Locally-Weighted Regression for Estimating the Forward Kinematics of a Geometric Vocal Tract Model

Adam C. Lammert[1], *Louis Goldstein*[2], *Khalil Iskarous*[3]

[1]Department of Computer Science, University of Southern California, California, USA
[2]Department of Linguistics, University of Southern California, California, USA
[3]Haskins Laboratories, New Haven, Connecticut, USA

lammert@usc.edu, louisgol@usc.edu, iskarous@haskins.yale.edu

Abstract

Task-space control is well studied in modeling speech production [1, 2, 3, 4]. Implementing control of this kind requires an accurate kinematic forward model. Despite debate about how to define the tasks for speech (i.e., acoustical vs. articulatory), a faithful forward model will be complex and infeasible to express analytically. Thus, it is necessary to learn the forward model from data. Artificial Neural Networks (ANNs) have previously been suggested for this [3, 4, 6]. We argue for the use of locally-linear methods, such as Locally-Weighted Regression (LWR). While ANNs are capable of learning complex forward maps, LWR is more appropriate. Common formulations of control assume locally-linearity, whereas ANNs fit a nonlinear model to the entire map. Likewise, training LWR is simple compared to the complex optimization for ANNs. We provide an empirical comparison of these methods for learning a vocal tract forward model, discussing theoretical and practical aspects of each.

Index Terms: speech production, task dynamics, forward kinematics, statistical machine learning, forward model estimation

1. Introduction

The task-dynamic model of speech production posits that the vocal tract is controlled at the level of tasks, rather than articulatory variables [1, 2]. This kind of task-space control has been well studied in the robotics community [13], and it is well supported by empirical data in the speech domain. Proper task-level control requires an accurate forward model, which describes the mapping between the low-level articulatory variables and relatively higher-level tasks. If the levels are disparate, the forward model for a given system can become very complex. For example, if the articulatory variables are individual muscle activations and the task variables are more abstract quantities like constriction degrees or formant frequencies, then the forward model represents a wide variety physical processes between those levels. As a result, the forward model will be a highly nonlinear mapping and, in practice, it will be infeasible to express analytically.

It is possible to estimate the forward model from data in these cases. Artificial neural networks (ANNs) are an obvious choice for learning models of this sort, since they have been theoretically verified as universal function approximators [12]. There is a long, successful history of using neural networks to estimate complex forward models from data. ANNs have been used to estimate forward and inverse maps in speech research [3, 4]. Saltzman [6] recently suggested the use of ANN-learned forwards and inverse models as a way to quantify the debate concerning the nature of the speech production tasks, using the uncontrolled manifold method [19]. The first step is to estimate the articulator-state dependent Jacobian for both the articulator-to-constriction task and articulator-to-acoustic maps. Articulatory and acoustic data from actual experiments can then be projected onto the null-spaces of this Jacobian (termed the uncontrolled manifold). This facilitates a comparison between the projection onto the null-space of the Jacobian and the projection onto its orthogonal complement (controlled manifold). The map showing greater projection onto the controlled manifold than the uncontrolled is deemed to be the one with the controlled task. But to achieve the quantification of this long debate in speech literature, first the maps have to be reliably learned from data. Saltzman [6] suggested using backpropagation.

However, ANNs have some drawbacks. They are notoriously difficult to design and slow to train [5]. Recent years have seen the successful use of locally-linear methods (e.g., Locally-Weighted Regression (LWR)) for estimating robotic forward models [8]. These methods have many desirable properties, including the lack of a long training phase and the presence of far fewer free parameters, as compared to artificial neural networks. In addition, using LWR is more appropriate for learning forward models in many applications. In control applications, local linearizations are ubiquitous. Elegant formulations of task-space control can be made by assuming that a local linearization of the forward model is appropriate. In particular, forward transformations are frequently expressed as a simple matrix multiplication with the Jacobian, an inherently linear transformation. LWR is estimating exactly what is needed for this common formulation. With a neural network, one must perform the additional step of locally linearizing the learned mapping.

We demonstrate the use of LWR to learn a forward kinematic model of the vocal tract. The data set is built by utilizing the TAsk Dynamic Application (TADA), an articulatory speech synthesizer [16]. At the core of TADA is the Configurable Articulatory Synthesizer (CASY) model [14, 15], for which the Jacobian can be derived analytically. This facilitates a comparison of each method's accuracy. The articulatory variables we consider are the same as CASY's input parameters (e.g., jaw height, tongue body position, tongue tip position, etc.). The task variables are those calculated by TADA, namely vocal tract constriction degrees and locations (e.g., tongue tip constriction, tongue body constriction, etc.). We demonstrate the effectiveness of LWR technique on a simple model of the vocal tract. We also illustrate the efficiency of LWR for this purpose. The current study is part of a larger one, where the maps from articulators and tasks to acoustics are also learnt and compared.

Section 2 will serve as a review of ANNs and LWR for learning forward models and estimating the Jacobian. In Section 3, we will discuss the experiments we conducted to test each learning method, and we present the results. In Section 4, we compare the performance of ANNs and LWR, and in Section 5, we present our concluding remarks.

2. Learning Forward Kinematics

Forward kinematic models map articulatory variables into the space of tasks. Mappings of this sort are, in general, highly nonlinear and difficult to express analytically. Thus, it can be practically useful to learn an approximation of the model from data. Many techniques are capable of this kind of learning. Choosing one comes down to several issues, including efficiency, ease of design, functional capability (i.e., what kind of functions it can learn) and also appropriateness to the application.

Our application of interest, for this speech-centered study, is the mapping from articulatory variables to constriction degrees and locations. Saltzman [6] suggested that this mapping could be learned by an artificial neural network. We implemented both artificial neural networks and locally-weighted regression to learn the forward kinematic model. Before discussing the relative merits of both methods for this application, we provide a brief review of both learning frameworks.

2.1. The Forward Model

To build an appropriate data set, we utilized the Task Dynamic Application (TADA) [16], an articulatory speech synthesizer. At the core of TADA is the CASY model [14, 15]. CASY is both (a) a geometric vocal tract model and (b) an implementation of simple transfer functions, which can produce reasonable acoustic output. As part of its operation, CASY also produces the analytically-derived Jacobian.

Thus, the forward model uses articulatory variables which are the same as CASY's input parameters. These articulatory variables are as follows: lip protrusion, jaw angle, vertical displacements of the upper lip and lower lip, tongue body position (an angle and a length), and tongue tip position (an angle and a length),. We built the data set by systematically manipulating a subset of the articulatory variables. The details are presented in Section 3, below.

Using TADA, we also calculated the task variables. These included the following: lip aperature (LA) and protrusion (PRO), tongue body constriction degree (TBCD) and location (TBCL), as well as tongue tip constriction degree (TTCD) and location (TTCL). Crucially, we also obtained the analytically-derived Jacobians for each of the articulatory poses.

Until now, we have only hinted at the complexity of the forward map. We present the equations for calculating the tongue body tasks from the articulatory variables.

$$x = (a_{cl} sin(a_{ja} + a_{ca}) - 0.7339)^2 \tag{1}$$

$$y = (-a_{cl} cos(a_{ja} + a_{ca}) + 0.4562)^2 \tag{2}$$

$$t_{TBCD} = 100 \cdot (0.6 - \sqrt{x + y}) \tag{3}$$

$$t_{TBCL} = \frac{180}{\pi} \cdot cos^{-1}(\frac{\sqrt{x}}{\sqrt{x + y}}) \tag{4}$$

where the articulatory variables are denoted by a, with subscripts indicating jaw angle (ja), tongue body length (cl) and tongue body angle (ca). The task variables are denoted by t. These task are highly nonlinear in the articulators. Still, they are relatively simpler than the tongue tip tasks, which has a similar form as TBCD and TBCL, but have greater redundancy, since five different articulators, represented geometrically as two lengths and three angles, contribute to them.

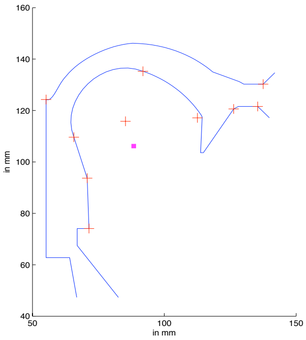

Figure 1: *A visualization of the Configurable Articulatory Synthesizer (CASY) in a neutral position, showing the outline of the vocal tract model. Crosses represent configurable portions of the vocal tract, which can be manipulated interactively.*

2.2. Artificial Neural Networks

We chose to implement a feedforward neural network and to train it using standard backpropagation. This network had linear input and output nodes, corresponding to each of the articulatory and task variables, respectively. Between were two layers of hidden units, all with sigmoidal activation functions.

Design parameters included the learning rate, the stopping criterion for training, and the number of units in each hidden layer. We did not consider the number of hidden layers to be a free parameter, and instead opted to fix the number at two.

Upon completion of training, a Jacobian (i.e., linearized) representation of the network can be obtained for any articulatory input vector. This can be done with the use of numerical methods [5].

However, the Jacobian can also be obtained analytically for a network of this kind using the feedforward formalism [11, 6]. If we note that each hidden node has a sigmoidal activation function, we can write the derivative of each node's activation with respect to its input as follows:

$$\delta = z(1 - z) \tag{5}$$

Then, we can arrange these values in a diagonal matrix, denoted Δ_i. The Jacobian for a given input posture is then

$$J = \Delta_{out} W_{out,H2} \Delta_{H2} W_{H2,H1} \Delta_{H1} W_{H1,in} \tag{6}$$

where W_{ij} is the weight matrix connecting layer i to layer j.

2.3. Locally-Weighted Regression

Locally-weighted regression is a memory-based, lazy learning method [7]. As such, the entire data set is remembered and used directly at prediction time, in order to calculate the parameters of interest. As such, we begin by assuming a data set with N feature vectors,

$$X = \{x_i\}_{i=1}^{N} \tag{7}$$

and an equal number of target vectors,

$$Y = \{y_i\}_{i=1}^{N} \tag{8}$$

It is assumed that the data were generated by a model following

$$y_i = f(x_i) + \epsilon \tag{9}$$

where f is a function which can be nonlinear, in general. The value ϵ represents the noise which follows

$$\epsilon \sim N(0, \sigma^2) \tag{10}$$

a normal distribution with mean 0 and variance σ^2.

We would like to fit the data in a local region defined by the data point x_i. The measure of locality K is taken to be a Gaussian kernel function

$$K(x_i, x_q, h) = exp\{-0.5(x_i - x_q)^T H(x_i - x_q)\} \tag{11}$$

although any such kernel can be utilized. H is a positive semi-definite diagonal matrix, with diagonal elements equal to h. The value h is a free parameter which determines the size of the kernel function. The larger it is, the narrower is the kernel, i.e., the smaller is the subset of the input space used.

The model we would like to fit in our local region is of the form

$$y_i = b^T x_i + \epsilon \tag{12}$$

where b is the vector of regression coefficients. The least squares solution can be found for b by computing:

$$b = (X^T W X)^{-1} X^T W Y \tag{13}$$

The matrix, W, is a diagonal weight matrix, formed from the outcomes of the kernel function.

Obtaining the Jacobian from this model is trivial, since it is already linear. The regression vector b contains the locally-relevant partial derivatives. In other words, the values of this vector are the elements of the Jacobian.

3. Experiments

We ran experiments to compare the accuracy of each method. To that end, we implemented both techniques in MATLAB and ran them on a data set, which we obtained by simulating TADA. The input and output variables consisted of the articulatory and tasks variables mentioned in Section 2.1. We manipulated the articulatory variables so as to evenly fill the input space. This generated an enormous number of data points, so we randomly selected a subset of 2000 points as a representative set. The analytically-derived Jacobians for each input vector were also collected, for the purposes of comparison.

For the sake of simplicity, we hand-tuned the LWR algorithm for our experiments. We found the optimal value for the width parameter, h, to be 300. We used this width value for all directions of the kernel function (i.e., homogeneous width). We also hand-designed the ANN, in terms of its topology and parameters. We used a network with 2 hidden layers, each containing 25 nodes. The learning rate was fixed to 0.0001 throughout the training period. The training period was capped at 200 iterations over the training data.

To compare the learned Jacobians to the analytically-derived Jacobian, we calculated the row-wise error for 25 randomly selected articulatory poses. For each pose, we calculated the magnitude and angle between each row vector of the analytic Jacobian and the estimated Jacobians. The results are shown in Table 1 (ANN) and Table 2 (LWR). To insure that the results are not highly sensitive to the particular parameters chosen, we evaluated estimation performance for seven different values of the LWR width parameter h from 100 to 700, in steps of 100, and seven different values of the number of nodes in both hidden layers from 5 to 65, in steps of 10. Increasing h and the number of nodes corresponds to decreasing the bias, from a bias-variance tradeoff view of model complexity. For each parameter value, we calculated the correlation coefficients in the same way as in Tables 1 and 2, but we averaged the correlation coefficients across all tasks. The results are in Fig 2.

J_i	r	$\angle(J_{A,i}, J_{B,i})$	$\parallel J_{A,i} \parallel$	$\parallel J_{B,i} \parallel$
J_{TBCL}	0.38	67	6.02	3.70
J_{TBCD}	0.68	46	1.00	1.12
J_{TTCL}	0.83	33	4.71	2.09
J_{TTCD}	0.33	70	1.89	1.85
J_{LA}	0.68	46	1.79	1.55
J_{PRO}	0.32	71	1.00	0.29

Table 1: *The mean correlation coefficient (r) between the rows of the analytically-derived Jacobian (J_A) and the Jacobian learned from the ANN (J_B). Also shown are the mean angle in degrees, the norm of each vector. The abbreviations that instantiate the subscript i of J correspond to the task variables described in Section 2.1*

J_i	r	$\angle(J_{A,i}, J_{B,i})$	$\parallel J_{A,i} \parallel$	$\parallel J_{B,i} \parallel$
J_{TBCL}	0.55	56	6.02	8.25
J_{TBCD}	0.86	30	1.00	1.05
J_{TTCL}	0.96	15	4.71	2.48
J_{TTCD}	0.36	68	1.89	2.51
J_{LA}	0.80	36	1.79	1.40
J_{PRO}	1.00	0.2	1.000	0.99

Table 2: *The mean correlation coefficient (r) between the rows of the analytically-derived Jacobian (J_A) and the Jacobian learned from the LWR (J_B). Also shown are the mean angle in degrees, the norm of each vector and difference between those norms. The abbreviations that instantiate the subscript i of J correspond to the task variables described in Section 2.1*

4. Results & Discussion

The results in Tables 1 and 2 show that LWR outperforms the ANN in terms of accuracy, for the map explored here and the particular parameters chosen. It is also notable that, for both methods, some rows of the Jacobian show fairly large angular differences as compared to the analytically-derived Jacobian. This highlights the complexity of the forward map we are trying to estimate. Specifically, for tongue tip constriction degree (TTCD), both methods perform quite poorly, probably due to the high degree of nonlinearity involved in that task.

As can be seen in Fig 2, it is possible to increase the accuracy of LWR by decreasing kernel width (increasing h) and the accuracy of the ANN by increasing the number of nodes, but with such an increase, the likelihood of overfitting increases. Moreover, LWR outperforms this particular ANN at the low and high ends of the parameters. Overall, it is notable that LWR and ANNs can both be used for learning complex forward models, since they are fundamentally different. LWR does have many desirable qualities which make it a better choice. LWR directly learns the appropriate values for generating the locally-linearized Jacobian, which means there is no need for an additional linearization step. Given the importance of local linearization in task-space control, this property is crucial.

There are many practical considerations, as well. ANNs

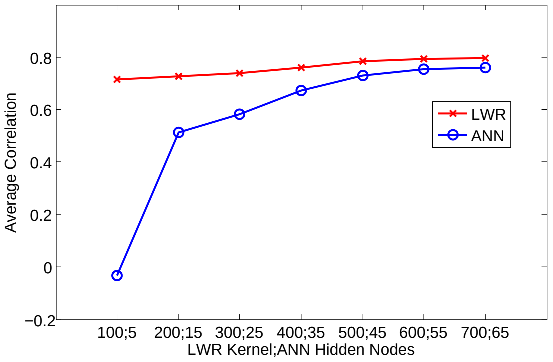

Figure 2: *A comparison of LWR and ANN performance as a function of the kernel parameter of LWR and the number of hidden nodes in each layer of the ANN. The dependent variable is the average correlation coefficient between the rows of the analytic Jacobian and each of LWR and ANN estimated Jacobians across all tasks and over 25 articulator poses.*

have many free parameters, and can be difficult to design. LWR, on the other hand, has only the locality constant. It should be noted that Bayesian methods have been developed which can optimize this free parameter automatically [8]. Training is one area where these methods are drastically different. In ANNs, training is done as an separate stage, before any predictions can be made from the model. Usually this stage is expensive in terms of computation time, since it involves optimization of a complex, non-convex cost function. In practice, it can be difficult to determine convergence of the training procedure. After training, the data set is essentially superfluous, unless training needs to be repeated (e.g., if a new data point arrives). Predictions are generally quick, and their efficiency depends on the (usually compact) network size.

5. Conclusion

We have demonstrated that a complex forward model of speech production can be learned from data. Both artificial neural networks and locally-weighted regression were tested on a speech production data set. We compared the accuracies of the two models by using them to compute a set of Jacobians for a variety of articulatory poses, and them comparing the learned Jacobians to an analytically-derived Jacobian. Similar accuracies were observed for each method with respect to this gold standard. However, the forward kinematics for some speech tasks, especially TTCD seems to be quite difficult to learn. Further work will attempt to improve learning for those tasks, using a bayesian approach [8], and to extend the learning to the estimation of acoustic tasks. We also plan to learn a relevant forward map from real (rtMRI) articulatory data [17, 18].

6. Acknowledgments

This work was supported by NIH NIDCD grant 02717, NIH R01 grant DC008780, NIH grant DC007124, as well as a graduate fellowship from the Annenberg Foundation. We would also like to acknowledge Elliot Saltzman for his technical insights.

7. References

[1] Saltzman, E. and Kelso, J.A.S., ”Skilled Actions: A Task Dynamic Appraoch”, Psychological Review, 94:84-106, 1987.

[2] Saltzman, E. and Munhall, K.G., ”A Dynamical Approach to Gestural Patterning in Speech Production”, Ecological Psychology, 1:333-382, 1989.

[3] Bailly, G., Laboissière, R., and Schwartz, J. L., ”Formant trajectories as audible gestures: an alternative to speech synthesis”, Journal of Phonetics, 19:9-23, 1991.

[4] Guenther, F., Hampson, L., and Johnson, D., ”A Theoretical Investigation of Reference Frames for the Planning of Speech Movements”, Psychological Review, 105:611-633, 1998.

[5] Bishop, C., ”Pattern Recognition and Machine Learning”, Springer, 2006.

[6] Saltzman, E., Kubo, M. and Tsao, C.C., ”Controlled Variables, the Uncontrolled Manifold, and the Task-Dynamic Model of Speech Production”, in Dynamics of Speech Production and Perception [Divenyi et al., eds.], IOS Press, 2006.

[7] Atkeson, C., Moore, A. and Schaal, S., ” Locally Weighted Learning”, AI Review, 11:11-73, 1997.

[8] Ting, J.A., D’Souza, A., Vijayakumar, S. and Schaal, S., ”A Bayesian Approach to Empirical Local Linearization for Robotics”, International Conferences on Robotics and Automation, Pasadena, CA., 2008.

[9] Jordan, M., ”Constrained Supervised Learning”, Journal of Mathematical Psychology, 36:396-425, 1992.

[10] Rumelhart, D., Hinton, G. and Williams, R., ”Learning Internal Representations by Error Propagation”, in Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1. Foundations [Rumelhart and McLelland, eds.] , MIT Press, 2008.

[11] Jordan, M. and Rumelhart, D., ”Forward Models: Supervised Learning with a Distal Teacher”, Cognitive Science, 16:307-354, 1992.

[12] Hornik, K., Stinchcombe, M. and White, H., ”Multilayer Feedforward Networks are Universal Approximators”, Neural Networks, 2, 1989.

[13] Sciavicco, L. and Siciliano, B., ”Modelling and Control of Robot Manipulators”, Springer, 2005.

[14] Rubin, P., Saltzman, E., Goldstein, L., McGowan, R., Tiede, M. and Browman, C. ”CASY and Extensions to the Task-Dynamic Model”, in 1st ETRW on Speech Production Modeling: From Control Strategies to Acoustics, Autrans, France.

[15] Iskarous, K., Goldstein, L., Whalen, D.H., Tiede, M. and Rubin, P., ”CASY: The Haskins Configurable Articulatory Synthesizer”, in Proceedings of the 15th International Congress of Phonetic Sciences, 2003.

[16] Nam, H., Goldstein, L., Browman, C., Rubin, P., Proctor, M. and Saltzman, E., ”TADA (TAsk Dynamics Application) manual”, Technical Report and Manual, 2006.

[17] Narayanan, S., Nayak, K., Lee, S., Sethy, A. and Byrd, D., “An approach to real-time magnetic resonance imaging for speech production”, JASA, 109:2446, 2004.

[18] Bresch, E., Nielsen, J., Nayak, K. and Narayanan, S., “Synchronized and noise-robust audio recordings during realtime MRI scans”, JASA, 120:1791, 2006.

[19] Scholz, J.P., and Schner, G., “The uncontrolled manifold concept: identifying control variables for a functional task”, Experimental Brain Research, 126:189-306, 1999.