Sparse Spectrum Gaussian Process Regression

Miguel Lázaro-Gredilla MIGUEL@TSC.UC3M.ES
Departamento de Teoría de la Señal y Comunicaciones
Universidad Carlos III de Madrid
28911 Leganés, Madrid, Spain

Joaquin Quiñonero-Candela JOAQUINC@MICROSOFT.COM
Microsoft Research Ltd.
7 J J Thomson Avenue
Cambridge CB3 0FB, UK

Carl Edward Rasmussen[∗] CER54@CAM.AC.UK
Deparment of Engineering
University of Cambridge, Trumpington st.
Cambridge CB2 1PZ, UK

Aníbal R. Figueiras-Vidal ARFV@TSC.UC3M.ES
Departamento de Teoría de la Señal y Comunicaciones
Universidad Carlos III de Madrid
28911 Leganés, Madrid, Spain

Editor: Tommi Jaakkola

Abstract

We present a new sparse Gaussian Process (GP) model for regression. The key novel idea is to sparsify the *spectral representation* of the GP. This leads to a simple, practical algorithm for regression tasks. We compare the achievable trade-offs between predictive accuracy and computational requirements, and show that these are typically superior to existing state-of-the-art sparse approximations. We discuss both the weight space and function space representations, and note that the new construction implies priors over functions which are always stationary, and can approximate any covariance function in this class.

Keywords: Gaussian process, probabilistic regression, sparse approximation, power spectrum, computational efficiency

1. Introduction

One of the main practical limitations of Gaussian processes (GPs) for machine learning (Rasmussen and Williams, 2006) is that in a direct implementation the computational and memory requirements scale as $O(n^2)$ and $O(n^3)$, respectively. In practice this limits the applicability of exact GP implementations to data sets where the number of training samples n does not exceed a few thousand.

A number of computationally efficient approximations to GPs have been proposed, which reduce storage requirements to $O(nm)$ and the number of computations to $O(nm^2)$, where m is much smaller than n. One family of approximations, reviewed in Quiñonero-Candela and Rasmussen (2005), is based on assumptions of conditional independence given a reduced set of m *inducing*

[∗] Also at Max Planck Institute for Biological Cybernetics, Spemannstr. 38, 72076 Tübingen, Germany .

inputs. Examples of such models are those proposed in Seeger et al. (2003), Smola and Bartlett (2001), Tresp (2000), Williams and Seeger (2001) and Csató and Opper (2002), as well as the Fully Independent Training Conditional (FITC) model, introduced as Sparse Pseudo-input GP (SPGP) by Snelson and Ghahramani (2006). Walder et al. (2008) introduced the Sparse Multiscale GP (SMGP), a modification of FITC that allows each basis function to have its own set of length-scales.[1] This additional flexibility typically yields some performance improvement over FITC, but it also requires learning twice as many parameters. SMGP can alternatively be derived within the unifying framework of Quiñonero-Candela and Rasmussen (2005) if we allow the inducing inputs to lie in a transformed domain, as shown in Lázaro-Gredilla and Figueiras-Vidal (2010).

Another family of approximations is based on approximate matrix-vector-multiplications (MVMs), where m is for example a reduced number of conjugate gradient steps to solve a system of linear equations. Some of these methods have been briefly reviewed in Quiñonero-Candela et al. (2007). Local mixtures of GP have been used by Urtasun and Darrell (2008) for efficient modelling of human poses.

In this paper we introduce a stationary trigonometric Bayesian model for regression that retains the computational efficiency of the aforementioned approaches, while improving performance. The model consists of a linear combination of trigonometric functions where both weights and phases are integrated out. All hyperparameters of the model (frequencies and amplitudes) are learned jointly by maximizing the marginal likelihood. This model is a stationary sparse GP that can approximate any desired stationary full GP. Sparse trigonometric expansions have been proposed in several contexts, for example, Lázaro-Gredilla et al. (2007) and Rahimi and Recht (2008), as discussed further in Section 4.3.

FITC, SMGP, and the model introduced in this paper focus on predictive accuracy at low computational cost, rather than on faithfully converging towards the full GP as the number of basis functions grows. Performance-wise, FITC and the more recent SMGP can be regarded as the current state-of-the-art sparse GP approximations, so we will use them as benchmarks in the performance comparisons.

In Section 2 we give a brief review of GP regression. In Section 3 we introduce the trigonometric Bayesian model, and in Section 4 we present the Sparse Spectrum Gaussian Process (SSGP) algorithm. Section 5 contains a comparative performance evaluation on several data sets.

2. Gaussian Process Regression

Regression is often formulated as the task of predicting the scalar output y_* associated to the D-dimensional input $\mathbf{x}_*$, given a training data set $\mathcal{D} \equiv \{\mathbf{x}_j, y_j | j = 1, \dots n\}$ of n input-output pairs. A common approach is to assume that the outputs have been generated by an unknown latent function $f(\mathbf{x})$ and independently corrupted by additive Gaussian noise of constant variance σ_n^2:

$$y_j \;=\; f(\mathbf{x}_j) + \varepsilon_j, \qquad \varepsilon_j \;\sim\; \mathcal{N}(0,\, \sigma_n^2)\,.$$

The regression task boils down to making inference about $f(\mathbf{x})$. Gaussian process (GP) regression is a probabilistic, non-parametric Bayesian approach. A Gaussian process prior distribution on $f(\mathbf{x})$ allows us to encode assumptions about the smoothness (or other properties) of the latent function

[1] Note that SMGP only extends FITC in the specific case of the anisotropic squared exponential covariance function, whereas FITC can be applied to any covariance function.

(Rasmussen and Williams, 2006). For any set of inputs $\{\mathbf{x}_i\}_{i=1}^n$ the corresponding vector of function evaluations $\mathbf{f} = [f(\mathbf{x}_1), \dots, f(\mathbf{x}_n)]^\top$ has a joint Gaussian distribution:

$$p(\mathbf{f}|\{\mathbf{x}_i\}_{i=1}^n) \;=\; \mathcal{N}(\mathbf{f}|\mathbf{0}, \mathbf{K_{ff}})\,.$$

This paper follows the common practice of setting the mean of the process to zero.[2] The properties of the GP prior over functions are governed by the covariance function

$$\left[\mathbf{K_{ff}}\right]_{ij} \;=\; k(\mathbf{x}_i, \mathbf{x}_j) \;=\; \mathbb{E}[f(\mathbf{x}_i) f(\mathbf{x}_j)]\,, \tag{1}$$

which determines how the similarity between a pair of function values varies as a function of the corresponding pair of inputs. A covariance function is *stationary* if it only depends on the difference between its inputs

$$k(\mathbf{x}_i, \mathbf{x}_j) \;=\; k(\mathbf{x}_i - \mathbf{x}_j) \;=\; k(\tau).$$

The elegance of the GP framework is that the properties of the function are conveniently expressed directly in terms of the covariance function, rather than implicitly via basis functions.

To obtain the predictive distribution $p(y_*|\mathbf{x}_*, \mathcal{D})$ it is useful to express the model in matrix notation by stacking the targets y_j in vector $\mathbf{y} = [y_1, \dots, y_n]^\top$ and writing the joint distribution of training and test targets:

$$\begin{bmatrix} \mathbf{y} \\ y_* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \mathbf{K_{ff}} + \sigma_n^2 \mathbf{I}_n & \mathbf{k_{f*}} \\ \mathbf{k_{f*}}^\top & k_{**} + \sigma_n^2 \end{bmatrix}\right),$$

where $\mathbf{k_{f*}}$ is the vector of covariances between $f(\mathbf{x}_*)$ and the training latent function values, and k_{**} is the prior variance of $f(\mathbf{x}_*)$. $\mathbf{I}_n$ is the $n \times n$ identity. The predictive distribution is obtained by conditioning on the observed training outputs:

$$p(y_*|\mathbf{x}_*, \mathcal{D}) \;=\; \mathcal{N}(\mu_*, \sigma_*^2), \;\; \text{where} \;\; \begin{cases} \mu_* = \mathbf{k_{*f}}(\mathbf{K_{ff}} + \sigma_n^2 \mathbf{I}_n)^{-1}\mathbf{y} \\ \sigma_*^2 = \sigma_n^2 + k_{**} - \mathbf{k_{*f}}(\mathbf{K_{ff}} + \sigma_n^2 \mathbf{I}_n)^{-1}\mathbf{k_{f*}}\,. \end{cases} \tag{2}$$

The covariance function is parameterized by *hyperparameters*. Consider for example the stationary anisotropic squared exponential covariance function

$$k_{\text{ARD}}(\tau) \;=\; \sigma_0^2 \exp(-\tfrac{1}{2}\tau^\top \Lambda^{-1} \tau), \quad \text{where} \quad \Lambda \;=\; \text{diag}([\ell_1^2,\; \ell_2^2,\; \dots\; \ell_D^2]). \tag{3}$$

The hyperparameters are the prior variance σ_0^2 and the lengthscales $\{\ell_d\}$ that determine how rapidly the covariance decays with the distance between inputs. This covariance function is also known as the ARD (Automatic Relevance Determination) squared exponential, because it can effectively prune input dimensions by growing the corresponding lengthscales.

It is convenient to denote all hyperparameters including the noise variance by θ. These can be learned by maximizing the evidence, or log marginal likelihood:

$$\log p(\mathbf{y}|\theta) \;=\; -\frac{n}{2}\log(2\pi) - \frac{1}{2}|\mathbf{K_{ff}} + \sigma_n^2 \mathbf{I}_n| - \frac{1}{2}\mathbf{y}^\top \left(\mathbf{K_{ff}} + \sigma_n^2 \mathbf{I}_n\right)^{-1} \mathbf{y}. \tag{4}$$

Provided there exist analytic forms for the gradients of the covariance function with respect to the hyperparameters, the evidence can be maximized by using a gradient-based search. Unfortunately, computing the evidence and the gradients requires the inversion of the covariance matrix $\mathbf{K_{ff}} + \sigma_n^2 \mathbf{I}_n$ at a cost of $\mathcal{O}(n^3)$ operations, which is prohibitive for large data sets.

[2] The extension to GPs with general mean functions is straightforward.

3. Trigonometric Bayesian Regression

In this section we present a Bayesian linear regression model with trigonometric basis functions, and related it to a full GP in the next section. Consider the model

$$f(\mathbf{x}) \;=\; \sum_{r=1}^{m} a_r \cos(2\pi \mathbf{s}_r^\top \mathbf{x}) + b_r \sin(2\pi \mathbf{s}_r^\top \mathbf{x}) \,, \tag{5}$$

where each of the m pairs of basis functions is parametrized by a D-dimensional vector $\mathbf{s}_r$ of spectral frequencies. Note that each *pair* of basis functions share frequencies, but each have independent amplitude parameters, a_r and b_r. We treat the frequencies as deterministic parameters and the amplitudes in a Bayesian way. The priors are independent Gaussian

$$a_r \;\sim\; \mathcal{N}\big(0, \frac{\sigma_0^2}{m}\big) \,, \quad b_r \;\sim\; \mathcal{N}\big(0, \frac{\sigma_0^2}{m}\big) \,,$$

where the variances are scaled down linearly by the number of basis functions. Under the prior, the distribution over functions from Equation (5) is Gaussian with mean function zero and covariance function (from Equation (1))

$$k(\mathbf{x}_i, \mathbf{x}_j) \;=\; \frac{\sigma_0^2}{m} \phi(\mathbf{x}_i)^\top \phi(\mathbf{x}_j) \;=\; \frac{\sigma_0^2}{m} \sum_{r=1}^{m} \cos\big(2\pi \mathbf{s}_r^\top (\mathbf{x}_i - \mathbf{x}_j)\big) \,, \tag{6}$$

where we define the column vector of length $2m$ containing the evaluation of the m pairs of trigonometric functions at $\mathbf{x}$

$$\phi(\mathbf{x}) \;=\; \big[\cos(2\pi \mathbf{s}_1^\top \mathbf{x}) \; \sin(2\pi \mathbf{s}_1^\top \mathbf{x}) \; \ldots \; \cos(2\pi \mathbf{s}_m^\top \mathbf{x}) \quad \sin(2\pi \mathbf{s}_m^\top \mathbf{x})\big]^\top .$$

Sparse linear models generally induce priors over functions whose variance depends on the input. In contrast, the covariance function in Equation (6) is stationary, that is, the prior variance is independent of the input and equal to σ_0^2. This is due to the particular nature of the trigonometric basis functions and implies that the predictive variances cannot be "healed", as proposed in Rasmussen and Quiñonero-Candela (2005) for the case of the Relevance Vector Machine.

The predictions and marginal likelihood can be evaluated using Equations (2) and (4), although direct evaluation is computationally inefficient when $2m < n$. For the predictive distribution we use the more efficient

$$\mathbb{E}[y_*] \;=\; \phi(\mathbf{x}_*)^\top \mathbf{A}^{-1} \Phi_\mathbf{f} \mathbf{y}, \quad \mathbb{V}[y_*] \;=\; \sigma_n^2 + \sigma_n^2 \phi(\mathbf{x}_*)^\top \mathbf{A}^{-1} \phi(\mathbf{x}_*), \tag{7}$$

where we have defined the $2m$ by n design matrix $\Phi_\mathbf{f} = [\phi(\mathbf{x}_1), \ldots, \phi(\mathbf{x}_n)]$ and $\mathbf{A} = \Phi_\mathbf{f} \Phi_\mathbf{f}^\top + \frac{m\sigma_n^2}{\sigma_0^2} \mathbf{I}_{2m}$. Similarly, for the log marginal likelihood

$$\log p(\mathbf{y}|\theta) \;=\; -\big[\mathbf{y}^\top \mathbf{y} - \mathbf{y}^\top \Phi_\mathbf{f}^\top \mathbf{A}^{-1} \Phi_\mathbf{f} \mathbf{y}\big] / (2\sigma_n^2) - \frac{1}{2} \log |\mathbf{A}| + m \log \frac{m\sigma_n^2}{\sigma_0^2} - \frac{n}{2} \log 2\pi\sigma_n^2 \,. \tag{8}$$

A stable and efficient implementation uses Cholesky decompositions, Appendix A. Both the predictive distribution and the marginal likelihood can be computed in $O(nm^2)$. The predictive mean and variance at an additional test point can be computed in $O(m)$ and $O(m^2)$ respectively. The storage costs are also reduced, since we no longer store the full covariance matrix (of size $n \times n$), but only the design matrix (of size $n \times 2m$).

3.1 Periodicity

One might be tempted to assume that this model would only be useful for modeling periodic functions, since strictly speaking a linear combination of periodic signals is itself periodic. However, if the individual frequencies are not all multiples of a common base frequency, then the period of the resulting signal will be very long, typically exceeding the range of the inputs by orders of magnitude. Thus, the model based on trigonometric basis functions has practical use for modeling non-periodic functions. The same principle is used (interchanging input and frequency domains) in uneven sampling to space apart frequency replicas an avoid aliasing, see for instance Bretthorst (2000). As our experimental results suggest, the model provides satisfactory predictive variances.

3.2 Representation

An alternative and equivalent representation of the model in Equation (5), which only uses half the number of trigonometric basis functions is possible, by writing the linear combination of a sine and a cosine as a cosine with an amplitude and a phase. Although these two representations are equivalent, *inference* based on them differs. Whereas we have been able to integrate out the amplitudes to arrive at the GP in Equation (6), this would not be possible analytically using the more parsimonious representation.

Optimization instead of marginalization of the phases has two important consequences. Firstly, we lose the property of stationarity of the prior over functions. Secondly we may expect that the model becomes more prone to overfitting. When considering the contribution from a basis function (pair) with a specific frequency, the optimization based scheme could fit arbitrarily the phase, whereas the integration based inference is constrained to use a flat prior over phases. In Section 5.3 we empirically verify that the computation vs accuracy tradeoff typically favors the less compact representation.

4. The Sparse Spectrum Gaussian Process

In the previous section we presented an explicit basis function regression model, but we did not discuss how to select the frequencies defining the basis functions. In this section, we present a sparse GP approximation view of this model, which shows how it can be understood as a computationally efficient approximation to any GP with stationary covariance function. In the next section we present experimental results showing that dramatic improvements over other state-of-the-art sparse GP regression algorithms are possible.

We will now take a generic GP with stationary covariance function and sparsify its power spectral density to obtain a sparse GP that approximates the full GP. The power spectral density (or power spectrum) $S(\mathbf{s})$ of a stationary random process expresses how the power is distributed over the frequency domain. For a stationary GP, the power is equal to the prior variance $k(\mathbf{x},\mathbf{x}) = k(\mathbf{0}) = \sigma_0^2$. The frequency vector $\mathbf{s}$ has the same length D as the input vector $\mathbf{x}$. The d-th element of $\mathbf{s}$ can be interpreted as the frequency associated to the d-th input dimension. The Wiener-Khintchine theorem (see for example Carlson, 1986, p. 162) states that the power spectrum and the autocorrelation of the random process constitute a Fourier pair. In our case, given that $f(\cdot)$ is drawn from a stationary Gaussian process, the autocorrelation function is equal to the stationary covariance function, and we have:

$$k(\tau) = \int_{\mathbb{R}^D} e^{2\pi i \mathbf{s}^\top \tau} S(\mathbf{s}) d\mathbf{s}, \qquad S(\mathbf{s}) = \int_{\mathbb{R}^D} e^{-2\pi i \mathbf{s}^\top \tau} k(\tau) d\tau. \tag{9}$$

We thus see that there are two equivalent representations for a stationary Gaussian process: the traditional one in terms of the covariance function in the (input) space domain, and a perhaps less usual one as the power spectrum in the frequency domain.

Bochner's theorem (Stein, 1999, p. 24) states that any stationary covariance function $k(\tau)$ can be represented as the Fourier transform of a positive finite measure. This means that the power spectrum in (9) is a positive finite measure, and in particular that it is *proportional* to a probability measure, $S(\mathbf{s}) \propto p_S(\mathbf{s})$. The proportionality constant can be directly obtained by evaluating the covariance function in (9) at $\tau = \mathbf{0}$. We obtain the relation:

$$S(\mathbf{s}) \;=\; k(\mathbf{0})\, p_S(\mathbf{s}) \;=\; \sigma_0^2\, p_S(\mathbf{s}). \tag{10}$$

We can use the fact that $S(\mathbf{s})$ is proportional to a multivariate probability density in $\mathbf{s}$ to rewrite the covariance function in (9) as an expectation:

$$\begin{aligned} k(\mathbf{x}_i,\mathbf{x}_j) &= k(\tau) = \int_{\mathbb{R}^D} e^{2\pi i \mathbf{s}^\top(\mathbf{x}_i-\mathbf{x}_j)} S(\mathbf{s})\, d\mathbf{s} \;=\; \sigma_0^2 \int_{\mathbb{R}^D} e^{2\pi i \mathbf{s}^\top \mathbf{x}_i} \left(e^{2\pi i \mathbf{s}^\top \mathbf{x}_j}\right)^* p_S(\mathbf{s}) d\mathbf{s} \\ &= \sigma_0^2 \mathbb{E}_{p_S}\left[e^{2\pi i \mathbf{s}^\top \mathbf{x}_i}\left(e^{2\pi i \mathbf{s}^\top \mathbf{x}_j}\right)^*\right], \end{aligned} \tag{11}$$

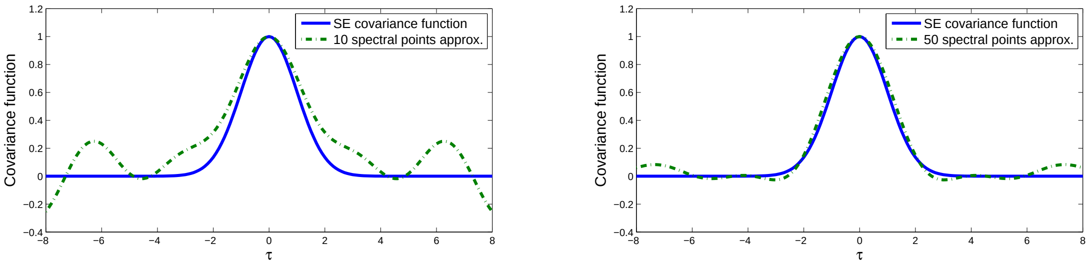
where $\mathbb{E}_{p_S}$ denotes expectation wrt. $p_S(\mathbf{s})$ and superscript asterisk[3] denotes complex conjugation. This last expression is an exact expansion of the covariance function as the expectation of a product of complex exponentials with respect to a particular distribution over their frequencies. This integral can be approximated by simple Monte Carlo by taking an average of a few samples corresponding to a finite set of frequencies, which we call *spectral points*.

Since the power spectrum is symmetric around zero, a valid Monte Carlo procedure is to sample frequencies always as a pair $\{\mathbf{s}_r, -\mathbf{s}_r\}$. This has the advantage of preserving the property of the exact expansion, Equation (11) that the imaginary terms cancel:

$$\begin{aligned} k(\mathbf{x}_i,\mathbf{x}_j) &\simeq \frac{\sigma_0^2}{2m}\sum_{r=1}^{m}\left[e^{2\pi i \mathbf{s}_r^\top \mathbf{x}_i}\left(e^{2\pi i \mathbf{s}_r^\top \mathbf{x}_j}\right)^* + \left(e^{2\pi i \mathbf{s}_r^\top \mathbf{x}_i}\right)^* e^{2\pi i \mathbf{s}_r^\top \mathbf{x}_j}\right] \\ &= \frac{\sigma_0^2}{m}\mathrm{Re}\Big[\sum_{r=1}^{m} e^{2\pi i \mathbf{s}_r^\top \mathbf{x}_i}\left(e^{2\pi i \mathbf{s}_r^\top \mathbf{x}_j}\right)^*\Big] \;=\; \frac{\sigma_0^2}{m}\sum_{r=1}^{m}\cos\left(2\pi \mathbf{s}_r^\top(\mathbf{x}_i-\mathbf{x}_j)\right), \end{aligned}$$

where $\mathbf{s}_r$ is drawn from $p_S(\mathbf{s})$ and $\mathrm{Re}[\cdot]$ denotes the real part of a complex number. Notice, that we have recovered exactly the expression for the covariance function induced by the trigonometric basis functions model, Equation (6). Further, we have given an interpretation of the frequencies as spectral Monte Carlo samples, approximating *any* stationary covariance function. This is a more general result than that of (MacKay, 2003, Ch. 45), which only applies to Gaussian covariances. The approximation is equivalent to replacing the original spectrum $S(\mathbf{s})$ by a set of Dirac deltas of amplitude σ_0^2 distributed according to $p_S(\mathbf{s})$. Thus, we “sparsify” the spectrum of the GP.

This convergence result can also be stated as follows: A stationary GP can be seen as a neural network with infinitely many hidden units and trigonometric activations if independent priors following Gaussian and $p_S(\mathbf{s})$ distributions are placed on the output and input weights, respectively. This is analogous to the result of Williams (1997) for the non-stationary multilayer perceptron covariance function.



(a)

(b)

Figure 1: Squared exponential covariance function and its approximation with (a) 10 and (b) 50 random spectral points respectively.

4.1 Example: The Squared Exponential Covariance Function

The probability density associated to the squared exponential covariance function of Equation (3) can be obtained from the Fourier transform

$$p_S^{\text{ARD}}(\mathbf{s}) \;=\; \frac{1}{k_{\text{ARD}}(\mathbf{0})}\int_{\mathbb{R}^D} e^{-2\pi i\mathbf{s}^\top\tau}k_{\text{ARD}}(\tau)d\tau \;=\; \sqrt{|2\pi\Lambda|}\exp(-2\pi^2\mathbf{s}^\top\Lambda\mathbf{s}), \tag{12}$$

which also has the form of a multivariate Gaussian distribution. For illustration purposes, we compare the exact squared exponential covariance function with its spectral approximation in Figure 1, where the spectral points are sampled from Equation (12). As expected, the quality of the approximation improves with the number of samples.

4.2 The SSGP Algorithm

One of the main goals of sparse approximations is to reduce the computational burden while retaining as much predictive accuracy as possible. Sampling from the spectral density constitutes a way of building a sparse approximation. However, we may suspect that we can obtain much sparser models if the spectral frequencies are learned by optimizing the marginal likelihood, an idea which we pursue in the following.

The algorithm we propose uses conjugate gradients to optimize the marginal likelihood (8) with respect to the spectral points $\{\mathbf{s}_r\}$ and the hyperparameters σ_0^2, σ_n^2, and $\{\ell_1,\ \ell_2,\ \dots\ \ell_D\}$. Optimizing with respect to the lengthscales in addition to the spectral points is effectively an overparametrization, but in our experience this redundancy proves helpful in avoiding undesired local minima. As is usual with this kind of optimization, the problem is non-convex and we cannot expect to find the global optimum. The goal of the optimization is to find a reasonable local optimum.

In detail, model selection for the SSGP algorithm consists in:

1. Initialize $\{\ell_d\}$, σ_0^2, and σ_n^2 to some sensible values. We use one half of the ranges of the input dimensions, the variance of $\{y_j\}$ and $\sigma_0^2/4$, respectively.

2. Initialize the $\{\mathbf{s}_r\}$ by sampling from (10).

[3] The superscript asterisk denotes complex conjugate and the subscript asterisk indicates test quantity.

3. Jointly optimize the marginal likelihood wrt. spectral points and hyperparameters.

The computational cost of training the SSGP algorithm is $O(nm^2)$ per conjugate gradient step. At prediction time, the cost is $O(m)$ for the predictive mean and $O(m^2)$ for the predictive variance per test point. These computational costs are of the same order as those of the majority of the sparse GP approximations that have recently been proposed (see Quiñonero-Candela and Rasmussen, 2005, for a review).

Learning the spectral frequencies by optimization departs from the original motivation of approximating a full GP. The optimization stage poses a risk of overfitting, which we assess in the experimental section that follows. However, the additional flexibility can potentially improve performance since it allows learning a covariance function suitable to the problem at hand.

4.3 Related Algorithms

Finite decompositions in terms of harmonic basis functions, such as Fourier series, are a classic idea. In the context of kernel machines recent work include Lázaro-Gredilla et al. (2007) for GPs and Rahimi and Recht (2008) for Support Vector Machines (SVMs). As we show in the experimental section, the details of the implementation turn out to have a critical impact on the performance of the algorithms. The SVM based approach uses projections onto a random set of harmonic functions, whereas the approach used in this paper uses the evidence framework to carefully craft an optimized sparse harmonic representation. As is revealed in the experimental section, optimization of the frequencies, amplitudes and noise offers dramatic performance improvements for comparable sparseness.

5. Experiments

In this section we investigate properties of the SSGP algorithm, and evaluate the computational complexity vs. accuracy tradeoff. We first relate the FITC and SSGP approximations. We then present empirical comparisons on several data sets, using FITC and SMGP as benchmarks. Finally, we revisit the alternative more compact representation of SSGP using phases, and discuss a data set where SSGP performs badly.

Our implementation of SSGP in matlab is available from `http://www.tsc.uc3m.es/~miguel/simpletutorialssgp.php` together with a simple usage tutorial and the data sets from this section. An implementation of FITC is available from Snelson's web page at `http://www.gatsby.ucl.ac.uk/~snelson`.

5.1 Comparing Predictive Distributions for SSGP and FITC

Whereas SSGP relies on a sparse approximation to the spectrum, the FITC approximation is sparse in a spatial sense: A set of *pseudo-inputs* is used as an information bottleneck. The only evaluations of the covariance function allowed are those involving a function value at a pseudo-input. For a set of m pseudo-inputs the computational complexity of FITC is of the same order as that of SSGP with m spectral points.

The covariance function induced by FITC has a constant prior variance, but it is not stationary. The original covariance of the full GP is only approximated faithfully in the vicinity of the pseudo-inputs and the covariance between any two function values that are both far apart from any pseudo-

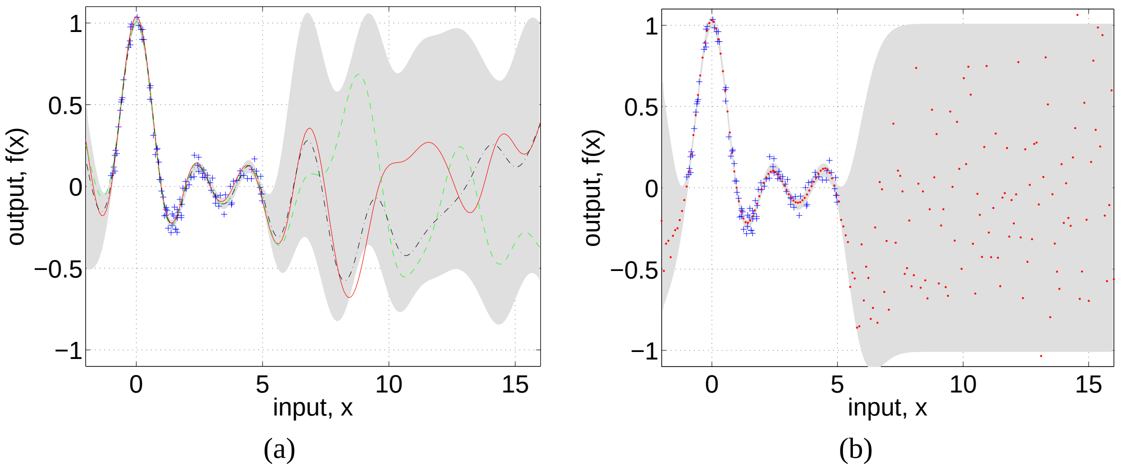


Figure 2: Learning the sinc(x) function from 100 noisy observations (plusses) using 40 basis functions with shaded area showing 95% (noise free) posterior confidence area. In panel (a) the SSGP method with three functions drawn from the posterior is shown. In (b) the same data for the FITC method with samples (dots) drawn from the joint posterior.

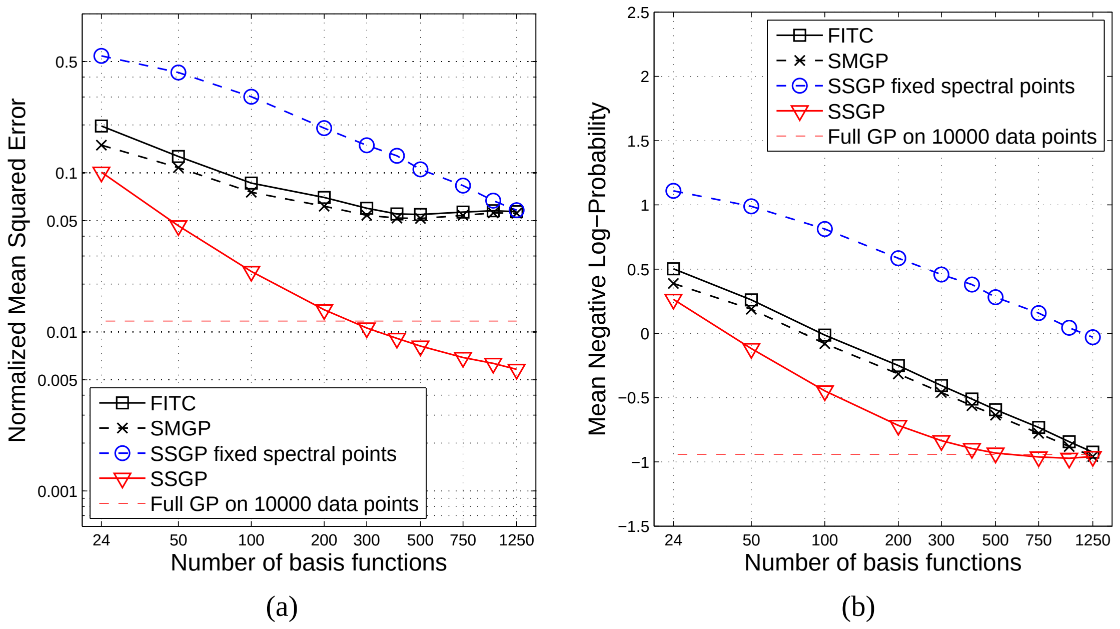
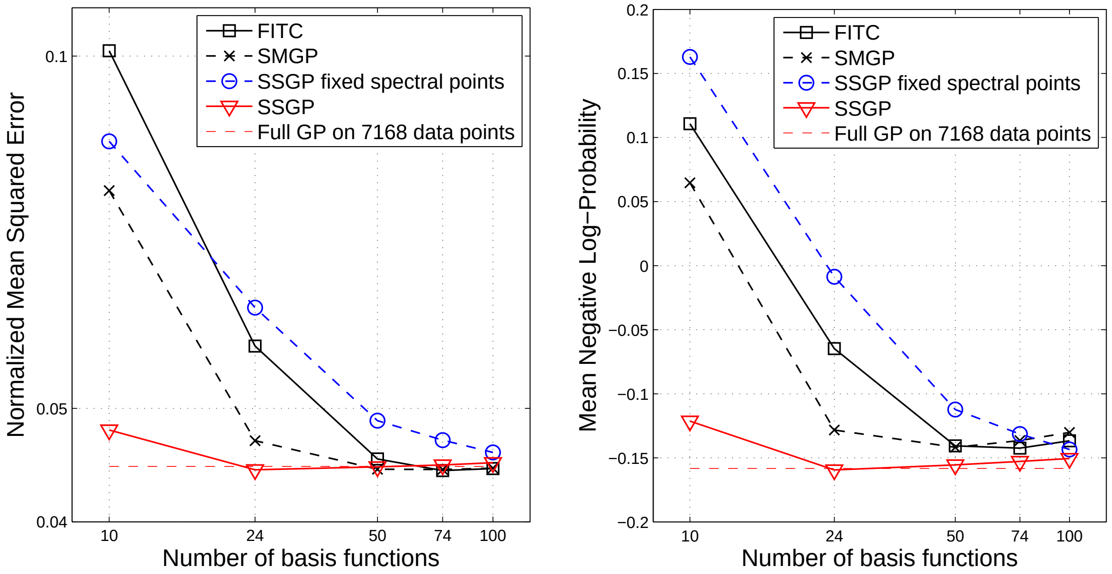


Figure 3: *Kin-40k* data set. (a) NMSE and (b) MNLP as a function of the number of basis functions.

input decays to zero. As a result, functions sampled from the GP prior induced by FITC tend to white Gaussian noise away from the pseudo-inputs.



(a)

(b)

Figure 4: *Pumadyn-32nm* data set. (a) NMSE and (b) MNLP as a function of the number of basis functions.

Figure 2 compares the predictive posterior distributions of SSGP and FITC for a simple synthetic data set. The training data is generated by evaluating the sinc function on 100 random inputs $x \in [-1,5]$ and adding white, zero-mean Gaussian noise of variance $\sigma_n^2 = 0.05^2$. SSGP is given 20 fixed spectral points sampled from the spectrum of a squared exponential covariance function, and FITC is given 40 fixed pseudo-inputs sampled uniformly from the range of the training inputs. The rest of the hyperparameters are optimized in both cases by maximizing the marginal likelihood. We plot the 95% confidence interval for both predictive distributions (mean ± two standard deviations), and draw three samples from the SSGP posterior and one sample from the FITC posterior.

Despite the different nature of the approximations, the figure shows that for an equal number of basis functions both predictive distributions are qualitatively very similar: the uncertainty grows away from the training data. In the following section, we verify empirically that the SSGP is a practical approximation for modelling non-periodic data.

5.2 Performance Evaluation

We will use two quantitative performance measures: the test Normalized Mean Square Error (NMSE) and the test Mean Negative Log Probability (MNLP) , defined as:

$$\text{NMSE} = \frac{\langle (y_{*j} - \mu_{*j})^2 \rangle}{\langle (y_{*j} - \bar{y})^2 \rangle} \quad \text{and} \quad \text{MNLP} = \frac{1}{2}\left\langle \left(\frac{y_{*j} - \mu_{*j}}{\sigma_{*j}}\right)^2 + \log \sigma_{*j}^2 + \log 2\pi \right\rangle , \tag{13}$$

where μ_{*j} and σ_{*j}^2 are, respectively, the predictive mean and variance for test sample j and y_{*j} is the actual test value for that sample. The average output value for training data is $\bar{y}$. We denote the average over test cases by $\langle\cdot\rangle$. For all experiments the values reported are averages over ten repetitions.

For each data set we report the performance of five different methods: first, the SSGP algorithm as presented in Section 4.2; second, a version of SSGP where the spectral points are "fixed" to samples from the spectral density of a squared exponential covariance function whose lengthscales are learned (SSGP fixed spectral points);[4] third, the FITC approximation, learning the pseudo-inputs; fourth, SMGP, trained as described in Walder et al. (2008); and finally as a base line comparison we report the result of a full GP trained on the entire training set. We plot the performance as a function of the number of basis functions. For FITC this is equivalent to the number of pseudo-inputs, whereas for SSGP a spectral point corresponds to two basis functions. The number of basis functions is a good proxy for computational cost.

We consider four data sets of size moderate enough to be tractable by a full GP, but still large enough that there is a motivation for computationally efficient approximations.

The two first data sets are both artificially generated using a robot arm simulator and are highly non-linear and have very low noise. They were both used in Seeger et al. (2003) and Snelson and Ghahramani (2006), but note that their definition of the NMSE measure differs by a factor of 2 from our definition in (13). We follow precisely their preprocessing and use the original splits. The first data set is *Kin-40k* (8 dimensions, 10000 training and 30000 testing samples) and the results are displayed in Figure 3. For both error measures SSGP outperforms FITC and SMGP by a large margin, and even improves on the performance of the full GP. The SSGP with fixed spectral points is inferior, proving that a greater sparsity vs. accuracy tradeoff can be achieved by optimizing the spectral points.

The *Pumadyn-32nm* problem (32 dimensions, 7168 training and 1024 testing samples) can be seen as a test of the ARD capabilities of a regression model, since only 4 out of the 32 input dimensions are relevant. Following Snelson and Ghahramani (2006), to avoid getting stuck at an undesirable bad local optimum, lengthscales are *initialized* from a full GP on a subset of 1024 training data points, for all compared methods. The results are shown in figure 4.

The conclusions are similar as for the *Kin-40k* data set. SSGP matches the full GP for a surprisingly small number of basis functions.

The *Pole Telecomm* and the *Elevators* data sets are taken from `http://www.liaad.up.pt/~ltorgo/Regression/DataSets.html`. In the *Pole Telecomm* data set we retain 26 dimensions, removing constants. We use the original split, 10000 data for training and 5000 for testing. Both the inputs and the outputs take a discrete set of values. In particular, the outputs take values between 0 and 100, in multiples of 10. We take into account the output quantization by lower bounding the value of σ_n^2 to the value of the quantization noise, $\text{bin}_{\text{spacing}}^2/12$. This lower bounding is applied to all the compared methods. The effect is to provide a better estimation for σ_n^2 and therefore, better MNLP measures, but we have observed that this modification has no noticeable effect on NMSE values. Resulting plots are in Figure 5.

SSGP is superior in terms of NMSE, getting very close to the full GP for more than 200 basis functions. In terms of MNLP, SSGP is between FITC and SMGP for small m, but slightly worse for more than 100 basis functions. This may be an indication that SSGP produces better predictive means than variances. We also see that SSGP with fixed spectral points is uniformly worse.

[4] In practice the spectral points are sampled from the spectral density of a squared exponential covariance function, and scaled as the lengthscales adapt.

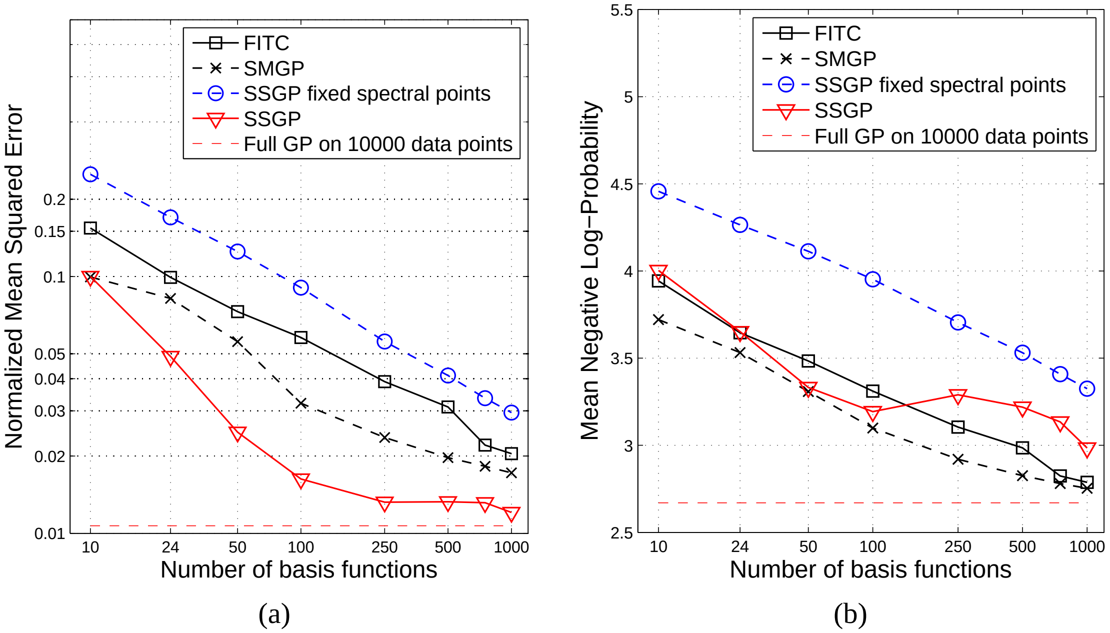
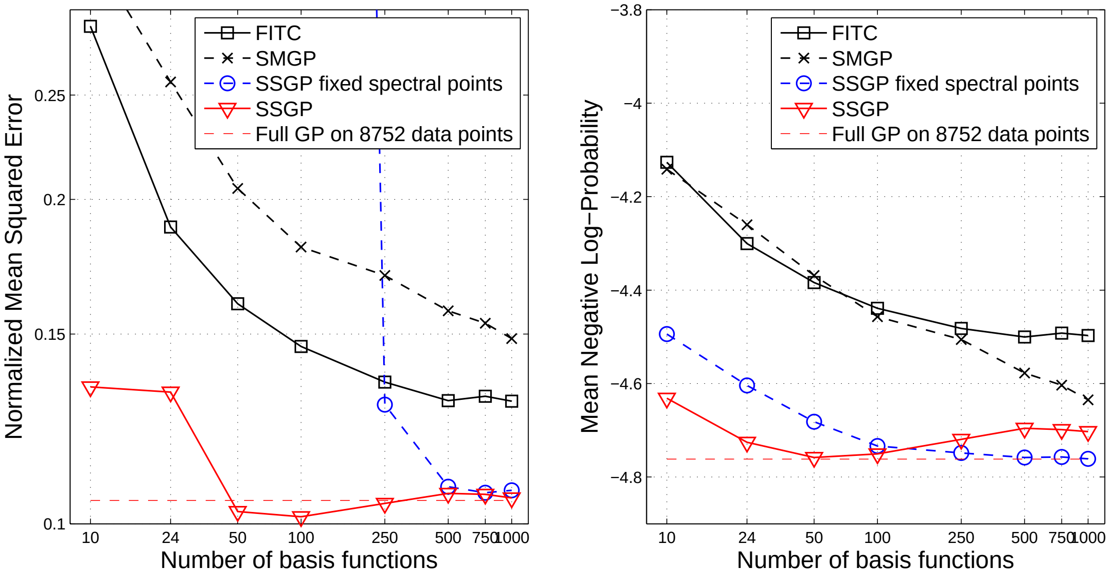


Figure 5: *Pole Telecomm* data set. (a) NMSE and (b) MNLP as a function of the number of basis functions.

The fourth data set, *Elevators*, relates to controlling the elevators of an F16 aircraft. After removing some constant inputs the data is 17-dimensional. We use the original split with 8752 data for training and 7847 for testing. Results are displayed in Figure 6. SSGP consistently outperforms FITC and SMGP and gets very close to the full GP using a very low number of basis functions. The large NMSE average errors incurred by SSGP with fixed spectral points for small numbers of basis functions are due to outliers that are present in a small number (about 10 out of 7847) of the test inputs, in some of the 10 repeated runs. The predictive variances for these few points are also big, so their impact on the MNLP score is small. Such an effect has not been observed in any of the other data sets.

5.3 Explicit Phase Representation

In Section 3.2 we considered an alternative representation of the SSGP model using only half the basis functions, but explicitly representing the phases. Bayesian inference in this representation is intractable, but one can optimize the phases instead, at the possibly increased risk of overfitting. As an example, we evaluate the performance of the cosine only expansion with explicit phases on the *Pole-Telecomm* data set in Figure 7. Whereas the performance for the two variants are comparable for small numbers of basis functions, the cosine only representation becomes worse when the number of basis functions gets larger, confirming our suspicion that optimization of the phases increases the risk of overfitting.



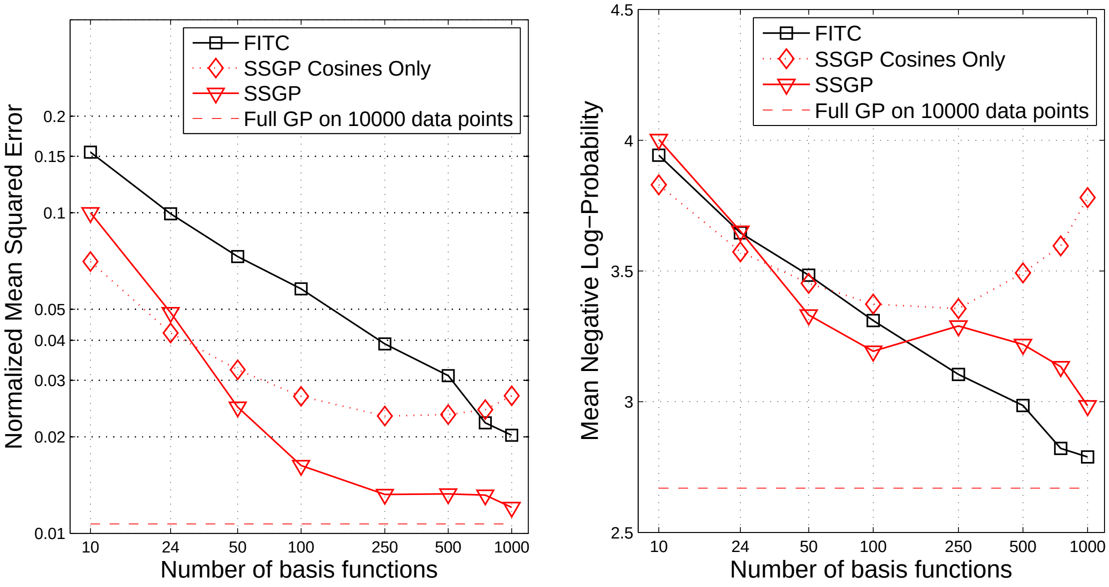
(a) (b)

Figure 6: *Elevators* data set. (a) NMSE and (b) MNLP as a function of the number of basis functions.

5.4 The Pendulum Data Set

So far we have seen data sets where SSGP consistently outperforms FITC and SMGP, and often approaches the performance of a full GP for quite small numbers of basis functions. In this section we present a counter example, showing that SSGP may occasionally fail, although we suspect that this is the exception rather than the norm.

The small data set *Pendulum* (9 dimensions, 315 training and 315 testing samples) represents the problem of predicting the change in angular velocity of a simulated mechanical pendulum over a short time frame (50 ms) as a function of various parameters of the dynamical system. The target variable depends heavily on all inputs and the targets are almost noise free. Figure 8 shows the results of our experiments. Note that we use up to 800 basis functions for investigation, although for computational reasons it would make sense to use the full GP rather than an approximation with more than 315 basis functions. Although the SSGP NMSE performance is good, we see that especially for large number of basis functions, the MNLP performance is spectacularly bad. A closer inspection shows that the mean predictions are quite accurate, the predictive variances are excessively small. This SSGP model thus exhibits overfitting in the form of being overconfident. Note, that the SSGP with fixed spectral points seems to suffer much less from this effect, as would be expected. Interestingly, re-running the SSGP algorithm with different random initializations gives very different predictions, the predictive distributions from separate runs disagreeing wildly. One could perhaps diagnose the occurrence of the problem in this way. The bottom line is that any algorithm which optimizes the marginal likelihood over a large number of parameters, will risk



(a) (b)

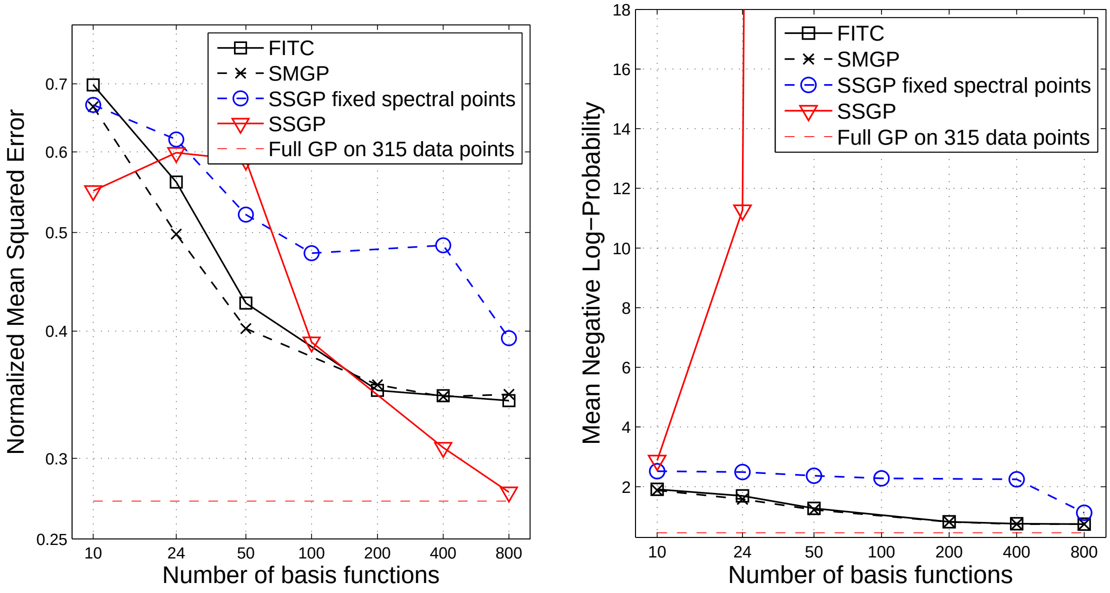
Figure 7: *Pole Telecomm* data set. (a) NMSE and (b) MNLP as a function of the number of basis functions, comparing SSGP with the version with cosines only and explicit phases.

falling in the overfitting trap. We nevertheless think that the SSGP algorithm will often have very good performance, and will be a practically important algorithm, although one must use it with care.

6. Discussion

We have introduced the Sparse Spectrum Gaussian Process (SSGP) algorithm, a novel perspective on sparse GP approximations where rather than the usual sparsity approximation in the spatial domain, it is the spectrum of the covariance function that is subject to a sparse approximation by means of a discrete set of samples, the spectral points. We have provided a detailed comparison of the computational complexity vs. accuracy tradeoff of SSGP to that of the state of the art GP sparse approximation FITC and its extension SMGP. SSGP shows a dramatic improvement in four commonly used benchmark regression data sets, including the two data sets used for evaluation in the paper where FITC was originally proposed (Snelson and Ghahramani, 2006). However, we found a small data set where SSGP badly fails, with good predictive means but with overconfident predictive variances. This indicates that although SSGP is practically a very appealing algorithm, care must be taken to avoid the occasional risk of overfitting.

Other algorithms, such as the variational approach of Titsias (2009) which focus on approaching the full GP in the limit of large numbers of basis functions are to a large degree safeguarded from overfitting. However, algorithms derived from GPs whose focus is on achieving good predictive accuracy on a limited computational budget, such as FITC, SMGP and the currently proposed SSGP,



(a) (b)

Figure 8: *Pendulum* data set. (a) NMSE and (b) MNLP as a function of the number of basis functions.

typically achieve superior performance, see Figure 3 in Titsias (2009), with some risk of overfitting. Note, that these algorithms don't generally converge toward the full GP.

An equivalent view of SSGP is as a sparse Bayesian linear combination of pairs of trigonometric basis functions, a sine and a cosine for each spectral point. The weights are integrated out, and at the price of having two basis functions per frequency, the phases are effectively integrated out as well. We have shown that although a representation in terms of a single basis function per frequency and an explicit phase is possible, learning the phases poses an increased risk of overfitting. If the spectral points are sampled from the power spectrum of a stationary GP, then SSGP approximates its covariance function. However, much sparser solutions can be achieved by learning the spectral points, which effectively implies learning the covariance function. The SSGP model is to the best of our knowledge the only sparse GP approximation that induces a stationary covariance function.

SSGP has been presented here as a Gaussian process prior for regression with a tractable likelihood function from the assumption of Gaussian observation noise. Extending to other types of analytically intractable likelihood functions, such as sigmoid for classification or Laplace for robust regression is possible by using the same approximation techniques as for full GPs. An example is the use of Expectation Propagation in the derivation of generalized FITC (Naish-Guzman and Holden, 2008). Further modifications and extensions of SSGP are discussed in Lázaro-Gredilla (2010).

The main differences between SSGP and most previous approaches to sparse GP regression is the stationarity of the prior and the non-local nature of the basis functions. It will be interesting to investigate more carefully in the future the exact conditions under which these spectacular sparsity vs. accuracy improvements can be expected.

Acknowledgments

This work has been partly supported by an FPU grant (first author) from the Spanish Ministry of Education and CICYT project TEC-2005-00992 (first and last author). Part of this work was developed while the first author was a visitor at the Computational and Biological Learning Lab, Department of Engineering, University of Cambridge.

Appendix A. Details of the Implementation

In practice, to improve numerical accuracy and speed, Equations (7) and (8) should be implemented using the Cholesky decomposition $\mathbf{R} = \text{chol}(\mathbf{A})$. Thus the predictive distribution is computed as

$$\mathbb{E}[y_*] \;=\; \phi(\mathbf{x}_*)^\top \mathbf{R}\backslash(\mathbf{R}^\top\backslash(\Phi_\mathbf{f}\mathbf{y})) \qquad \mathbb{V}[y_*] \;=\; \sigma_n^2 + \sigma_n^2 ||\mathbf{R}^\top\backslash\phi(\mathbf{x}_*)||^2,$$

and the log evidence as

$$\log p(\mathbf{y}|\theta) \;=\; -\frac{1}{2\sigma_n^2}\left[||\mathbf{y}||^2 - ||\mathbf{R}^\top\backslash(\Phi_\mathbf{f}\mathbf{y})||^2\right] - \frac{1}{2}\sum_i \log \mathbf{R}_{ii}^2 + m \log \frac{m\sigma_n^2}{\sigma_0^2} - \frac{n}{2}\log 2\pi\sigma_n^2\,,$$

where $\mathbf{R}_{ii}$ refers to the diagonal elements of $\mathbf{R}$.

References

G. L. Bretthorst. Nonuniform sampling: Bandwidth and aliasing. In *Maximum Entropy and Bayesian Methods*, pages 1–28. Kluwer, 2000.

A. B. Carlson. *Communication Systems*. McGraw-Hill, 3rd edition, 1986.

L. Csató and M. Opper. Sparse online Gaussian processes. *Neural Computation*, 14(3):641–669, 2002.

M. Lázaro-Gredilla. *Sparse Gaussian Processes for Large-Scale Machine Learning*. PhD thesis, Universidad Carlos III de Madrid, 2010. URL `http://www.tsc.uc3m.es/~miguel/publications.php`.

M. Lázaro-Gredilla and A.R. Figueiras-Vidal. Inter-domain Gaussian processes for sparse inference using inducing features. In *Advances in Neural Information Processing Systems 22*, pages 1087–1095. MIT Press, 2010.

M. Lázaro-Gredilla, J. Quiñonero-Candela, and A. Figueiras-Vidal. Sparse spectral sampling Gaussian processes. Technical report, Microsoft Research, 2007.

D. J. C. MacKay. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2003.

A. Naish-Guzman and S. Holden. The generalized FITC approximation. In *Advances in Neural Information Processing Systems 20,* pages 1057–1064. MIT Press, 2008.

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

J. Quiñonero-Candela, C. E. Rasmussen, and C. K. I. Williams. Approximation methods for Gaussian process regression. In L. Bottou, O. Chapelle, D. DeCoste, and J. Weston, editors, *Large-Scale Kernel Machines*, pages 203–223. MIT Press, 2007.

A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems 20,* pages 1177–1184. MIT Press, Cambridge, MA, 2008.

C. E. Rasmussen and Joaquin Quiñonero-Candela. Healing the relevance vector machine through augmentation. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 689–696, 2005.

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

M. Seeger, C. K. I. Williams, and N. D. Lawrence. Fast forward selection to speed up sparse Gaussian process regression. In *Proceedings of the 9th International Workshop on AI Stats*, 2003.

A. J. Smola and P. Bartlett. Sparse greedy Gaussian process regression. In *Advances in Neural Information Processing Systems 13,* pages 619–625. MIT Press, 2001.

E. Snelson and Z. Ghahramani. Sparse Gaussian processes using pseudo-inputs. In *Advances in Neural Information Processing Systems 18,* pages 1259Ű–1266. MIT Press, 2006.

M. L. Stein. *Interpolation of Spatial Data*. Springer Verlag, 1999.

M. K. Titsias. Variational learning of inducing variables in sparse Gaussian processes. In *Proceedings of the 12th International Workshop on AI Stats*, 2009.

V. Tresp. A Bayesian committee machine. *Neural Computation*, 12:2719–2741, 2000.

R. Urtasun and T. Darrell. Sparse probabilistic regression for activity-independent human pose inference. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8, 2008.

C. Walder, K. I. Kim, and B. Schölkopf. Sparse multiscale Gaussian process regression. In *25th International Conference on Machine Learning*. ACM Press, New York, 2008.

C. K. I. Williams. Computing with infinite networks. In *Advances in Neural Information Processing Systems 9*, pages 1069–1072. MIT Press, 1997.

C. K. I. Williams and M. Seeger. Using the Nyström method to speed up kernel machines. In *Advances in Neural Information Processing Systems 13,* pages 682–688. MIT Press, 2001.