Bayesian Backfitting

Aaron D'Souza ADSOUZA@USC.EDU
Sethu Vijayakumar SETHU@USC.EDU
Stefan Schaal SSCHAAL@USC.EDU
Computer Science & Neuroscience, University of Southern California, Los Angeles, CA 90089-2520, USA, and ATR Human Information Sciences Lab, 2-2 Hikaridai, Seika-cho, Soraku gun, 619-02 Kyoto, Japan.

Abstract

Whenever a graphical model contains connections from multiple nodes to a single node, statistical inference of model parameters may require the evaluation and possibly the inversion of the covariance matrix of all variables contributing to such a fan-in, particularly in the context of regression and classification. Thus, for high dimensional fan-ins, statistical inference can become computationally rather expensive and numerically brittle. In this paper, we propose an EM-based estimation method that statistically decouples the inputs by the introduction of hidden variables in each branch of the fan-in. As a result, the algorithm has a per-iteration complexity that is only *linear* in the order of the fan-in. Interestingly, the resulting algorithm can be interpreted as a probabilistic version of *backfitting*, and consequently, is ideally suited for applications of backfitting that require to cleanly propagate probabilities, as in Bayesian inference. We demonstrate the effectiveness of Bayesian Backfitting in dealing with extremely high-dimensional, underconstrained regression problems. In addition we highlight its connection to probabilistic partial least squares regression, and its extensions to nonlinear datasets through variational Bayesian mixture of experts regression, and nonparametric locally weighted learning.

1. Introduction

In many statistical learning problems one finds elements that can be characterized in terms of generalized linear models

$$y(\mathbf{x}) = \sum_{m=1}^{d} b_m f_m(\mathbf{x}; \theta_m) + \epsilon \qquad (1)$$

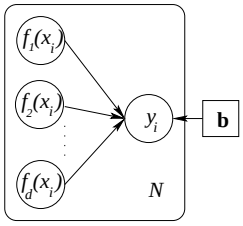


Figure 1. Graphical model for generalized linear regression.

i.e., multiple predictors $f_m(\mathbf{x}; \theta_m)$ (where $1 \leq m \leq d$) that are generated by an adjustable nonlinear transformation with parameters θ_m, and that are fed linearly to an output y by an inner product with a parameter vector $\mathbf{b} = [b_1, \ldots, b_d]^T$ and additive noise ϵ (Atkeson et al., 1997; Hastie & Tibshirani, 1990; Jordan & Jacobs, 1994; Poggio & Girosi, 1990). We can depict such models graphically as shown in Fig. 1. It is easy to see that the optimal estimate of the parameters b_m (in a least-squares or maximum likelihood sense) is $\mathbf{b} = \left(\mathbf{F}^T\mathbf{F}\right)^{-1}\mathbf{F}^T\mathbf{y}$ where $\mathbf{F}$ denotes a matrix whose columns contain the $f_{m,i}$ of all training data points $i = 1, \ldots, N$. With increasing number of fan-in variables, the inversion $\left(\mathbf{F}^T\mathbf{F}\right)^{-1}$ becomes computationally expensive and numerically brittle.

Various strategies are possible to combat these problems. First, robust methods of matrix inversion can be employed (Belsley et al., 1980), usually leading to at least $O(d^2)$ computational complexity. As these methods normally operate solely based on the input data, they may ignore low variance input dimensions that are important for the regression in favor of high variance inputs that are irrelevant — a typical example of such an algorithm is principal component regression (PCR) (Schaal et al., 1998). Alternatively, one can try to find a reduced cost solution by projecting the input space along specially chosen one-dimensional projection directions. For example, Partial Least Squares (PLS) regression (Wold, 1975) computes projections based on directions of high input-output correlation.

Even though this method performs extremely well in practice (Schaal et al., 1998), it misses a clean probabilistic interpretation (Frank & Friedman, 1993). Yet another framework for estimating generalized additive models of the form $y = \sum_{m=1}^{d} g_m(\mathbf{x})$ is *backfitting* (Hastie & Tibshirani, 1990), where the functions g_m absorb the parameters b_m in form of $g_m = b_m f_m$ (see Algorithm 1). Backfitting decomposes the statistical estimation problem into d individual estimation problems by creating "fake supervised targets" $\left(\mathbf{y} - \sum_{k \neq m} \mathbf{g}_k\right)$ for each function g_m. At the cost of an iterative procedure, this strategy effectively reduces the computational complexity of fan-ins, by decomposing it into d individual 1-dimensional estimation procedures, and allows easier numerical robustness control since no matrix inversion is required.

1: Init: $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N]^T$, $g_{m,i} = g_m(\mathbf{x}_i; \theta_m, b_m)$, $\mathbf{g}_m = [g_{m,1}, g_{m,2}, ..., g_{m,N}]^T$
2: **repeat**
3: $\quad \mathbf{g}_m \leftarrow \min_{b_m, \theta_m} \left\| \mathbf{g}_m - (\mathbf{y} - \sum_{k \neq m} \mathbf{g}_k) \right\|^2 \; \forall 1 \leq m \leq d$
4: **until** convergence

Algorithm 1: Backfitting

Using this notion of decoupling the problem along each branch of the fan-in, Sec. 2 motivates and describes a modification to the graphical model of Fig. 1, which performs a similar decoupling, and allows an EM-based estimation of its parameters. We highlight the connection of the resulting algorithm (called *Probabilistic Backfitting*) to traditional backfitting, and show that it indeed converges to an ordinary least squares (OLS) regression solution. Sec. 3 discusses two Bayesian extensions (Bayesian Backfitting-I and II) to the algorithm, and evaluates their efficacy on severely underconstrained regression problems. Sec. 4 introduces and evaluates extensions of both Probabilistic and Bayesian Backfitting for probabilistic partial least squares regression, mixture model estimation, and nonparametric locally weighted learning.

2. Backfitting and Graphical Models

Although backfitting provides a very general framework for estimating additive models, it has no probabilistic derivation and is thus hard to cleanly insert into statistical learning algorithms that emphasize the estimation of confidence, posterior distributions, and model complexity. A simple modification of the graphical model of Fig. 1, however, enables us to create the desired algorithmic decoupling of the predictor functions, and gives backfitting a probabilistic interpretation.

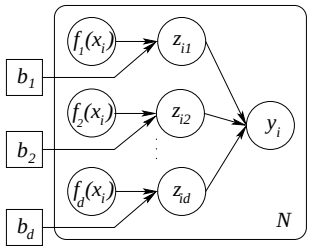


Figure 2. Graphical model with modified fan-in.

Consider the introduction of a hidden random variable z_{im} in the graphical model, as shown in Fig. 2. This variable is analogous to the output of the $g_m(x_{im})$ function of Algorithm 1. For the derivation of our algorithm, we assume that z_{im} and y_i are conditionally normally distributed:

$$y_i|\mathbf{z}_i \sim \mathcal{N}\left(\mathbf{1}^T\mathbf{z}_i, \psi_y\right) = \mathcal{N}\left(\sum_{m=1}^{d} z_{im}, \psi_y\right)$$
$$z_{im}|\mathbf{x}_i \sim \mathcal{N}\left(g_m(\mathbf{x}_i), \psi_{zm}\right) = \mathcal{N}\left(b_m f_m(\mathbf{x}_i), \psi_{zm}\right)$$

where $\mathbf{1} = [1, 1, \ldots, 1]^T$. Note that the regression coefficients b_m are now *before* the fan in. We will see that this results in the optimization of each coefficient requiring only local computation within its branch.

2.1. EM-based Parameter Estimation

Given the data set $\mathcal{D} = \{\mathbf{x}_i, y_i\}_{i=1}^N$, and the graphical model of Fig. 2, we wish to estimate the parameters b_m and (possibly) optimize the individual functions $f_m(\mathbf{x}; \theta_m)$ with respect to the parameters θ_m. This is easily formulated as an EM algorithm, which maximizes the *incomplete* log likelihood $\log p(\mathbf{y}|\mathbf{X})$:

$$\log p(\mathbf{y}|\mathbf{X}) = -\frac{N}{2}\log\psi_y - \frac{1}{2}\sum_{i=1}^{N}\left(y_i - \mathbf{b}^T\mathbf{f}(\mathbf{x}_i)\right)^2 + const \quad (2)$$

by maximizing the expected *complete* log likelihood $\langle \log p(\mathbf{y}, \mathbf{Z}|\mathbf{X}) \rangle$, where:

$$\log p(\mathbf{y}, \mathbf{Z}|\mathbf{X}) = -\frac{N}{2}\log\psi_y - \frac{1}{2\psi_y}\sum_{i=1}^{N}\left(y_i - \mathbf{1}^T\mathbf{z}_i\right)^2 - \sum_{m=1}^{d}\left[\frac{N}{2}\log\psi_{zm} + \frac{1}{2\psi_{zm}}\sum_{i=1}^{N}\left(z_{im} - b_m f_m(\mathbf{x}_i; \boldsymbol{\theta}_m)\right)^2\right] + const \quad (3)$$

As this maximization is solely based on standard manipulations of normal distributions, we omit derivations and just summarize the EM update equations for b_m and the noise variances ψ_y and ψ_{zm} as follows:

M-Step :

$$b_m = \frac{\sum_{i=1}^N \langle z_{im} \rangle f_m(\mathbf{x}_i)}{\sum_{i=1}^N f_m(\mathbf{x}_i)^2}$$

$$\psi_y = \frac{1}{N} \sum_{i=1}^N \left(y_i - \mathbf{1}^T \langle \mathbf{z}_i \rangle \right)^2 + \mathbf{1}^T \mathbf{\Sigma_z} \mathbf{1}$$

$$\psi_{zm} = \frac{1}{N} \sum_{i=1}^N \left(\langle z_{im} \rangle - b_m f_m(\mathbf{x}_i) \right)^2 + \sigma_{zm}^2$$

E-Step :

$$\mathbf{1}^T \mathbf{\Sigma_z} \mathbf{1} = \left(\sum_{m=1}^d \psi_{zm} \right) \left[1 - \frac{1}{s} \left(\sum_{m=1}^d \psi_{zm} \right) \right]$$

$$\sigma_{zm}^2 = \psi_{zm} \left(1 - \frac{1}{s} \psi_{zm} \right)$$

$$\langle z_{im} \rangle = b_m f_m(\mathbf{x}_i) + \frac{1}{s} \psi_{zm} \left(y_i - \mathbf{b}^T \mathbf{f}(\mathbf{x}_i) \right)$$

where we define $s = \psi_y + \sum_{m=1}^d \psi_{zm}$, and $\mathbf{\Sigma_z} = \mathrm{Cov}(\mathbf{z}|\mathbf{y}, \mathbf{X})$. In addition, the parameters θ_m of each function f_m can be updated by setting $\sum_{i=1}^N \left(\langle z_{im} \rangle - b_m f_m \left(\mathbf{x}_i; \theta_m \right) \right) \frac{\partial f_m(\mathbf{x}_i; \theta_m)}{\partial \theta_m} = 0$ and solving for θ_m. As this step depends on the particular choice of f_m, e.g., splines, kernel smoothers, parametric models, etc., we will not pursue it any further in this paper and just note that *any* statistical approximation mechanism could be used.

Two items in the above EM algorithm are of special interest. First, all equations are algorithmically $O(d)$ where d is the number of predictor functions f_m. Second, if we substitute the expression for $\langle z_{im} \rangle$ in the maximization equation for b_m we get the following update equation:

$$b_m^{(n+1)} = b_m^{(n)} + \frac{\psi_{zm}}{s} \frac{\sum_{i=1}^N \left(y_i - \sum_{k=1}^d b_k^{(n)} f_k(\mathbf{x}_i) \right) f_m(\mathbf{x}_i)}{\sum_{i=1}^N f_m(\mathbf{x}_i)^2} \tag{4}$$

Thus each EM cycle updates the mth regression coefficient by an amount proportional to the correlation between the mth predictor and the residual error. Hence the residual can be interpreted as forming a "fake target" for the mth branch of the fan-in. As the next section shows, this enables us to place this algorithm in the context of *backfitting*.

2.2. Interpreting the EM Solution as Probabilistic Backfitting

In the context of understanding Eq. (4) as Probabilistic Backfitting, we note that backfitting can be viewed as a formal Gauss-Seidel algorithm; an equivalence that becomes exact in the special case of linear models (Hastie & Tibshirani, 1990). For the linear system $\mathbf{F}^T \mathbf{F} \mathbf{b} = \mathbf{F}^T \mathbf{y}$, the Gauss-Seidel updates for the individual b_m are:

$$b_m = \frac{\sum_{i=1}^N \left(y_i - \sum_{k \neq m}^d b_k f_k(\mathbf{x}_i) \right) f_m(\mathbf{x}_i)}{\sum_{i=1}^N f_m(\mathbf{x}_i)^2} \tag{5}$$

A well-known extension to the Gauss-Seidel algorithm called *successive relaxation* adds a fraction $(1 - \omega)$ of b_m to the update and giving us:

$$b_m^{(n+1)} = (1 - \omega) b_m^{(n)} + \omega \frac{\sum_{i=1}^N \left(y_i - \sum_{k \neq m}^d b_k f_k(\mathbf{x}_i) \right) f_m(\mathbf{x}_i)}{\sum_{i=1}^N f_m(\mathbf{x}_i)^2} \tag{6}$$

which has improved convergence rates for *overrelaxation* ($1 < \omega < 2$), or improved stability for *underrelaxation* ($0 < \omega < 1$). For $\omega = 1$, the standard Gauss-Seidel/backfitting of Eq. (5) is recovered. Setting $\omega = \omega_m = \psi_{zm}/s$ in Eq. (6), it can be shown that (after some algebraic rearrangement,) we obtain exactly our EM update in Eq. (4), i.e., we indeed derive a probabilistic version of backfitting.

2.3. Convergence of Probabilistic Backfitting

While the incomplete likelihood in Eq. (2) has only a *global* maximum w.r.t. its parameters (corresponding to OLS), could the introduction of the hidden variables and additional parameters in Eq. (3) introduce *local* maxima in the likelihood landscape? Note that for examining convergence properties, we only focus on the estimation of the parameters $\boldsymbol{\phi} = [\mathbf{b}, \psi_{z1}, \ldots, \psi_{zd}, \psi_y]^T$, as the functions f_m cannot be treated in general without knowing their structure. We start with the assumption that we have reached a stationary point $\boldsymbol{\phi}^*$ in the EM algorithm, which implies:

$$\left. \frac{\partial \langle \log p(\mathbf{y}, \mathbf{Z}|\mathbf{X}; \boldsymbol{\phi}) \rangle}{\partial \boldsymbol{\phi}} \right|_{\boldsymbol{\phi} = \boldsymbol{\phi}^*} = \mathbf{0} \tag{7}$$

Using Jensen's inequality, it is easy to show that for an arbitrary distribution $Q(\mathbf{Z})$ over the hidden variables:

$$\log p(\mathbf{y}|\mathbf{X}; \boldsymbol{\phi}) \geq \langle \log p(\mathbf{y}, \mathbf{Z}|\mathbf{X}; \boldsymbol{\phi}) \rangle_{Q(\mathbf{Z})} + \mathcal{H}\left[Q(\mathbf{Z}) \right] = \mathcal{F}(Q, \boldsymbol{\phi}) \tag{8}$$

where $\mathcal{H}[\cdot]$ denotes entropy. EM alternately maximizes $\mathcal{F}$ w.r.t. Q (in the E-step) and $\boldsymbol{\phi}$ (in the M-step). Differentiating $\mathcal{F}(Q, \boldsymbol{\phi})$ w.r.t. $\boldsymbol{\phi}$ at the stationary point $\boldsymbol{\phi}^*$, and noting that $\mathcal{H}[\cdot]$ is independent of $\boldsymbol{\phi}$, gives:

$$\left.\frac{\partial \mathcal{F}(Q,\boldsymbol{\phi})}{\partial \boldsymbol{\phi}}\right|_{\boldsymbol{\phi}=\boldsymbol{\phi}^*} = \left.\frac{\partial \left\langle \log p(\mathbf{y},\mathbf{Z}|\mathbf{X};\boldsymbol{\phi})\right\rangle}{\partial \boldsymbol{\phi}}\right|_{\boldsymbol{\phi}=\boldsymbol{\phi}^*} = \mathbf{0} \quad (9)$$

Note however, that the preceding E-step sets $Q(\mathbf{Z})$ to the true posterior distribution $p(\mathbf{Z}|\mathbf{y},\mathbf{X};\boldsymbol{\phi}^*)$ which raises the lower bound in Eq. (8) to an equality, from which it follows that:

$$\left.\frac{\partial \log p(\mathbf{y}|\mathbf{X};\boldsymbol{\phi})}{\partial \boldsymbol{\phi}}\right|_{\boldsymbol{\phi}=\boldsymbol{\phi}^*} = \left.\frac{\partial \mathcal{F}(Q,\boldsymbol{\phi})}{\partial \boldsymbol{\phi}}\right|_{\boldsymbol{\phi}=\boldsymbol{\phi}^*} = 0 \quad (10)$$

i.e. we have reached a maximum in the *incomplete* likelihood as well. Given that the *incomplete* log likelihood $\log p(\mathbf{y}|\mathbf{X};\boldsymbol{\phi})$ in Eq. (2) has *only* a global maximum (i.e., the OLS solution), reaching the stationary point of Eq. (7) in our EM algorithm for Probabilistic Backfitting must correspond to finding the OLS solution. Therefore, Probabilistic Backfitting is indeed performing true linear regression with a global optimum.

3. Bayesian Backfitting

Having a probabilistic interpretation of backfitting, allows us to use a Bayesian framework to regularize its OLS solution against overfitting. We achieve this by placing a prior distribution over the regression coefficients $\mathbf{b}$. As the following two sections demonstrate, our choice of prior structure results in two different, yet important forms of regularization.

3.1. Regularization over Input Dimensionality (Bayesian Backfitting-I)

We place a Gaussian prior over *each* element b_m of $\mathbf{b}$, with mean 0 and precision α_m, and a Gamma prior over each precision variable α_m:

$$\begin{aligned} p(\mathbf{b}|\boldsymbol{\alpha}) &= \prod_{m=1}^{d} \left(\frac{\alpha_m}{2\pi}\right)^{1/2} \exp\left\{-\frac{\alpha_m}{2} b_m^2\right\} \\ p(\boldsymbol{\alpha}) &= \prod_{m=1}^{d} \frac{b_\alpha^{a_\alpha}}{\Gamma(a_\alpha)} \alpha_m^{(a_\alpha-1)} \exp\left(-b_\alpha \alpha_m\right) \end{aligned} \quad (11)$$

Using a factorial variational approximation (Ghahramani & Beal, 2000), we can derive the modified update equations for the variables in the model. Due to space constraints, we omit the derivation, and only summarize the update equations for the mean of $\mathbf{b}$ and $\boldsymbol{\alpha}$:

$$\begin{aligned} \langle b_m \rangle^{(n+1)} &= \left(\frac{\sum_{i=1}^N f_m(\mathbf{x}_i)^2}{\sum_{i=1}^N f_m(\mathbf{x}_i)^2 + \psi_{zm}\langle \alpha_m \rangle}\right) \langle b_m \rangle^{(n)} \\ &+ \frac{\psi_{zm} \sum_{i=1}^N \left(y_i - \langle \mathbf{b} \rangle^{(n)^T} \mathbf{f}(\mathbf{x}_i)\right) f_m(\mathbf{x}_i)}{s\left(\sum_{i=1}^N f_m(\mathbf{x}_i)^2 + \psi_{zm}\langle \alpha_m \rangle\right)} \end{aligned} \quad (12)$$

$$\langle \alpha_m \rangle = \frac{2a_\alpha + 1}{2b_\alpha + \langle b_m^2 \rangle} \quad (13)$$

This formulation results in a regression solution which regularizes over the *number of retained input dimensions* in the final regression vector, similar to Automatic Relevance Determination (ARD) (Neal, 1994).

3.2. Regularization over Regression Vector Length (Bayesian Backfitting-II)

Alternatively, we may choose to use a *single* precision variable over the entire regression vector $\mathbf{b}$:

$$\begin{aligned} p(\mathbf{b}|\alpha) &= \left(\frac{\alpha}{2\pi}\right)^{d/2} \exp\left\{-\frac{\alpha}{2}\mathbf{b}^T\mathbf{b}\right\} \\ p(\alpha) &= \frac{b_\alpha^{a_\alpha}}{\Gamma(a_\alpha)} \alpha^{(a_\alpha-1)} \exp\left(-b_\alpha \alpha\right) \end{aligned} \quad (14)$$

which results in following update equations:

$$\begin{aligned} \langle b_m \rangle^{(n+1)} &= \left(\frac{\sum_{i=1}^N f_m(\mathbf{x}_i)^2}{\sum_{i=1}^N f_m(\mathbf{x}_i)^2 + \psi_{zm}\langle \alpha \rangle}\right) \langle b_m \rangle^{(n)} \\ &+ \frac{\psi_{zm} \sum_{i=1}^N \left(y_i - \langle \mathbf{b} \rangle^{(n)^T} \mathbf{f}(\mathbf{x}_i)\right) f_m(\mathbf{x}_i)}{s\left(\sum_{i=1}^N f_m(\mathbf{x}_i)^2 + \psi_{zm}\langle \alpha \rangle\right)} \end{aligned} \quad (15)$$

$$\langle \alpha \rangle = \frac{2a_\alpha + d}{2b_\alpha + \langle \|\mathbf{b}\|^2 \rangle} \quad (16)$$

This formulation results in a regression solution which regularizes over the *length of the regression vector* $\mathbf{b}$. This is similar in operation to *shrinkage* methods such as Ridge Regression which favor minimum-norm solutions from the solution space.

Comparing Eqs. (12) and (15) with Eq. (4), we see that in the absence of any correlation between the mth input and the residual error, the first term in Eqs. (12) and (15) causes the expected value $\langle b_m \rangle$ to decay with each iteration, thus providing the required regularization. It is important to note that the Bayesian extensions do not increase the computational complexity of the algorithm beyond $O(d)$.

3.3. Comparison with Ridge Regression

We evaluated both versions of Bayesian Backfitting against Ridge Regression on two sets of problems. In each set we considered the special case of $f_m(\mathbf{x}_i) = x_{im}$, i.e., an OLS problem.

3.3.1. Experiment Set I

The first set of experiments was designed to test the performance of these algorithms on datasets which had

a large number of correlated (redundant) inputs. Each dataset consisted of N points of 50-dimensional input data and 1-dimensional output data, generated in the following manner:

1. Generate N normally distributed vectors of dimensionality $d_i \leq 50$.
2. Generate output data using a random d_i-dimensional regression vector ($b_m \sim \mathcal{N}(0, 100)$). Add Gaussian noise to output data (signal-to-noise SNR=50).
3. Pad the input data with $50 - d_i$ zero dimensions, and rotate the padded input by a random 50-dimensional rotation matrix. This spreads the d_i input dimensions across the entire 50-dimensional space resulting in highly correlated inputs.

100 datasets from each of 3 categories of (d_i, N) were generated. The first, (50,1000), was a fully constrained regression problem. The second, (5,1000), was ill-conditioned with an intrinsic input dimensionality of just 5. The third, (5,20), was severely underconstrained with just 20 data points from an intrinsic dimensionality of 5, rotated into a 50-dimensional space. For each dataset, a noise-free test set of 1000 points was generated with the same parameters. In all experiments, the ridge parameter λ in Ridge Regression was optimized using a line search on the mean squared error obtained on these test sets (not used in training). The Bayesian Backfitting methods had no access to the test set until evaluation.

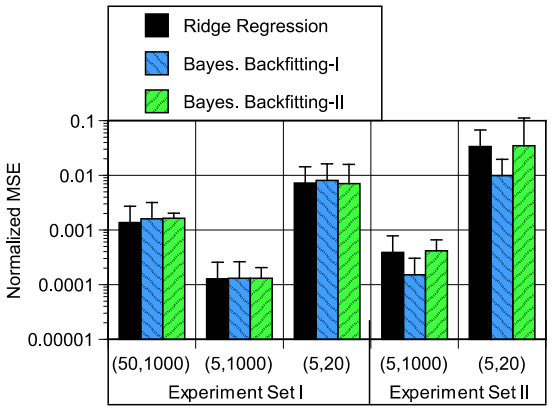


Figure 3. Normalized MSE results on test set

Fig. 3 summarizes the results. The normalized MSE of the three algorithms is statistically indistinguishable on all three categories of problems. It should be noted that for the second and third dataset categories — (5,1000) and (5,20) — Bayesian Backfitting-I correctly retained only 5 predictor dimensions for each dataset, indicating a correct regularization of the number of input dimensions required to predict the output.

Exp. I category	mean # of iterations	
	B. Backfitting-I	B. Backfitting-II
(50,1000)	242.98	165.53
(5,1000)	173.30	103.72
(5,20)	93.69	43.11

Table 1. Mean # of iterations reqd. for 95% ridge regression accuracy

We also examined the number of $O(d)$ (where $d = 50$) Bayesian Backfitting cycles required to reach 95% of the accuracy of ridge regression. Comparing the typical $O(d^3)$ computational complexity of matrix inversion — $d^3 = 50^2 d = 2500d$ in our tests — with the results in Table 1, it is apparent that Bayesian Backfitting is well suited to applications that quickly require a good approximate solution, but 100% accuracy is not of immediate importance. Such is the case in many large scale probabilistic inference systems where many parameters are optimized simultaneously, such that a complete maximization for individual parameter estimation is often useless due to other non-converged variables. It should also be noted that the expensive line search of Ridge Regression to find the appropriate value for the regularizing constant λ makes it particularly expensive for high dimensional datasets.

3.3.2. Experiment Set II

The second set of experiments tested the algorithms on datasets which had a large number of irrelevent inputs. The data generation process is identical to that in Sec. 3.3.1, except for step 3. Instead of padding with zero dimensions, the input data is padded with an uncorrelated Gaussian signal at 1/10th the standard deviation of the relevant inputs. Also, the rotation step is skipped, since we wish to determine if the algorithms can ignore irrelevant dimensions, and rotating the relevant inputs would spread them over all dimensions.

Datasets from 2 categories of (d_i, N) are generated. The first (5,1000), contains 1000 data points with 5 relevant inputs padded with 45 noise dimensions. The second (5,20) is underconstrained and has only 20 datapoints. As shown in the last two columns of Fig. 3, Bayesian Backfitting-I performs significantly better than the other two methods due to its ability to regularize each input dimension independently and eliminate irrelevant inputs.

3.4. *Very* High Dimensional Regression

To demonstrate the capability of Bayesian Backfitting to operate robustly in very high-dimensional spaces, we created a training set of 1000 data points in a man-

ner similar to that in Sec. 3.3.1 for category (5,1000), only in this case, the regression vector is a known simple transformation $[1, 2, 3, 4, 5]^T$, and the input data was padded with 99,995 alternating zero and irrelevant dimensions for a *total of 100,000 dimensions*. The first 100 dimensions were then rotated by a random rotation matrix.

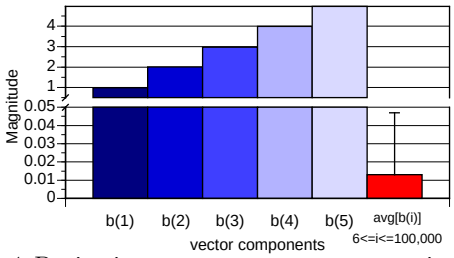


Figure 4. Projection vector component magnitudes after inverse rotation

After training with Bayesian Backfitting-II, the regression solution $\mathbf{b}$ was rotated back by the inverse of the transformation used in the creation of the dataset. Fig. 4 shows the magnitudes of the vector components. As shown, the first 5 components have correctly estimated the $[1, 2, 3, 4, 5]^T$ transformation used to generate the output. The projection components corresponding to the remaining 99,995 dimensions converged to extremely small values. The mean of these values (along with their standard deviation) are plotted as the 6th bar in the figure. The normalized mean squared error after 10,000 updates of Bayesian Backfitting-II was consistently about 0.0001 for 10 runs of the algorithm measured on a noiseless test set.

3.5. Performance on a Chemometrics Dataset

As the previous section shows, backfitting performs extremely well on severely underconstrained, high-dimensional datasets. One typical example is the estimation of concentrations of substances based on their near infra-red (NIR) absorption spectra (Frank & Friedman, 1993). The input variables are the spectral responses at various wavelengths (typically several hundred in number), while the number of calibration solutions (data points) is much smaller (< 100).

The following dataset is a calibration set that aims to estimate the moisture content in a sample of corn. The input data is 700 dimensional while the number of samples for which readings were taken is merely 80.

Table 2 shows the leave-one-out cross validation errors obtained by running Ridge Regression, Bayesian Backfitting, and PLS (with 26 projection directions) on this data set. The error measure for Bayesian Backfitting was obtained by running the algorithms on each cross validation training set (created by leaving out a single data point) until convergence (until the relative change in the functional $F(Q, \boldsymbol{\theta})$ was less than 1e-5) before testing on the left out data point. The Ridge Regression parameters and the number of projection directions for PLS were determined by leave-one-out cross validation. The motivation for obtaining the cross validation errors was to determine whether or not this severly underconstrained problem would be overfit by Bayesian Backfitting. From the results it is clear that even though we are guaranteed to converge to an OLS solution, Bayesian Backfitting is a numerically robust formulation that largely resists overfitting the data in cases where the model is overparameterized (when $N \ll d$), as expected from a Bayesian method.

Algorithm	nMSE (CV)
Ridge Regression	3.009e-4
Bayesian Backfitting-I	3.015e-4
Bayesian Backfitting-II	3.230e-4
PLS-26	2.007e-4

Table 2. Summary of leave-one-out cross validation errors on the chemometrics data set.

4. Extensions of Bayesian Backfitting

4.1. Probabilistic Partial Least Squares

One of the most straightforward applications of Bayesian Backfitting is in high-dimensional linear regression, characterized by $f_m(\mathbf{x}) = x_m$ and $m \gg 1$. Partial Least Squares regression (PLS) (Wold, 1975) has found widespread application in this domain. The essence of PLS is to find a projection direction $\mathbf{u}$ in the input space of the regression that has high correlation with the outputs, i.e., $\mathbf{u} = \mathbf{X}^T\mathbf{y}$, and, after projecting all input data onto this projection, i.e., $\mathbf{s} = \mathbf{X}\mathbf{u}$, to perform a univariate regression between $\mathbf{s}$ and $\mathbf{y}$. Multiple iterations using orthogonal projections, and the residual error of the previous univariate fit as a target variable may be required. We note that the iterative determination of the parameters b_m in Bayesian Backfitting corresponds to estimating a single optimal projection direction, not just a heuristic one, and the way backfitting estimates this projection direction avoids the need for an additional univariate regression along the projection. From this perspective, Bayesian Backfitting can be seen as a probabilistic version of PLS, although the it is not "partial" anymore but rather complete, since it recovers the correct projection direction with a single regression vector.

4.2. Growing Variational Mixture of Experts

Consider a mixture-of-experts regression, where the prior probability of each expert's contribution to

the regression is modeled by a Gaussian gating network (Xu et al., 1995). For high-dimensional input spaces, updating the regression parameters by ordinary weighted least squares regression quickly becomes a computational bottleneck, and is numerically brittle. This situation can be remedied by using the Bayesian Backfitting algorithm in each of the experts. Exploiting the full power of probabilistic inference, we will also automatically select the number of mixture components K by growing K based on Bayesian model comparison.

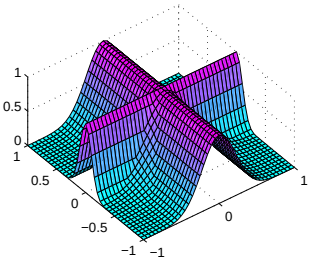
Within the locality of each mixture component, we hypothesize the existence of two "child" components. We place a joint Normal-Wishart prior over the mean $\boldsymbol{\mu}$ and precision $\mathbf{B}$ of each child-component's Gaussian gating network. Each data point within the locality of the parent component is assigned a binomial variable ξ_i, which decides which child-component the point belongs to. The probability parameter π of the binomial distribution is given a Beta distribution prior.

$$\begin{aligned}
\mathbf{x}_i|\xi_{im}=1,\boldsymbol{\mu}_m,\mathbf{B}_m &\sim \mathcal{N}\left(\boldsymbol{\mu}_m,\mathbf{B}_m^{-1}\right)\\
\boldsymbol{\mu}_m|\mathbf{B}_m,\alpha_0 &\sim \mathcal{N}\left(\mathbf{0},\mathbf{B}_m^{-1}/\alpha_0\right)\\
\mathbf{B}_m|\nu,\mathbf{S} &\sim \text{Wishart}_\nu\left(\mathbf{B}_m;\mathbf{S}\right)\\
p(\xi_{i1}=1)=\pi;\quad p(\xi_{i2}=1) &= (1-\pi)\\
\pi &\sim \text{Beta}\left(\pi;u_1,u_2\right)
\end{aligned}$$

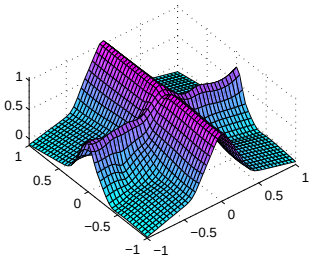
These distributions are in addition to the Bayesian regularizing priors of Eqs. (11) and (14) over each of the backfitting experts. Computing the true evidence requires integration over all the model parameters — an analytically infeasable task. Using a factorial variational approximation, we can derive an EM-like algorithm to maximize a lower bound to the true model evidence. During this maximization process, if both child-components survive in maximizing the local evidence, they replace the parent model, effectively "growing" the overall number of mixture components by one. The algorithm terminates when the local evidence within each component supports one model instead of two.

Between the splitting of individual components, we perform a similar variational evidence maximization on the collection of all K components. This allows components to die away if a split based on local evidence caused the overall model evidence to decrease. This results in a method similar to the one used by Ghahramani and Beal in the context of variational factor analyzers (Ghahramani & Beal, 2000), except that their method for selecting a component to split was slightly heuristic, while our evidence based approach guarantees that we do not locally overfit.

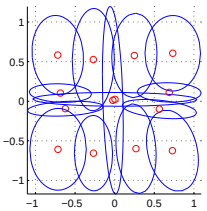
We tested this algorithm on 20 "cross" datasets



(a) Original function



(b) Estimated function (avg. NMSE=0.0525)



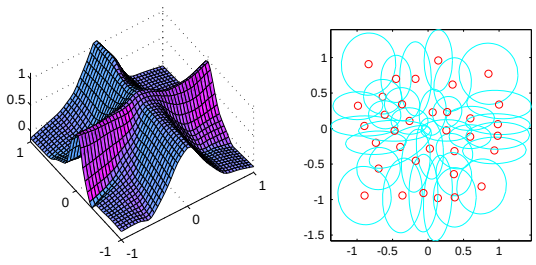
(c) Component locations

Figure 5. Growing mixtures of Bayesian Backfitters

shown in Fig. 5(a). Each dataset consisted of 2000 2-dimensional input data points (x_1, x_2), randomly generated in the interval [-1,1]. The output was generated using the function $y = \max\left[e^{-10x_1^2}, e^{-50x_2^2}, e^{-5(x_1^2+x_2^2)}\right]$ along with additive Gaussian noise (SNR=50). Subsequently, the input was augmented with 18 zero dimensions, and rotated by a random rotation matrix into a 20-dimensional space. The algorithm was initialized with 5 mixture components, which grew to an average of 14.95 on convergence. Figs. 5(b) and 5(c) show the learned function and the typical final configuration of the model components after rotating back to a 2-dimensional input space. The configuration of experts is intuitively reasonable, demonstrating the feasibility of inserting Bayesian Backfitting into nonlinear learning systems.

4.3. Probabilistic Backfitting in Locally Weighted Regression

Bayesian Backfitting is also applicable in a non-parametric locally weighted learning (LWL) scenario for function approximation with locally linear models (Vijayakumar & Schaal, 2000). Here, each training data point is assigned a weight $w_{k,i} = exp(-0.5(\mathbf{x}_i - \mathbf{c}_k)^T\mathbf{D}_k(\mathbf{x}_i - \mathbf{c}_k))$ such that the locally linear model is obtained from a weighted regression analysis $\beta_k = (\mathbf{X}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}\mathbf{W}\mathbf{y}$. The computationally expensive linear regression model in LWL can be replaced by Bayesian Backfitting by employing a



(a) Estimated function

(b) Component locations

Figure 6. Function estimation with LWL

heteroscedastic backfitting model, in which $z_{im}|\mathbf{x}_i \sim \mathcal{N}\left(b_m f_m(\mathbf{x}_i), \psi_{zm}/w_i\right)$ and $y_i|\mathbf{z}_i \sim \mathcal{N}\left(\mathbf{1}^T\mathbf{z}_i, \psi_y/w_i\right)$. From a Bayesian standpoint, we can interpret the weighting to signify that a datapoint with very low weight has correspondingly high noise variance, such that it has little influence on the model estimation. We can derive an EM backfitting algorithm for this model in much the same manner as that done in Sec. 2.1. In addition, the decoupling of the input dimensions allows us to perform efficient adjustment of the distance metric $\mathbf{D}_k$ for the Gaussian kernels along each of the input dimensions using stochastic leave-one-out cross-validation (Vijayakumar & Schaal, 2000). Fig. 6 demonstrates function approximation results and automatic kernel shaping for the same datasets as in the previous section. Again, very efficient computational complexity and robust numerical performance was achieved due to Bayesian Backfitting.

5. Discussion

Starting with the idea that, with the introduction of hidden variables, we can reduce the computational complexity of statistical inference in the fan-ins of graphical models, we have developed a probablilistic version of *backfitting* — a very general framework for estimating generalized additive models. Although the Probabilistic and Bayesian Backfitting methods are iterative, our experimental evaluations show that in practice, they require rather few iterations to achieve approximately accurate solutions — especially in ill-conditioned, underconstrained datasets. This makes them well suited to be inserted into various statistical learning frameworks. While the original backfitting algorithm within the framework of nonparametric statistics did not make any assumptions about the distribution of the variables in the model, Bayesian Backfitting required to be explicit about distributions, chosen to be Gaussian as in many other probabilistic derivations. Ongoing evaluations in application areas of robot control and human-computer interaction will provide insights into the potential impacts of such assumptions.

Acknowledgements

This work was made possible by support from the US National Science Foundation (Awards 9710312 and 0082995), an AFOSR grant to Barron Associates, the ATR Human Information Science Laboratories, and the ERATO Kawato Dynamic Brain Project funded by the Japan Science and Technology Corporation.

References

Atkeson, C. G., Moore, A. W., & Schaal, S. (1997). Locally weighted learning. *Artificial Intelligence Review*, *11*, 11–73.

Belsley, D. A., Kuh, E., & Welsch, R. E. (1980). *Regression diagnostics: Identifying influential data and sources of collinearity*. New York: Wiley.

Frank, I. E., & Friedman, J. H. (1993). A statistical view of some chemometrics regression tools. *Technometrics*, *35*, 109–135.

Ghahramani, Z., & Beal, M. J. (2000). Variational inference for Bayesian mixtures of factor analysers. *Advances in Neural Information Processing Systems* (pp. 509–514). MIT Press.

Hastie, T. J., & Tibshirani, R. J. (1990). *Generalized additive models*. No. 43 in Monographs on Statistics and Applied Probability. Chapman & Hall.

Jordan, M. I., & Jacobs, R. A. (1994). Hierarchical mixtures of experts and the EM algorithm. *Neural Computation*, *6*, 181–214.

Neal, R. M. (1994). *Bayesian learning for neural networks*. Doctoral dissertation, Dept. of Computer Science, University of Toronto.

Poggio, R., & Girosi, F. (1990). Regularization algorithms for learning that are equivalent to multilayer networks. *Science*, *247*, 213–225.

Schaal, S., Vijayakumar, S., & Atkeson, C. G. (1998). Local dimensionality reduction. *Advances in Neural Information Processing Systems 10* (pp. 633–639). Cambridge, MA: MIT Press.

Vijayakumar, S., & Schaal, S. (2000). An $O(n)$ algorithm for incremental real time learning in high dimensional space. *Proceedings of the Seventeenth International Conference on Machine Learning (ICML2000)* (pp. 1079–1086). Stanford, CA.

Wold, H. (1975). Soft modeling by latent variables: The nonlinear iterative partial least squares approach. In J. Gani (Ed.), *Perspectives in probability and statistics, papers in honour of M. S. Bartlett*, 520–540. London: Academic Press.

Xu, L., Jordan, M. I., & Hinton, G. E. (1995). An alternative model for mixtures of experts. *Advances in Neural Information Processing Systems* (pp. 633–640). Cambridge, MA: MIT Press.