Heikki Peura 62678U

Sparse Approximate Gaussian Process Regression

Mat-2.4108 Independent Research Projects in Applied Mathematics

Espoo, April 15, 2010.
Contents

1 Introduction 2

2 Gaussian process regression 4
   2.1 Motivation 4
   2.2 Regression for Gaussian data 5
   2.3 Covariance functions 9

3 Sparse approximations 11
   3.1 Earlier approaches 11
   3.2 Variational model selection 14

4 Case study 18

5 Conclusions 22

A Matrix operations 24
1 Introduction

In the digital era, there is a growing demand for models which make sense of the wealth of data available in virtually every field of our society. Supervised learning is concerned with this problem: learning input-output mappings from empirical data. Depending on the nature of the output, it is called either regression (for continuous variables) or classification (for discrete variables). Examples of both tasks are plentiful, from econometrics (regression) to email spam filtering (classification). To make inferences, proposed models are fitted to data, then used to make predictions about further inputs.

In general, the models used for these tasks are parametric, meaning that the model assumes a certain functional form, for example commonly used linear regression takes the effect between inputs and outputs as linear. These restrictions on the class of functions may yield poor predictive performance if the effect cannot be captured by a linear model. Here, Gaussian process (GP) models provide an interesting alternative: instead of choosing a function class, we essentially place prior probabilities to all possible functions (with higher probabilities to those that we consider more likely) and use a Bayesian framework to perform inferences.

A Gaussian process is a generalization of the Gaussian (i.e., normal) probability distribution to a stochastic process, that is, from a random variable to a function. Due to certain important properties, such as consistency of the inferences, GP models provide an elegant framework for probabilistic inference. While Gaussian processes are not a new idea, their use for supervised learning only dates back to the 1990s, when the machine learning community was seeking ways to better apply neural networks: the probabilistic GP framework soon gained momentum on its own. Under certain conditions, Gaussian processes can in fact be mathematically equivalent to neural networks, as well as Bayesian linear models, splines and time series models.

An important question in GP literature is the computational cost of the inference. GP models involve matrix inversion, which has complicated their use for large datasets. Several approximative methods have been proposed to deal with this issue through the selection of a set of inputs to represent the whole dataset. While these methods have been able to reduce the computational burden, they have been subject to criticism for other reasons, namely overfitting to data. Recently, Titsias (2009) proposed a variational method taking into account this issue.
This work seeks to compare the various sparse approximation methods for Gaussian process models. In section 2, we go through the GP regression framework, generally following Rasmussen and Williams (2006) and Kuss (2006), while section 3 provides a review of the available sparse approximation methods, detailing the novel variational method by Titsias. In section 4, the methods are compared for a regression problem with spatial data: the records of precipitation in the US for the year 1995. Section 5 concludes the work.
2 Gaussian process regression

2.1 Motivation

Consider a set of noisy pairwise observations, \( \mathcal{D} = \{ (x_1, y_1), \ldots, (x_n, y_n) \} \), where \( x \in \mathbb{R}^n \). The dependency of target variable \( y \) on corresponding inputs \( x \) is described through a latent function to give the likelihood

\[
p(y|f(x), w),
\]

where \( w \) contains additional parameters of the likelihood. We are interested in making inference about \( f \) using training data and then making predictions on a test dataset with the model. Under the Bayesian treatment, we need to make prior assumptions about the latent function \( f \). These are then updated to posterior probabilities via the systematic use of Bayes' rule

\[
p(w|\mathcal{D}, \mathcal{M}) = \frac{p(w|\mathcal{M}) p(\mathcal{D}|w, \mathcal{M})}{p(\mathcal{D}|\mathcal{M})},
\]

which gives a formal description of how the beliefs about \( w \) change, given \( \mathcal{D} \). The evidence term \( p(\mathcal{D}|\mathcal{M}) = \int p(w|\mathcal{M}) p(\mathcal{D}|w, \mathcal{M}) dw \) is a normalizing constant which can also be interpreted as the marginal likelihood of the model. With several possible models \( \mathcal{M} \), we would assign each its own likelihood \( p(\mathcal{D}|\mathcal{M}_i) \).

In a parametric approach, we could use for example a basic linear model \( f(x) = x^T \alpha \), and make prior assumptions about the parameters \( \alpha \). But since we don’t know the function form, the model appears quite restrictive. Instead, we can make inference about \( f \) directly using a non-parametric approach, such as a Gaussian process model, where a Gaussian prior is put on function values.

A Gaussian process \( \mathcal{GP}(m(x), k(x, x')) \) is a generalisation of multivariate Gaussian random variables to infinite sets. It is fully specified by its mean \( m(x) \) and covariance function \( k(x_1, x_j) \):

\[
m(x) = E[f(x)]
\]

\[
k(x_1, x_j) = E[(f(x_1) - m(x_1))(f(x_j) - m(x_j))].
\]
The process is Gaussian if all joint distributions are multivariate normal, that is for any given inputs \( \{x_1, \ldots, x_n\} \) the corresponding variables \( \{f(x_1), \ldots, f(x_n)\} \) have a joint \( n \)-dimensional normal distribution

\[
p(f|X, w) = N(m, K),
\]

where the terms of the mean \( m \) and the covariance matrix \( K \) correspond to those of (3) and (4). Here, \( w \) represents possible hyperparameters defining the properties of \( K \). From hereon, for notational convenience, we will only consider GPs with zero mean. Note that this assumption means that we assume the latent function \( f \) values equally likely to be positive or negative. Most often, this assumption is only fair if we transform the output values \( y \), for example by subtracting the mean or removing trends, such that our assumption holds. It is, however, also possible to use a parametric function \( m(x) \) for the inference.

### 2.2 Regression for Gaussian data

As long as the observational noise is normally distributed, we find that Gaussian process regression is analytically tractable using Bayesian inference. If we take the likelihood (1) as normal, we have the classical regression model \( y_i = f(x_i) + \epsilon_i \), where \( \epsilon_i \sim N(0, \sigma_n^2) \) and \( \sigma_n^2 \) is the noise variance. Given the inputs \( X \), the joint likelihood becomes

\[
p(y|f(x), w) = \prod_{i=1}^n N(y_i|f_i, \sigma_n^2) = N(y|f, \sigma_n^2 I).
\]

For the latent function \( f \) prior we use a zero-mean Gaussian process

\[
p(f|X, w) = N(0, K),
\]

so the posterior for \( f \) becomes

\[
p(f|D, w) \propto p(y|f)p(f|w) = N(y|f, \sigma_n^2 I)N(f|0, K)
\]

\[
= N(KK^{-1} + \sigma_n^2 I)^{-1}y, (K + \sigma_n^{-2} I)^{-1}
\]

as the product of two Gaussians is also a Gaussian. Now, we wish to make predictions on test variables \( f_* \) (and \( y_* \)) given some additional test inputs \( X_* \). The joint prior
distribution of \( f \) and \( f_* \) is also multivariate Gaussian

\[
p(f, f_*) = \mathcal{N}\left(0, \begin{bmatrix} K_{f,f} & K_{f,*} \\ K_{*,f} & K_{*,*} \end{bmatrix} \right),
\]

(10)

where the covariance matrix has been partitioned so that \( K_{*,f} \), for example, describes the covariance between the test and training inputs. To derive the joint posterior, we use Bayes’ rule with the joint prior (10) and the normal likelihood (6)

\[
p(f, f_*|y) = p(f, f_*)p(y|f) \]

(11)

where \( p(y) = \int p(y|f)p(f)df \) is the evidence term. To find the desired posterior predictive distribution, we need to marginalize over the latent variables \( f \)

\[
p(f_*|y) = \frac{1}{p(y)} \int p(f, f_*)p(y|f)df,
\]

(12)

which is tractable since both the distributions in the integrand are Gaussian, giving the posterior predictive distribution \( p(f_*|y) = \mathcal{N}(\mu_x, \sigma_x^2) \), where

\[
\mu_x = K_{*,f}(K_{f,f} + \sigma_n^2 I)^{-1}y
\]

(13)

\[
\sigma_x^2 = K_{*,*} - K_{*,f}(K_{f,f} + \sigma_n^2 I)^{-1}K_{f,*}.
\]

(14)

To find the predictive distribution for test targets \( y_* \), we can simply add the noise term into the variance (14). Thus, given a covariance function, we can calculate the predictive distribution for any given input \( x_* \).

Above, we have ignored the hyperparameters of the Gaussian process prior. These covariance function parameters are important in incorporating prior knowledge in the model. As these parameters are typically not known beforehand, we should also perform Bayesian inference over them. Assigning a prior over all the hyperparameters \( w \) slightly modifies (11) and (12) to give

\[
p(f, f_*, w|y) = \frac{p(y|f)p(f, f_*|w)p(w)}{p(y)}
\]

(15)

\[
p(f_*|y) = \frac{1}{p(y)} \int p(y|f)p(f, f_*|w)p(w)df dw,
\]

(16)

which is no longer analytically tractable. Approximations for the inference, such as
expectation propagation, are considered in, for example, Minka (2001). Similarly, if we wanted to employ a non-Gaussian likelihood, for example a logistic one for classification, we would need to use these approximations.

Instead of full Bayesian inference over the hyperparameters, we can find their maximum a posterior (MAP) point estimates by maximising the posterior density

$$w^* = \arg \max_w p(y|w)p(w),$$

with $f$ integrated out. By fixing $w$ to the point estimate in (16), we can integrate out $f$ and we have

$$p(f_*|y) \approx p(f_*|y, w^*)$$

The logarithm of the conditional evidence is

$$E = \log(p(y|w)p(w)) = \log(p(y|f)) + \log(p(f|w)) - \log(p(w)),$$

which we can calculate analytically:

$$E = -\frac{1}{2} \log \left| K_{f,f} + \sigma^2_n I \right| - \frac{1}{2} y^T \left( K_{f,f} + \sigma^2_n I \right)^{-1} y - \frac{n}{2} \log(2\pi) - \log(p(w)).$$

This log posterior can be maximised with respect to $w$ to find the point estimates for the hyperparameters. The optimization can be performed using for example standard gradient descent methods. For this, the gradient of the energy function can be found through fairly straightforward differentiation. Here, the log transform allows us to bypass numerical issues associated with for example extremely small probabilities. Furthermore, the logarithm of a normal distribution is of quadratic form, a convenient form for the quadratic algorithms used in the optimization.

Figure 1 shows an example of a prediction task using GP regression. In (a), a MAP estimate for the model hyperparameters has been calculated, given the observed data, and used to make a prediction for additional inputs. Near the observations, the fit is good, but further away, the variance quickly grows. Figure (b) illustrates the importance of hyperparameter optimisation: for different hyperparameter values, the predictions can be significantly different. Here, the hyperparameter governing the correlation distance is changed, which results in either quick deterioration in
Figure 1: (a): GP model predictions after hyperparameter optimisation using MAP point estimate. Given the data points (circles), the predicted model (line) fits the underlying process (dashed line) well when near the observations, but the variance of the prediction (grey area) grows significantly larger when moving further away. (b): Predictions with three different hyperparameter values for the same data points as in (a). The short lengthscale (thin line) quickly adapts to changes, while the long lengthscale (dashed line) takes time to adjust. The thick line represents optimized length-scale.

prediction performance (for small parameter values) or failure to fit the data (for large parameter values).
2.3 Covariance functions

The covariance function is in the heart of the Gaussian process model. It is used to describe dependencies and similarity between function values. Since we consider only zero mean processes, the covariance function alone determines the properties of functions generated by the process and, for regression, how much the data are smoothed. For a Gaussian process prior, the choice of the function depends on the possible knowledge on the properties of the data. For example, for a regression problem of seasonal data, we would be inclined to include a periodic component in the covariance function.

Above we have seen that given a covariance function it is fairly straightforward to carry out the inference and make predictions. However, as we do not generally know which covariance function to use, we need to choose a parametric family of functions and estimate the parameters. A common example is the squared exponential (SE) covariance function

\[ k(x, x') = \sigma^2_M \exp(-\frac{||x - x'||^2}{l^2}), \quad (21) \]

which uses two parameters to describe the similarity in the data: the magnitude \( \sigma^2_M \) and the characteristic length-scale \( l \) which dictates how quickly the covariance decays with distance.

In principle, any kind of function can be used as a covariance function, as long as it satisfies certain conditions. For \( k(x, x') \) to be a valid covariance function, it has to be symmetric and positive semi-definite. To be positive semi-definite, a function has to satisfy

\[ \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j k(x_i, x_j) = \alpha^T K \alpha \geq 0 \quad (22) \]

for any \( n, \ \alpha \in \mathbb{R}^n \) and \( x_1, \ldots, x_n \). Proofs of positive semi-definiteness are generally not straightforward, but luckily, different covariance functions can also be constructed from existing ones using simple rules: that is, addition and multiplication. Thus several different covariance structures can be used together to capture different aspects of data variation. Additionally to the common squared exponential covariance function described above, several covariance function types can be used, such as the Matérn family of functions

\[ k_\nu(r) = \frac{2^{1-\nu}}{\Gamma(\nu)} \left( \frac{\sqrt{2\nu r}}{l} \right)^\nu K_\nu \left( \frac{\sqrt{2\nu r}}{l} \right), \quad (23) \]
where \( r = \|x - x'\| \) and \( K_\nu \) is a modified Bessel function of the second kind of order \( \nu > 0 \). The class has been named after the work of Matérn (1960) by Stein (1999). Note that when \( \nu \to \infty \) we find the SE covariance function above. By choosing the value of \( \nu \), we can control the smoothness of the function; most used are the simplest forms of the function which can be obtained with \( \nu \) half-integer. The smoothness of a Gaussian process determines how it can adapt to quickly changing effects in the data: the SE function, being infinitely differentiable, is very smooth, whereas the Matérn functions can adjust more quickly to changes, see figure 2.3.
3 Sparse approximations

While the Gaussian process framework is attractive in its flexibility and probabilistic approach, the matrix inversion required for inference in eq. (37) is computationally expensive for large datasets. Matrix inversion scales to $\mathcal{O}(n^3)$, where $n$ is the number of training cases, effectively limiting the size of the problems to the range of thousands. To deal with this issue, several sparse approximation algorithms have been proposed, see for example Csató and Opper (2002), Seeger et al. (2003), Snelson and Ghahramani (2006). An overview of these methods can be found in Quinonero-Candela and Rasmussen (2005), while later developments include the variational approach of Titsias (2009). We will briefly go through the earlier developments and focus on the work of Titsias comparing the methods.

3.1 Earlier approaches

The common theme in all sparse approximations is the use of a small set of latent variables to perform inferences, that is, we choose $m$ inducing inputs that are treated exactly, while the other latent variables are given some approximate treatment. This causes the computational complexity of the matrix inversion to reduce to $\mathcal{O}(nm^2)$, which can thus be controlled by the number of inducing inputs. The additional set of latent inducing variables $u$ corresponds to a set of inducing inputs $X_u$. Now the joint prior for all the latent variables can be written as $p(f, f^*, u)$.

Due to the consistency of Gaussian processes, the original prior can be found by integrating over $u$ in the joint GP prior:

$$p(f, f^*) = \int p(f, f^*, u)dud = \int p(f, f^*|u)p(u)du,$$

where $p(u) = N(0, K_{u,u})$. The above exact expression is approximated by assuming that $f$ and $f^*$ are conditionally independent given $u$ and we can thus write

$$p(f, f^*) \approx q(f, f^*) = \int q(f^*|u)q(f|u)p(u)du.$$

Due to the conditional independence assumption, the latent variables $f$ and $f^*$ communicate only through $u$, which thus induces any dependencies between training and test variables. Introducing the notation $Q_{a,b} = K_{a,u}K_{u,u}^{-1}K_{u,b}$, the exact expressions
for the training and test conditionals can be written as

\[ p(f|u) = N(K_{f,u}K_{u,u}^{-1}u, K_{f,f} - Q_{f,f}) \]  
\[ p(f_*|u) = N(K_{f,u}K_{u,u}^{-1}u, K_{s,s} - Q_{s,s}) \],

which can be seen as noise-free special cases of the standard predictive equation (12), with \( u \) as “observations”.

**Deterministic Training Conditional.** The most simple approximation would be to select a subset of data to perform inference with, reducing the computational cost from \( O(n^3) \) to \( O(m^3) \) with \( m \) data points chosen. In this approach however, the omitted data points are completely lost and therefore the uncertainties of the model become hard to model. Therefore, more sophisticated sparse approximations have been developed. For example, the deterministic training conditional (DTC) (or projected process approximation, developed by Csató and Opper (2002) and Seeger et al. (2003)), can make use of all \( n \) data points. In DTC, the training conditional from equation (26) is replaced by a deterministic training conditional while the test conditional remains exact:

\[ q_{DTC}(f|u) = N(K_{f,u}K_{u,u}^{-1}u, 0) \]  
\[ q_{DTC}(f_*|u) = p(f_*|u). \]

The joint prior implied by DTC is

\[ q_{DTC}(f, f_*) = N(0, \begin{bmatrix} Q_{f,f} & Q_{f,*} \\ Q_{*,f} & K_{s,*} \end{bmatrix}) \]  

and the posterior predictive distribution and energy function are given by

\[ q_{DTC}(f_*|y) = N(Q_{*,f}(Q_{f,f} + \sigma_n^2I)^{-1}y, Q_{s,s} - Q_{s,f}(Q_{f,f} + \sigma_n^2I)^{-1}Q_{f,*}) \]  
\[ E_{DTC} = -\frac{1}{2} \log \left| Q_{f,f} + \sigma_n^2I \right| - \frac{1}{2}y^T \left( Q_{f,f} + \sigma_n^2I \right)^{-1}y - \frac{n}{2} \log(2\pi) - \log(p(w)). \]

**Fully independent training conditional.** Presented by Snelson and Ghahramani (2006) under the name sparse pseudo-input Gaussian process, the fully independent
training conditional (FITC) approximation is more sophisticated than the DTC in that it doesn’t require a deterministic relation between \( f \) and \( u \). In FITC, the inducing conditionals are

\[
q_{\text{FITC}}(f|u) = \mathcal{N}(K_{f,u}K_{u,u}^{-1}u, \text{diag}[K_{f,f} - Q_{f,f}]) \tag{33}
\]
\[
q_{\text{FITC}}(f_*|u) = p(f_*|u), \tag{34}
\]

implying the effective prior as

\[
q_{\text{FITC}}(f, f_*) = \mathcal{N}\left(0, \begin{bmatrix}
Q_{f,f} - \text{diag}[Q_{f,f} - K_{f,f}] & Q_{f,*} \\
Q_{*,f} & K_{*,*}
\end{bmatrix}\right). \tag{35}
\]

Thus the difference between DTC and FITC is that FITC uses exact covariances on the diagonal in the top left corner of the implied covariance, leading to improved accuracy. To get the predictive distribution and the energy function, we define \( \Lambda = \text{diag}[K_{f,f} - Q_{f,f}] \) to yield

\[
q_{\text{FITC}}(f_*|y) = \mathcal{N}(K_{*,f}(Q_{f,f} + \Lambda + \sigma_n^2 I)^{-1}y, K_{*,*} - K_{*,f}(Q_{f,f} + \Lambda + \sigma_n^2 I)^{-1}K_{f,*}) \tag{36}
\]

\[
E_{\text{FITC}} = -\frac{1}{2} \log \left| Q_{f,f} + \Lambda + \sigma_n^2 I \right| - \frac{1}{2} y^T \left( Q_{f,f} + \Lambda + \sigma_n^2 I \right)^{-1} y - \frac{n}{2} \log(2\pi) - \log(p(w)). \tag{37}
\]

In GP models, the bulk of the computational cost is caused by the matrix inversion of the \( n \times n \) covariance matrix. Here, both in DTC and FITC, this cost can be reduced by manipulating the predictive expression to deal with a smaller matrix using the matrix inversion lemma and the matrix determinant lemma, see appendix A. Taking for example the inversion of \( Q_{f,f} + \Lambda \) and using the matrix inversion lemma (47), we have

\[
(Q_{f,f} + \Lambda)^{-1} = \Lambda^{-1} - \Lambda^{-1}K_{u,u}(K_{u,u} + K_{u,f}\Lambda^{-1}K_{f,u})^{-1}K_{u,f}\Lambda^{-1}, \tag{38}
\]

where the diagonal \( \Lambda^{-1} \) can be performed elementwise and the matrix to be inverted, \( K_{u,u} + K_{u,f}\Lambda^{-1}K_{f,u} \), is only \( m \times m \). Snelson and Ghahramani (2006) note that the computationally most expensive part here is the matrix multiplication \( K_{u,f}\Lambda^{-1}K_{f,u} \), which is \( \Theta(nm^2) \). Therefore, the presented approximations reduce the computational
burden from $\mathcal{O}(n^3)$ to $\mathcal{O}(nm^2)$. The determinant in the energy function can be evaluated similarly using the matrix determinant lemma (48).

### 3.2 Variational model selection

Above, the choice of inducing inputs has not been considered. Often, they are carefully chosen as a subset of training inputs, but the construction of the approximations doesn’t limit their choice. Therefore, their locations can also be optimized: for example, Snelson and Ghahramani (2006) choose the inducing inputs by maximizing the marginal likelihood with respect to a fixed number of them, essentially treating them as additional hyperparameters of the model. This approach offers increased flexibility when fitting the data, but it can potentially lead to overfitting when optimizing over all hyperparameters, as noted by Titsias (2009). To reduce the problem of overfitting, Titsias proposes a variational method in which the inducing variables and the hyperparameters are selected jointly by maximizing a lower bound to the exact marginal likelihood. Here, the inducing inputs are not treated as hyperparameters but variational parameters which are selected by minimizing the Kullback-Leibler (KL) divergence between an approximate variational GP and the true posterior GP.

The variational learning approach is somewhat different from the DTC and FITC approaches above. Titsias notes that the predictive distribution

$$p(f_*|y) = \int p(f_*, f, u)p(f|u, y)p(u|y)dfdu$$  \hspace{1cm} (39)

and the problem lies in the selection of the distribution $\phi(u)$ and the inducing inputs $X_u$.

---

1Note that this is in fact a general expression for all sparse approximations: by specific choices of means and covariances, we would find both DTC and FITC. For details, see Titsias (2009).
The selection is performed by forming a variational distribution \( q(f) \) and the exact posterior \( p(f|y) \) on the training function values and minimizing the distance between them, or, equivalently, augmenting these distributions with the inducing inputs \( u \):

\[ q(f, u) = p(f|u)\phi(u). \]

Here, the augmented probability model is flexible in the sense that the inducing inputs \( X_u \) do not affect the exact GP model \( p(y, f) \), since \( p(f) \) does not change with the values of \( u \), even though \( p(f, u) \) and \( p(u) \) do. Thus the inducing inputs are not model parameters like the hyperparameters, and can be turned into variational parameters.

In variational inference (see e.g., Jordan et al., 1999), we approximate the true posterior \( p(f, u|y) = p(f|u, y)p(u|y) \) with a variational distribution \( q(f, u) \). To do this, we minimize the Kullback-Leibler divergence (Kullback and Leibler, 1951)

\[
KL(q(f, u)||p(f, u|y)) = \int q(f, u) \log \frac{q(f, u)}{p(f, u|y)} df du. \tag{41}
\]

This minimization is equivalent to maximizing the variational lower bound on the true log marginal likelihood (substituting the expressions for the distributions):

\[
\log p(y) \geq F_V(X_u, \phi) = \int p(f|u)\phi(u) \log \frac{p(y|f)p(f|u)p(u)}{p(f|u)\phi(u)} df du \tag{42}
\]

\[
= \int \phi(u) \left[ \int p(f|u) \log p(y|f) df + \log \frac{p(u)}{\phi(u)} du \right], \tag{43}
\]

where \( \int p(f|u) \log p(y|f) df = \log[N(y|\alpha, \sigma^2 I)] - \frac{1}{2\sigma^2} Tr(K_{f,f} - Q_{f,f}) \), where \( \alpha = E(f|u) = K_{f,u}K_{u,u}^{-1}u \). The bound is maximized with respect to \( \phi \) by moving the log outside the integral (reversing Jensen’s inequality), to give

\[
F_V(X_u) = \log[N(y|0, \sigma^2 I + Q_{f,f})] - \frac{1}{2\sigma^2} Tr(K_{f,f} - Q_{f,f}). \tag{44}
\]

For more details on the derivation, see Titsias (2009). Here, the computational complexity is \( \mathcal{O}(nm^2) \) as in DTC and FITC. Similarly, we can optimize over the inducing inputs \( X_u \). The optimal distribution \( \phi^* \) required for predictions can be found by differentiating the variational lower bound with respect to \( \phi(u) \). Omitting the derivation here, the predictive distribution turns out to be exactly the one used in DTC above. However, even though the predictive distribution is the same, the selection of inducing inputs and kernel hyperparameters is very different in the methods due to the trace term appearing in the variational lower bound which is
absent in both DTC and FITC. Essentially, the variational model is the sum of the DTC log likelihood (energy function) and the regularization trace term: the variational method simultaneously maximizes the log likelihood and minimizes the trace. The trace represents the total variance of the conditional prior $p(f|u)$ and can also be interpreted as the squared error of predicting $f$ from $u$. Essentially, the trace term regularizes the hyperparameters of the model, adjusting them with respect to the uncertainty of the estimates (the squared error) to avoid overfitting.

A clear distinction between the models is that in DTC and FITC, $X_u$ are treated as additional kernel hyperparameters, whereas in the variational method they are variational parameters due to the minimized KL divergence. Figure 3 shows a comparison of the three predictions of the three methods on a simple one-dimensional toy data set. This example illustrates the differences in the methods: with relatively little training data, the DTC clearly overfits the data, producing a more rough prediction than the full GP. This is evidenced also by the DTC uncertainty estimate, which varies wildly depending on the distance to the nearest training data point.

The FITC model, on the other hand, produces much better results (and it is, in general, preferred over DTC), matching the full GP prediction relatively well. However, the uncertainty estimate is not as accurate, reflecting some overfitting. The variational model (VAR), in the bottom panel of figure 3, produces essentially the same result as the full GP, both in terms of prediction and uncertainty. While the predictive distribution of VAR is exactly the same as in the DTC, the regularization trace term helps the model avoid overfitting: when the trace term is large because there are too few inducing inputs, the term favours hyperparameters that give a smoother function. Notably, the optimized places of inducing inputs somewhat differ between the methods: the DTC and FITC approaches place inducing inputs further from the training inputs than the VAR approach.

Implementation. The calculations were performed in Matlab (R2009a) with the GPstuff toolbox for Gaussian process models developed at the Centre of Excellence for Computational Complex Systems Research (COSY) / Department of Biomedical Engineering and Computational Science at the Aalto University School of Science and Technology. While the FITC was already implemented in the toolbox, the new variational approach along with the DTC method were added to it as a part of this work. The toolbox is available online at [http://www.lce.hut.fi/gpstuff](http://www.lce.hut.fi/gpstuff).
Figure 3: An illustration of the three methods on a toy data set with 20 training points and 15 inducing inputs, from top to bottom: DTC, FITC, VAR. The thick line represents the model prediction and the green dashed lines its uncertainty. The black dashed lines represent the full GP prediction and uncertainty. The data points used are blue dots; the original inducing points are marked with blue crosses, the optimized result with red ones.
4 Case study

In this section, we apply the sparse approximate Gaussian process regression methods to a spatial data set of rainfall in the United States. The US precipitation data set consists of monthly precipitation measures recorded across the country from 1895 to 1977\(^2\). The data consist of the spatial coordinates and the elevations of the weather stations and the monthly rainfall in millimeters. The total number of stations is 11918, but a high number of the measurements are missing. In this work, we use the stations for which all the measurements in 1995 are available, in total 5776 stations. The data have been previously used by, for example, Vanhatalo and Vehtari (2008).

Since the number of data points is infeasible for full GP regression, we need to employ sparse approximations. To more easily compare the sparse approximate methods with the full GP, a subset of the data consisting of the Western part of the US is selected for more detailed analysis. The locations of the stations, as well as the area selected, are shown in figure 4. The predictions of a full GP model on the whole data set is shown in figure 4 panel (b), showing the existence of phenomena along both east-west and north-south axis. Therefore, it makes sense to model give both directions their own lengthscales in the model. Further, we see that the Western part chosen for closer inspection encompasses both some of the highest and lowest precipitation values in the data.

As covariance functions, we used a combination of a squared exponential and a Matérn function with \(\nu = 3/2\) to capture effects of different nature: the Matérn function is somewhat rougher than the SE function and should therefore adapt to changes more quickly. Hyperpriors for the lengthscales and magnitudes of the covariance functions were chosen to be half Student’s-\(t\), with ten degrees of freedom for the lengthscales and 15 degrees for the magnitudes and variance of one for both.

For each of the three methods (DTC, FITC, VAR), predictive performance was measured as a function of the number of inducing inputs. A ten-fold cross-validation was performed to find the uncertainties of the predictive estimates. The chosen area contained around 1700 data points, giving training and test sets of sizes 1530 and 170.

\(^2\)Available online at [http://www.image.ucar.edu/GSP/Data/US.monthly.met/](http://www.image.ucar.edu/GSP/Data/US.monthly.met/)
Figure 4: The rainfall measurement points are plotted in panel (a), with the chosen subset in red, and the predictions for the yearly rainfall from a full GP model in panel (b).

The models in this work are compared using the mean squared error (MSE) and the mean log predictive density (MLPD) measures:

\[
\text{MSE} = \frac{1}{n} \sum_i (E[p(y_{*i}|x_{*i}, \mathcal{D})] - y_i)^2
\]

\[
\text{MLPD} = \frac{1}{n} \sum_i \log p(y_{*i}|x_{*i}, \mathcal{D}),
\]

where the sums go through the \( n \) test points and \( p(y_\ast|x_\ast, \mathcal{D}) \) is the posterior predictive distribution of \( y_\ast \) given test input \( x_\ast \) and \( y_i \) are the test outputs. MSE, measuring the difference of the test outputs and the mean of the predictions is a standard measure for model fit, but it doesn’t take into account the uncertainty in the estimates. MLPD, on the other hand, takes the uncertainty into account by using the posterior predictive distribution.

Figure 5 presents the predictive maps for the Western part of the United States for each of the sparse approximations for 29 inducing inputs, along with the full GP
result. While the optimized locations of the inducing inputs somewhat differ, results for the three approximate models are similar, not entirely able to predict the more subtle variations in rainfall across the area. Here, especially the FITC and VAR models produce resembling predictions, while the DTC model predictions appear somewhat smoother.

In figure 6, the predictive performance of the three methods is presented for a varying number of inducing inputs, along with the baseline result from the full GP. As expected, as the number of inducing inputs grows, the results become more accurate. The performance difference between the methods is small on small numbers of inducing points, in which case the full GP clearly outperforms all the approximate methods in terms of both errors. When the number of inducing inputs is increased to around 120, the errors for the approximations are close to that of the full GP. Here, the variational model tends to the full GP result, similarly to the results obtained by Titsias (2009) on different data, but the results, with the errors obtained from cross-validation, are similar for all three models. Notably, as the number of
inducing inputs is increased, the FITC model even outperforms the full GP in terms of the MLPD error. Here, it is important to note that the optimization over a large number of variables significantly increases the computational cost of the inference: the results obtained for over 100 inducing inputs are in this sense impractical, requiring even more computational power than the full GP model. Thus, in practice, we are limited to keeping the number of inducing inputs much lower, or choosing them without optimization. Titsias (2009) deals with this issue by using a greedy selection scheme for the inducing inputs among the training inputs, providing a variational EM algorithm for this. This approach was not implemented here to limit the scope of the work.
5 Conclusions

In this work, we examined different approaches for sparse approximate Gaussian process regression. In GP models, the computational cost of inference has often caused concern when dealing with large datasets; the most expensive part of the inference being the inversion of the covariance matrix. The available sparse approximations seek to reduce the computational burden by treating only some data points exactly, diminishing the size of the matrix to be inverted while retaining essential information from the whole data set.

Several methods have been proposed for sparse approximations. In this work, the methods compared were the deterministic training conditional (DTC), the fully independent training conditional (FITC) and the variational learning of inducing inputs (VAR). Of these three, FITC has become the standard for sparse approximations, while both it and DTC are known to be susceptible to overfitting; this is what the novel VAR approach seeks to improve on. This was seen on experiments on a toy data set: the VAR approach was able to exactly replicate the full GP result, while the others failed to do the same.

The three methods were tested on a spatial data set on precipitation in the United States in 1995. The models were compared in terms of their predictive accuracy using different measures. The results show that with these data, the three methods provide similar performance for small numbers of inducing inputs. The variational model implemented in this work performed reasonably well, tending to the result obtained by the full GP model. With a high number of inducing inputs, the FITC model seemed to outperform the other methods. The overfitting associated with the FITC and DTC models was not readily apparent with the spatial data set. Thus, while the novel variational method for learning inducing inputs clearly provides notable advantages in terms of the reliability of prediction performance by finding results close to those of the full GP model, it will not always outperform the previous methods.

In this work, the variational learning of inducing inputs was tested only for a regression example: in the future, the method is to be implemented for classification problems (and other non-Gaussian likelihoods) as well, for example for the expectation propagation algorithm. Another interesting development is the use of sparse approximations to allow for the development of multi-output Gaussian process mod-
els, see for example Alvarez et al. (2010). Traditionally, a multi-output GP has required matrix inversion with time $O(n^3d^3)$, where $n$ and $d$ are the numbers of training cases and outputs, but the sparse convolution method of Alvarez et al. reduces this to $O(ndm^2)$, where $m$ is the number of inducing inputs.
A Matrix operations

Lemma 1 Let $Z$ be an $n \times n$ matrix, $W$ be an $m \times m$ matrix, $U$ be an $n \times m$ matrix and $V$ be an $n \times m$ matrix. If $Z$ and $W$ are nonsingular, then $UWZ + Z$ is nonsingular if and only if $W^{-1} + VZ^{-1}U$ is nonsingular, in which case

$$(UWV + Z)^{-1} = Z^{-1} - Z^{-1}U(W^{-1} + VZ^{-1}U)^{-1}VZ^{-1}. \quad (47)$$

Lemma 2 Let $Z$ be an $n \times n$ matrix, $W$ be an $m \times m$ matrix, $U$ be an $n \times m$ matrix and $V$ be an $n \times m$ matrix. If $Z$ and $W$ are nonsingular, then

$$|UWV + Z| = |Z||W||W^{-1} + VZ^{-1}U|. \quad (48)$$

The proofs for both lemmas can be found in, for example, [Harville](2008).
References


