Function factorization using warped Gaussian processes

Mikkel N. Schmidt MNS@IMM.DTU.DK
University of Cambridge, Department of Engineering, Trumpington Street, Cambridge, CB2 1PZ, UK

Abstract

We introduce a new approach to non-linear regression called function factorization, that is suitable for problems where an output variable can reasonably be modeled by a number of multiplicative interaction terms between non-linear functions of the inputs. The idea is to approximate a complicated function on a high-dimensional space by the sum of products of simpler functions on lower-dimensional subspaces. Function factorization can be seen as a generalization of matrix and tensor factorization methods, in which the data are approximated by the sum of outer products of vectors. We present a non-parametric Bayesian approach to function factorization where the priors over the factorizing functions are warped Gaussian processes, and we do inference using Hamiltonian Markov chain Monte Carlo. We demonstrate the superior predictive performance of the method on a food science data set compared to Gaussian process regression and tensor factorization using PARAFAC and GEMANOVA models.

1. Introduction

In many regression problems, the output variable can only be reasonably explained by interactions between the input variables. An example, which we shall return to in our experiments, is measurements of the color of meat under different storage conditions. Color is an important quality that affects the consumers choice, and it is thus important to understand and model how the color is influenced by different explanatory factors such as storage time, temperature, oxygen content in the atmosphere, and exposure to light. It is reasonable to assume, that the color of meat does not vary linearly with each of the explanatory variables, but that it depends on interactions of the explanatory variables in a non-linear fashion.

In this paper we present a new approach to non-linear regression that is suitable for problems where the output variable can be reasonably explained by non-linear interactions of the inputs. The goal in regression analysis is to infer a mapping function, $y : \mathcal{X} \rightarrow \mathbb{R}$, based on N observed input-output pairs, $\mathcal{D} = \{(\boldsymbol{x}_n, y_n)\}_{n=1}^N$, where $\boldsymbol{x}_n \in \mathcal{X}$ are inputs and $y_n \in \mathbb{R}$ are outputs. The Bayesian approach to the regression problem is to formulate a prior distribution over mapping functions and combine this with the data using a suitable observation model to infer the posterior distribution over the mapping function. The posterior can then, for example, be used to make inference about the value of the output, y_*, at a previously unseen position in input space, $\boldsymbol{x}_*$, by computing the predictive distribution which involves integrating over the posterior.

The main idea in function factorization is to approximate a complicated function, $y(\boldsymbol{x})$, on a high-dimensional space, $\mathcal{X}$, by the sum of products of a number of simpler functions, $f^{i,k}(\boldsymbol{x}^i)$, on lower dimensional subspaces, $\mathcal{X}^i$,

$$y(\boldsymbol{x}) \approx \sum_{k=1}^{K} \prod_{i=1}^{I} f^{i,k}(\boldsymbol{x}^i). \tag{1}$$

We refer to this as a K-component I-factor function factorization model.

In the model, we assume that the inputs, $\boldsymbol{x}_n \in \mathcal{X}$, can naturally be divided into I inputs that lie in subspaces of $\mathcal{X}$, $\boldsymbol{x}_n^1 \in \mathcal{X}^1, \ldots, \boldsymbol{x}_n^I \in \mathcal{X}^I$, and that the output is reasonably modeled by a number of multiplicative interaction terms between non-linear functions of the inputs. The subspaces need not be chosen to be disjoint; for example, two subspaces could be identical which makes it possible for the model to capture non-stationary modulation effects.

Bayesian inference in function factorization models requires the specification of a likelihood function as well as priors over the factorizing functions. These priors

Appearing in *Proceedings of the 26th International Conference on Machine Learning*, Montreal, Canada, 2009.

could for example be chosen by selecting a suitably flexible parameterized family of functions and assume prior distributions over the parameters. Another approach, which we will pursue in this paper, is to assume a non-parametric distribution over the functions. Specifically, we choose warped Gaussian process priors.

2. Relation to other methods

Function factorization with warped Gaussian process priors (FF-WGP) generalizes a number of other well known machine learning techniques including matrix and tensor factorization, linear regression, and warped Gaussian process regression. In the following, we give an overview of the relation to these methods.

2.1. Relation to matrix and tensor factorization

Function factorization generalizes matrix and tensor factorization models, as illustrated in Figure 1. In matrix factorization, a data matrix, $\boldsymbol{Y}$, is approximated by the product of two matrices, $\boldsymbol{F}_1$ and $\boldsymbol{F}_2$,

$$\boldsymbol{Y} \approx \boldsymbol{F}_1^\top \boldsymbol{F}_2. \tag{2}$$

To make the relation to function factorization explicit, we can rewrite this as

$$y_{n_1,n_2} \approx \sum_{k=1}^{K} \prod_{i=1}^{2} f_{n_i}^{i,k}, \tag{3}$$

where y_{n_1,n_2} is element (n_1, n_2) of $\boldsymbol{Y}$ and $f_n^{i,k}$ is element (k, n) of $\boldsymbol{F}_i$. Comparing this with Eq. (1), we note that the main difference between matrix factorization and function factorization is that in the former the goal is to learn a set of parameters, $f_{n_i}^{i,k}$, whereas in the latter a set of functions, $f^{i,k}(\boldsymbol{x}^i)$, are learned.

In matrix factorization data points lie on a regular grid in the joint space of column and row indices, (n_1, n_2), and are approximated by the sum of outer products of two factors, which are vectors defined on the column and row indices respectively. In function factorization, data points lie in the space $\mathcal{X}$ and are approximated by the sum of products of functions defined on subspaces of $\mathcal{X}$.

In function factorization the priors can be chosen, for example, to have support over the non-negative numbers or to have a sub- or super-Gaussian density, which will lead to generalizations of certain Bayesian formulations of non-negative matrix factorization (NMF) and independent component analysis (ICA). Similar analogies exists between function factorization and higher-order decompositions of tensors such as the parallel factor analysis (PARAFAC) model.

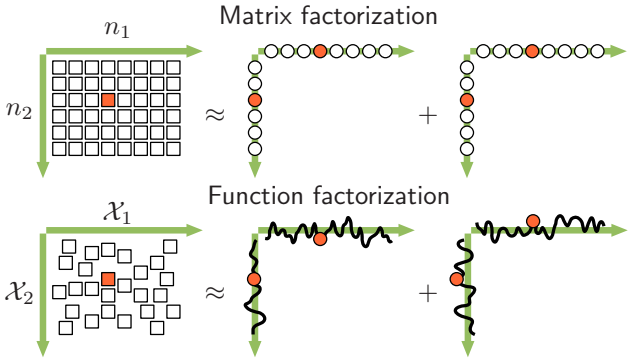

Figure 1. Illustration of the relation between matrix factorization and function factorization. In matrix factorization, data lie on a regular grid and is approximated by a product-sum of vectors. In function factorization, data lie in $\mathcal{X}$ and are approximated by a product-sum of functions over subspaces of $\mathcal{X}$.

Schmidt and Laurberg's (2008) method for non-negative matrix factorization with Gaussian process priors can be seen as a K-component two-factor function factorization model, where the data is required to be in the form of a matrix, and inference is done by computing a maximum a posteriori estimate.

2.2. Relation to linear regression

Function factorization can also be seen as a generalization of linear regression. To show this, we start from the basic linear regression equation, where the outputs are modeled by a linear combination of the inputs,

$$y_n \approx \sum_{i=k}^{K} x_n^k \beta_k, \tag{4}$$

where x_n^k denotes the kth element of $\boldsymbol{x}_n$ and β_k are regression coefficients. Since this model is linear and additive in the inputs, it does not model non-linear effects and interactions between the inputs. To overcome these issues, linear regression can be performed on non-linear interactive terms,

$$y_n \approx \sum_{k=1}^{K} f^k(\boldsymbol{x}_n^i)\beta_k, \tag{5}$$

where f^k are non-linear functions of all the input variables. To show the relation to function factorization, we choose f^k as a product of non-linear transforma-

tions of the inputs,

$$y_n \approx \sum_{k=1}^{K} \left(\prod_{i=1}^{I} f^{i,k}(\boldsymbol{x}_n^i) \right) \beta_k. \qquad (6)$$

This expression is very similar to Eq. (1); however, in linear regression the objective is to learn the regression coefficients, β_k, for fixed non-linear transformations of the inputs, whereas in function factorization the aim is to learn the non-linear functions themselves. Function factorization generalizes this formulation of linear regression, since the regression coefficients without loss of generality can be incorporated into the non-linear functions.

2.3. Relation to warped Gaussian processes

Function factorization generalizes warped Gaussian process (WGP) regression, when WGP priors are assumed over the factorizing functions. Obviously, when $K = I = 1$, FF-WGP collapses to WGP regression; however, in models with multiple factors the two methods differ in the assumptions that are made about the data.

A simple illustration is given in Figure 2 for a two-dimensional toy data problem generated as the product of two cosine functions. Fifteen data points were chosen from the function and regression analysis was performed using a GP and a one-component two-factor function factorization with GP priors. In regions of the function that are far away from any of the observed input points, the GP tends to its zero mean prior. The function factorization method on the other hand assumes that the data has a multi-linear structure, and since that assumption in this case is correct, the data is modeled accurately even in regions with no observations. In many data sets, the assumption that the outputs are well modeled by multiplicative interactive terms is reasonable, and when the assumptions holds, function factorization can provide better results than Gaussian process regression based methods.

Adams and Stegle's (2008) Gaussian process product model, in which a function is modeled as the point-wise product of two Gaussian processes for the purpose of capturing non-stationarities, can be seen as a one-component two-factor function factorization model with Gaussian process priors, where the two factorizing functions are both defined over the entire space, $\mathcal{X}$.

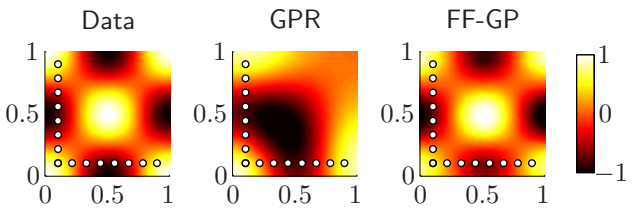

Figure 2. A simple toy example that illustrates important differences between Gaussian process regression and function factorization. Left: A two-dimensional toy data set is constructed as the multiplication of two cosines and 15 data points are chosen at the indicated positions. Middle: Gaussian process regression on the data points fits the data well in the region close to the observation, and far away from the observations it tends to its zero mean prior. Right: A one-component function factorization on the same data fits the data well in the entire region, due to the correct assumption that the data comes from a product of two functions.

3. Function factorization using warped Gaussian processes

In the following, we describe a non-parametric Bayesian approach to function factorization using warped Gaussian process (WGP) priors over the factorizing functions. First, we give a summary of the WGP, and then we describe function factorization using WGP priors. Finally, we present an inference procedure based on Hamiltonian Markov chain Monte Carlo (HMCMC).

3.1. Warped Gaussian processes

A Gaussian process[1] (GP) is a flexible and practical method for specifying a non-parametric distribution over a function, $g(\boldsymbol{x})$. It is fully characterized by its mean function, $m(\boldsymbol{x}) = E\big[g(\boldsymbol{x})\big]$, and its covariance function, $c(\boldsymbol{x}, \boldsymbol{x}') = E\big[\big(g(\boldsymbol{x}) - m(\boldsymbol{x})\big)\big(g(\boldsymbol{x}') - m(\boldsymbol{x}')\big)\big]$. We use the notation $g(\boldsymbol{x}) \sim \mathcal{GP}\big(m(\boldsymbol{x}), c(\boldsymbol{x}, \boldsymbol{x}')\big)$ to denote a random function drawn from a GP.

The GP is limited in the sense that it assumes that any finite subset of values of the function $g(\boldsymbol{x})$ follow a joint multivariate Gaussian distribution. The idea behind the warped Gaussian process (WGP) (Snelson et al., 2004) is to overcome this limitation by mapping the GP through a non-linear warp function, $h(g)$, parameterized by $\boldsymbol{\theta}_h$,

$$y(\boldsymbol{x}) = h\big(g(\boldsymbol{x})\big), \quad g(\boldsymbol{x}) \sim \mathcal{GP}\big(m(\boldsymbol{x}), c(\boldsymbol{x}, \boldsymbol{x}')\big), \qquad (7)$$

[1] See Rasmussen and Williams (2006) for a comprehensive introduction to Gaussian processes in machine learning.

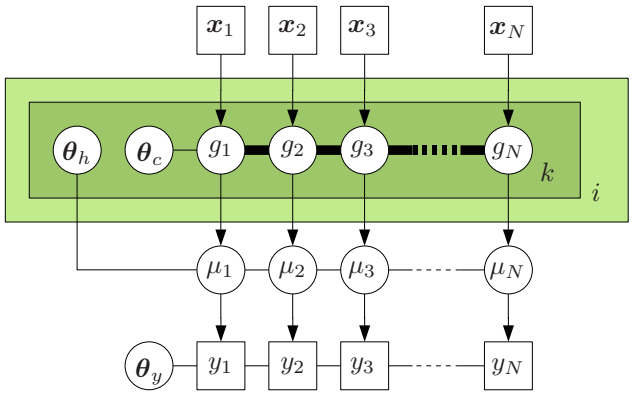

Figure 3. Graphical model for the function factorization model with warped Gaussian process priors. Squares represent observed variable and circles denote unobserved variables and parameters. The bold line indicates that the nodes $g_1^{i,k}, \ldots, g_N^{i,k}$ are fully connected.

and jointly learn the parameters of the GP and the warp function. Snelson et al. (2004) learn the parameters of the WGP by maximum likelihood, but they note that priors can be included to learn maximum a posteriori estimates or Markov chain Monte Carlo can be used to integrate out the parameters.

3.2. The FF-WGP model

Let $\mu(\boldsymbol{x})$ denote a function factorization model,

$$\mu(\boldsymbol{x}) = \sum_{k=1}^{K} \prod_{i=1}^{I} f^{i,k}(\boldsymbol{x}), \tag{8}$$

where $f^{i,k}(\boldsymbol{x})$ are modeled by zero mean WGPs. In the most general formulation, each factor in each component has distinct warp and covariance functions,

$$f^{i,k}(\boldsymbol{x}) = h^{i,k}\big(g^{i,k}(\boldsymbol{x})\big), \tag{9}$$

$$g^{i,k}(\boldsymbol{x}) \sim \mathcal{GP}\big(0, c^{i,k}(\boldsymbol{x}, \boldsymbol{x}')\big). \tag{10}$$

We then model the outputs, y_n, as independent and identically distributed given $\mu_n = \mu(\boldsymbol{x}_n)$, using some likelihood function, $p(y_n|\mu_n)$, parameterized by $\boldsymbol{\theta}_y$.

A graphical model of the FF-WGP is shown in Figure 3. The unknowns in the model are the parameters of the likelihood function, $\boldsymbol{\theta}_y$, the warp functions, $\boldsymbol{\theta}_h$, and the covariance functions, $\boldsymbol{\theta}_c$, as well as the latent variables, $g_n^{i,k} = g^{i,k}(\boldsymbol{x}_n)$.

3.3. Change of parameters

The latent variables, $g_n^{i,k}$, have a multivariate Gaussian distribution a priori, and are thus highly correlated. We have found empirically that MCMC inference in the model is more efficient when we perform a change of variables, such that the latent variables are uncorrelated a priori. We do this by defining a new latent variable, $z_{n'}^{i,k}$, related to $g_n^{i,k}$ by

$$g_n^{i,k} = \sum_{n'=1}^{N} C_{n',n}^{i,k} z_{n'}^{i,k}, \tag{11}$$

where $C_{n,n'}^{i,k}$ holds the Cholesky decomposition of the covariance matrix of the (i,k)th Gaussian process,

$$\sum_{n''=1}^{N} C_{n,n''}^{i,k} C_{n',n''}^{i,k} = c_{n,n'}^{i,k} = c^{i,k}(\boldsymbol{x}_n, \boldsymbol{x}_{n'}). \tag{12}$$

With this change of variables, the model can be written as

$$\mu_n = \sum_{k=1}^{K} \prod_{i=1}^{I} h^{i,k}\left(\sum_{n'=1}^{N} C_{n',n}^{i,k} z_{n'}^{i,k}\right), \tag{13}$$

and the prior over the new latent variables are i.i.d. standard Gaussian, $z_{n'}^{i,k} \sim \mathcal{N}(0,1)$. We emphasize that this change of variables does not change the model—only its parameterization.

3.4. Posterior and predictive distribution

The joint posterior distribution of the latent variables and the parameters conditioned on the data is given by

$$p(\boldsymbol{z}, \boldsymbol{\theta}|\mathcal{D}) \propto p(\boldsymbol{\theta}) \prod_{n=1}^{N} \left(p(y_n|\mu_n) \prod_{i=1}^{I} \prod_{k=1}^{K} p(z_n^{i,k}) \right), \tag{14}$$

where $\boldsymbol{z} = \{z_n^{i,k}\}$ denotes all latent variables, $\boldsymbol{\theta} = \{\boldsymbol{\theta}_y, \boldsymbol{\theta}_h, \boldsymbol{\theta}_c\}$ denotes all parameters, and μ_n is defined in Eq. (13).

To infer the value of the output y_* at a previously unseen point $\boldsymbol{x}_*$ we must evaluate the predictive distribution, which requires integrating the posterior distribution over the latent variables and parameters,

$$p(y_*|\boldsymbol{x}_*, \mathcal{D}) = \iint p(y_*|\boldsymbol{x}_*, \boldsymbol{z}, \boldsymbol{\theta}) p(\boldsymbol{z}, \boldsymbol{\theta}|\mathcal{D}) d\boldsymbol{z} d\boldsymbol{\theta}. \tag{15}$$

Since this integral is analytically intractable, we approximate it by Monte Carlo sampling, i.e., we draw M samples, $\{\boldsymbol{z}^m, \boldsymbol{\theta}^m\}_{m=1}^{M}$, from the posterior distribution in Eq. (14) and approximate Eq. (15) by the sum

$$p(y_*|\boldsymbol{x}_*, \mathcal{D}) \approx \sum_{m=1}^{M} p(y_*|\boldsymbol{x}_*, \boldsymbol{z}^m, \boldsymbol{\theta}^m). \tag{16}$$

This expression can be directly evaluated, if the input point, $\boldsymbol{x}_*$, coincides with at least one data point, $\boldsymbol{x}_n$,

on all subspaces, $\mathcal{X}^i$, because then all required latent variables are instantiated in the posterior sample. In the toy example in Figure 2, for example, this means that Eq. 16 can be directly evaluated on the 8-by-8 grid formed by the 15 input points. To evaluate the predictive distribution outside these points we instead draw samples from the predictive distribution.

3.5. Inference using Hamiltonian Markov chain Monte Carlo

Since we can not directly draw samples from the posterior distribution in Eq. (14), we use a Markov chain Monte Carlo sampling procedure. Hamiltonian Markov chain Monte Carlo (HMCMC) (Duane et al., 1987) is an attractive method for this problem, because it improves on the sometimes slow convergence rates due to the random walk behavior of other MCMC methods such as Gibbs sampling and Metropolis-Hastings. HMCMC requires the computation of derivatives of the logarithm of the posterior distribution with respect to all variables and parameters, which can be done as we show in the following.

We define $\mathcal{L}$ which is proportional to the negative log posterior up to an additive constant

$$\begin{aligned}\mathcal{L} = & -\log p(\boldsymbol{\theta}) - \sum_{n=1}^{N}\Bigg(\log p(y_n|\mu_n) \\ & - \sum_{i=1}^{I}\sum_{k=1}^{K}\frac{\left(z_n^{i,k}\right)^2}{2}\Bigg).\end{aligned} \tag{17}$$

We now need to compute the derivatives of $\mathcal{L}$ with respect to the latent variables and all parameters of the model. The derivative with respect to the latent variables requires the computation of the derivate of the log likelihood and of the warp functions,

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial z_{n'}^{i,k}} = & -\sum_{n=1}^{N}\frac{\partial \log p(y_n|\mu_n)}{\partial \mu_n}\left(\prod_{i'\neq i} h_n^{i',k}\right)\times \\ & \frac{\partial h_n^{i,k}}{\partial g_n^{i,k}} C_{n',n}^{i,k} + z_{n'}^{i,k}.\end{aligned} \tag{18}$$

The derivative with respect to the parameters of the likelihood function is straightforward, and has two terms: the derivative of the prior and the derivative of the log likelihood,

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{\theta}_y} = -\frac{\partial \log p(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_y} - \sum_{n=1}^{N}\frac{\partial \log p(y_n|\mu_n)}{\partial \boldsymbol{\theta}_y}. \tag{19}$$

Similarly, the derivate with respect to the parameters of the warp functions has two terms: the derivative of the prior and the derivative of the log likelihood, where we use the chain rule,

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial \boldsymbol{\theta}_h} = & -\frac{\partial \log p(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_h} - \sum_{n=1}^{N}\frac{\partial \log p(y_n|\mu_n)}{\partial \mu_n}\times \\ & \sum_{k=1}^{K}\sum_{i=1}^{I}\left(\prod_{i'\neq i} h_n^{i',k}\right)\frac{\partial h_n^{i,k}}{\partial \boldsymbol{\theta}_h}.\end{aligned} \tag{20}$$

The derivative with respect to the parameters of the covariance function is a bit more involved, since it requires the computation of the derivative of a Cholesky decomposition. We begin with the derivative of the log posterior with respect to the Cholesky decomposition itself

$$\frac{\partial \mathcal{L}}{\partial C_{n',n}^{i,k}} = \frac{\partial \log p(y_n|\mu_n)}{\partial \mu_n}\left(\prod_{i'\neq i} h_n^{i',k}\right)\frac{\partial h_n^{i,k}}{\partial g_n^{i,k}} z_{n'}^{i,k}. \tag{21}$$

Using this, we can compute the backward derivative (Smith, 1995) of the Cholesky decomposition, $F_{n,n'}^{i,k}$, and finally the desired derivative can be evaluated as

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{\theta}_c} = \sum_{n=1}^{N}\sum_{n'=1}^{N} F_{n,n'}^{i,k}\frac{\partial c_{n,n'}^{i,k}}{\partial \boldsymbol{\theta}_c}. \tag{22}$$

The computation of the derivative of the Cholesky decomposition is approximately as computationally expensive as computing a Cholesky decomposition which is the most expensive computation in the WGP method. For that reason, the use of the backwards derivative is attractive when we need to compute the derivative with respect to several parameters of a covariance function, since the backward derivate needs only be computed once.

4. Experiments

We evaluated the proposed model on a food science data set (Bro & Jakobsen, 2002) that consists of measurements of the color of fresh beef as it changes during storage under different conditions. The storage conditions are determined by the values of five independent variables summarized in Table 1. In a reduced factorial experimental design, measurements were taken at a subset of the possible combinations of values of the independent variables, such that the data can be represented as a five-dimensional tensor with 60% of the values missing. A detailed description of the experimental design and the data is given by Bro and Jakobsen (2002) who analyze the data using generalized multiplicative analysis of variance (GEMANOVA).

We note, that because of the factorial experimental design, the input data points lie on a regular grid which

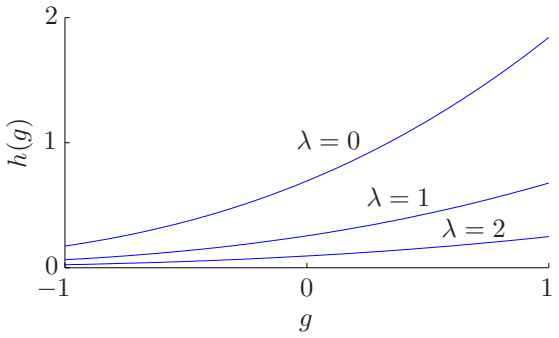

Figure 4. Warp function that transforms a standard Gaussian distribution into an exponential distribution with log scale λ.

is a requirement in order to analyze the data using tensor factorization methods, suitably tailored to handle missing data (Tomasi & Bro, 2002). The function factorization method does not require the data to lie on a grid, and its applicability thus extends beyond multiway array data.

4.1. Model choice

In our experiments we use a Gaussian likelihood,

$$p(y_n|\mu_n) = \frac{1}{\sqrt{2\pi \exp(v)}} \exp\left(-\frac{(y_n-\mu_n)^2}{2\exp(v)}\right), \quad (23)$$

parameterized by the log variance, $\boldsymbol{\theta}_y = \{v\}$, which ensures that the variance is always positive. This likelihood function allows us to directly compare our results with those of Bro and Jakobsen (2002) who use least squares PARAFAC and GEMANOVA models to analyze the same data.

The purpose of the warp functions is to map the Gaussian outputs of the GPs into another desired distribution. In the present data we expect the factorizing functions to be non-negative, because the output variable that measures the red color of the meat is inherently non-negative. We use a particular warp function, illustrated in Figure 4,

$$h^{i,k}(g) = -\exp(-\lambda^i)\log\left(\frac{1}{2} - \frac{1}{2}\mathrm{erf}\Big(\frac{g}{\sqrt{2}}\Big)\right), \quad (24)$$

parameterized by log scale parameters $\boldsymbol{\theta}_h = \{\lambda^i\}$. This warp function was suggested by Schmidt and Laurberg (2008) and has the property that it transforms a standard Gaussian variable to an exponentially distributed variable.

For the first four independent variables, shown in Table 1, we choose a Gaussian covariance function,

$$c^{i,k}(\boldsymbol{x}, \boldsymbol{x}') = \exp\left(-\exp(\ell^i)\ ||\boldsymbol{x}-\boldsymbol{x}'||^2\right), \quad (25)$$

parameterized by log length-scale parameters $\boldsymbol{\theta}_c = \{\ell^i\}$. We choose the covariance function for the fifth independent variable, the muscle number, as a delta function,

$$c^{i,k}(\boldsymbol{x}, \boldsymbol{x}') = \delta(\boldsymbol{x}-\boldsymbol{x}'), \quad (26)$$

such that this factor is effectively modeled as an exponentially distributed random variable.

For simplicity, we choose a non-informative flat (improper) prior over all parameters, $p(\boldsymbol{\theta}) \propto 1$, but we note that it is straightforward to choose proper priors over the parameters in a hierarchical fashion, and include any hyper-parameters in the HMCMC inference procedure.

4.2. Cross-validation

We divided the data set into ten subsets and performed ten-fold cross-validation. We included three different FF-WGP models in our experiments: $K = \{1, 2, 3\}$. In each run, we ran the HMCMC sampler for 5000 iterations using 20 leapfrog steps in each iteration and discarded the first half of the samples to allow the sampler to burn in. Plots of the samples of the parameters indicated that the sampling procedure had stabilized after a few hundred iterations. As an example, the log variance parameter, v, as a function of the iteration number is shown in Figure 5 for one of the HMCMC runs. We then computed the posterior mean estimate of the held-out data. For comparison we fitted $K = \{1, 2, 3\}$ component PARAFAC models using the N-way toolbox (Andersson & Bro, 2000). We also fitted a standard Gaussian process with Gaussian covariance by maximum likelihood and computed the posterior mean of the held-out data. For all of the models, we then computed the root mean squared error of the held out data.

4.3. Results

The results of the experiments are given in Table 2. A one-component PARAFAC model yielded a relatively high cross-validation error, whereas a two-component PARAFAC yields a relatively low error. A three-component PARAFAC leads to over-fitting, i.e., it fitted the training data better but gave a higher cross-validation error.

The two-component GEMANOVA model is suggested by (Bro & Jakobsen, 2002) as a good tradeoff between model complexity and predictive quality, and corresponds to a restricted PARAFAC model with some of

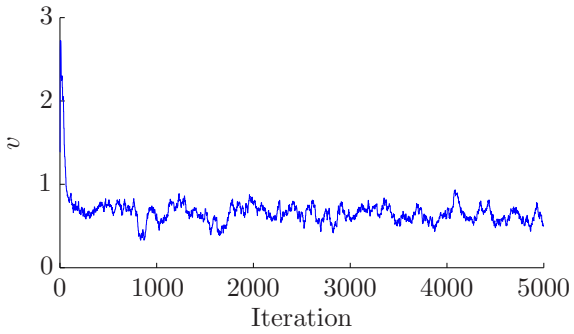

Figure 5. Log variance parameter, v, as a function of iteration number in one run of the HMCMC sampler for a two-component FF-WGP model.

Table 1. Independent variables that affect the color of beef as it changes during storage.

Independent variable	Unit	Values
Storage time	Days	$0, 3, 7, 8, 10$
Temperature	°C	$2, 5, 8$
Oxygen content in head-space	%	$40, 60, 80$
Exposure time to light	%	$0, 50, 100$
Muscle number		$1, 2, 3, 4, 5, 6$

the factors fixed. In the experiments of Bro and Jakobsen (2002), the GEMANOVA model yields a cross-validation error comparable to our results on a two-component PARAFAC model, but using fewer parameters.

Our result for the one-component FF-WGP model was similar to the one-component PARAFAC, which again suggests that one multiplicative component is not enough to adequately describe the data. Two- and three-component FF-WGP models yielded better predictions and had no problems with over-fitting, since all parameters are integrated out using MCMC. Gaussian process regression performed slightly worse than the factorization based models, possibly because the factorial structure of the problem is not exploited.

Figure 6 shows the learned factor pertaining to storage time in the one-component PARAFAC and FF-WGP model. In the PARAFAC model, the value of the factor can only be estimated at the five input positions at which data points were available. The function factorization approach on the other hand gives a full posterior distribution over a function, which can be evaluated anywhere. The two methods agree that

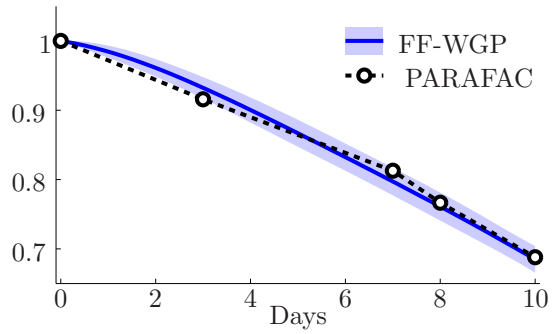

Figure 6. Relative effect of storage time: The factor pertaining to storage time in one-factor PARAFAC and FF-WGP models. The PARAFAC model estimates the value of the factor at the five points at which data was recorded. The FF-WGP outputs a posterior distribution over functions. The plot shows the posterior mean and standard deviation.

storage time negatively influences the color of beef in a near-linear manner.

5. Conclusions

We have presented a new approach to non-linear regression called function factorization. The method is based on approximating a function in a high-dimensional space as the sum of products of functions on subspaces. Using warped Gaussian processes as non-parametric priors, we have presented a Bayesian inference procedure based on Hamiltonian Markov chain Monte Carlo sampling.

Factorization based methods such as the PARAFAC model, that model data as a product of factors, can lead to intuitive and interpretable results when the factors have a physical meaning. Non-parametric Bayesian regression methods, such as the warped Gaussian process, make fewer assumptions on the structure of the data, but lack the same interpretability. The function factorization method presented here combines the idea of a factorized model with the flexibility of non-parametric Bayesian regression.

On a food science data set we have shown that the function factorization method using warped Gaussian process priors leads to superior performance in terms of cross-validated root mean squared error on a pre-

[2]Our results differ from Bro and Jakobsen (2002) who note that the model is degenerate but report a leave-one-out cross-validation RMSE of 1.50.

[3]Leave-one-out cross-validation result from (Bro & Jakobsen, 2002) with one multiplicative component plus one main component.

Table 2. Ten-fold cross-validation root mean squared error results on beef color dataset for parallel factor analysis (PARAFAC), generalized multiplicative analysis of variance (GEMANOVA), function factorization using warped Gaussian processes (FF-WGP), and Gaussian process regression (GPR).

Model	Components	Cross-val. RMSE
PARAFAC	1	2.95
	2	1.71[2]
	3	2.36
GEMANOVA	2	1.75[3]
FF-WGP	1	2.94
	2	1.50
	3	**1.45**
GPR	n/a	1.80

diction task compared with tensor factorization and Gaussian process regression. Also, we have demonstrated that the function factorization method provides full posterior distributions over the factorizing functions, which can improve on the interpretability of the model.

References

Adams, R., & Stegle, O. (2008). Gaussian process product models for nonparametric nonstationarity. *Machine Learning, International Conference on (ICML)* (pp. 1–8).

Andersson, C., & Bro, R. (2000). The n-way toolbox for matlab. *Chemometrics & Intelligent Laboratory Systems*, *52*, 1–4.

Bro, R., & Jakobsen, M. (2002). Exploring complex interactions in designed data using gemanova. color changes in fresh beef during storage. *Chemometrics, Journal of*, *16*, 294–304.

Duane, S., Kennedy, A. D., Pendleton, B. J., & Roweth, D. (1987). Hybrid monte carlo. *Physics Letters B*, *195*, 216–222.

Rasmussen, C. E., & Williams, C. K. I. (2006). *Gaussian processes for machine learning.* MIT Press.

Schmidt, M., & Laurberg, H. (2008). Nonnegative matrix factorization with gaussian process priors. *Computational Intelligence and Neuroscience*, *2008*. 10.1155/2008/361705.

Smith, S. P. (1995). Differentiation of the cholesky algorithm. *Computational and Graphical Statistics, Journal of*, *4*, 134–147.

Snelson, E., Rasmussen, C. E., & Ghahramani, Z. (2004). Warped gaussian processes. *Neural Information Processing Systems, Advances in (NIPS)* (pp. 337–344).

Tomasi, G., & Bro, R. (2002). Parafac and missing values. *Chemometrics and Intelligent Laboratory Systems*, *75*, 163–180. 10.1016/j.chemolab.2004.07.003.