A Bayesian Approach to Empirical Local Linearization For Robotics

Jo-Anne Ting*, Aaron D'Souza†, Sethu Vijayakumar‡ and Stefan Schaal*§

*Computer Science, University of Southern California, Los Angeles, CA 90089, USA
†Google, Inc. Mountain View, CA 94043, USA
‡University of Edinburgh, Edinburgh, EH9 3JZ, UK
§ATR Computational Neuroscience Labs, Kyoto 619-0288, Japan
Email: joanneti@usc.edu, adsouza@google.com, sethu.vijayakumar@ed.ac.uk, sschaal@usc.edu

***Abstract*— Local linearizations are ubiquitous in the control of robotic systems. Analytical methods, if available, can be used to obtain the linearization, but in complex robotics systems where the dynamics and kinematics are often not faithfully obtainable, empirical linearization may be preferable. In this case, it is important to only use data for the local linearization that lies within a "reasonable" linear regime of the system, which can be defined from the Hessian at the point of the linearization—a quantity that is not available without an analytical model. We introduce a Bayesian approach to solve statistically what constitutes a "reasonable" local regime. We approach this problem in the context local linear regression. In contrast to previous locally linear methods, we avoid cross-validation or complex statistical hypothesis testing techniques to find the appropriate local regime. Instead, we treat the parameters of the local regime probabilistically and use approximate Bayesian inference for their estimation. The approach results in an analytical set of iterative update equations that are easily implemented on real robotics systems for real-time applications. As in other locally weighted regressions, our algorithm also lends itself to complete nonlinear function approximation for learning empirical internal models. We sketch the derivation of our Bayesian method and provide evaluations on synthetic data and actual robot data where the analytical linearization was known.**

I. INTRODUCTION

Locally linear methods have been shown to be useful for robot control, especially in the context of learning of internal models of high-dimensional robotic systems for feedforward control and for learning local linearizations for the purpose of optimal control and reinforcement learning [1]–[3]. One of the key problems of these methods is finding the right size of the local region for a linearization, as in locally weighted regression. Existing methods, such as supersmoothing [4], locally weighted projection regression (LWPR) [3] and those developed by Fan et al. [5], [6], to name a few, use either cross-validation techniques or complex statistical hypothesis testing methods and require significant manual parameter tuning by the user for good and stable performance. Some are only applicable for very low-dimensional data.

In this paper, we introduce a Bayesian formulation of spatially local adaptive kernels for locally weighted regression, which automatically determines the local regime for linearization from Bayesian statistics. Our new approach treats all open parameters probabilistically and uses variational approximations [7] to produce an analytically tractable Bayesian algorithm. In particular, we use the Bernoulli distribution to model the weights generated by the locally adaptive weighted kernel—a key detail that allows us to learn the appropriate local regime for linearization. We evaluate our algorithm on synthetic data sets to demonstrate its competitiveness with other state-of-the-art nonlinear function approximation methods like Gaussian process regression (GPR) [8]. We also evaluate the algorithm on a direct kinematics problem for a 7 degree-of-freedom (DOF) robotic arm for the purpose of estimating the Jacobian matrix, showing that it can produce results that are comparable to the analytical Jacobian. The main purpose of this paper is to introduce the new Bayesian treatment of local linearization and to demonstrate its functionality. Future work will address the application of this method in problems of reinforcement learning and nonlinear robot control on humanoid robots.

II. LOCALLY WEIGHTED REGRESSION

For nonparametric locally weighted regression [1], let us assume we have a data set of N training samples, $D = \{\mathbf{x}_i, y_i\}_{i=1}^N$, drawn from a nonlinear function f: $y_i = f(\mathbf{x}_i) + \epsilon$ (where $\mathbf{x}_i \in \Re^{d\times 1}$ is the input vector, y_i is the scalar output, ϵ is additive mean-zero Gaussian noise, and d is the number of input dimensions). If we consider a local regime of the input space around a query point $\mathbf{x}_q \in \Re^{d\times 1}$ and choose the locality appropriately, we can use a low order polynomial to fit this local subset of data samples in a fast and efficient manner. Consider the local model:

$$y_i = \mathbf{b}^T\mathbf{x}_i + \epsilon \quad (1)$$

where $\mathbf{b} \in \Re^{d\times 1}$ is the regression vector, i.e., the slope of the tangent, and $\epsilon \sim \text{Normal}(0, \sigma^2)$ is output noise with a variance of σ^2. Our goal is to approximate a locally linear model at a query point $\mathbf{x}_q$ in order to make the prediction y_q, where $y_q = \mathbf{b}^T\mathbf{x}_q$.

The measure of locality for each data sample i, $\{\mathbf{x}_i, y_i\}$, is computed from a weighting kernel K, $w_i = K(\mathbf{x}_i, \mathbf{x}_q, \mathbf{h})$, such that there is a scalar weight w_i associated with each sample i, according to the sample's Euclidean distance in input space from the query input point $\mathbf{x}_q$. $\mathbf{h} \in \Re^+$ is a d by 1 vector that represents how wide the weighting kernel is and dictates the quality of fit of the locally linear model. $\mathbf{h}$ is a form of distance metric, a measure that determines the size and shape of the weighting kernel. It is the size of the local regime in input space to be linearized. A smaller $\mathbf{h}$ indicates that the weighting kernel is broader. We assume that the further a data sample is from $\mathbf{x}_q$ in input space, the more it should be downweighted. A popular choice of the function K is the Gaussian kernel $w_i = \exp\left\{-0.5\left(\mathbf{x}_i - \mathbf{x}_q\right)^T \mathbf{H}\left(\mathbf{x}_i - \mathbf{x}_q\right)\right\}$, where $\mathbf{H}$ is a positive semi-definite diagonal matrix with $\mathbf{h}$ on its diagonal.

The distance metric of the kernel, parameterized by $\mathbf{h}$, must be chosen carefully. If $\mathbf{h}$ is too large, then we risk overfitting the data, i.e., fitting noise. If $\mathbf{h}$ is too small, we may oversmooth the data, i.e., not fitting enough structure in the data. In general, $\mathbf{h}$ is chosen as a function of the local curvature of $f(\mathbf{x})$ and of the data density around the query point $\mathbf{x}_q$. If we can find the right distance metric value, as a function of $\mathbf{x}_q$, nonlinear function approximation may be solved accurately and efficiently. Past work has involved use of cross-validation, statistical hypothesis testing or search to find this optimal distance metric value. However, these methods may be sensitive to initialization values (for gradient descent), require manual meta-parameter tuning or be quite computationally involved. Next, we propose a variational Bayesian algorithm that learns both $\mathbf{b}$ and $\mathbf{h}$ simultaneously in an Expectation-Maximization-like (EM) [9] framework.

III. Bayesian Locally Weighted Regression

A. Model

Given the local model in (1), we assume that the following prior distributions are used:

$$\begin{aligned} p(y_i|\mathbf{x}_i) &\sim \text{Normal}\left(\mathbf{b}^T\mathbf{x}_i, \sigma^2\right) \\ p(\mathbf{b}|\sigma^2) &\sim \text{Normal}\left(0, \sigma^2\boldsymbol{\Sigma}_{\mathbf{b}_0}\right) \\ p(\sigma^2) &\sim \text{Scaled-Inv-}\chi^2\left(n, \sigma_N^2\right) \end{aligned} \tag{2}$$

where $\boldsymbol{\Sigma}_{\mathbf{b}_0}$ is the prior covariance of $\mathbf{b}$, and n and σ_N^2 are parameters of the Scaled-inverse-χ^2 distribution (n is the number of degrees of freedom parameter and σ_N^2 is the scale parameter). A Scaled-inverse-χ^2 distribution was selected for σ^2 since it is the conjugate prior for the variance parameter of a Gaussian distribution. Fig. 1 depicts the graphical model proposed, compactly describing the full multi-dimensional system in plate notation. The longer vertical plate shows that there are N samples of observed $\{\mathbf{x}_i, y_i\}$ data, while the wider horizontal plate shows d duplications of random variables for the d input dimensions of the data.

We assume that each data sample i, $\{\mathbf{x}_i, y_i\}$, in D has a **scalar indicator-like** weight, $0 \leq w_i \leq 1$, associated with it. If $w_i = 1$, then the data sample is fully included in the local linear regression problem. Otherwise, if $w_i = 0$, then the data sample is excluded from the regression. In contrast to Sec. II, where the weight for each data sample w_i is an explicit function K, we treat the weights probabilistically, defining the weight w_i to be $w_i = \prod_{m=1}^{d} \langle w_{im} \rangle$. w_{im} is a random variable representing the weight of data sample i in the mth input dimension:

$$p(w_{im}) \sim \text{Bernoulli}\left(\frac{1}{1 + |x_{im} - x_{qm}|^r h_m}\right) \tag{3}$$

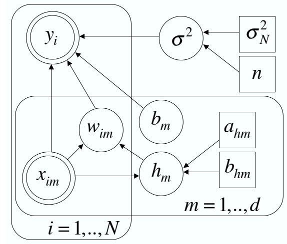


Fig. 1. Graphical model of Bayesian locally weighted regression in plate notation. Random variables are in circles, observed random variables are in double circles, and point-estimated parameters are in squares.

where x_{im} is the mth coefficient of the sample input $\mathbf{x}_i$, x_{qm} is the mth coefficient of the query vector $\mathbf{x}_q$, and $r > 0$ is a scalar. The weight w_{im} is a function of the distance of the sample i from the query point x_{qm} in input space and the distance metric h_m, defined as:

$$p(h_m) \sim \text{Gamma}\left(a_{hm}, b_{hm}\right) \tag{4}$$

where a_{hm} and b_{hm} are parameters of the Gamma distribution[1]. Modeling h_m as a Gamma distribution ensures that the inferred width of the weighting kernel remains positive. r controls the curvature of the weighting kernel. For smaller values of r, the weighting kernel takes on a shape with a narrower peak, but longer tails. For our experiments, we use $r = 4$ and, with this initial curvature, learn the width/distance metric of each local weighting kernel.

B. Inference

We can treat the entire regression problem as an EM learning problem [7], [9]. Defining $\mathbf{X}$ to be a matrix with input vectors $\mathbf{x}_i$ arranged in its rows and $\mathbf{y}$ as a vector with coefficients y_i, we would like to maximize the log likelihood $\log p(\mathbf{y}|\mathbf{X})$ (also known as the "incomplete" log likelihood) for generating the observed data. Due to analytical issues, we do not have access to the incomplete log likelihood, but only a lower bound of it. The lower bound is based on an expected value of the "complete" data likelihood $\left\langle \log p(\mathbf{y}, \mathbf{b}, \mathbf{w}, \mathbf{h}, \sigma^2|\mathbf{X})\right\rangle$[2], where $p(\mathbf{y}, \mathbf{b}, \mathbf{w}, \mathbf{h}, \sigma^2|\mathbf{X}) = \prod_{i=1}^{N} p(y_i, \mathbf{b}, w_i, \mathbf{h}, \sigma^2|\mathbf{x}_i)$. In our model, each y_i of data sample i has an indicator-like scalar weight w_i associated with it. We can express the complete log likelihood L as:

$$\begin{aligned} L = &\sum_{i=1}^{N} \log p(y_i|\mathbf{x}_i, \mathbf{b}, \sigma^2)^{w_i} + \sum_{i=1}^{N}\sum_{m=1}^{d} \log p(w_{im}) \\ &+ \log p(\mathbf{b}|\sigma^2) + \log p(\sigma^2) + \log p(\mathbf{h}) \end{aligned}$$

Expanding the $\log p(w_{im})$ term above, we notice that there is a problematic $\log\left(1 + (x_{im} - x_{qm})^r\right)$ term that prevents us from deriving an analytically tractable expression for

[1] Note that the model in its current form does not address input data that has irrelevant and redundant dimensions. Modifications can be made, through the use of Automatic Relevance Determination (ARD) [10], to introduce such an ability, but this is left to another paper. For a redundant or irrelevant dimension m, h_m should reflect this redundancy/irrelevancy and take on a very low value.

[2] Note that $\langle\rangle$ denotes the expectation operator

the posterior of h_m. To address this, we use a variational approach on concave/convex functions suggested by Jaakkola et al. [11] in order to produce analytically tractable expressions. We can lower bound the term $\log\left(1+\left(x_{im}-x_{qm}\right)^r\right)$ so that $\log p(w_{im}) \geq (1-w_{im})\log\left(x_{im}-x_{qm}\right)^r h_m - \lambda_{im}\left(x_{im}-x_{qm}\right)^r h_m$, where λ_{im} is a variational parameter to be optimized in the M-step of our final EM-like algorithm. The lower bound to L is then:

$$\begin{aligned}\hat{L} = &-\frac{1}{2}\sum_{i=1}^{N} w_i \log\sigma^2 - \sum_{i=1}^{N}\frac{w_i\left(y_i-\mathbf{b}^T\mathbf{x}_i\right)^2}{2\sigma^2}\\ &+\sum_{i=1}^{N}\sum_{m=1}^{d}(1-w_{im})\log\left(x_{im}-x_{qm}\right)^r h_m\\ &-\sum_{i=1}^{N}\sum_{m=1}^{d}\lambda_{im}\left(x_{im}-x_{qm}\right)^r h_m - \frac{1}{2}\log\sigma^2\\ &+\frac{1}{2}\log|\boldsymbol{\Sigma}_{\mathbf{b}_0}^{-1}| - \frac{\mathbf{b}^T\boldsymbol{\Sigma}_{\mathbf{b}_0}^{-1}\mathbf{b}}{2\sigma^2} - \left(\frac{n_0}{2}+1\right)\log\sigma^2 - \frac{n_0\sigma_{N0}^2}{2\sigma^2}\\ &+\sum_{m=1}^{d}(a_{hm0}-1)\log h_m - \sum_{m=1}^{d} b_{hm0}h_m + \text{const}\end{aligned}$$

We would like to maximize the lower bound to the log likelihood and find the corresponding parameter values. The expectation of $\hat{L}$ should be taken with respect to the true posterior distribution of all hidden variables $Q(\mathbf{b},\sigma^2,\mathbf{h})$. Since this is an analytically intractable expression, a lower bound can be formulated using a technique from variational calculus where we make a factorial approximation [7] of the true posterior as follows: $Q(\mathbf{b},\sigma^2,\mathbf{h}) = Q(\mathbf{b},\sigma^2)Q(\mathbf{h})$. While losing a small amount of accuracy, all resulting posterior distributions over hidden variables become analytically tractable. The posterior distributions of w_{im}, $p(w_{im} = 1|y_i,\mathbf{x}_i,\boldsymbol{\theta},w_{i,k\neq m})$, are inferred using Bayes' rule:

$$\frac{p(y_i|\mathbf{x}_i,\boldsymbol{\theta},w_{i,k\neq m},w_{im}=1)^{\prod_{t=1,t\neq m}^{d}\langle w_{it}\rangle}p(w_{im}=1)}{p(y_i|\mathbf{x}_i,\boldsymbol{\theta},w_{i1},w_{i2},...,w_{id})}$$

where $\boldsymbol{\theta} = \left\{\mathbf{b},\sigma^2,\mathbf{h}\right\}$, $w_{i,k\neq m}$ denotes the weights for sample i in all dimensions except for dimension m. The posterior mean of w_{im} is then $\langle p(w_{im}=1|y_i,\mathbf{x}_i,\boldsymbol{\theta},w_{i,k\neq m})\rangle$. The final posterior EM update equations are listed below:

E-step:

$$\boldsymbol{\Sigma}_{\mathbf{b}} = \left(\boldsymbol{\Sigma}_{\mathbf{b}_0}^{-1} + \sum_{i=1}^{N} w_i\mathbf{x}_i\mathbf{x}_i^T\right)^{-1} \tag{5}$$

$$\langle\mathbf{b}\rangle = \boldsymbol{\Sigma}_{\mathbf{b}}\left(\sum_{i=1}^{N} w_i y_i\mathbf{x}_i\right) \tag{6}$$

$$\sigma_N^2 = \frac{\begin{pmatrix}\sum_{i=1}^{N} w_i\left\langle\left(y_i-\mathbf{b}^T\mathbf{x}_i\right)^2\right\rangle\\ +\left\langle\mathbf{b}^T\boldsymbol{\Sigma}_{\mathbf{b}}^{-1}\mathbf{b}\right\rangle + n_0\sigma_{N0}^2\end{pmatrix}}{n_0+\sum_{i=1}^{N} w_i} \tag{7}$$

$$\langle w_{im}\rangle = \frac{q_{im}A_i^{\prod_{k=1,k\neq m}^{d}\langle w_{ik}\rangle}}{q_{im}A_i^{\prod_{k=1,k\neq m}^{d}\langle w_{ik}\rangle}+1-q_{im}} \tag{8}$$

$$\langle h_m\rangle = \frac{a_{hm,0}+N-\sum_{i=1}^{N}\langle w_{im}\rangle}{b_{hm,0}+\sum_{i=1}^{N}\lambda_{im}\left(x_{im}-x_{qm}\right)^r} \tag{9}$$

M-step:

$$\lambda_{im} = \frac{1}{1+\left(x_{im}-x_{qm}\right)^r\langle h_m\rangle} \tag{10}$$

where $q_{im} = 1/\left(1+\left(x_{im}-x_{qm}\right)^r\langle h_m\rangle\right)$, and $A_i = \text{Normal}\left(y_i:\langle\mathbf{b}\rangle^T\mathbf{x}_i,\sigma_N^2\right)$. Examining (5) and (6), we see that when the data sample i has a lower weight w_i, it will be downweighted in the regression problem[3]. (7) shows that the output variance is calculated in a weighted fashion as well. (9) reveals that the distance metric h_m is a function of the number of samples that have a low weight (i.e., are almost excluded from the local model). Assuming the prior distribution of the weight kernel is initialized to be broad and wide (e.g., $a_{hm,0} = b_{hm,0} = 10^{-8}$—see next paragraph for more details), if all samples are included in the local model, then the numerator of h_m will be $a_{hm,0}$, leading to a very small h_m (i.e., a wide broad kernel that encompasses all samples) if the second term of the denominator dominates.

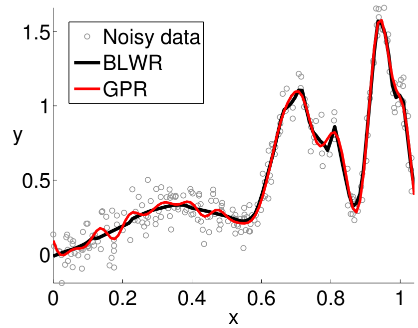
Note that an inversion of a $d\times d$ matrix needs to be done in (5), and this results in (5)-(10) having a computational complexity of $O(d^3)$ per EM iteration. To deal with problems with very high input dimensionality, we can introduce intermediate variables between the inputs and outputs, as done in [12], in order to get fast EM update equations that are $O(d)$ per EM iteration. We omit this derivation due to lack of space.

A few remarks should be made regarding the initialization of priors used in (5)-(10). We can set the initial covariance matrix of $\mathbf{b}$ to a large enough value (e.g., $10^6\mathbf{I}$, where $\mathbf{I}$ is the identity matrix) to indicate a large enough of uncertainty associated with the prior distribution of $\mathbf{b}$. n_0, the degrees of freedom parameter, should be set to the number of samples in the training set, and σ_{N0}^2, the initial noise variance, can be set to some small value (e.g., 0.1). Finally, the initial distance metric of the weighting kernel should also be set so that the kernel is broad and wide. For example, values of $a_{hm0} = b_{hm0} = 10^{-8}$ mean that the initial value of h_m is 1 with high uncertainty. These values can be used if no informative prior knowledge is available. Otherwise, if prior information is available, both parameters should be set to reflect this. In the event that more noise is present in the training data, the initial weighting kernel can be made to be broader, with less uncertainty associated with its initial bandwidth h_m value. Note that some sort of initial belief about the noise level is needed; otherwise, it will be impossible to distinguish between noise and structure in the training data.

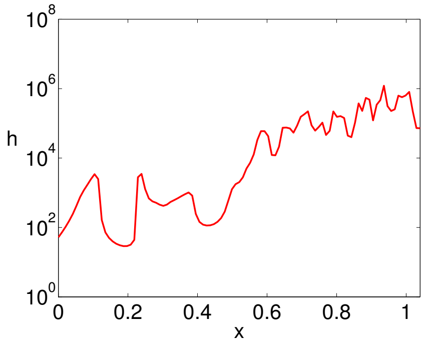
IV. Experimental Results

We evaluate our Bayesian locally weighted regression algorithm (BLWR), first, on synthetic data sets, in order to

[3] To avoid computational problems resulting from division by zero, note that during implementation, w_i needs to be capped to some small non-zero value.

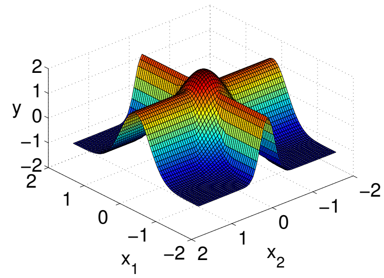


(a) Predicted output

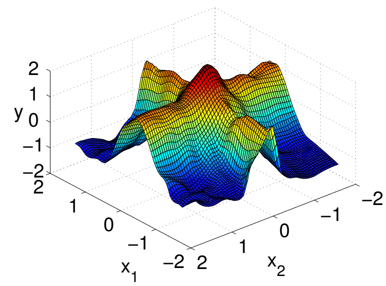


(b) h

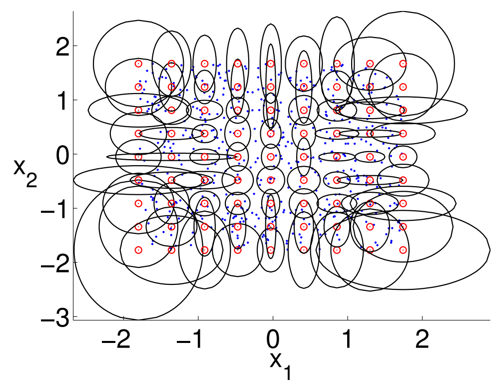
Fig. 2. 1-d function with varying curvature, shown for GPR and BLWR. Training data has an output noise variance of 0.01 and 250 samples.



(a) Target function



(b) Predicted function by BLWR



(c) Automatically learned distance metric values

Fig. 3. a) Target nonlinear 2-d CROSS target function; b) Predicted function produced by BLWR; c) Learned weighting kernels in input space, where the small red circles indicate the test input points and centers of the weighting kernels. Training data has an output noise variance of 0.01 and 500 samples.

establish that its performance is competitive to other state-of-the-art techniques for nonlinear regression such as locally weighted projection regression (LWPR) [3] and Gaussian process regression (GPR) [8]. Then, we demonstrate the effectiveness of our algorithm at estimating the Jacobian matrix in a kinematics problem for a 7 DOF robotic arm, comparing it with results from locally weighted regression (LWR) [3] with an optimally hand-tuned distance metric and with the analytically derived Jacobian.

A. *Synthetic Data*

First, we demonstrate the locally adaptive kernel property of our Bayesian locally weighted regression algorithm on a data set with scalar inputs for ease of visualization and compare it to GPR. GPR is a nonparametric technique for nonlinear function approximation that is generally acknowledged to be have excellent performance, but it is not computationally efficient for very large data sets. Since the BLWR model presented in this paper cannot deal with redundant and irrelevant dimensions, we use only low-dimensional synthetic small data sets for the purpose of demonstrating the competitive performance of BLWR to GPR on data with these characteristics. Future evaluations will address the application of BLWR to very large high-dimensional data sets.

Fig. 2(a) shows the predicted output for GPR and BLWR on the first data set (composed of 250 noisy training data samples), which was generated from the equation $y = x - \sin^3\left(2\pi x^3\right)\cos(2\pi x^3)\exp(x^4)$, with mean-zero noise with variance of 0.01 added to the outputs. GPR and BLWR perform similarly when predicting the outputs from noiseless test inputs, with GPR overfitting slightly more in the flatter areas on the left side of the data plot, as shown in red. In comparison, BLWR does not overfit the data in the flatter areas in order to accommodate the high curvature on the right side of the data plot. As Fig. 2(b) shows, it correctly adjusts the distance metric h with the curvature of the function (with h increasing as the curvature of the function increases) and does not display any overfitting or oversmoothing trends.

We also evaluated BLWR on a 500-sample data set consisting of the 2-dimensional function (CROSS), $y = \max\{\exp(-10x_1^2), \exp(-50x_2^2), 1.25\exp(-5(x_1^2+x_2^2))\}$, as previously examined in [2], [3]. Mean-zero noise with a

TABLE I

AVERAGE NORMALIZED MEAN SQUARED ERROR COMPARISONS, OVER 10 TRIALS, BETWEEN GPR, LWPR AND BLWR FOR THE NONLINEAR 2-D CROSS FUNCTION.

Algorithm	nMSE	std-dev
GPR	0.01991	0.00314
LWPR	0.02556	0.00416
BLWR	0.02609	0.00532

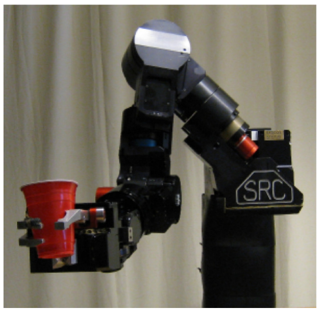


Fig. 4. Sarcos anthropomorphic arm

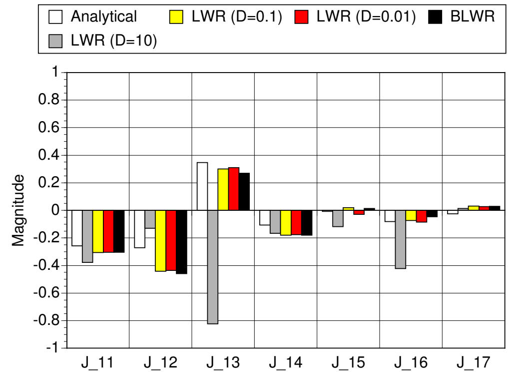
variance of 0.01 was added to the outputs. Fig. 3(a) shows the target function, evaluated over a noiseless test input (giving 1681 data points on a 41×41 grid in the 2×2 square in input space). Fig. 3(b) shows the predicted outputs for BLWR, and Fig. 3(c) depicts the learned distance metric values h over a subset of the test data points scattered over the 41×41 grid (shown as red circles). As before, we see that the width of the weighting kernel adjusts according to the curvature of the function. Table. I compares the performance of BLWR to GPR and LWPR, averaged over 10 randomly chosen training data sets. Performance was quantified in terms of normalized mean squared prediction error (nMSE) value on the noiseless test sets. We see that BLWR performs competitively to LWPR, with GPR doing slightly better.

B. Robotic Arm Data

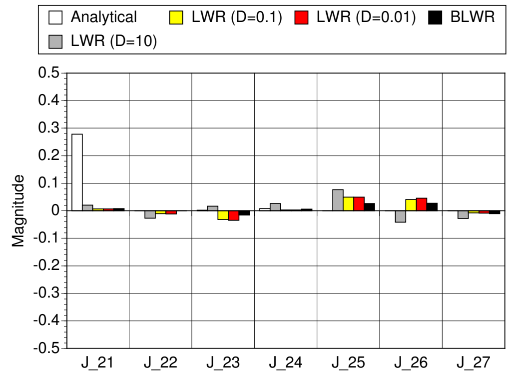
We collected 10,800 data samples from a 7 DOF anthropomorphic robotic arm made by Sarcos, as shown in Fig. 4, while performing a trajectory tracking task in Cartesian space. The input data, $\boldsymbol{\theta}$, consists of 7 arm joint angles, while output data is the resulting position, $\mathbf{p} = [x\ y\ z]^T$, of the arm's end effector in Cartesian space. For the purpose of establishing that BLWR does the right thing for each local regression problem, we would like to solve the kinematics problem, $\mathbf{p} = f(\boldsymbol{\theta})$, in order to find the Jacobian J for a local linearization problem, as defined below:

$$\frac{\partial \mathbf{p}}{\partial t} = \underbrace{\frac{\partial f}{\partial \boldsymbol{\theta}}}_{J} \frac{\partial \boldsymbol{\theta}}{\partial t}$$

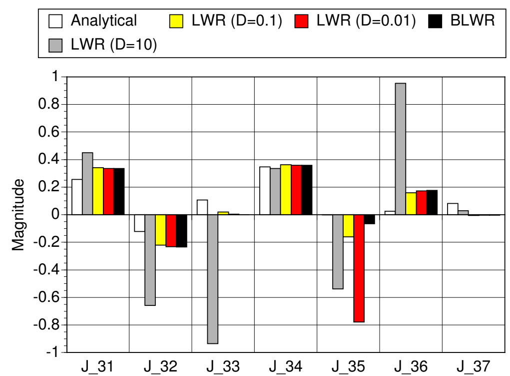
We compare the estimated Jacobian values to the analytically computed Jacobian J_A for a particular local linearization problem, given a query input vector of joint angles: $[0\ \ -0.3\ \ 0\ \ 1.57\ \ 0\ \ 0\ \ 0]^T$. Since locally weighted regression (LWR) with cross-validation (to find the optimal distance metric) on a 7-dimensional data set would be computationally prohibitive, we instead choose to compare our Bayesian algorithm with LWR where the distance metric is manually hand-tuned to be optimal. We run LWR using a range of distance metric values where, for simplicity, the weighting kernel had a homogeneous (i.e., the same) width in all dimensions of the input space. We explored a wide range of different distance metric values ($\sim 100 - 200$ values) for LWR—a painstaking procedure—and report results from a representative set of these values. We can visualize the coefficients of the 3×7 Jacobian matrix using bar plots,



(a) J_1: coefficients of the first row of the Jacobian



(b) J_2: coefficients of the second row of the Jacobian



(c) J_3: coefficients of the third row of the Jacobian

Fig. 5. Analytically derived versus inferred values of the Jacobian matrix, shown for each of the 3 rows of the matrix (corresponding to the x, y and z positions of the robotic arm's end effector).

TABLE II

ANGULAR AND MAGNITUDE DIFFERENCE BETWEEN THE ANALYTICAL JACOBIAN J_A AND THE INFERRED JACOBIAN OF BLWR, J_B. J_i IS THE iTH ROW OF J, AND $|J_{A,i}|$ IS THE MAGNITUDE OF THE iTH ROW OF J_A.

J_i	$\angle J_{A,i} - \angle J_{B,i}$	abs $(\lvert J_{A,i}\rvert - \lvert J_{B,i}\rvert)$	$\lvert J_{A,i}\rvert$	$\lvert J_{B,i}\rvert$
J_1	19 degrees	0.1129	0.5280	0.6464
J_2	79 degrees	0.2353	0.2780	0.0427
J_3	25 degrees	0.1071	0.4687	0.5758

TABLE III

ANGULAR AND MAGNITUDE DIFFERENCE BETWEEN THE ANALYTICAL JACOBIAN J_A AND THE INFERRED JACOBIAN OF LWR, J_L (WITH A MANUALLY TUNED OPTIMAL DISTANCE METRIC OF $D = 0.1$).

J_i	$\angle J_{A,i} - \angle J_{L,i}$	abs $(\lvert J_{A,i}\rvert - \lvert J_{L,i}\rvert)$	$\lvert J_{A,i}\rvert$	$\lvert J_{L,i}\rvert$
J_1	16 degrees	0.1182	0.5280	0.6411
J_2	85 degrees	0.2047	0.2780	0.0734
J_3	27 degrees	0.1216	0.4687	0.5903

shown in Figs. 5(a)-5(c). Each bar plot shows the coefficients of a row of J, where $J = [J_1 \ J_2 \ J_3]^T$, and compares the inferred coefficient values from BLWR and LWR. Of all the distance metric values we experimented with for LWR, a distance metric value of 0.1 appeared to be most suitable for this particular linearization problem, as the bar plots show. Notice that the coefficients for $J_{A,2}$ are particularly small, as seen in Fig 5(b). This can be explained by the lack of exploration and movement by the robotic arm in the y-coordinate of Cartesian space while the training data was collected.

To evaluate the difference between the coefficients of the analytical Jacobian J_A and the coefficients of the inferred Jacobian by BLWR and LWR (denoted by J_B and J_L, respectively), we calculate the angle between the analytical and inferred row vectors of the Jacobian matrix. Tables II-III report this angular difference, as well as the magnitude difference between vectors and the individual magnitudes of each vector. We observe that, in general, BLWR and LWR with an optimally hand-tuned distance metric perform similarly, with angular differences ranging from 16 to 30 degrees with the analytical Jacobian row vectors. Notice that the angular differences for J_2 are particularly large and that the magnitudes of BLWR's and LWR's inferred row vectors are rather small (0.0427 and 0.0734, respectively, compared to 0.2780 for the second row vector of the analytical Jacobian). The large J_2 angular difference for BLWR and LWR is not particularly worrisome, given the relatively small magnitudes of all row vectors $J_{A,2}$, $J_{B,2}$ and $J_{L,2}$. As such, the large angular difference for J_2 can be explained and discounted in the evaluation of BLWR and LWR algorithms. Note that the condition number associated with the input data is a very large 10^5, indicating that the problem is ill-conditioned and not as easy to solve as it may appear.

In this robotic experiment, we see the advantages offered by our Bayesian locally weighted algorithm. BLWR performed as well as LWR with an optimally hand-tuned distance metric, but without the need for any meta-parameter tuning, cross-validation or involved hypothesis testing.

V. CONCLUSIONS

We introduced a Bayesian formulation of spatially local adaptive kernels for locally weighted regression. Our approach treats all open parameters probabilistically and learns the appropriate local regime for each linearization problem. We present experimental results on synthetic low-dimensional data, showing competitiveness with Gaussian process regression, a state-of-the-art nonparametric nonlinear function approximation method. On a 7-dimensional linearization problem for a robotic arm, we demonstrate that our Bayesian algorithm performs just as well as a locally weighted algorithm where the distance metric is hand-tuned to be optimal. However, our algorithm does not require the painstaking and time-consuming process of cross-validating or hypothesis testing. Future work will address the application of this Bayesian locally linear algorithm to high-dimensional function approximation where the input data contains numerous redundant and irrelevant dimensions—a common scenario in problems of reinforcement learning and nonlinear humanoid robot control.

VI. ACKNOWLEDGMENTS

This research was supported in part by National Science Foundation grants ECS-0325383, IIS-0312802, IIS-0082995, ECS-0326095, ANI-0224419, a NASA grant AC#98 − 516, an AFOSR grant on Intelligent Control, the ERATO Kawato Dynamic Brain Project funded by the Japanese Science and Technology Agency, and the ATR Computational Neuroscience Laboratories.

REFERENCES

[1] C. Atkeson, A. Moore, and S. Schaal, "Locally weighted learning," *AI Review*, vol. 11, pp. 11–73, April 1997.

[2] S. Schaal, S. Vijayakumar, and C. Atkeson, "Local dimensionality reduction," in *Advances in Neural Information Processing Systems*, M. Jordan, M. Kearns, and S. Solla, Eds. MIT Press, 1998.

[3] S. Vijayakumar, A. D'Souza, and S. Schaal, "Incremental online learning in high dimensions," *Neural Computation*, pp. 1–336, 2005.

[4] J. H. Friedman, "A variable span smoother," Stanford University, Tech. Rep., 1984.

[5] J. Fan and I. Gijbels, "Variable bandwidth and local linear regression smoothers," *Annals of Statistics*, vol. 20, pp. 2008–2036, 1992.

[6] ——, "Data-driven bandwidth selection in local polynomial fitting: Variable bandwidth and spatial adaptation," *Journal of the Royal Statistical Society B*, vol. 57, pp. 371–395, 1995.

[7] Z. Ghahramani and M. Beal, "Graphical models and variational methods," in *Advanced Mean Field Methods - Theory and Practice*, D. Saad and M. Opper, Eds. MIT Press, 2000.

[8] C. E. Rasmussen, "Evaluation of Gaussian Processes and other methods for non-linear regression," Ph.D. dissertation, University of Toronto, 1996.

[9] A. Dempster, N. Laird, and D. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of Royal Statistical Society. Series B*, vol. 39, no. 1, pp. 1–38, 1977.

[10] R. Neal, "Bayesian learning for neural networks," Ph.D. dissertation, Dept. of Computer Science, University of Toronto, 1994.

[11] T. S. Jaakkola and M. I. Jordan, "Bayesian parameter estimation via variational methods," *Statistics and Computing*, vol. 10, pp. 25–37, 2000.

[12] J. Ting, A. D'Souza, K. Yamamoto, T. Yoshioka, D. Hoffman, S. Kakei, L. Sergio, J. Kalaska, M. Kawato, P. Strick, and S. Schaal, "Predicting EMG data from M1 neurons with variational Bayesian least squares," in *Proceedings of Advances in neural information processing systems 18*. MIT Press, 2005.