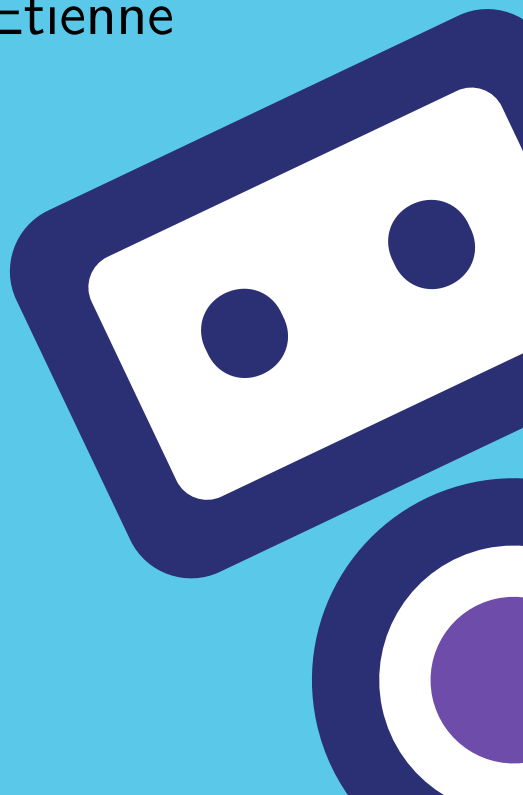
Gaussian process models using banded precisions matrices

Nicolas Durrande, PROWLER.io – Mines St-Étienne
(nicolas@prowler.io)

Second workshop on Gaussian processes
St-Étienne, Oct. 2018



This talk presents the paper

Banded Matrix Operators for Gaussian Markov Models in the Automatic Differentiation Era

Author 1 Institution 1 | **Author 2** Institution 2 | **Author 3** Institution 3 | **Author 4** Institution 4 | **Author 5** Institution 5

Abstract

Banded matrices can be used as precision matrices in several models including linear state-space models, some Gaussian processes, distribution is not tractable when the likelihood is not conjugate. These two limitations have been thoroughly studied over the past decades and several approaches have been proposed to overcome them. The most popular method for reducing computational complexity is

This talk presents the paper

Banded Matrix Operators for Gaussian Markov Models in the Automatic Differentiation Era

Author 1 Institution 1 | **Author 2** Institution 2 | **Author 3** Institution 3 | **Author 4** Institution 4 | **Author 5** Institution 5

Abstract

Banded matrices can be used as precision matrices in several models including linear state-space models, some Gaussian processes, distribution is not tractable when the likelihood is not conjugate. These two limitations have been thoroughly studied over the past decades and several approaches have been proposed to overcome them. The most popular method for reducing computational complexity is

which is a joint work with

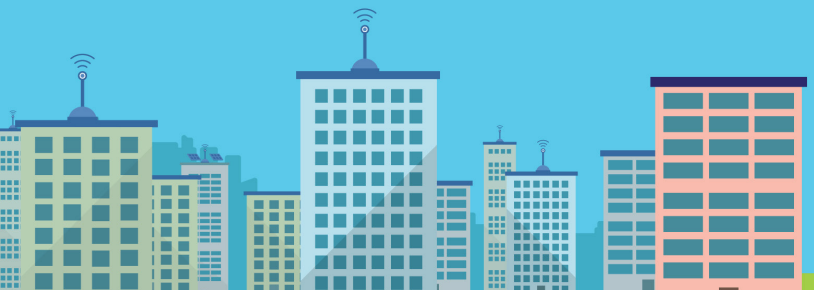
Vincent Adam

Lucas Bordeaux

Stefanos Eleftheriadis

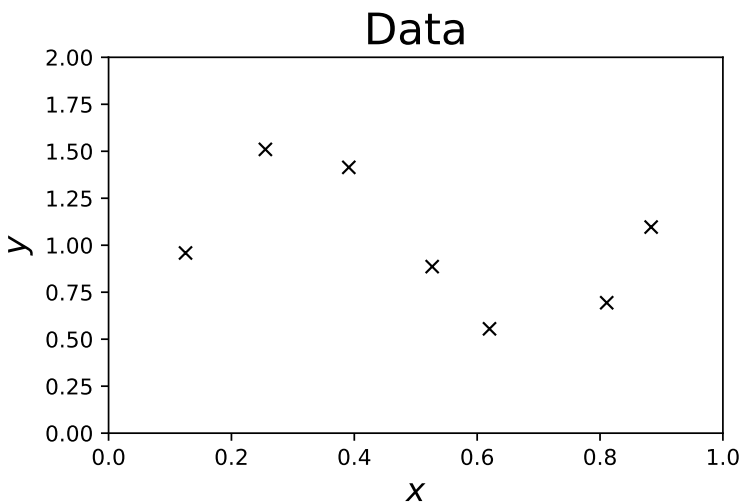
James Hensman

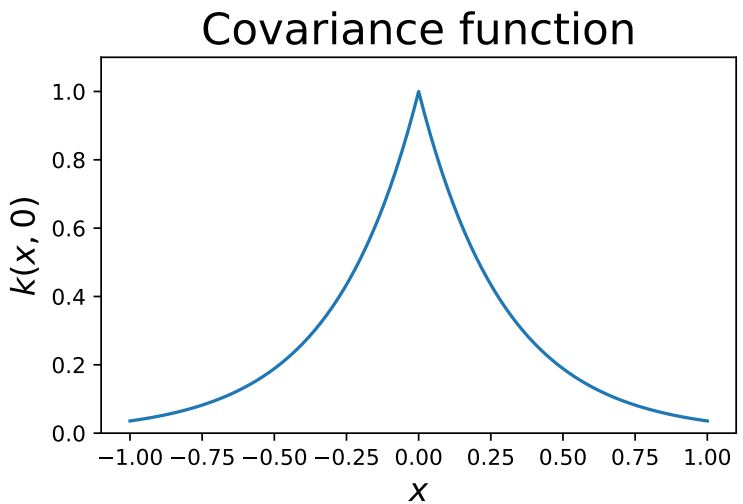
Motivating example



We want to build a GP regression model with dataset (X, Y), and a GP $f \sim \mathcal{N}(0, k)$ with exponential kernel:

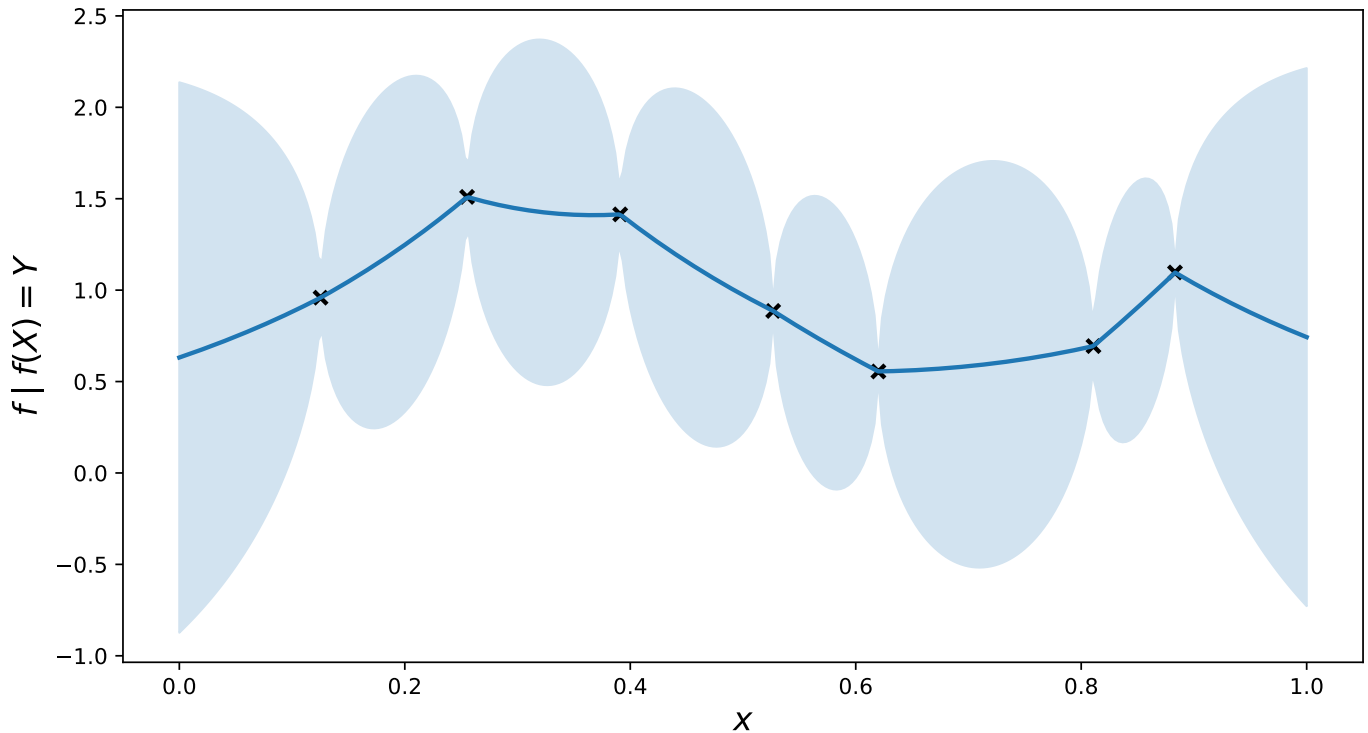
$$k(x, y) = \sigma^2 \exp\left(-\frac{|x - y|}{\theta}\right)$$





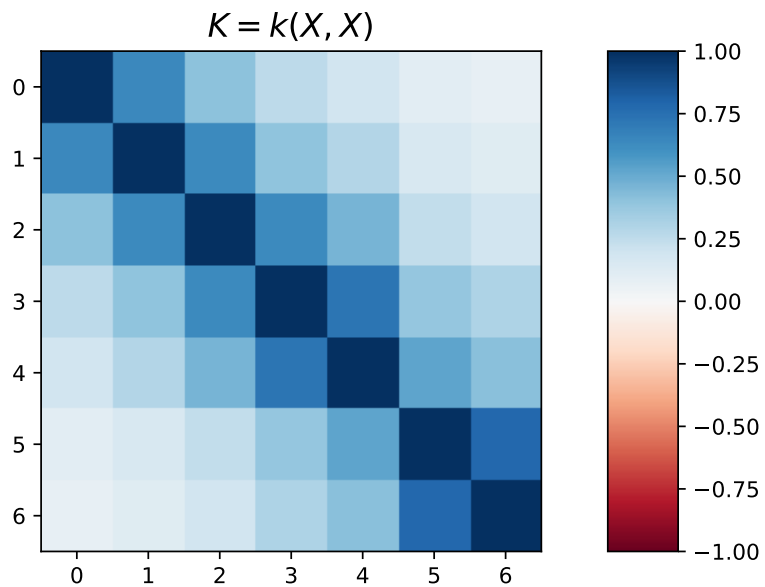
Given the data, f has conditional mean m and covariance c:

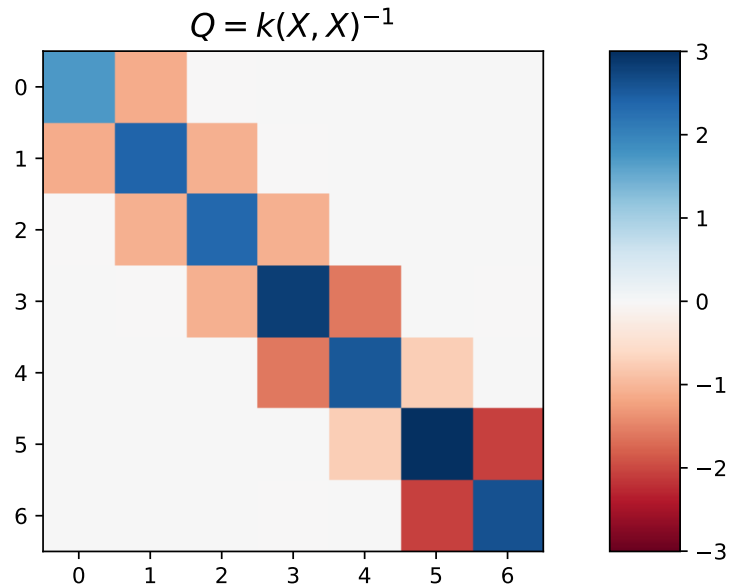
$$m(x) = k(x, X)k(X, X)^{-1}Y$$
$$c(x, y) = k(x, y) - k(x, X)k(X, X)^{-1}k(X, y)$$



The computationally expensive steps are computing the matrix $k(X, X)$ (which is $\mathcal{O}(n^2)$) and inverting it ($\mathcal{O}(n^3)$).

Let's have a look at $K = k(X, X)$ and its inverse $Q = K^{-1}$:





The precision matrix Q is tridiagonal... This is due to the Markov property of f.

An element of a covariance matrix is $K(x_i, x_j) = \mathrm{cov}\left(f(x_i), f(x_j)\right)$.

$\Rightarrow$ It depends on the **marginal distribution**.

Things are different for a precision matrix. for $I = \{i, j\}$ we have:

$$Q_{I,I} = \mathrm{cov}\left(f(X_I), f(X_I) \mid f(X_k),\ k \notin I\right)^{-1}$$

$\Rightarrow$ It depends on the **conditional distribution**.

As a consequence:

- $Q_{i,i} = \mathrm{var}\left(f(X_i) \mid f(X_k),\ k \neq i\right)^{-1}$
- $Q_{i,j} = 0 \Leftrightarrow f(X_i)$ and $f(X_j)$ are conditionally independent given the other observations.
- There is no equivalent of the $k(.,.)$ for the precision matrix

It is possible to compute directly the elements of the precision matrix.

For a GP with Matérn 1/2 kernel evaluated at X, we get:

$$Q = \sigma^{-2} \begin{pmatrix} \frac{1}{1-\lambda_1^2} & \frac{-\lambda_1}{1-\lambda_1^2} & & & \\ \frac{-\lambda_1}{1-\lambda_1^2} & \frac{1-\lambda_1^2\lambda_2^2}{(1-\lambda_1^2)(1-\lambda_2^2)} & \ddots & & \\ & \frac{-\lambda_2}{1-\lambda_2^2} & \ddots & \frac{-\lambda_{n-2}}{1-\lambda_{n-2}^2} & \\ & & \ddots & \frac{1-\lambda_{n-2}^2\lambda_{n-1}^2}{(1-\lambda_{n-2}^2)(1-\lambda_{n-1}^2)} & \\ & & & \frac{-\lambda_{n-1}}{1-\lambda_{n-1}^2} & \frac{1}{1-\lambda_{n-1}^2} \end{pmatrix}$$

where $\lambda_i = \exp(-(X_{i+1} - X_i)/\theta)$.

Note that Q has a **band structure**. Complexity is ~~$\mathcal{O}(n^2)$~~ $\mathcal{O}(n)$.

Sampling

Let $g \sim \mathcal{N}(0, Q^{-1})$ be a vector of length N.

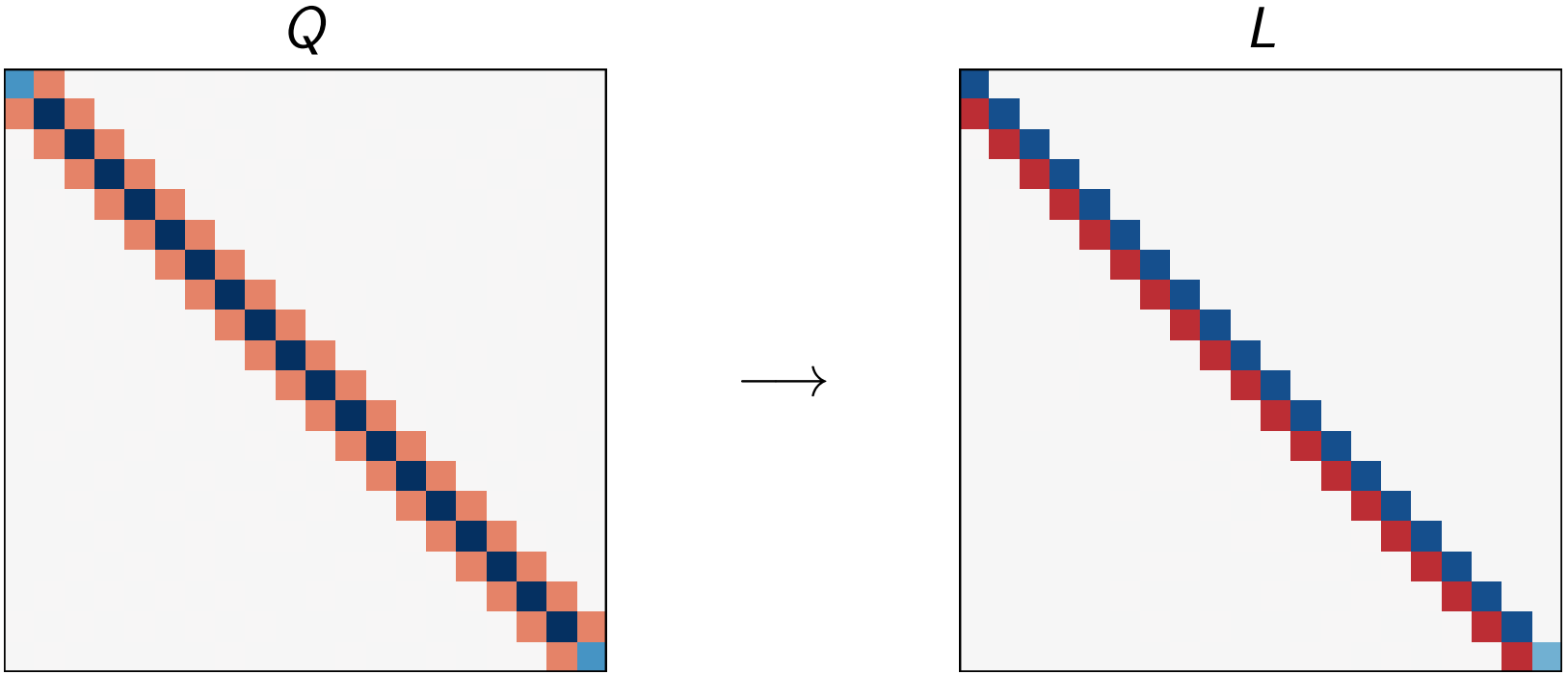
Given μ and Q, one can generate a sample of g by:

1. computing the Cholesky factorisation: $Q = LL^T$ $\mathcal{O}(N^3)$
2. sampling N independent variables $v_i \sim \mathcal{N}(0, 1)$ $\mathcal{O}(N)$
3. computing $\mu + L^{-T}v$ $\mathcal{O}(N^2)$

Let's do it!

Sampling

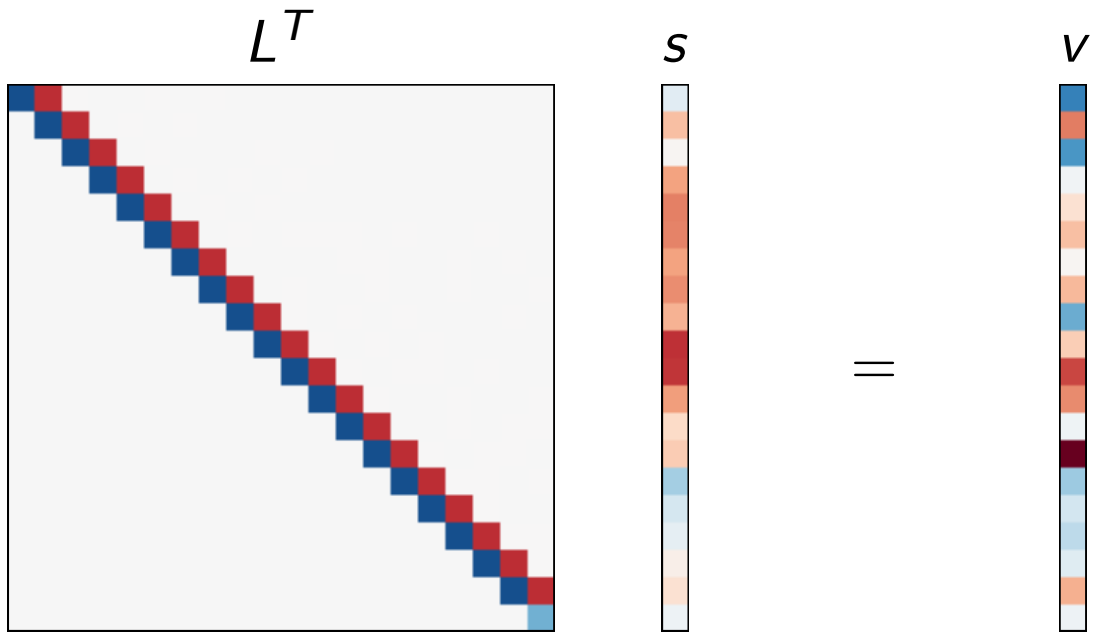
We start with a regular grid of 20 points on $[0, 1]$. We compute the Matérn 1/2 precision and its Cholesky factor:



It turns out that L is also a **banded matrix**!
Cholesky is ~~$\mathcal{O}(n^3)$~~ $\mathcal{O}(n)$.

Sampling

Computing L^{-T} yields a dense triangular matrix ($\mathcal{O}(n^3)$)... but solving $L^T s = v$ is easy ($\mathcal{O}(n)$)!



-Is this result surprising?

-Not really given the Markov property...

The banded structure of the precision allowed us to sample in $\mathcal{O}(N)$ time instead of $\mathcal{O}(N^3)$!

The banded structure of the precision allowed us to sample in $\mathcal{O}(N)$ time instead of $\mathcal{O}(N^3)$!

Question: Can we do the same for inference?

Inference with banded precision




There are many models leading to sparse/banded precisions:

Continuous time

- GP with the Markov property
  $\Rightarrow$ Brownian motion, ...
- Linear Stochastic differential equations $df(t) = Ff(t) + dB(t)$
  $\Rightarrow$ Brownian motion, GPs from the Matérn family

Discrete

- State Space Models: $f_t = A_{t-1} f_{t-1} + \varepsilon_t$
  $\Rightarrow$ Autoregressive models
- Gaussian Graphical models
- Gaussian Markov Random Fields (GMRF).

GMRF are Gaussian vectors indexed by the nodes of a graph. They are defined directly via their precision matrix, which is typically sparse.

Example

Let $G = (\{1, \ldots, N\}, E)$ be an undirected graph with adjacency matrix A, and degree matrix D.

We introduce the following norm for vectors indexed by the graph nodes:

$$||f||^2 = \sum_{(i,j)\in E} (f_i - f_j)^2 = f^t Q f$$

where $Q = D - A$. This can be seen as a RKHS with kernel Q^{-1}.

GP regression

Let $g \sim \mathcal{N}(0, Q^{-1})$ be a vector of length N and $\{A, B\}$ be a partition of $\{1, \ldots, N\}$.

The conditional distribution of $g_A \mid g_B$ is $\mathcal{N}(m, P^{-1})$ with:

$$m = Q_{A,A}^{-1} Q_{A,B} g_B$$
$$P = Q_{A,A}$$

If Q is banded (say bandwidth l), we can make this efficient by implementing dedicated operators:

- Cholesky factorisation ~~$\mathcal{O}(n^3)$~~ $\mathcal{O}(nl^2)$
- Triangular Solve ~~$\mathcal{O}(n^2)$~~ $\mathcal{O}(nl)$

Given a graph with N nodes and a vector Y of n observations at a subset of nodes X, we consider the following model:

$$
\begin{aligned}
f &\sim \mathcal{N}(0, Q^{-1}) \\
y_i &= f(x_i) + \varepsilon_i \qquad \text{with } \varepsilon_i \sim \mathcal{N}(0, \tau^2) \text{ i.i.d.}
\end{aligned}
$$

Where Q is banded and depends on some parameters θ.

Question: Can we efficiently estimate the model parameters θ, τ^2?

If we have observations only at a subset of node, then we can write $K = EQ^{-1}E^T$. The marginal likelihood of the model is then:

$$
\begin{aligned}
L(\theta, \tau^2) &= -\frac{n}{2}\log(2\pi) - \frac{1}{2}\log|EQ^{-1}E^T + \tau^2 I| - \frac{1}{2}Y^T(EQ^{-1}E^T + \tau^2 I)^{-1}Y \\
&= -\frac{n}{2}\log(2\pi) - \frac{1}{2}\log|Q + \tau^{-2}E^T E| + \frac{1}{2}\log|Q| - \frac{1}{2}\log|\tau^2 I| \\
&\quad -\frac{1}{2\tau^2}Y^T Y + \frac{1}{2\tau^4}Y^T E(Q + \tau^{-2}E^T E)^{-1}E^T Y \\
&= -\frac{n}{2}\log(2\pi) - \log|L| + \log|L_Q| - \frac{n}{2}\log\tau^2 - \frac{1}{2\tau^2}Y^T Y \\
&\quad + \frac{1}{2\tau^4}Y^T E L^{-T}L^{-1}E^T Y
\end{aligned}
$$

with $LL^T = (Q + \tau^{-2}E^T E)$ and $L_Q L_Q^T = Q$.

$\Rightarrow$ Previous operators allow doing this in $\mathcal{O}(Nl^2)$!

More fancy models...

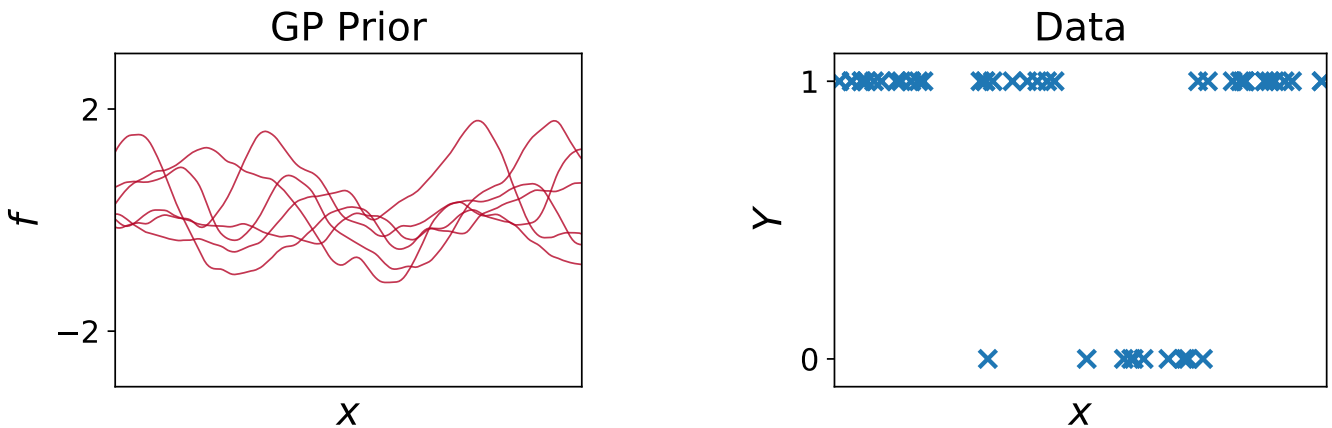
Another model we are interested in is:

$$f \sim \mathcal{N}(0, Q^{-1})$$

$$p(y \mid f) = \prod_{i+1}^{n} p_i(y_i \mid f_i)$$

Example

$y_i \mid f(x_i) \sim \mathcal{B}(\phi(x))$ with $\phi(x) = \frac{\exp(f(x))}{1+\exp(f(x))}$:

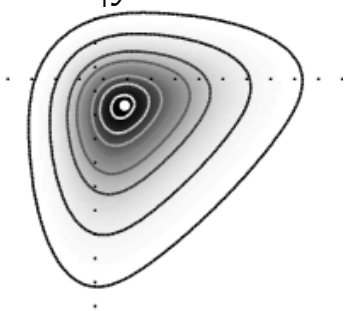


In this case, the conditional distribution of f given the data is not Gaussian anymore, and there is usually no analytical solution.

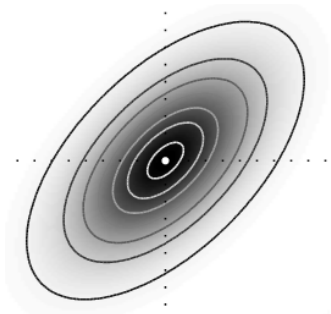
Example

If we consider the classification example with only two observations $Y = (1, 0)^T$, we have the following distributions over (f_1, f_2).

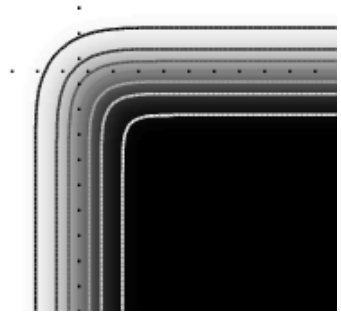
posterior $p_{f|y=Y}(F)$



prior $p_f(F)$



likelihood $p_{y|f=F}(Y)$



Source: Nickisch and Rasmussen, Approximations for Binary Gaussian Process Classification, JMLR 2008.

Variational Inference

Variational inference consists in optimizing the parameters of a distribution (say q) such that it approximates $p_{f|y=Y}$.

The objective function is a lower bound to the marginal likelihood:

$$\begin{aligned}\log\, p_y(Y) &\geq \mathrm{E}_{F\sim q} \log \frac{p_{y,f}(Y,F)}{q(F)} \\ &= \sum_{i=1}^{n} \mathrm{E}_{F\sim q} \log\, p_{y_i|f_i=F_i}(Y_i) - \mathrm{KL}[q \,\|\, p].\end{aligned}$$

The distribution q is typically chosen to be to be multivariate Gaussian.

In our settings, p_f has a banded precision, and we choose q to be the pdf of $\mathcal{N}(m_q, (L_q L_q^T)^{-1})$ where m_q is a vector and L_q a banded lower triangular matrix.

Variational Inference

Can we compute efficiently $\sum_{i=1}^{n} \mathrm{E}_{F\sim q} \log\, p_{y_i|f_i=F_i}(Y_i) - \mathrm{KL}[q \,\|\, p]$?

- The first term depends on the marginal distribution of q: we thus need m_q and the diagonal terms of Q_q^{-1}. The expectation may be obtained analytically or numerically.
- The second one is

$$\mathrm{KL}[q \,\|\, p] = \frac{1}{2}\Big(\mathrm{tr}(Q_q^{-1}Q_p) + 2\sum_i \log[L_q]_{ii} - \log[L_p]_{ii} + (m_p - m_q)^T L_p L_p^T (m_p - m_q) - N\Big).$$

  The trace term looks challenging, but it boils down to the sum of an element-wise product, where Q_p is banded. Compute Q_q^{-1} only inside the band is sufficient!

The new operators required are **sparse inverse subset**, and a **banded-matrix product**. They are $\mathcal{O}(Nl^2)$ and $\mathcal{O}(Nl)$.

MCMC

MCMC is the classical approach to perform exact inference when no analytical solutions are available.

Since f typically has high correlations, it is common to apply whitening $f = L^{-T} v$ and to do the sampling directly on v.

Given a prior on the hyper-parameters, the log joint density is then:

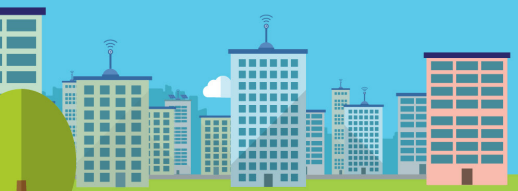
$$\log\, p(v, \theta, y) = \log\, p(v) + \log\, p(\theta) + \sum_{i=1}^{n} \log\, p(y_i | \theta, (L_Q^{-\top} v)_i).$$

One can see that it does not require any supplementary operator.

For all the operators we have listed so far, a version dedicated to banded matrices can already be found in the literature :)

If we want the methods to be efficient, we need to have access to the gradients of our objectives :(

Implementation in an autodifferentiation framework



The principle of autodifferentiation libraries such as Theano, PyTorch or TensorFlow is that each operator comes with an implementation of its derivative.

Using the chain rule, the autodifferentiation framework can then compute the derivative of any variable with respect to any other.

In practice, these frameworks typically use reverse mode differentiation: given a chain of operations $X \to Y \to \cdots \to c$ (with X, Y matrices), they use $\left[\frac{\partial c}{\partial Y_{ij}}\right]$ to compute $\left[\frac{\partial c}{\partial X_{ij}}\right]$.

With the notation $\bar{Y} = \left[\frac{\partial c}{\partial Y_{ij}}\right]$, we have

$$dc = \sum_{ij} \frac{\partial c}{\partial Y_{ij}} dY_{ij} = tr(\bar{Y}^T dY).$$

$\bar{X}$ can be obtained using the relation between dY and dX and permutations inside the trace.

Example

Consider the following operations: $X \to Y = XX^T \to c = sum(Y)$.

- $\bar{Y}$ is the matrix with entries $\frac{\partial c}{\partial Y_{ij}} = 1$
- $\bar{X}$ is $(\bar{Y} + \bar{Y}^T)X$:

$$\begin{aligned} dc &= tr(\bar{Y}^T dY) = tr(\bar{Y}^T (dX\, X^T + X\, dX^T)) \\ &= tr(\bar{Y}^T dX\, X^T) + tr(\bar{Y}^T X\, dX^T) \\ &= tr(X^T \bar{Y}^T dX) + tr(dX X^T \bar{Y}) \\ &= tr(X^T \bar{Y}^T dX) + tr(X^T \bar{Y} dX) \\ &= tr(X^T (\bar{Y}^T + \bar{Y}) dX) \end{aligned}$$

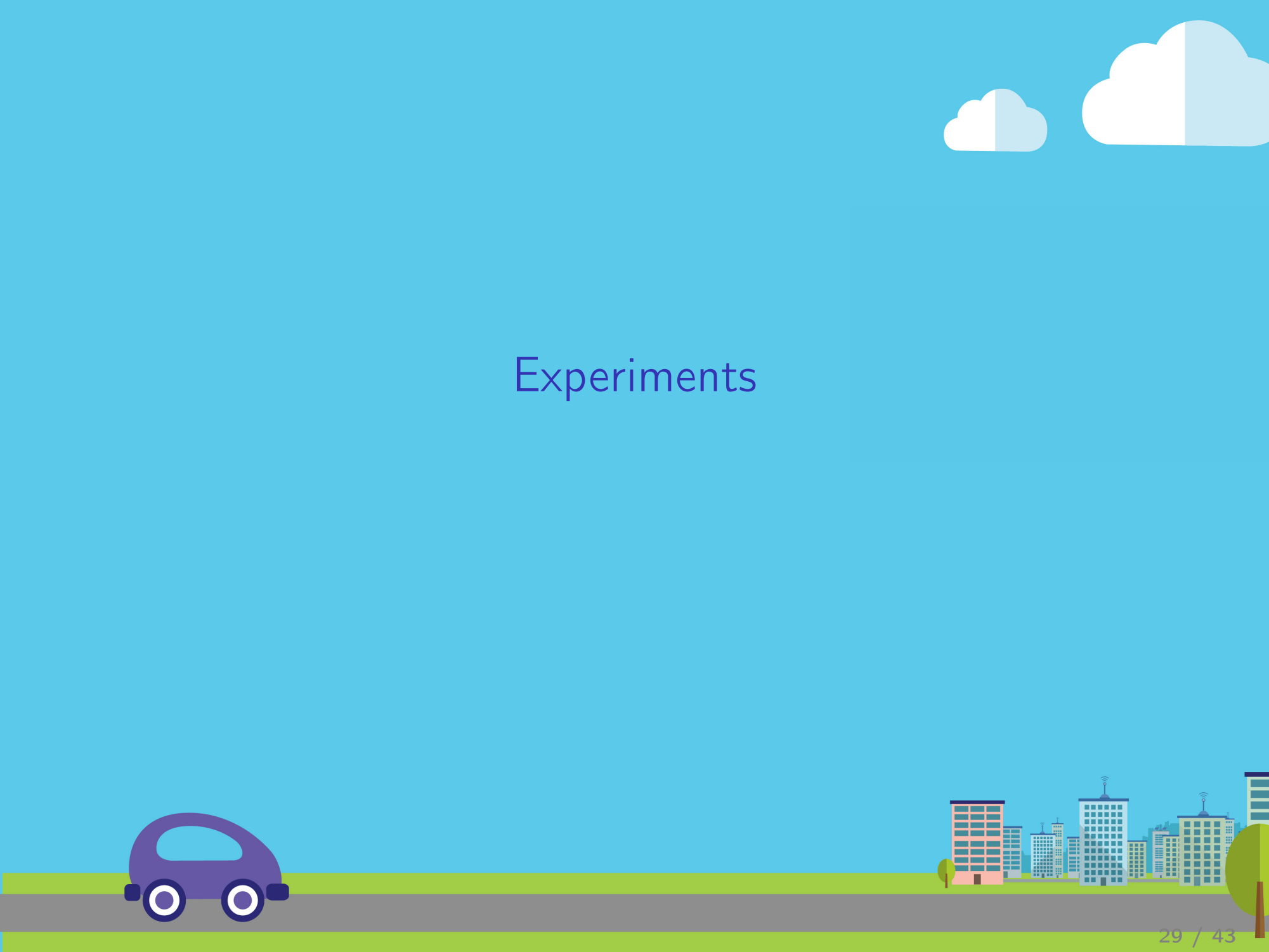
Applying this to our operators gives

Symbol	Input	Forward	Reverse Mode Sensitivities	
$\mathbb{P}$	B_1, B_2	$P = B_1 B_2$	$\bar{B}_1 = \mathbb{P}(\bar{P}, B_2^T)$	$\bar{B}_2 = \mathbb{P}(B_1^T, \bar{P})$
$\mathbb{P}$	B, v	$p = Bv$	$\bar{B} = \mathbb{O}(\bar{p}, v)$	$\bar{v} = \mathbb{P}(B^T, \bar{p})$
$\mathbb{O}$	m, v	$O = mv^T$	$\bar{m} = \mathbb{P}(\bar{O}, v)$	$\bar{v} = \mathbb{P}(\bar{O}^T, m)$
$\mathbb{S}$	L, v	$s = L^{-1}v$	$\bar{v} = \mathbb{S}(L^T, \bar{s})$ $\bar{L}^T = -\mathbb{O}(\mathbb{S}(L, v), \mathbb{S}(L^T, \bar{s}))$	
$\mathbb{S}$	L, B	$S = L^{-1}B$	$\bar{B} = \mathbb{S}(L^T, \bar{S})$ $\bar{L}^T = -\mathbb{P}(\mathbb{S}(L, B), \mathbb{S}(L^T, \bar{S})^T)$	

These expressions require the forward operators we already have!

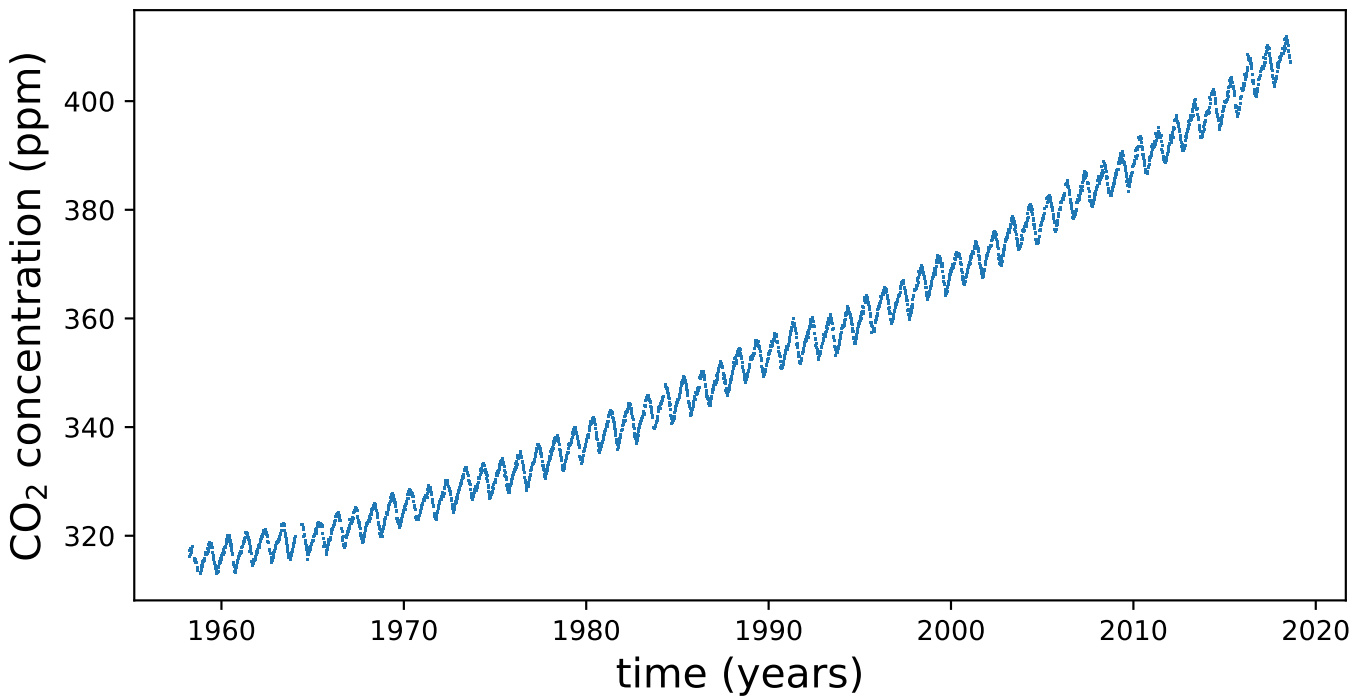
For Cholesky and sparse inverse subset, things are a bit tricky:
$\Rightarrow$ we modified existing non banded code and we used 'tangent'.

Experiments



Exp. 1: Mauna Loa CO_2 dataset

We looked at the weekly measurement of CO_2 concentration at the Mauna Loa observatory (Hawaii):



This dataset consists of 3082 observations.

Exp. 1: Mauna Loa CO_2 dataset

We consider 3 implementations of a GP model with exponential kernel:

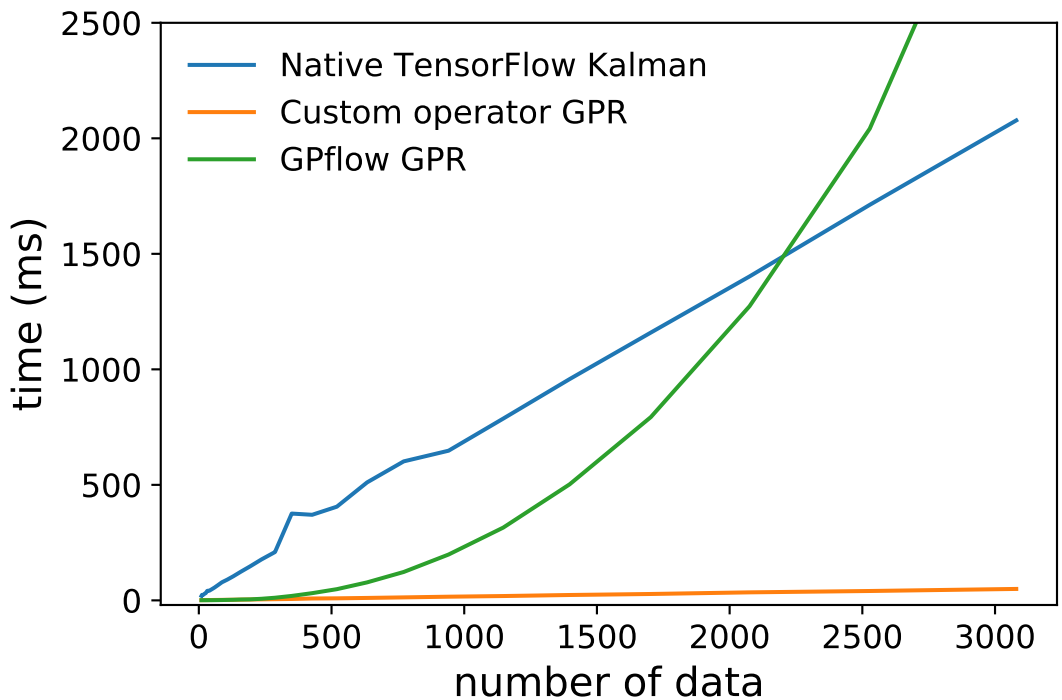
GPflow	a classic GPR implementation based on covariances
Kalman	a Kalman filter implemented in TensorFlow
Custom	the proposed framework with banded prec and custom ops

Note that the model settings correspond to a bandwidth of one.

Exp. 1: Mauna Loa CO_2 dataset

We then compare the execution time for computing the log-likelihood and its gradients (w.r.t the kernel parameters).

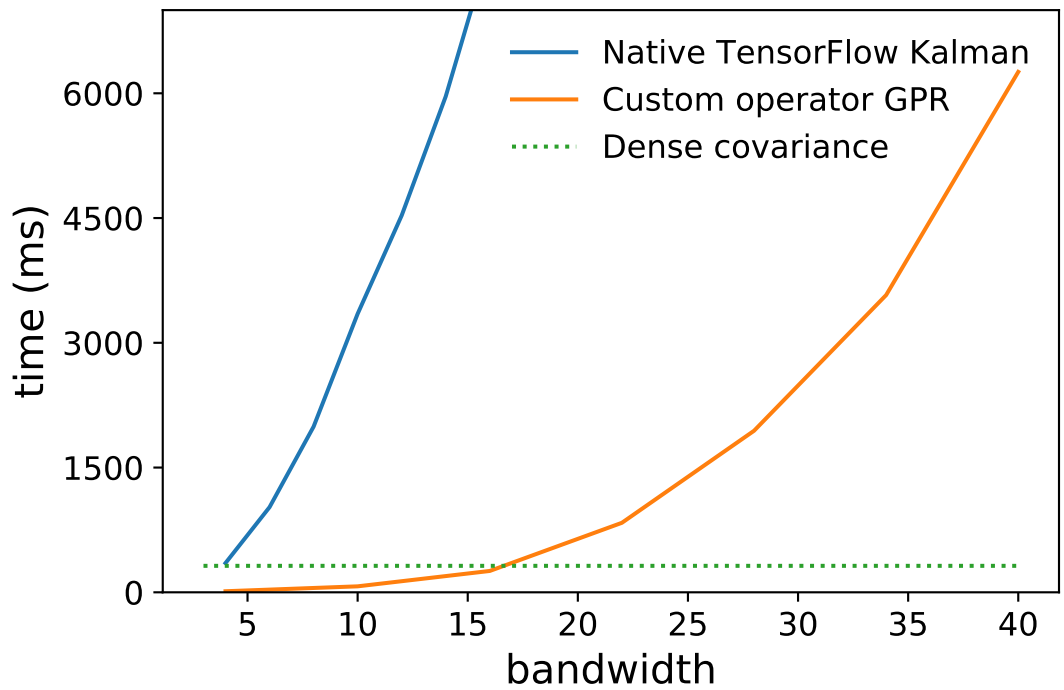
Considering subsets of the data allows us to study the influence of the number of observations:



Exp. 1: Mauna Loa CO_2 dataset

Similarly, we can study the influence of the **precision bandwidth** on the log-likelihood execution time.

To do so we consider more complex models with quasi-periodic components.



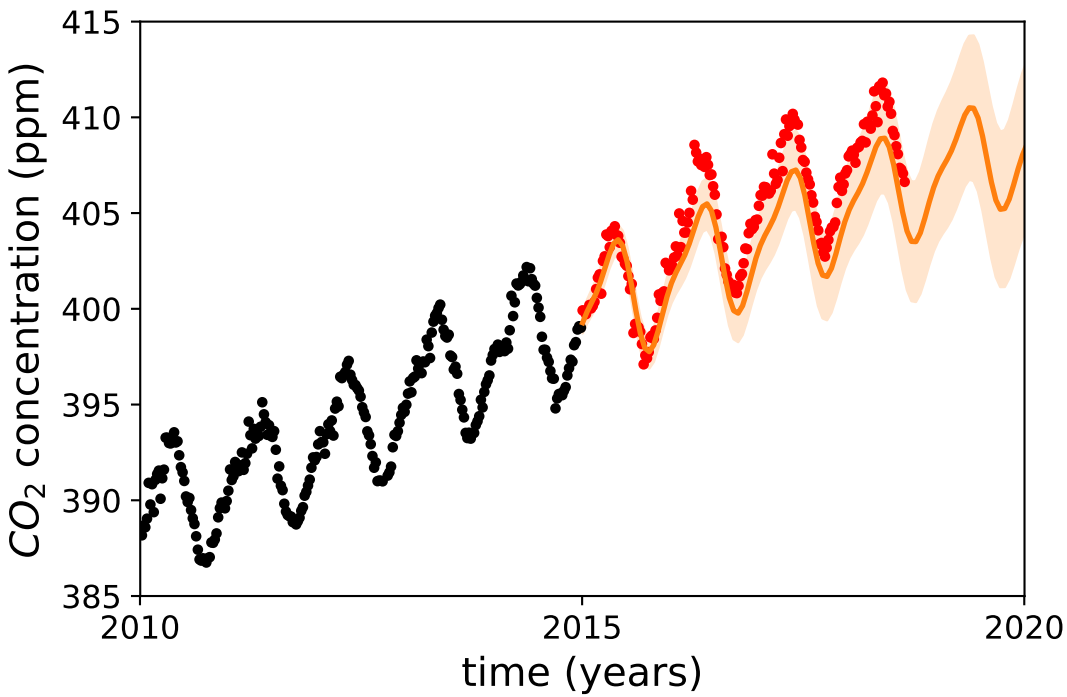
On this example the number of observations is fixed to 1500.

Exp. 1: Mauna Loa CO_2 dataset

The following figure shows the predictions for the model with kernel

$$k(d) = k_{3/2}(d) + k_{1/2}(d)\cos(\omega d) + k_{1/2}(d)\cos(2\omega d)$$

which has a bandwidth of 5:



Exp. 2: Porto taxi dataset

This dataset consists of taxi GPS location for a year.

Location of passenger pick-ups.

Exp. 2: Porto taxi dataset

It has already been successfully modelled by Cox processes:

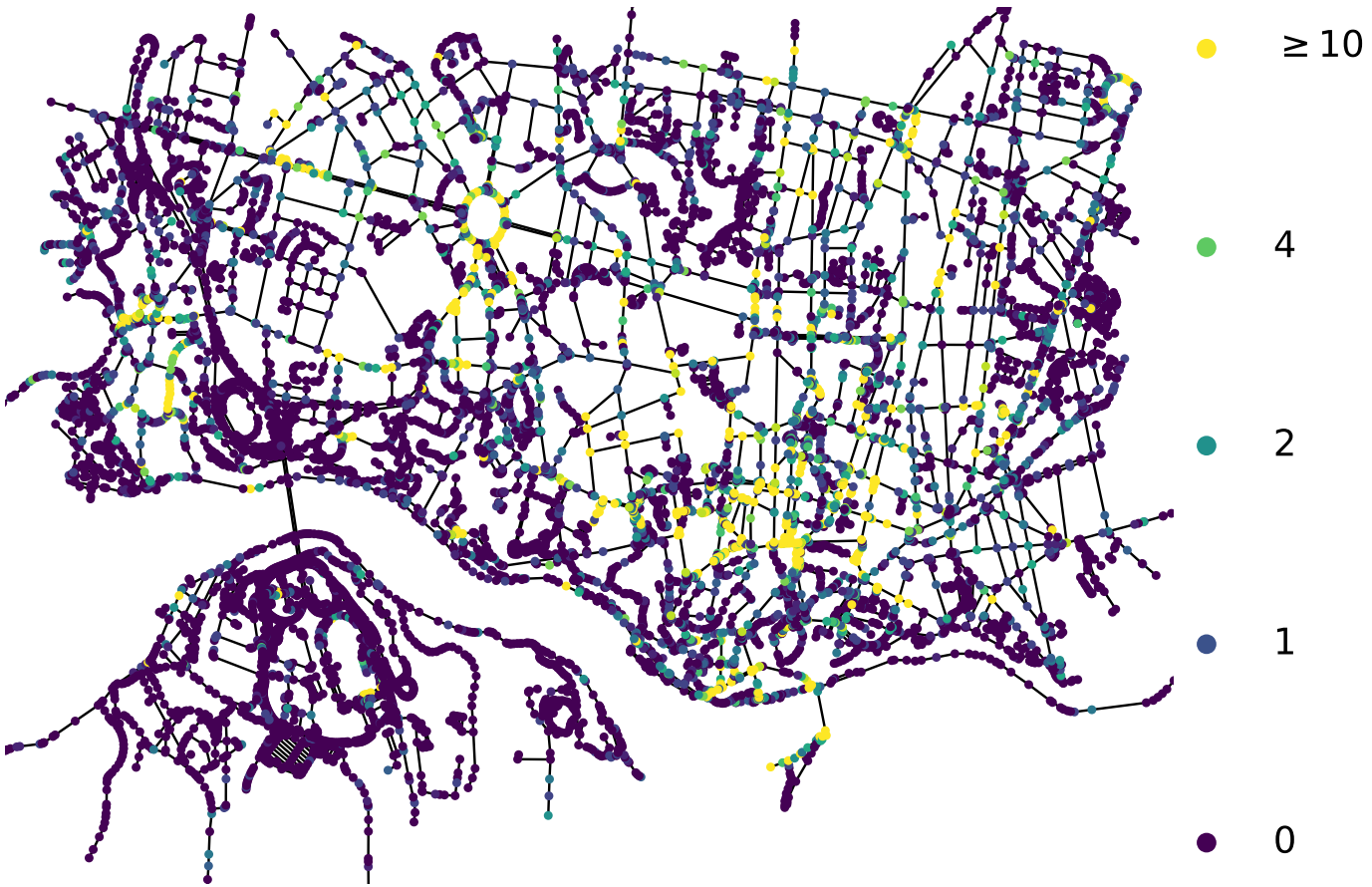
$$f \sim \mathcal{GP}(0, k)$$
$$y_{\mathcal{D}} \sim \mathcal{P}\left(\int_{\mathcal{D}} f(x)^2 dx\right)$$

This model assumes there is a smooth underlying rate.

Ref: S. John and J. Hensman, Large-scale Cox process with variational Fourier Features, ICML 2018

Exp. 2: Porto taxi dataset

We introduce a graph corresponding to the road network and clip the data to the graph nodes (first three weeks):



Exp. 2: Porto taxi dataset

We choose the following Cox processes model with a GP indexed by the graph nodes:

$$f \sim \mathcal{N}(0, Q^{-1})$$
$$y_i \sim \mathcal{P}\left(\exp(f_i) w_i\right)$$

where w_i is the length of the edges leading to node i.

Q is define such that the norm it generates is the sum of Matérn 1/2 norm on each edge:

$$g^T Q h = \frac{1}{\sigma^2} \sum_{(i,j)\in E} \frac{1}{1-\lambda_{i,j}} (g_i \; g_j) \begin{pmatrix} 1 & -\lambda_{i,j} \\ -\lambda_{i,j} & 1 \end{pmatrix} \begin{pmatrix} h_i \\ h_j \end{pmatrix} - \frac{1}{2} g_i h_i - \frac{1}{2} g_j h_j$$

where $\lambda_{i,j} = \sigma^2 \exp(-d_{i,j}/\ell)$ with $\sigma^2 = 10$, $\ell = 10^4$.

Exp. 2: Porto taxi dataset

Using variational inference, we obtain the following posterior on f:

Exp. 2: Porto taxi dataset

It corresponds to the following predictions for y:

Exp. 2: Porto taxi dataset

Using the next three weeks of the data as a test set, we compare the likelihood of three candidates:

Variational inference	-15778.5
Hamiltonian Monte Carlo	-15873.6
baseline using empirical rates	-17146.6

Our models perform better than the baseline, which means that our initial assumptions make sense!

Conclusion

In a nutshell...

The proposed framework is applicable to a wide class of models such as *State space models*, *Gaussian Markov Random Fields* or *Continuous Markovian processes*.

It allows to perform state of the art inference: *Maximum likelihood*, *Variational inference* or *Hamiltonian Monte Carlo*.

The resulting models show **interesting behaviours** and inference is **fast**!