} \\ \mathbf{0} & -\mathbf{A}[\Delta t_2] & \mathbf{I} & \ldots & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & -\mathbf{A}[\Delta t_n] & \ldots & \mathbf{I} \end{bmatrix}$$

of block tridiagonal (BTD) structure allowing to write

$$\mathcal{K}^{-1} = \mathbf{T}^\top\mathbf{Q}^{-1}\mathbf{T}, \text{ and } \mathcal{K} = \mathbf{A}\mathbf{Q}\mathbf{A}^\top,$$

where it becomes obvious that $\mathcal{K}^{-1}$ is a symmetric BTD; which is in essence the structure exploited in the SpInGP framework by Grigorievskiy et al. (2017).

2.5.1. Linear systems

Using the the matrix inversion lemma, we can rewrite $\left(\mathbf{K} + \mathbf{W}^{-1}\right)^{-1}$ as

$$= \mathbf{W} - \mathbf{W}\mathbf{G}\left(\mathcal{K}^{-1} + \mathbf{G}^\top\mathbf{W}^{-1}\mathbf{G}\right)^{-1}\mathbf{G}^\top\mathbf{W}$$
$$= \mathbf{W} - \mathbf{W}\mathbf{G}\mathbf{R}^{-1}\mathbf{G}^\top\mathbf{W},\ \mathbf{R} = \mathbf{T}^\top\mathbf{Q}^{-1}\mathbf{T} + \mathbf{G}^\top\mathbf{W}\mathbf{G}.$$

This reveals that we have to solve a system with a symmetric BTD system matrix $\mathbf{R}$, where $\mathbf{G}^\top\mathbf{W}\mathbf{G} = \mathrm{diag}([0; \mathbf{W}]) \otimes (\mathbf{H}^\top\mathbf{H})$. The only (numerical) problem could be the large condition of any of the constituent matrices of $\mathbf{Q}$ as it would render the multiplication with $\mathcal{K}^{-1}$ a numerical endeavour. Adding a small ridge α^2 to the individual constituents of $\mathbf{Q}$ i.e. use $\tilde{\mathbf{Q}}_i = \mathbf{Q}_i + \alpha^2\mathbf{I}$ instead of $\mathbf{Q}_i$ is a practical remedy. Finally, we have

$$\mathrm{solve}_{\mathbf{K}}(\mathbf{W}, \mathbf{R}) = \mathbf{W}\mathbf{R} - \mathbf{W}\mathbf{G}\mathbf{R}^{-1}\mathbf{G}^\top\mathbf{W}\mathbf{R}.$$

2.5.2. Matrix-vector multiplications

Using the identity $\mathcal{K} = \mathbf{A}\mathbf{Q}\mathbf{A}^\top$ from Grigorievskiy et al. (2017) and $\mathbf{K} = \mathbf{G}\mathcal{K}\mathbf{G}^\top$, we can write

$$\mathrm{mvm}_{\mathbf{K}}(\mathbf{R}) = \mathbf{G}\mathbf{T}^{-1}\mathbf{Q}\mathbf{T}^{-\top}\mathbf{G}^\top\mathbf{R}$$

where all constituents allow for fast matrix-vector multiplications. The matrix $\mathbf{G}$ is sparse, the matrix $\mathbf{Q}$ is block diagonal and the linear system with $\mathbf{T}$ is of BTD type. Hence, overall runtime is $\mathcal{O}(nd^2)$. For the derivatives $\frac{\mathrm{mvm}_\mathbf{K}(\mathbf{r})}{\partial\theta_i}$, we proceed component-wise using $\frac{\mathbf{Q}}{\partial\theta_i}$ and $\frac{\mathbf{T}^{-1}}{\partial\theta_i} = -\mathbf{T}^{-1}\frac{\mathbf{T}}{\partial\theta_i}\mathbf{T}^{-1}$. The derivative $\mathrm{d}\exp(\mathbf{X})$ of the matrix exponential $\exp(\mathbf{X})$ is obtained via a method by Najfeld & Havel (1995, Eqs. 10&11) using a matrix exponential of twice the size

$$\exp\left(\begin{bmatrix}\mathbf{X} & \mathbf{0}\\ \mathrm{d}\mathbf{X} & \mathbf{X}\end{bmatrix}\right) = \begin{bmatrix}\exp(\mathbf{X}) & \mathbf{0}\\ \mathrm{d}\exp(\mathbf{X}) & \exp(\mathbf{X})\end{bmatrix}.$$

2.5.3. Log determinants

The Kalman filter (Alg. 2) can be used to compute the log determinant $\mathrm{ld}_\mathbf{K}(\mathbf{W}) = \sum_i \log z_i$ in $\mathcal{O}(nd^3)$.

There are two kinds of derivatives of the log determinant required for learning (see Sec. 2.2). First, the hyperparameter derivatives $\frac{\partial \mathrm{ld}_\mathbf{K}(\mathbf{W})}{\partial\boldsymbol{\theta}}$ are computed component-wise using a differential version of the Kalman filter (Alg. 2) as described in Särkkä (2013, Appendix), the matrix exponential derivative algorithm by Najfeld & Havel (1995, Eqs. 10&11) and the identity $\frac{\partial \mathrm{ld}_\mathbf{K}(\mathbf{W})}{\partial\theta_j} = \sum_i \frac{1}{z_i}\frac{\partial z_i}{\partial\theta_j}$.

Second, the noise precision derivative is computed using the matrix determinant lemma

$$\frac{\partial \mathrm{ld}_\mathbf{K}(\mathbf{W})}{\partial\mathbf{W}} = \mathrm{diag}(\mathbf{G}\mathbf{R}^{-1}\mathbf{G}^\top)$$

where $\mathbf{R}$ and $\mathbf{G}$ are as defined in Sec. 2.5.1. Since $\mathbf{G}$ is a Kronecker product, we do not need to know $\mathbf{R}^{-1}$ completely; only the block diagonal part needs to be evaluated (Grigorievskiy et al., 2017, Sec. 3.1), which we achieve using the `sparseinv` package (Davis, 2014).

2.5.4. Predictions

Once the parameters $\boldsymbol{\alpha}$ and $\mathbf{W}$ have been obtained from one of the inference algorithms, predictions can be computed using Kalman filtering (Alg. 2) followed by RTS smoothing (Alg. 3) in linear time. The unseen test input(s) t_* are simply included into the data set, then the latent distribution can be extracted via $\mu_{f,i} = \mathbf{H}\mathbf{m}_i$ and $\sigma^2_{f,i} = \mathbf{H}\mathbf{P}_i\mathbf{H}^\top$. Assumed density filtering can be achieved by switching on the ADF flag in Algorithm 2.

Now that we have detailed the computational primitives, we describe how to use them to drive different approximate inference methods.

2.6. Laplace approximation (LA)

The GP Laplace approximation (Williams & Barber, 1998) is essentially a second order Taylor expansion of the GP posterior $\mathbb{P}(\mathbf{F}|\mathbf{y}) \propto \mathcal{N}(\mathbf{F}|\mathbf{m},\mathbf{K})\prod_i \mathbb{P}(y_i|f_i)$ around its mode $\hat{\mathbf{F}} = \arg\max_\mathbf{F} \mathbb{P}(\mathbf{F}|\mathbf{y})$ with $W_{ii} = -\partial^2 \log\mathbb{P}(y_i|f_i)/\partial f_i^2$ the likelihood curvature and

$$\log Z_{LA} = -\frac{1}{2}\left[\boldsymbol{\alpha}^\top \mathrm{mvm}_\mathbf{K}(\boldsymbol{\alpha}) + \mathrm{ld}_\mathbf{K}(\mathbf{W}) - 2\sum_i \log\mathbb{P}(y_i|\hat{f}_i)\right]$$

being an approximation to the (log) marginal likelihood. In practice, we use a Newton method with line searches. Similar primitives have been used in Kalman-based demand forecasting (Seeger et al., 2016) with linear models. Note that for log-concave likelihoods, the mode finding is a convex program.

2.7. Variational Bayes (VB)

The VB method uses convex duality to exactly represent the individual (log) likelihoods as a maximum over quadratics $\ell(f_i) = \log\mathbb{P}(y_i|f_i) = \max_{W_{ii}} b_i f_i - W_{ii} f_i^2/s + h(W_{ii})$ given that the likelihood is super Gaussian (e.g. Laplace, Student's t, logistic) (Gibbs & MacKay, 2000). Finally, inference can be interpreted as a sequence of Laplace approximations (Seeger & Nickisch, 2011) with the smoothed log likelihood $\ell_{\mathrm{VB}}(f_i) = \ell(g_i) + b_i(f_i - g_i)$ with smoothed latent $g_i = \mathrm{sign}(f_i - z_i)\sqrt{(f_i - z_i)^2 + v_i} + z_i$. The parameters (z_i, b_i) depend on the likelihood only e.g. $(z_i, b_i) = (y_i, 0)$ for Student's t and Laplace and $(z_i, b_i) = (0, y_i/2)$ for logistic likelihood and v_i is the marginal variance. The marginal likelihood lower bound takes the form

$$\log Z \geq \log Z_{\mathrm{VB}} = -\frac{1}{2}\left[\boldsymbol{\alpha}^\top \mathrm{mvm}_\mathbf{K}(\boldsymbol{\alpha}) + \mathrm{ld}_\mathbf{K}(\mathbf{W}) - 2\sum_i \ell_{\mathrm{VB}}(f_i) - 2\rho_{\mathrm{VB}}\right],$$

where ρ_{VB} collects a number of scalar terms depending on $(\mathbf{z}, \mathbf{b}, \mathbf{W}, \boldsymbol{\alpha}, \mathbf{m})$.

2.8. Direct Kullback–Leibler minization (KL)

Finding the best Gaussian posterior approximation $\mathcal{N}(\mathbf{b}|\boldsymbol{\mu}, \mathbf{V})$ by minimizing its Kullback–Leibler divergence to the exact posterior is a very generic inference approach (Opper & Archambeau, 2009) which has recently been made practical via a conjugate variational inference algorithm (Khan & Lin, 2017) operating as a sequence of GP regression steps. In particular, GP regression problems $j = 1, \ldots, J$ are solved for a sequence of Gaussian pseudo observations whose mean and precision $(\tilde{\mathbf{y}}_j, \tilde{\mathbf{W}}_j)$ are iteratively updated based on the first and second derivative of the convolved likelihood $\ell_{\mathrm{KL}}(f_i) = \int \ell(t)\mathcal{N}(f_i|t, v_i)\,\mathrm{d}t$ where v_i is the marginal variance until convergence. The marginal

likelihood lower bound takes the form

$$\log Z \geq \log Z_{\mathrm{KL}} = -\frac{1}{2}\left[\boldsymbol{\alpha}^\top \mathrm{mvm}_{\mathbf{K}}(\boldsymbol{\alpha}) + \mathrm{ld}_{\mathbf{K}}(\mathbf{W}) - 2\sum_i \ell_{\mathrm{KL}}(f_i) - 2\rho_{\mathrm{KL}}\right],$$

where the remainder $\rho_{\mathrm{KL}} = \mathrm{tr}(\mathbf{W}^\top \partial \mathrm{ld}_{\mathbf{K}}(\mathbf{W})/\partial \mathbf{W})$ can be computed using computational primitive 4.

2.9. Assumed density filtering (ADF) a.k.a. single-sweep Expectation propagation (EP)

In expectation propagation (EP) (Minka, 2001), the non-Gaussian likelihoods $\mathbb{P}(y_i|f_i)$ are replaced by unnormalized Gaussians $t_i(f_i) = \exp(b_i f_i - W_{ii} f_i^2/2)$ and their parameters (b_i, W_{ii}) are iteratively (in multiple passes) updated such that $\mathbb{Q}_{\neg i}(f_i)\mathbb{P}(y_i|f_i)$ and $\mathbb{Q}_{\neg i}(f_i)t(f_i)$ have $k = 0, \ldots, 2$ identical moments $z_i^k = \int f_i^k \mathbb{Q}_{\neg i}(f_i)t(f_i)\,\mathrm{d}f_i$. Here, $\mathbb{Q}_{\neg i}(f_i) = \int \mathcal{N}(\mathbf{f}|\mathbf{m}, \mathbf{K}) \prod_{j\neq i} t_j(f_j)\,\mathrm{d}\mathbf{f}_{\neg i}$ denotes the cavity distribution. Unlike full state space EP using forward and backward passes (Heskes & Zoeter, 2002), there is a single-pass variant doing only one forward sweep that is know as assumed density filtering (ADF). It is very simple to implement in the GP setting. In fact, ADF is readily implemented by Algorithm 2 when the flag "ADF" is switched on. The marginal likelihood approximation takes the form

$$\log Z_{\mathrm{ADF}} = -\frac{1}{2}\left[\boldsymbol{\alpha}^\top \mathrm{mvm}_{\mathbf{K}}(\boldsymbol{\alpha}) + \mathrm{ld}_{\mathbf{K}}(\mathbf{W}) - 2\sum_i \log z_i^0 - 2\rho_{\mathrm{ADF}}\right],$$

where the remainder ρ_{ADF} collects a number of scalar terms depending on $(\mathbf{b}, \mathbf{W}, \boldsymbol{\alpha}, \mathbf{m})$.

3. Experiments

The experiments focus on showing that the state space formulation delivers the exactness of the full naïve solution, but with appealing computational benefits, and wide applicability over GP regression and classification tasks. Sec. 3.1 assesses the effects of the fast approximations of $\mathbf{A}_i$ and $\mathbf{Q}_i$. Sec. 3.2 demonstrates the unprecedented computational speed, and Sec. 3.3 presents a comparison study including 12 likelihood/inference combinations. Finally, two large-scale real-data examples are presented and solved on a standard laptop in a matter of minutes.

3.1. Effects in fast computation of $\mathbf{A}_i$ and $\mathbf{Q}_i$

In the first experiment we study the validity of the interpolation to approximate matrix exponential computation (Sec. 2.4). The input time points of observations t_i were randomly selected from the interval [0, 12] and outputs y_i were generated from the sum of two sinusoids plus Gaussian noise: $y_i = 0.2\sin(2\pi\, t_i + 2) + 0.5\sin(0.6\pi\, t_i + 0.13) + 0.1\,\mathcal{N}(0, 1)$. The Δt_is were exponentially distributed since the time points followed a Poisson point process generation scheme. All results were calculated over 20 independent realizations.

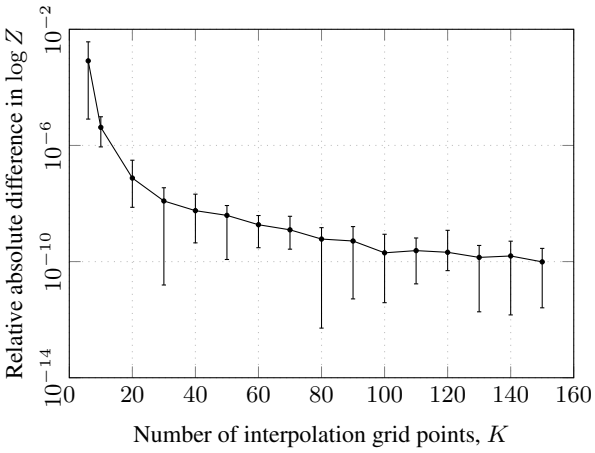


Figure 1. Relative differences in $\log Z$ with different approximation grid sizes for $\mathbf{A}_i$ and $\mathbf{Q}_i$, K, of solving a GP regression problem. Results calculated over 20 independent repetitions, mean±min/max errors visualized.

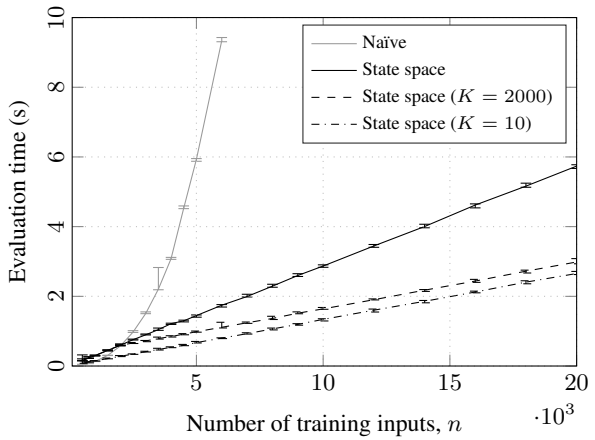


Figure 2. Empirical computational times of GP prediction using the GPML toolbox implementation as a function of number of training inputs, n, and degree of approximation, K. For all four methods the maximum absolute error in predicted means was 10^{-9}. Results calculated over ten independent runs.

For each generated dataset we considered GP regression (in the form of Sec. 2.5) with a Gaussian likelihood and Matérn ($\nu = 5/2$) covariance function. Initially, all the matrices $\mathbf{A}_i$ and $\mathbf{Q}_i$ were computed exactly. The results were compared to the approximate results of those matrices with various number of interpolation grid points K. The absolute relative difference between the approximated and not approximated marginal likelihood and its derivatives were computed. The results are given in Figure 1. The figure shows that the relative difference is decreasing with the number of grid

Table 1. A representative subset of supported likelihoods and inference schemes (for a full list, see Rasmussen & Nickisch, 2010). Results for simulated data with $n = 1000$ (around the break-even point of computational benefits). Results compared to respective naïve solution in mean absolute error (MAE). †The results for EP are compared against ADF explaining the deviation and speed-up.

Likelihood	Inference	MAE in $\boldsymbol{\alpha}$	MAE in $\mathbf{W}$	MAE in $\boldsymbol{\mu}_{f,*}$	$-\log Z$	$-\log Z_{\text{ss}}$	t/t_{ss}	Description
Gaussian	Exact	$< 10^{-4}$	$< 10^{-16}$	$< 10^{-14}$	-1252.29	-1252.30	2.0	Regression
Student's t	Laplace	$< 10^{-7}$	$< 10^{-6}$	$< 10^{-3}$	2114.45	2114.45	1.4	Regression,
Student's t	VB	$< 10^{-6}$	$< 10^{-6}$	$< 10^{-7}$	2114.72	2114.72	2.7	robust
Student's t	KL	$< 10^{-4}$	$< 10^{-4}$	$< 10^{-5}$	2114.86	2114.86	4.6	
Poisson	Laplace	$< 10^{-6}$	$< 10^{-4}$	$< 10^{-6}$	1200.11	1200.11	1.2	Poisson regression,
Poisson	EP/ADF†	$< 10^{-1}$	$< 10^{0}$	$< 10^{-2}$	1200.11	1206.59	39.5	count data
Logistic	Laplace	$< 10^{-8}$	$< 10^{-7}$	$< 10^{-7}$	491.58	491.58	1.3	Classification,
Logistic	VB	$< 10^{-6}$	$< 10^{-6}$	$< 10^{-6}$	492.36	492.36	2.3	logit regression
Logistic	KL	$< 10^{-7}$	$< 10^{-6}$	$< 10^{-7}$	491.57	491.57	4.0	
Logistic	EP/ADF†	$< 10^{-1}$	$< 10^{0}$	$< 10^{-1}$	491.50	525.46	48.1	
Erf	Laplace	$< 10^{-8}$	$< 10^{-6}$	$< 10^{-7}$	392.01	392.01	1.2	Classification,
Erf	EP/ADF†	$< 10^{0}$	$< 10^{0}$	$< 10^{-1}$	392.01	433.75	37.1	probit regression

points and finally saturates. Hence, increasing accuracy of approximation with the growing size of the interpolation grid. More figures with the accuracies of the derivatives computations can be found in the Supplementary material.

3.2. Computational benefits

The practical computational benefits of the state space form in handling the latent were evaluated in the following simulation study. We consider GP regression with a Matérn ($\nu = 3/2$) covariance function with simulated data from a modified sinc function ($6\sin(7\pi\, x)/(7\pi\, x + 1)$) with Gaussian measurement noise and input locations x drawn uniformly. The number of data points was increased step-wise from $n = 500$ to $n = 20{,}000$. The calculations were repeated for 10 independent realizations of noise.

The results (including results in following sections) were run on an Apple MacBook Pro (2.3 GHz Intel Core i5, 16 Gb RAM) laptop in Mathworks Matlab 2017b. All methods were implemented in the GPML Toolbox framework, and the state space methods only differed in terms of solving the continuous-time model for $\mathbf{A}_i$ and $\mathbf{Q}_i$ (see Sec. 2.4).

Figure 2 shows the empirical computation times for the $\mathcal{O}(n^3)$ naïve and $\mathcal{O}(n)$ state space results. The state space results were computed with no interpolation, and $\mathbf{A}_i$ and $\mathbf{Q}_i$ interpolated with $K = 2000$ and $K = 10$. The computation times with $K = 2000$ follow the exact state space model up to $n = 2000$. In terms of error in predictive mean over a uniform grid of 200 points, the maximum absolute error of state space results compared to the naïve results was 10^{-9}.

3.3. Numerical effects in non-Gaussian likelihoods

The previous section focused on showing that the latent state space computations essentially exact up to numerical errors or choices of interpolation factors in solving the continuous-time model. Delivering the computational primitives for approximate inference using LA, VB, KL, or EP should thus give the same results as if run through naïvely.

Table 1 shows a representative subset of combinations of likelihoods and inference scheme combinations (for a full list, see Rasmussen & Nickisch, 2010). For each model, appropriate data was produced by modifying the simulation scheme explained in the previous section (Student's t: 10% of observations outliers; Poisson: counts followed the exponentiated sinc function; Logistic/Erf: the sign function applied to the sinc). The mean absolute error in $\boldsymbol{\alpha}$, $\mathbf{W}$, and $\boldsymbol{\mu}_{f,*}$ between the state space and naïve solution are shown. The results are equal typically up to 4–6 decimals. It is probable that the state space approach shows accumulation of numerical errors. The large offsets in the EP values are due to our state space implementation being single-sweep (ADF). Here only $n = 1000$ data points were used, while Figure 2 shows that for regression the computational benefits only really kick-in in around $n = 2000$. For example in KL, the speed-up is clear already at $n = 1000$.

3.4. Robust regression of electricity consumption

We present a proof-of-concept large-scale robust regression study using a Student's t likelihood for the observations, where the data is inherently noisy and corrupted by outlying observations. We consider hourly observations of log electricity consumption (Hébrail & Bérard, 2012) for one household (in log kW) over a time-period of 1,442 days ($n = 34{,}154$, with 434 missing observations). We use a GP with a Student's t likelihood (with one degree of freedom) and a Matérn ($\nu = 3/2$) covariance function for predicting/interpolating values for missing days (state dimensionality $d = 2$). For inference we use direct KL minimization (Sec. 2.8). We evaluate our approach by 10-fold cross-validation over complete days, in this experiment with fixed hyperparameters, and obtain a predictive RMSE of 0.98 ± 0.02 and NLPD of 1.47 ± 0.01.

3.5. Airline accidents

Finally, we study the regression problem of explaining the time-dependent intensity of accidents and incidents of commercial aircraft. The data consists of dates of incidents that were scraped form (Wikipedia, 2018), and it covers 1210 incidents over the time-span of 1919–2017. We use a log Gaussian Cox process, an inhomogeneous Poisson process model for count data. The unknown intensity function $\lambda(t)$ is modeled with a log-Gaussian process such that $f(t) = \log \lambda(t)$. The likelihood of the unknown function corresponds to $\mathbb{P}(\{t_i\}|f) = \exp\left(-\int \exp(f(t))\,\mathrm{d}t + \sum_{i=1}^{n} f(t_i)\right)$. However, this likelihood requires non-trivial integration over the exponentiated GP. Møller et al. (1998) propose a locally constant intensity in subregions based on discretising the interval into bins. This approximation corresponds to having a Poisson model for each bin. The likelihood becomes $\mathbb{P}(\{t_i\}|f) \approx \prod_{j=1}^{N} \mathrm{Poisson}(y_j|\exp(f(\hat{t}_j)))$, where $\hat{t}_j$ is the bin coordinate and y_j the number of data points in it. This model reaches posterior consistency in the limit of bin width going to zero ($N \to \infty$) (Tokdar & Ghosh, 2007). Thus it is expected that the results improve the tighter the binning is.

We use a bin width of one day leading to $N = 35{,}959$ observations, and a prior covariance structure $k(t,t') = k_{\text{Matérn}}(t,t') + k_{\text{periodic}}(t,t')\,k_{\text{Matérn}}(t,t')$ capturing a slow trend and decaying time-of-year effect (period one year). The model state dimension is $d = 30$. For inference we used ADF (single-sweep EP, Sec. 2.9). All hyperparameters (except the period length) were optimized w.r.t. marginal likelihood, such that we first obtained a ball-park estimate of the parameters using one-month binning, and then continued optimizing with the full data set.

Figure 3 shows the time-dependent intensity $\lambda(t)$ that show a clear trend and pronounced periodic effects. The time course of the periodic effects are better visible in Figure 4 that show the gradual formation of the periodicity, and the more recent decay of the winter mode. We omit speculation of explaining factors in the data, but assume the effects to be largely explained by the number of operating flights. We further note that a wider bin size would deteriorate the analysis of the periodic peaks (they become 'smoothed' out), thus justifying the need for the large N as speculated above.

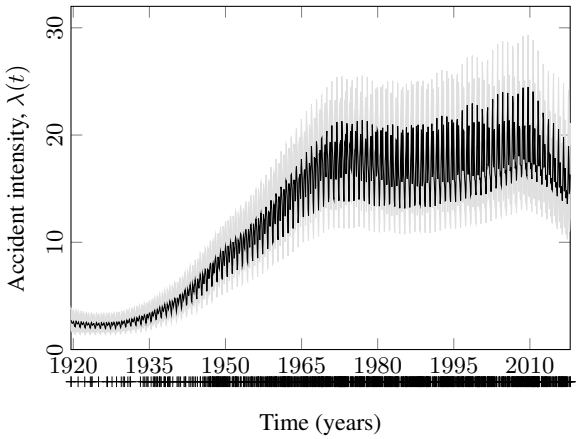


Figure 3. Intensity of aircraft incident modeled by a log Gaussian Cox process with the mean and approximate 90% confidence regions visualized ($N = 35{,}959$). The observed indicent dates are shown by the markers on the bottom.

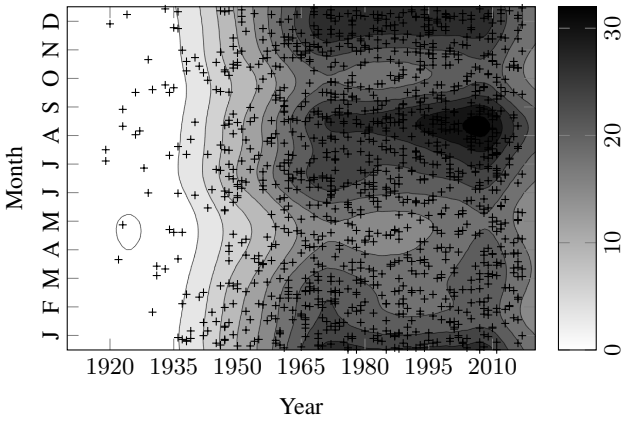


Figure 4. The time course of the seasonal effect in the airline accident intensity, plotted in a year vs. month plot (with wrap-around continuity between edges). Markers show incident dates. The bimodal yearly effect has started receding in the previous years.

4. Discussion and conclusion

Motivated by the computational constraints imposed by analytic intractability in the non-conjugate setting and cubic scaling, we propose to extend the state space representation of Gaussian processes to the non-Gaussian setting. We cast a range of approximate inference schemes using a small set of generic computational primitives to enable a unified treatment and show how to implement them using scalable algorithms relying on Kalman filters and dynamical system theory. We propose to use convolution interpolation to accelerate the expensive matrix exponential computations, which further reduces the runtime by a factor of two. We demonstrate computational benefits on a number of time series datasets to illustrate the tradeoffs and the achievable accuracy as compared to the dense setting.

Possible drawbacks are related to the cubic computational complexity in model state dimension (e.g. when considering several products of covariance functions), and problems related to floating point precision accumulating in the recursions when n is very large.

Overall, we conclude that for accurate scalable inference in GP time series, the state space viewpoint adds a valuable alternative to the computational toolbox of the modeling practitioner using our reference implementation.

References

Aravkin, A. Y., Burke, J. V., and Pillonetto, G. Sparse/robust estimation and Kalman smoothing with nonsmooth log-concave densities: Modeling, computation, and theory. *Journal of Machine Learning Research (JMLR)*, 14(1): 2689–2728, 2013.

Aravkin, A. Y., Burke, J. V., and Pillonetto, G. Optimization viewpoint on Kalman smoothing with applications to robust and sparse estimation. In *Compressed Sensing & Sparse Filtering*, pp. 237–280. Springer, 2014.

Bui, T. D., Yan, J., and Turner, R. E. A unifying framework for Gaussian process pseudo-point approximations using power expectation propagation. *Journal of Machine Learning Research (JMLR)*, 18(104):1–72, 2017.

Davis, T. `sparseinv`: Sparse inverse subset, 2014. URL https://mathworks.com/matlabcentral/fileexchange/33966-sparseinv--sparse-inverse-subset.

Frigola, R., Chen, Y., and Rasmussen, C. E. Variational Gaussian process state-space models. In *Advances in Neural Information Processing Systems*, pp. 3680–3688. Curran Associates, Inc., 2014.

Gibbs, M. N. and MacKay, D. J. Variational Gaussian process classifiers. *IEEE Transactions on Neural Networks*, 11(6):1458–1464, 2000.

Grigorievskiy, A. and Karhunen, J. Gaussian process kernels for popular state-space time series models. In *International Joint Conference on Neural Networks (IJCNN)*, pp. 3354–3363. IEEE, 2016.

Grigorievskiy, A., Lawrence, N., and Särkkä, S. Parallelizable sparse inverse formulation Gaussian processes (SpInGP). In *Proceedings of the IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, 2017.

Hartikainen, J. and Särkkä, S. Kalman filtering and smoothing solutions to temporal Gaussian process regression models. In *Proceedings of the IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, pp. 379–384, 2010.

Hartikainen, J., Riihimäki, J., and Särkkä, S. Sparse spatio-temporal Gaussian processes with general likelihoods. In *Proceedings of the International Conference on Artificial Neural Networks (ICANN)*, pp. 193–200, 2011.

Hébrail, G. and Bérard, A. Individual household electric power consumption data set, 2012. URL https://archive.ics.uci.edu/ml/datasets/individual+household+electric+power+consumption. Online: UCI Machine Learning Repository.

Hensman, J., Fusi, N., and Lawrence, N. D. Gaussian processes for big data. In *Uncertainty in Artificial Intelligence (UAI)*, pp. 282–290. AUAI Press, 2013.

Hensman, J., Durrande, N., and Solin, A. Variational Fourier features for Gaussian processes. *Journal of Machine Learning Research (JMLR)*, 18(151):1–52, 2018.

Heskes, T. and Zoeter, O. Expectation propagation for approximate inference in dynamic Bayesian networks. In *Uncertainty in Artificial Intelligence (UAI)*, pp. 216–223. Morgan Kaufmann Publishers Inc., 2002.

Keys, R. G. Cubic convolution interpolation for digital image processing. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 29(6):1153–1160, 1981.

Khan, M. and Lin, W. Conjugate-computation variational inference: Converting variational inference in non-conjugate models to inferences in conjugate models. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 54 of *PMLR*, pp. 878–887, 2017.

Krauth, K., Bonilla, E. V., Cutajar, K., and Filippone, M. AutoGP: Exploring the capabilities and limitations of Gaussian process models. 2017.

Kuss, M. and Rasmussen, C. E. Assessing approximate inference for binary Gaussian process classification. *Journal of Machine Learning Research (JMLR)*, 6(Oct):1679–1704, 2005.

Lázaro-Gredilla, M., Quiñonero-Candela, J., Rasmussen, C. E., and Figueiras-Vidal, A. R. Sparse spectrum Gaussian process regression. *Journal of Machine Learning Research (JMLR)*, 11:1865–1881, 2010.

Minka, T. Expectation propagation for approximate Bayesian inference. In *Uncertainty in Artificial Intelligence (UAI)*, volume 17, pp. 362–369, 2001.

Møller, J., Syversveen, A. R., and Waagepetersen, R. P. Log Gaussian Cox processes. *Scandinavian Journal of Statistics*, 25(3):451–482, 1998.

Naish-Guzman, A. and Holden, S. The generalized FITC approximation. In *Advances in Neural Information Processing Systems*, pp. 1057–1064. Curran Associates, Inc., 2008.

Najfeld, I. and Havel, T. F. Derivatives of the matrix exponential and their computation. *Advances in Applied Mathematics*, 16(3):321–375, 1995.

Nickisch, H. and Rasmussen, C. E. Approximations for binary Gaussian process classification. *Journal of Machine Learning Research (JMLR)*, 9(10):2035–2078, 2008.

Opper, M. and Archambeau, C. The variational Gaussian approximation revisited. *Neural Computation*, 21(3):786–792, 2009.

Quiñonero-Candela, J. and Rasmussen, C. E. A unifying view of sparse approximate Gaussian process regression. *Journal of Machine Learning Research (JMLR)*, 6(Dec): 1939–1959, 2005.

Rasmussen, C. E. and Nickisch, H. Gaussian processes for machine learning (GPML) toolbox. *Journal of Machine Learning Research (JMLR)*, 11:3011–3015, 2010. URL http://www.gaussianprocess.org/gpml/code.

Rasmussen, C. E. and Williams, C. K. I. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

Reece, S. and Roberts, S. An introduction to Gaussian processes for the Kalman filter expert. In *Proceedings of the 13th Conference on Information Fusion (FUSION)*. IEEE, 2010.

Särkkä, S. *Bayesian Filtering and Smoothing*, volume 3. Cambridge University Press, 2013.

Seeger, M. W. and Nickisch, H. Large scale Bayesian inference and experimental design for sparse linear models. *SIAM Journal on Imaging Sciences*, 4(1):166–199, 2011.

Seeger, M. W., Salinas, D., and Flunkert, V. Bayesian intermittent demand forecasting for large inventories. In *Advances in Neural Information Processing Systems*, pp. 4646–4654. Curran Associates, Inc., 2016.

Snelson, E. and Ghahramani, Z. Sparse Gaussian processes using pseudo-inputs. In *Advances in Neural Information Processing Systems*, pp. 1257–1264. Curran Associates, Inc., 2006.

Solin, A. *Stochastic Differential Equation Methods for Spatio-Temporal Gaussian Process Regression*. Doctoral dissertation, Aalto University, Helsinki, Finland, 2016.

Solin, A. and Särkkä, S. Explicit link between periodic covariance functions and state space models. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 33 of *PMLR*, pp. 904–912, 2014.

Solin, A. and Särkkä, S. Gaussian quadratures for state space approximation of scale mixtures of squared exponential covariance functions. In *Proceedings of the IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, 2014a.

Solin, A. and Särkkä, S. Hilbert space methods for reduced-rank Gaussian process regression. *arXiv preprint arXiv:1401.5508*, 2014b.

Titsias, M. K. Variational learning of inducing variables in sparse Gaussian processes. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 5 of *PMLR*, pp. 567–574, 2009.

Tokdar, S. T. and Ghosh, J. K. Posterior consistency of logistic Gaussian process priors in density estimation. *Journal of Statistical Planning and Inference*, 137(1): 34–42, 2007.

Wikipedia, 2018. URL https://en.wikipedia.org/wiki/List_of_accidents_and_incidents_involving_commercial_aircraft. [Online; retrieved 11-Jan-2018].

Williams, C. K. and Barber, D. Bayesian classification with Gaussian processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(12):1342–1351, 1998.

Wilson, A. G. and Nickisch, H. Kernel interpolation for scalable structured Gaussian processes (KISS-GP). In *International Conference on Machine Learning (ICML)*, volume 37 of *PMLR*, pp. 1775–1784, 2015.

Wilson, A. G., Dann, C., and Nickisch, H. Thoughts on massively scalable Gaussian processes. *arXiv preprint arXiv:1511.01870*, 2015.

Supplementary Material for State Space Gaussian Processes with Non-Gaussian Likelihood

In this appendix we provide further identities that are made possible by the recursive formulation together with some additional plots addressing the effects of possible approximations. The results in A. follow as a by-product of the algorithms presented in the main paper, and are provided here as additional material.

A. Recursions for $\boldsymbol{\alpha}$ and $\mathbf{L}$

The lower-triangular Cholesky factor $\mathbf{L} \in \mathbb{R}^{n \times n}$ given by

$$\mathbf{L}\,\mathbf{L}^\top = \mathbf{K} + \mathbf{W}^{-1} \tag{12}$$

can in the general case be solved efficiently in $\mathcal{O}(n^3)$. If the covariance function is Markovian, the following recursion can be used for forming the Cholesky factor in $\mathcal{O}(n^2)$ time complexity:

$$\mathbf{L}_{i,i} = \sqrt{s_i} \tag{13}$$

with $s_i = z_i / W_{ii}$ the innovation variance of Algorithm 2 for the diagonal and

$$\mathbf{L}_{i,j} = \mathbf{H} \left[\prod_{k=i}^{j-1} \mathbf{A}_k \right] \mathbf{k}_j \, \sqrt{s_i} \tag{14}$$

for the lower-triangular off-diagonal elements, $i = 1, 2, \ldots, n$ and $j < i$. The matrix product is constructed by iterated right-side multiplication.

The matrix-inverse of the Cholesky factor is also interesting as it gives the inverse of the original expression:

$$\mathbf{L}^{-1}\,\mathbf{L}^{-\mathsf{T}} = (\mathbf{K} + \mathbf{W}^{-1})^{-1}. \tag{15}$$

The inverse Cholesky factor can be constructed as follows in $\mathcal{O}(n^2)$ time complexity:

$$[\mathbf{L}^{-1}]_{i,i} = 1/\sqrt{s_i} \tag{16}$$

for the diagonal and

$$[\mathbf{L}^{-1}]_{j,i} = -\mathbf{H} \left[\prod_{k=i}^{j-1} (\mathbf{I} - \mathbf{k}_k \, \mathbf{H}) \right] \mathbf{k}_j / \sqrt{s_i} \tag{17}$$

for the lower-triangular off-diagonal elements, $i = 1, 2, \ldots, n$ and $j < i$.

Rather than directly solving $\boldsymbol{\beta} = \mathbf{L} \backslash \mathbf{r}$ or $\boldsymbol{\alpha} = \mathbf{L}^\top \backslash (\mathbf{L} \backslash \mathbf{r})$ by solving the linear systems by forward and backward substitution (in $\mathcal{O}(n^2)$) using the Cholesky factor $\mathbf{L}$ obtained in the previous section, the vectors $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ can be formed in $\mathcal{O}(n)$ time complexity (and $\mathcal{O}(n)$ memory) by the following forward and backward recursions using the filter forward and smoother backward passes.

The recursion for forward solving $\boldsymbol{\beta} \in \mathbb{R}^n$:

$$\beta_i = v_i / \sqrt{s_i}, \tag{18}$$

where $v_i = -c_i / W_{ii}$ and $s_i = z_i / W_{ii}$ are the Kalman filter (Alg. 2) innovation mean and variance at step $i = 1, 2, \ldots, n$.

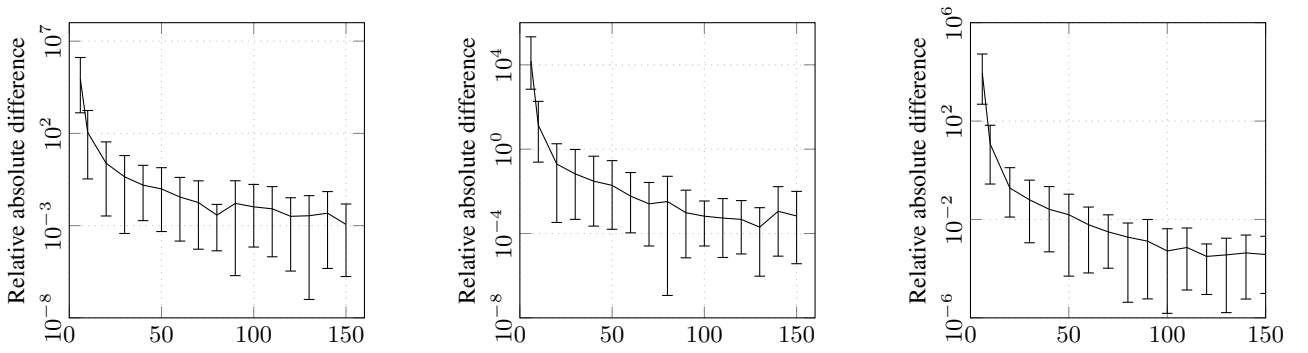
The calculation of $\boldsymbol{\alpha}$ can most easily be done by utilizing the Rauch–Tung–Striebel mean and gain terms as follows:

$$\alpha_i = \beta_i / \sqrt{s_i} - W_{ii} \mathbf{H} \Delta \mathbf{m}_i = \gamma_i - W_{ii} \mathbf{H} \Delta \mathbf{m}_i, \tag{19}$$

for $i = 1, 2, \ldots, n-1$ and $\alpha_n = \beta_n / \sqrt{s_n} = \gamma_n$.

B. Extra results of experiments in Sec. 3.1

Figure 5 provides additional plots for the interpolation experiment study in the main paper. The effects induced by approximations in solving $\mathbf{A}_i$ and $\mathbf{Q}_i$ are more pronounced for small K, when comparing the derivative terms (w.r.t. hyperparameters) of $\log Z$. Even for the derivative terms the errors drop quickly as a function of approximation degree K.



(a) Derivative w.r.t. ℓ (b) Derivative w.r.t. σ_f (c) Noise scale derivative

Figure 5. Relative absolute differences in derivatives of $\log Z$ with respect to covariance hyperparameters and noise scale. Different approximation grid sizes, K, for solving $\mathbf{A}_i$ and $\mathbf{Q}_i$ regression are evaluated. Results calculated over 20 independent repetitions, mean$\pm$min/max errors visualized.