

000

001

002

003

004

005

006

007

008

009

010

011

012

013

014

015

016

017

018

019

020

021

022

023

024

025

026

027

028

029

030

031

032

033

034

035

036

037

038

039

040

041

042

043

044

045

046

047

048

049

050

051

052

053

# Gaussian Processes with Probabilistic Programming

**Anonymous Author(s)**

Affiliation

Address

email

## Abstract

Abstract

## 1 Introduction

MCMC lends itself to Bayesian interpretations of Gaussian Processes since they can provide a vehicle to express otherwise intractable integrals necessary for a fully Bayesian representation.

## 2 Gaussian Processes

In the following, we will introduce GP related theory and notations. We will exclusively work on two variable regression problems. Let the data be real-valued scalars  $\{x_i, y_i\}_{i=1}^n$  (complete data will be denoted by column vectors  $\mathbf{x}, \mathbf{y}$ ). GPs present a non-parametric way to express prior knowledge on the space of possible functions  $f$  that we assume to have generated the data.  $f$  is assumed latent and the GP prior is given by a multivariate Gaussian with mean and covariance  $f(\mathbf{x}) \sim \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}'))$ , where  $m(\mathbf{x})$  is a function of the mean of all functions that map to  $y_i$  at  $x_i$  and  $k(\mathbf{x}, \mathbf{x}')$  is a kernel or covariance function that summarizes the covariance of all functions that map to  $y_i$  at  $x_i$ . We can absorb the mean function into the covariance function so without loss of generality we can set the mean to zero. The marginal likelihood can be expressed as:

$$p(\mathbf{y}|\mathbf{x}) = \int p(\mathbf{y}|\mathbf{f}, \mathbf{x}) p(\mathbf{f}|\mathbf{x}) d\mathbf{f} \quad (1)$$

where the prior is Gaussian  $\mathbf{f}|\mathbf{x} \sim \mathcal{N}(0, k(\mathbf{x}, \mathbf{x}'))$ . For a zero mean Gaussian Process this results in a Gaussian posterior  $\mathcal{N}(\boldsymbol{\mu}, \Sigma)$  with mean:

$$\boldsymbol{\mu} = \mathbf{K}(\mathbf{x}, \mathbf{x}^*) \mathbf{K}(\mathbf{x}^*, \mathbf{x}^*)^{-1} \mathbf{y} \quad (2)$$

and covariance

$$\Sigma = \mathbf{K}(\mathbf{x}, \mathbf{x}) + \mathbf{K}(\mathbf{x}, \mathbf{x}^*) \mathbf{K}(\mathbf{x}^*, \mathbf{x}^*)^{-1} \mathbf{K}(\mathbf{x}^*, \mathbf{x}). \quad (3)$$

where  $\mathbf{K}$  is a covariance function. The covariance function covers general high-level properties of the observed data such as linearity, periodicity and smoothness. The most widely used type of covariance function is the squared exponential covariance function:

$$k(x, x') = \sigma^2 \exp\left(-\frac{(x - x')^2}{2\ell^2}\right) \quad (4)$$

where  $\sigma$  and  $\ell$  are hyper-parameters.  $\sigma$  is a scaling factor and  $\ell$  is the typical length-scale. Smaller variations can be achieved by exchanging these hyper-parameters. Below, we see how we can express simple GP smoothing with a few

054 lines of Venture code while allowing users to custom design covariance functions.  
055

056 Listing 1: GP Smoothing

```
057 [ASSUME 1 1]
058 [ASSUME sf 2]
059
060  $k(x, x') := \sigma^2 \exp\left(-\frac{(x-x')^2}{2\ell^2}\right)$ 
061
062 [ASSUME f VentureFunction(k, σ, ℓ) ]
063 [ASSUME SE make-se (apply-function f 1 sf) ]
064 [ASSUME (make-gp 0 SE ) ]
```

065  
066 In the case where hyper-parameters are unknown they can be found deterministically by optimizing  
067 the marginal likelihood using a gradient based optimizer. Non-deterministic, Bayesian representa-  
068 tions of this case are also known (Neal, 1997). Extending the program described in listing 1 for a  
069 Bayesian treatment of hyper-parameters is simple using the build in stochastic procedure that simu-  
070 lates drawing samples from a gamma distribution:

071 Listing 2: Bayesian GP Smoothing

```
072 [ASSUME 1 (gamma 1 3)]
073 [ASSUME sf (gamma 1 2)]
074
075  $k(x, x') := \sigma^2 \exp\left(-\frac{(x-x')^2}{2\ell^2}\right)$ 
076
077 [ASSUME f VentureFunction(k, σ, ℓ) ]
078 [ASSUME SE make-se (apply-function f 1 sf) ]
079 [ASSUME (make-gp 0 SE ) ]
```

080  
081 Larger variations are achieved by changing the type of the covariance function structure. A different  
082 type could be a linear covariance function:  
083

$$k(x, x') = \sigma^2(x - \ell)(x' - \ell). \quad (5)$$

084  
085  
086  
087  
088  
089 Note that covariance function structures are compositional. We can add covariance functions if we  
090 want to model globally valid structures  
091

$$k_3(x, x') = k_1(x, x') + k_2(x, x') \quad (6)$$

092  
093  
094  
095  
096  
097 and we can multiply covariance functions if the data is best explained by local structure  
098

$$k_4(x, x') = k_1(x, x') \times k_2(x, x'); \quad (7)$$

100  
101  
102  
103  
104 both,  $k_3$  and  $k_4$  are valid covariance function structures. This leads to an infinite space of possible  
105 structures that could potentially explain the observed data best (e.g. Fig. ??). In the following, we  
106 will refer to covariance functions that are not composite as base covariance functions. Note that this  
107 form of composition can be easily expressed in Venture, for example if one wishes to add a linear  
and a periodic kernel:

108

109

Listing 3: LIN × PER

```

110 [ASSUME 1 (gamma 1 3)]
111 [ASSUME sf (gamma 1 2)]
112 [ASSUME a (gamma 2 2)]
113
114  $k_{LIN}(x, x') = \sigma_1^2(x - \ell)(x' - \ell)$ 
115  $k_{PER}(x, x') := \sigma_2^2 \exp\left(-\frac{2 \sin^2(\pi(x-x')/p)}{\ell^2}\right)$ 
116
117 [ASSUME fLIN VentureFunction( $k_{LIN}, \sigma_1$ ) ]
118 [ASSUME fPER VentureFunction( $k_{PER}, \sigma_2, \ell, p$ ) ]
119 [ASSUME LIN (make-LIN (apply-function fLIN a)) ]
120 [ASSUME PER (make-PER (apply-function fPER 1 sf)) ]
121 [ASSUME (make-gp 0 (function-times LIN PER)) ]
122

```

123 Knowledge about the composite nature of covariance functions is not new, however, until recently,  
124 the choice and the composition of covariance functions were done ad-hoc. The Automated Statisti-  
125 cian Project came up with an approximate search over the possible space of kernel structures (Du-  
126 venuaud et al., 2013; Lloyd et al., 2014).

127

## 128 2.1 A Bayesian interpretation

129

130 In the following, we will explore a Bayesian representation of GP. The probability of the hyper-  
131 parameters of a GP with assumptions as above and given covariance function structure  $\mathbf{K}$  can be  
132 described as:

$$133 P(\boldsymbol{\theta} | \mathbf{D}, \mathbf{K}) = \frac{P(\mathbf{D} | \boldsymbol{\theta}, \mathbf{K})P(\boldsymbol{\theta} | \mathbf{K})}{P(\mathbf{D} | \mathbf{K})}. \quad (8)$$

134

135 We are interested in the case where covariance structure is not given. Our probabilistic programming  
136 based MCMC framework approximates the following intractable integrals of the expectation for the  
137 prediction:

$$138 \mathbb{E}[y^* | x^*, D, \mathbf{K}_\Omega^s] = \iint f(x^*, \boldsymbol{\theta}, \mathbf{K}) P(\boldsymbol{\theta} | \mathbf{D}, \mathbf{K}) P(\mathbf{K} | \Omega, s, n) d\boldsymbol{\theta} d\mathbf{K}. \quad (9)$$

140

141 This is done by sampling from the posterior probability distribution of the hyper-parameters and the  
142 possible kernel:

$$143 y^* \approx \frac{1}{T} \sum_{t=1}^T f(x^* | \boldsymbol{\theta}^{(t)}, \mathbf{K}^{(t)}). \quad (10)$$

145

146

## 3 Stochastic Processes

147

148

In order to provide the sampling of the kernel, we introduce a stochastic process to the SP that  
simulates the grammar for algebraic expressions of kernel algebra. Here, we start with a set of  
possible kernels and draw a random subset. For this subset of size  $n$ , we sample a set of possible  
operators that operate on the base kernels.

151

152

The marginal probability of a kernel structure which allows us to sample is characterized by the  
probability of a uniformly chosen subset of the set of  $n$  possible covariance functions times the  
probability of sampling a global or a local structure which is given by a binomial distribution:

155

156

$$P(\mathbf{K} | \Omega, s, n) = P(\Omega | s, n) \times P(s | n) \times P(n), \quad (11)$$

157

with

158

159

$$P(\Omega | s, n) = \binom{n}{r} p_{+ \times}^k (1 - p_{+ \times})^{n-k} \quad (12)$$

160

and

161

$$P(s | n) = \frac{n!}{|s|!} \quad (13)$$

162 where  $P(n)$  is a prior on the number of base kernels used. It is possible to also assign a prior for the  
 163 probability to sample global or local priors, however, we have assigned complete uncertainty to this  
 164 with the binomial  $p = 0.5$ .

## 166 4 Experiments

168 Neal suggested the treatment of outliers as a use-case for a Bayesian treatment of Gaussian pro-  
 169 cesses (1997). He evaluates his MCMC setting using the following synthetic data problem. Let  $f$   
 170 be the underlying function that generates the data:

$$172 \quad f(x) = 0.3 + 0.4x + 0.5 \sin(2.7x) + \frac{1.1}{(1+x^2)} + \eta \quad \text{with } \eta \sim \mathcal{N}(0, \sigma) \quad (14)$$

174 We synthetically generate outliers by setting  $\sigma = 0.1$  in 95% of the case and to  $\sigma = 1$  in the  
 175 remaining cases. Venture GPs can capture the true underlying function within only 100 MH steps  
 176 (see Fig. 2). Note that Neal devices an additional noise model and performs large numbe of Hybrid-  
 177 Monte Carlo and Gibbs steps.

### 179 4.1 Structure Learning

181 . To inspect the posterior of the structure of the covariance function we convert each kernel expres-  
 182 sion into a sum of products and subsequently simplify expressions using the following grammar:

183 Listing 4: Grammar to simplify expressions

185 SE × SE	→ SE
186 {SE, PER, C, WN} × WN	→ WN
187 LIN + LIN	→ LIN
188 {SE, PER, C, WN, LIN} × C	→ {SE, PER, C, WN, LIN}

189 We defined a set of covariance structures so that we could reproduce results for covariance function  
 190 structure learning as in the Automated Statistician. Our results are very similar to what has been  
 191 reported by previous work ( Duvenaud et al., 2013; see Fig. 3).

### 193 4.2 Log-Likelihood

### 195 4.3 Residuals

## 197 References

- 199 Duvenaud, D., Lloyd, J. R., Grosse, R., Tenenbaum, J., and Ghahramani, Z. (2013). Structure  
 200 discovery in nonparametric regression through compositional kernel search. In *Proceedings of  
 201 the 30th International Conference on Machine Learning (ICML-13)*, pages 1166–1174.
- 202 Lloyd, J. R., Duvenaud, D., Grosse, R., Tenenbaum, J., and Ghahramani, Z. (2014). Automatic  
 203 construction and natural-language description of nonparametric regression models. In *Twenty-  
 204 Eighth AAAI Conference on Artificial Intelligence*.
- 205 Neal, R. M. (1997). Monte carlo implementation of gaussian process models for bayesian regression  
 206 and classification. *arXiv preprint physics/9701026*.

207  
 208  
 209  
 210  
 211  
 212  
 213  
 214  
 215

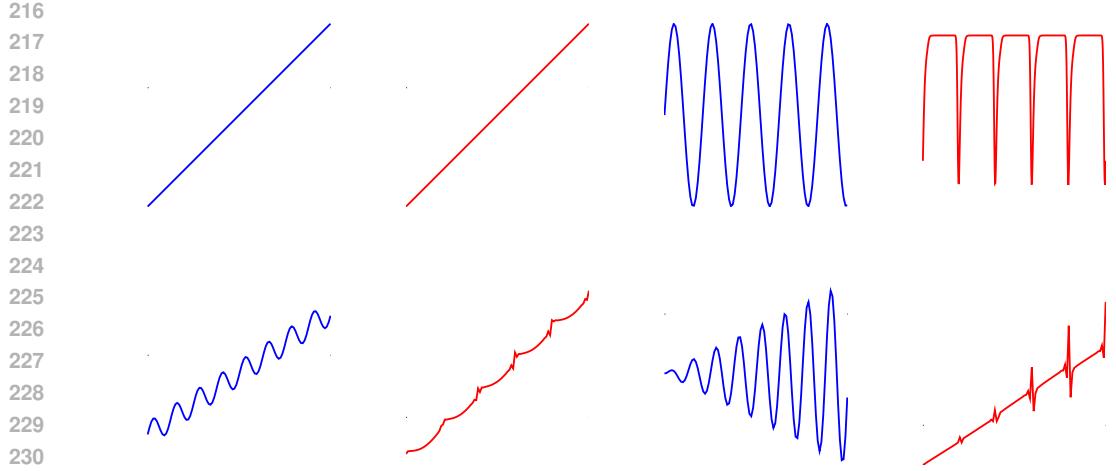


Figure 1: Composition of covariance functions (blue, left) and samples from the distribution of curves they can produce (red, right).

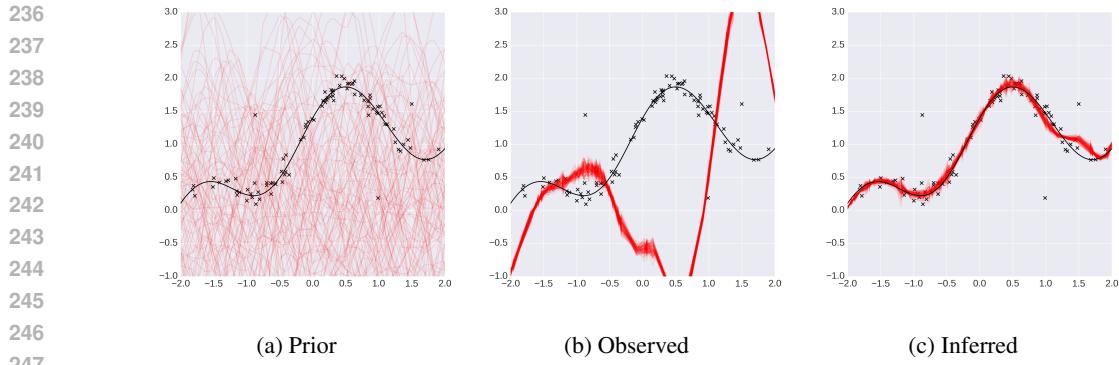


Figure 2: Running a Venture GP on Neal’s example for MCMC showing the prior, after having observed the data and after performing inference on the hyper-parameters. Note how the GP is choosing outliers to smooth instead of essential data.

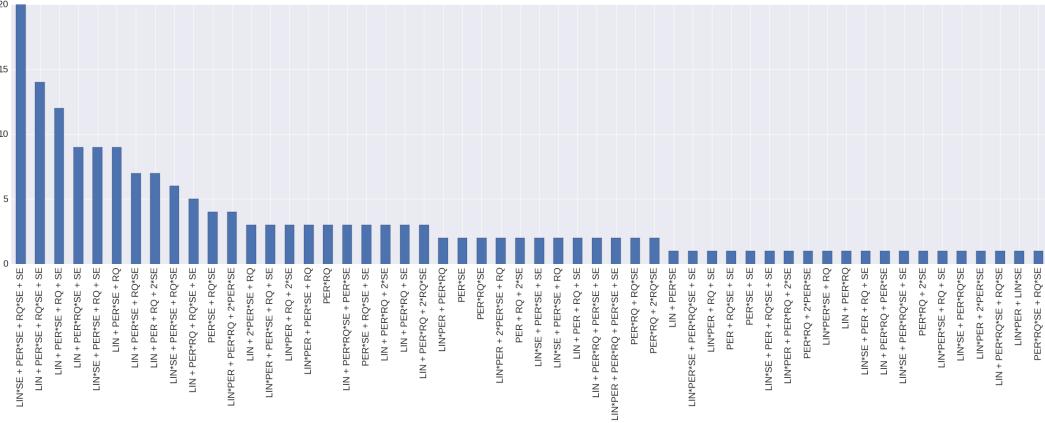
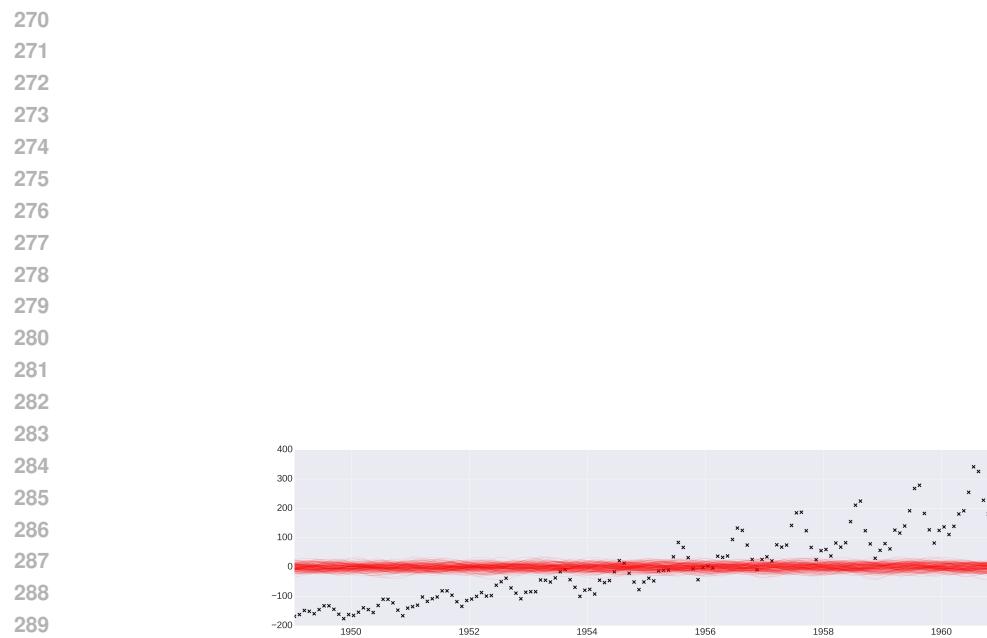
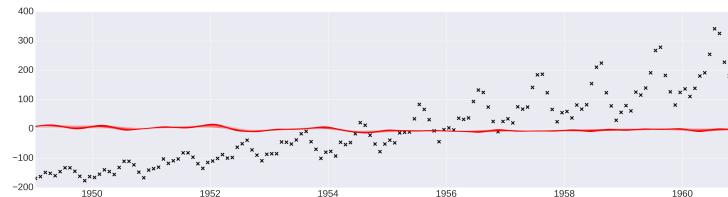


Figure 3: Preliminary results from the cross-validation on the CO<sub>2</sub> data. Note that Duvenaud et al. (2013) report LIN × SE + PER × SE + RQ × SE. We have run a leave one out cross-validation on this data set. Above we see the preliminary results on 181 validations (of a total of 545 × 2 runs).



290 (a) Prior, before having seen any data.  
291



299 (b) After having seen any data but before inference  
300



310 (c) After 1000 MH steps.  
311

312 Figure 4: Running a Venuter GP with covariance structure PER x SE on the airline data  
313

314  
315  
316  
317  
318  
319  
320  
321  
322  
323