LETTER Communicated by Reza Shadmehr

Failure of Motor Learning for Large Initial Errors

Terence D. Sanger
sanger@stanford.edu
Department of Neurology and Neurological Sciences, Stanford University, Stanford, CA 94305-5235, U.S.A.

For certain complex motor tasks, humans may experience the frustration of a lack of improvement despite repeated practice. We investigate a computational basis for failure of motor learning when there is no prior information about the system to be controlled and when it is not practical to perform a thorough random exploration of the set of possible commands. In this case, if the desired movement has never yet been performed, then it may not be possible to learn the correct motor commands since there will be no appropriate training examples. We derive the mathematical basis for this phenomenon when the controller can be modeled as a linear combination of nonlinear basis functions trained using a gradient descent learning rule on the observed commands and their results. We show that there are two failure modes for which continued training examples will never lead to improvement in performance. We suggest that this may provide a model for the lack of improvement in human skills that can occur despite repeated practice of a complex task.

1 Introduction

Although humans are extraordinarily adept at learning new and complex skills, it is not unusual for a person learning a new skill to find that he or she is making no improvement despite repeated practice. For example, in the absence of effective coaching, a beginner learning to play golf may attempt many hundreds of swings without significantly improving performance. The same experience is common for many sports, musical skills, and other skilled tasks. A more pathological form of this phenomenon may occur for children and adults with movement disorders, for whom a lifetime of practice at movement does not lead to improvement in the quality of their motor skills. Since there are many examples of adaptive algorithms that are able to improve skill through repeated practice, these phenomena lead to the question of whether there is a particular limitation that can explain the lack of success with repetitive practice for complex skills.

Successful control of an unknown environment requires the ability to determine the motor commands that lead to particular desired results. If a motor plan is specified in terms of a sequence of desired states, then at

some point, it becomes necessary to perform the inverse dynamics transform that will estimate the appropriate control signals needed to achieve the desired states. Learning an inverse dynamics transform leads to an internal model of the behavior of the environment (Scheidt, Reinkensmeyer, Conditt, Rymer, & Mussa-Ivaldi, 2000; Dingwell, Mah, & Mussa-Ivaldi, 2002; Flanders, Hondzinski, Soechting, & Jackson, 2003). There are many examples of successful algorithms for learning inverse dynamics models from observations of the behavior of a plant (Kawato, Furukawa, & Suzuki, 1987; Atkeson, 1989; Kawato & Gomi, 1992; Jordan & Rumelhart, 1992). Experiments in humans and animals (Gandolfo, Li, Benda, Schioppa, & Bizzi, 2000) have shown strong support for the formation of internal models (Dingwell et al., 2002) and the ability to adapt, with practice, to unknown or unusual dynamic environments (Krakauer, Pine, Ghilardi, & Ghez, 2000; Thoroughman & Shadmehr, 1999; Wang, Dordevic, & Shadmehr, 2001). Further experiments have indicated that generalization to previously unseen conditions is possible (Goodbody & Wolpert, 1998; Flanders et al., 2003; Criscimagna-Hemminger, Donchin, Gazzaniga, & Shadmehr, 2003), and that there are limits on generalization that are consistent with models based on local basis-function approximations (Shadmehr & Moussavi, 2000; Thorughman & Shadmehr, 2000; Donchin & Shadmehr, 2003).

However, essentially all computational algorithms and biological models for solving the inverse dynamics problem assume either that an approximate inverse model is available prior to learning or that a sufficient number of examples of motor commands and their resulting effects are available for training. If an approximate inverse model is available, then it may be possible to extrapolate from observed examples to determine the necessary changes in the motor command for desired performance. Even if an approximate model is not available, some algorithms can choose motor commands randomly in order to explore the space of reachable states. This has been proposed as an explanation of the apparently random movements of infants. But there are at least three problems with random exploration. First, for any real robotic or natural control system, random exploration can lead to injury. Second, the dimensionality of the control or state space may be high enough that complete exploration is not feasible in a reasonable amount of time. Third, there are clearly some states that are more useful than others, and it would be advantageous to explore only near regions of desired behavior. For example, there is no point in learning to play golf by randomly swinging the club about one's head.

Here, we investigate limitations on motor learning when no approximation to the plant is available and when random exploration is dangerous or otherwise not feasible. We model learning by choosing a particular desired effect s^* that the system will transform into an estimated motor command m. This motor command will lead to an actual effect s that will in general be different from s^*. The system then uses the trio (s^*, m, s) to improve its future performance (Sanger, 1994). The goal of the system is to map s^* into

the "correct" command m^* that achieves $s = s^*$. If an approximation to the plant or its inverse were available, then the error $s - s^*$ could be converted to an approximate motor error $m - m^*$ so that modification of the motor command will improve performance. Since in this article we assume that no approximation to the plant is available, we instead must use the observed command-result pair (m, s) to learn the value of the inverse at s. We show that if the initial performance is such that s is sufficiently far from the desired s^*, then under certain circumstances, no amount of practice will ever be sufficient to learn the correct value of m^*.

2 Method

Let S be the set of all observable states of the environment, and let M be the set of all possible motor commands that can be applied. Let the environment to be controlled be represented by a memoryless plant P that maps motor commands $m \in M$ onto (sensory) effects or states $s \in S$ so that $P : m \to s$. The animal or robot transforms desired effects s^* into motor commands m through an approximate inverse dynamics mapping F for which $F : s^* \to m$. Thus, we can write the complete system as

$$s^* \to^F m \to^P s. \tag{2.1}$$

The goal of learning is to modify the approximate inverse dynamics F to produce the "correct" command m^* for which

$$s^* \to^F m^* \to^P s^*, \tag{2.2}$$

so that the exact desired effect s^* occurs. If this is achieved, then PF is the identity on the desired set of states $S^* \subset S$, and F will be a partial right inverse of P. In order to achieve this, the mapping F can be modified based on observed values (s^*, m, s). In particular, since $s = P(m)$, modification of F such that $m = F(s)$ will lead to a partial inverse for which $s = PF(s)$.

Under certain conditions, it can be shown that iteration of this procedure will lead to convergence on the desired states so that $s^* = PF(s^*)$ for all $s^* \in S^*$ (Sanger, 1994). However, this is not guaranteed under all conditions. In particular, learning can occur only for those states s that are actually observed. This set is given by $\{PF(s^*) : s^* \in S^*\}$, which we can write as the set $\hat{S} = PF(S^*)$. Successful learning may satisfy $s = PF(s)$ for all $s \in PF(S^*)$. But this is not the desired behavior. The desired behavior is that $s^* = PF(s^*)$ for all $s^* \in S^*$. Thus, the system may successfully learn the inverse dynamics but on an inappropriate or undesired set of states. Figure 1 illustrates the relationship between the various subsets $\hat{S}$, S^*, and M.

If this form of learning converges to a steady state, then for all $s^* \in S^*$, we have $PF(s^*) \in \hat{S}$, and for all $s \in \hat{S}$, we have $PF(s) = s \in \hat{S}$, so we note that

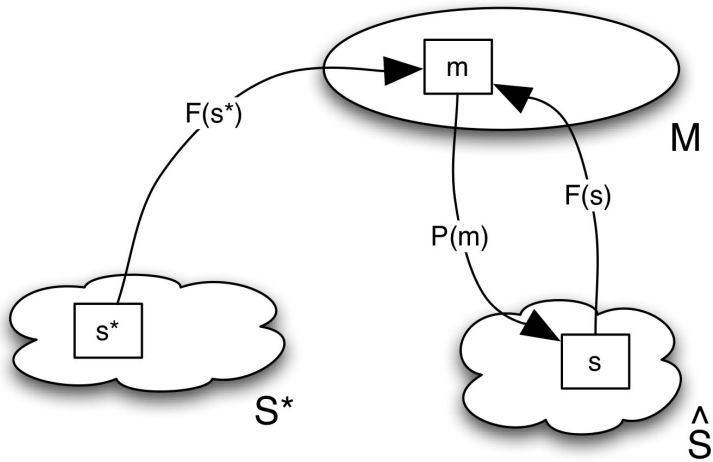

Figure 1: Illustration of the relationship between variables in the model. s^* is the desired state of the plant, and S^* is the set of all desired states. $m = F(s^*)$ is the motor command produced by the animal or robot's internal inverse dynamics model. $s = P(m)$ is the resulting state that occurs when the command m is applied to the unknown plant P. If the inverse dynamics model F trains successfully, then it will set $m = F(s)$ in order to map the observed state s into the motor command m that actually generated it.

the mapping PF is a projection onto the set $\hat{S}$. Points in S^* that are not in $\hat{S}$ will never be mapped to the correct motor commands.

The mapping F can be approximated to arbitrary accuracy as a weighted sum of a large but finite set of basis functions, so without loss of generality, we assume that F takes the form

$$F(s^*) = \sum_{i=1}^{n} w_i \phi_i(s^*), \tag{2.3}$$

where $\phi_i()$ is a set of fixed scalar basis functions, such as radial basis functions, Fourier components, or other reasonable basis for function space. (For a similar development, refer to Thoroughman & Shadmehr, 2000; Donchin & Shadmehr, 2003.) w_i is a set of vectors that specify the "weights" for combining the basis functions, and this computational structure has a straightforward neural network implementation (Sanger, 2003). We can rewrite equation 2.3 in vector notation as

$$F(s^*) = W\phi(s^*), \tag{2.4}$$

where W is a matrix of weights, and $\phi()$ is a column vector representing the basis function outputs. Modification of F due to learning is accomplished

by changing the weight matrix W. Although we have chosen a specific functional form for F, the following discussion will apply to any adaptive controller that can be approximated in this way.

We can thus perform the analysis for this specific functional form of the controller F. Given a desired state s^*, we have $s = P(m) = P(W\phi(s^*))$, and learning will modify the weights W based on the observed values s and m. If we assume that F is modified using an iterative learning rule, then after a single training pair (s, m), we have

$$F \to F + \Delta F = (W + \Delta W)\phi, \tag{2.5}$$

and the weights W are modified according to

$$\Delta W = r(W, m, s, s^*), \tag{2.6}$$

where $r()$ describes the learning rule, and there is a possible additional normalization step to maintain bounded weights.

If we define the error vector as $\delta_s = s - s^*$ and an energy function as $\mathcal{E} = \|\delta_s\|^2$, then gradient descent on $\mathcal{E}$ with respect to the weights W is given by

$$\Delta W = -\gamma \frac{\partial \mathcal{E}}{\partial \mathcal{W}} = -\gamma \left[\left(\frac{\partial P(m)}{\partial m} \right)^T \delta_s \right] \phi(s^*)^T, \tag{2.7}$$

where γ is the learning rate. Since we usually do not know $\partial P(m)/\partial m$ (otherwise, we would have a complete model of the plant and would not need to learn it), algorithms for motor learning often approximate this based on reasonable assumptions about the form of the plant. For example, approximation using a constant linear function $K \approx \partial P(m)/\partial m$ is used in feedback-error learning (Kawato & Gomi, 1992), while others have used backpropagation through a learned approximate forward model (Jordan & Rumelhart, 1992). However, if the plant is nonlinear, then a constant approximation may be valid only in a small region, and a learned forward model will be valid only near observed data points. If the approximation is sufficiently incorrect, then the sign of elements of $(\partial P(m)/\partial m)^T \delta_s$ may be incorrect, and modification by ΔW could actually increase the error $\mathcal{E}$.

In the following, we assume that the plant is nonlinear and unknown, so that no approximation to $\partial P(m)/\partial m$ is available. In this case, it is reasonable to take samples of observed plant inputs and outputs (m, s) and train on these, since no other information is available. In particular, there is no information about the desired inputs and outputs (m^*, s^*) or about the value or sign of the motor error $m - m^*$.

In this case, the gradient-descent learning rule that converges to the solution $m = F(s)$ is given by the least-mean-squared (LMS) algorithm applied to the observed data,

$$r(W, m, s) = \Delta W = (m - W\phi(s))\phi(s)^T, \tag{2.8}$$

so that the functional form of the update in this case depends only on the product of the basis function outputs $\phi(s)$ and the output error vector $\delta = m - W\phi(s)$ of the network.

After a single learning iteration, the output of the inverse dynamics network is modified so that

$$F(s^*) \rightarrow F(s^*) + \Delta F(s^*) = (W + \Delta W)\phi(s^*). \tag{2.9}$$

The network has converged when there are no further changes in the network in response to the desired inputs. There will be no change in the output $F(s^*)$ if $\Delta W\phi(s^*) = 0$. Substituting the gradient descent learning rule, we obtain

$$\Delta F(s^*) = \Delta W\phi(s^*) = \delta\phi(s)^T\phi(s^*) = 0. \tag{2.10}$$

Normally, we say that when $\Delta F(s^*) = 0$, then the network has converged, and for many applications of a gradient descent algorithm, this condition guarantees successful solution of the problem. However, in the case of motor learning in our formulation, there are two other possibilities. Since $\phi(s)^T\phi(s^*)$ is a scalar, equation 2.10 will be zero if and only if either

1. $\delta = 0$, or
2. $\phi(s)^T\phi(s^*) = 0$.

The first case may represent failure of learning, since $\delta = 0$ does not imply that $PF(s^*) = s^*$. In particular, $\delta = 0$ implies only that $F(s) = m$ or, equivalently, $PF(s) = s$. Thus, learning stops, but the error at s^* is nonzero. If we substitute $s = PF(s^*)$, we see that $PF(PF(s^*)) = PF(s^*)$, which means that this situation will occur whenever PF is a projection operator near s^*.

The second case represents a failure of learning due to the fact that if the basis function representations $\phi(s)$ and $\phi(s^*)$ are orthogonal, then training at s will not change the value of $F(s^*)$ despite continuing changes in the network output for other input values. Therefore, whether the learning eventually converges (to $\delta = 0$), the value of $F(s^*)$ will not have been modified, and errors will remain.

These two failure modes correspond to two different mechanisms that impede learning at s^*. In the first case, the network has already learned the mapping at s (the error δ is zero), and therefore it is unaware of the error at s^* and makes no further corrections. In the second case, despite continuing errors and modification of the network at s, there is no change in the output at s^*. In both cases, continued attempts at achieving the desired s^* will fail.

3 Simulations

The first type of failure is produced whenever the combination of the plant and the network forms a projection operator. This can occur if the input

m results in a plant output $s = P(m)$ such that one or more elements $\phi_i(s)$ are always zero, and no learning will occur since the LMS algorithm, 2.8, applied to a zero input causes no change in the weights connected to that input. Alternatively, if the network ignores an input such that $\partial\phi/\partial s_i = 0$ for some element s_i of s, then it will never be possible to control the behavior of s_i. The network may converge, but it will not achieve the desired values.

In order to demonstrate the first type of failure, we simulate a linear network F that attempts to invert a linear plant P with four inputs and four outputs. The plant is represented by a 4×4 random matrix with entries chosen from a uniform distribution between 0 and 1. The network is represented by a 4×4 matrix F with initially random entries, but with one of the initial eigenvalues set to zero so that F is degenerate with rank 3. Therefore, PF is a projection operator from R^4 to a three-dimensional hypersurface. Training occurs according to equation 2.8, with a learning rate of 0.01, and 1000 random examples drawn from a uniform distribution on the positive unit cube in R^4. The eigenvalues are not constrained during training. The time series of mean-squared command error $\|m - m^*\|$ and performance error $\|s - s^*\|$ is calculated and smoothed with a 20-point moving average. The process is repeated for 100 different randomly chosen plants and initial weights, and the mean and standard deviations of the command and performance errors are calculated for each point in time.

The results are displayed in Figure 2. Note that the command error converges to zero as expected with the LMS algorithm, but that the performance error remains nonzero. The network learns the mapping $s = PF(s)$ almost perfectly on a three-dimensional subspace, but the error for randomly chosen examples in the full four-dimensional space remains significant. Because the network training samples $s = PF(s^*)$ always lie in the subspace, the matrix F remains low rank despite the lack of any constraint forcing it to do so. Eventually, the error on the subspace becomes sufficiently small that learning effectively stops, yet significant errors in performance remain.

In order to show that the second type of failure can occur even for extremely simple problems, we simulate a network that attempts to compute the identity mapping from a single-dimensional s to a single-dimensional command m. For this simulation, a nonlinear network F has 10 narrow gaussian basis functions ϕ_i with standard deviation 0.05 and centers evenly spaced between 0.0 and 1.0. Here, we choose the plant to be the identity, so $P(m) = s = m$, and the correct solution is $F(s^*) = s^*$. Weights w_i are initialized to random values uniformly distributed between 0.45 and 0.75. The desired response is a value of $s^* = 0.2$, but during training, some local exploration is allowed so that training examples use s^* uniformly distributed between 0.1 and 0.3. For each example s^*, the values of $m = W\phi(s^*)$ and $s = P(m) = m$ are calculated. Gradient descent training of W on pairs (s, m) occurs for 1000 examples with a learning rate of 0.05 according to equation 2.8.

Figure 3A shows the resulting network after training. The dashed line is the desired function $F(s^*) = s^*$. The solid line is the learned value of

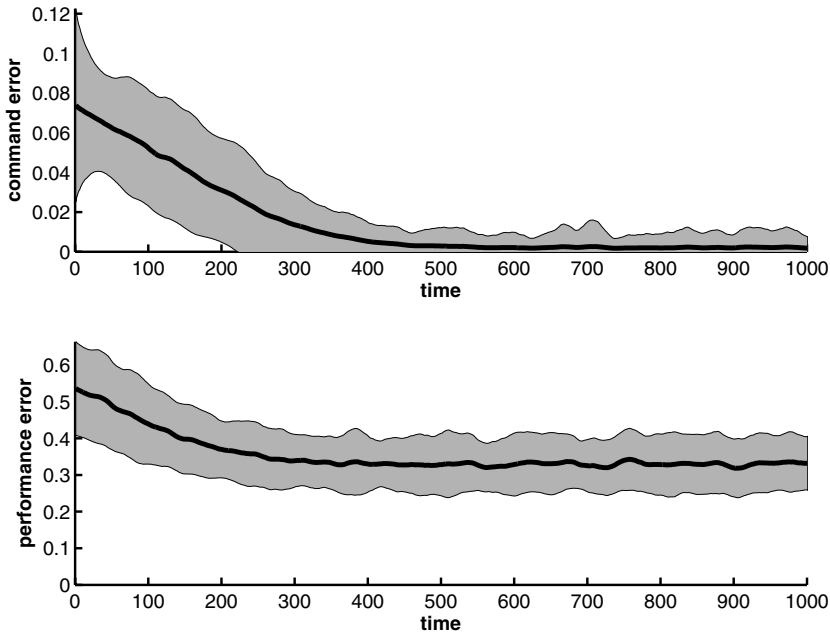

Figure 2: Simulation of the first type of failure of motor learning when PF is a projection operator. The top graph shows the command (motor) error $\|m - m^*\|$, and the bottom graph shows the performance (sensory) error $\|s - s^*\|$ (Mean-squared error in arbitrary units; time in units of iterations of the learning algorithm). Solid lines show the mean error averaged over 100 trials for 100 different random linear plants P, and the gray region shows $+/-$ one standard deviation.

$F(s) = \sum_i w_i \phi_i(s)$, and the thin lines at the bottom show a plot of each basis function multiplied by its corresponding weight $w_i \phi_i(s)$. Note that the result approximates the correct result only in the neighborhood of s, but not near s^*. Figure 3B shows the evolution of the weights during training. Note that only 3 of the 10 weights are modified by the training procedure.

This network demonstrates the particular case for which $\phi(s)^T \phi(s^*) = 0$. Because the basis functions are local, the representation for $\phi(s^*)$ excites only the first few basis functions, while the representation $\phi(s)$ excites only the last few. This situation occurred because the initial values of the weights for basis functions near s^* were between 0.45 and 0.75, and thus far from the correct value of approximately 0.2. During learning, the only observed values of s used for training were between 0.7 and 0.9 (since $s = PF(s^*) = \sum w_i \phi_i(s^*)$), and so the network learned the correct solution only in this neighborhood.

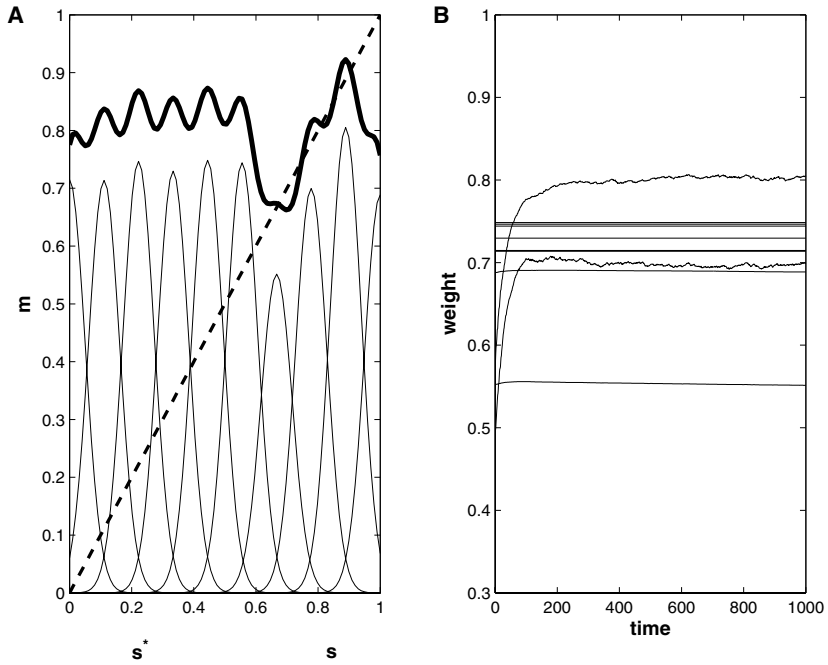

Figure 3: Simulation of the second type of failure of motor learning when $\phi(s^*)\phi(s) = 0$. See the explanation in the text. (A) The solid line is the learned network $\sum_i w_i\phi_i(s)$, the dashed line is the desired function, and basis functions $w_i\phi_i(s)$ are shown as well. The network approximates the desired function only in the neighborhood of s. (B) The evolution of the weights w_i during 1000 steps of training.

Although these are extremely simple examples, they are important because they demonstrate that the phenomenon of failure of motor learning can occur even for networks and plants of low complexity. Because of the larger number of variables, a more realistic plant with a high-dimensional control or state space will be even more likely to suffer this problem if the set of desired values of the states does not overlap with the set of observed values or if some modes of the system are not observed at all.

4 Discussion

The analysis shows that an animal or autonomous robot learning to control an unknown environment by observing the consequences of its own actions may fail to achieve its goal despite the use of a network learning algorithm with normally guaranteed convergence. There are two different

ways in which learning may fail, and in both cases, failure occurs because the network is not trained on the desired states of the system. The first failure mode occurs when the network converges on an undesired set and therefore stops learning. The second failure mode occurs when training on the observed states of the system does not cause changes in performance on the desired states. In both cases, the weights cease to change despite lack of convergence to the globally optimal solution. The learning algorithm therefore behaves as if it has achieved a "local minimum," but it should be noted that the reason for the failure of convergence may be due less to the nonlinear structure of the problem to be solved and more to the lack of availability of appropriate training examples. These two types of failure occur only if no approximate plant inverse is available and random exploration of the complete control space is not permitted.

Although we have performed the analysis for the particular case of a basis function network trained using gradient descent, many other network structures and training algorithms (but not all) will be susceptible to this type of failure of learning. We note that if an approximate plant inverse is available or can be learned, then algorithms that minimize the error $s - s^*$ may be able to converge even for large initial errors. If the weight matrix W maintains full rank and the plant P is controllable, then the combination PF may never become a projection operator, and thus the first type of failure may not occur. If the basis functions ϕ_i are broadly tuned, then orthogonality may be unlikely or impossible, so that the second type of failure may not occur. Therefore, although we show that two types of failure are possible, this will not necessarily occur for all plants or for learning with prior knowledge about the plant.

This type of failure can also be avoided if it is feasible to explore the set of possible controls thoroughly. If sufficient time is available to choose large numbers of examples of m randomly (and if doing so does not lead to injury), then an accurate plant inverse can be estimated throughout almost all of space, including the desired set S^*. The number of examples required will depend on the nature of the plant, the dimensionality of the control space, and the nature of the network approximating the inverse. If an approximately correct motor command can be produced, then full exploration may not be necessary. In particular, local exploration or small, random changes in the neighborhood of an approximately correct motor command may allow estimation of the plant inverse in a region that includes the desired performance.

In some cases, a forward model of the plant may be available, so that a complete exploration of state space can occur off-line. An approximate forward model can be learned by observation of the plant and used to estimate an inverse model (Jordan & Rumelhart, 1992). However, in the absence of prior information about the plant, a learned forward model will be accurate only in the neighborhood of observed pairs (m, s), and thus an estimated inverse will also be accurate only in such regions and will be

susceptible to the second type of motor learning failure. If an approximate example near the desired (m^*, s^*) has never been observed, then the forward and inverse models may not provide useful information about the region of desired performance.

Our development is very similar to analysis of generalization experiments (Thoroughman & Shadmehr, 2000; Shadmehr & Moussavi, 2000; Donchin & Shadmehr, 2003). When studying generalization, a particular movement s is trained until it can be performed accurately. Then it is asked for which s^* there has been a change in performance. The set of s^* with changed performance indicates the extent of the representation $\phi(s)$ and can be used to investigate the form of the basis functions ϕ_i. In contrast, when studying failure of learning, the goal is the movement s^*, and accurate performance on s is a by-product but not the purpose of training. If training at s does not generalize to s^*, then learning may fail.

Real-world examples of the first type of learning failure occur when a particular movement is not in the repertoire of known actions, or when the results of a particular movement cannot be detected by the controller (in both cases, the system behaves as a projection operator; it ignores or fails to produce certain elements of the desired performance). For example, a nonnative English speaker attempting to produce the *th* sound may never successfully produce this sound despite years of practice. All attempts at producing the foreign consonant are projected onto the set of consonants that are within the existing motor repertoire. Although the speaker is aware of the error and can produce an incorrect sound repeatedly on command, the sound never approximates the desired one. Another example of a projection operator is that of a nonnative student learning to speak Chinese but who is not yet able to distinguish between two spoken vowels. Since the two different vowels produce the same sensory response s, the speaker is unable to correct errors in order to learn to differentiate the two different vowels.

Real-world examples of the second type of learning failure occur when all components of a particular movement are achievable but do not occur properly in context, so that the desired goal has never been observed. In this case, the system is not a projection operator, but there are no relevant training examples. For example, a novice high diver who attempts to learn a complex dive may never be successful without coaching despite having all the necessary muscular ability to achieve the required forces. Repeated practice will produce a result that is so far from the desired result that no successful examples are available for learning. This is also a situation where random exploration is not a good idea.

How then is learning possible at all? In many cases, an approximation to the plant inverse may be available, particularly if the plant obeys uncomplicated Newtonian mechanics. Even when no inverse is available, if the network is able to achieve approximate performance, then s and s^* may be sufficiently close that there is some overlap in the set of excited basis functions. If $\phi(s)^T\phi(s^*) \neq 0$, then training at s will lead to changes in $F(s^*)$. As

we have shown previously, under reasonable smoothness assumptions on F and P, convergence can then be guaranteed (Sanger, 1994). The broader the tuning of the basis functions, the further apart s and s^* can be and still allow successful learning. Failure will be more likely when basis functions ϕ_i have low overlap for different values of s, as would occur for very narrow locally tuned functions. But this can also occur for broadly tuned functions if the patterns of activity $\phi(s)$ and $\phi(s^*)$ are orthogonal. For narrowly tuned basis functions, the product $\phi(s)\phi(s^*)$ is more likely to be zero particularly when there is a large number of basis functions (e.g., if s is in a high-dimensional space). We therefore expect to see this phenomenon more when complex trajectories need to be learned. This is exactly the situation in which generalization may be most difficult. Thus, there is a relationship between the ability to generalize (from $F(s)$ to $F(s^*)$) and the ability to learn reliably through iterative practice.

It is interesting to speculate on the role of coaching in this context. A coach could redirect attention to a particular sensory representation (perhaps a kinesthetic rather than a visual representation) for which the desired and actual movements do not lead to orthogonal basis function outputs. A coach could help limit attention to only a few important sensory variables, thereby reducing the dimensionality and reducing the likelihood of orthogonality of the basis functions. A coach could draw attention to errors that might have been ignored, and thereby ensure that learning does not converge prematurely. A coach could guide exploration from areas in which the inverse is known to regions in which it is not known, while maintaining reasonable margins of safety.

We hope that this formulation of limitations in motor learning from experience may help to further the understanding of why certain types of skill learning can fail in advanced skill learning and for children and adults with motor disorders. Certainly, there are tasks that are so complex that almost any conceivable learning algorithm will fail to converge to the desired solution. In this letter, we have shown that under certain circumstances, learning even a simple task can fail to converge. Understanding the failure of simple tasks may help to explain why some subjects with motor impairments are unable to perform tasks that others find easy. In the future, this understanding may lead to a mathematical description of the role of coaching that could be helpful for improving motor performance for children and adults with and without motor impairments.

Acknowledgments

I was supported in part by grant K23-NS41243-02 from the NINDS and in part by a faculty scholars award from Pfizer Pharmaceuticals. I also wish to thank the reviewers for their insightful comments and suggestions.

References

Atkeson, C. G. (1989). Learning arm kinematics and dynamics. *Ann. Rev. Neurosci.*, *12*, 157–183.

Criscimagna-Hemminger, S. E., Donchin, O., Gazzaniga, M. S. Shadmehr, R. (2003). Learned dynamics of reaching movements generalize from dominant to nondominant arm. *J. Neurophysiol.*, *89*(1), 168–176.

Dingwell, J. B., Mah, C. D. Mussa-Ivaldi, F. A. (2002). Manipulating objects with internal degrees of freedom: Evidence for model-based control. *J. Neurophysiol.*, *88*(1), 222–235.

Donchin, O. Shadmehr, R. (2003). Linking motor learning to function approximation: Learning in an unlearnable force field. In T. G. Dietterich, S. Becker Z. Ghahramani (Eds.), *Advances in neural information processing systems*, *14*. Cambridge, MA: MIT Press.

Flanders, M., Hondzinski, J. M., Soechting, J. F. Jackson, J. C. (2003). Using arm configuration to learn the effects of gyroscopes and other devices. *J. Neurophysiol.*, *89*(1), 450–459.

Gandolfo, F., Li, C., Benda, B. J., Schioppa, C. P. Bizzi, E. (2000). Cortical correlates of learning in monkeys adapting to a new dynamical environment. *Proc. Natl. Acad. Sci. U.S.A.*, *97*(5), 2259–2263.

Goodbody, S. J. Wolpert, D. M. (1998). Temporal and amplitude generalization in motor learning. *J. Neurophysiol.*, *79*(4), 1825–1838.

Jordan, M. I. Rumelhart, D. E. (1992). Forward models: Supervised learning with a distal teacher. *Cognitive Science*, *16*, 307–354.

Kawato, M., Furukawa, K. Suzuki, R. (1987). A hierarchical neural-network model for control and learning of voluntary movement. *Biol. Cybern.*, *57*(3), 169–185.

Kawato, M. Gomi, H. (1992). A computational model of four regions of the cerebellum based on feedback-error learning. *Biol. Cybern.*, *68*(2), 95–103.

Krakauer, J. W., Pine, Z. M., Ghilardi, M. F. Ghez, C. (2000). Learning of visuomotor transformations for vectorial planning of reaching trajectories. *J. Neurosci.*, *20*(23), 8916–8924.

Sanger, T. D. (1994). Neural network learning control of robot manipulators using gradually increasing task difficulty. *IEEE Trans. Robotics and Automation*, *10*(3), 323–333.

Sanger, T. D. (2003). Neural population codes. *Curr. Opin. Neurobiol.*, *13*(2), 238–249.

Scheidt, R. A., Reinkensmeyer, D. J., Conditt, M. A., Rymer, W. Z. Mussa-Ivaldi, F. A. (2000). Persistence of motor adaptation during constrained, multi-joint, arm movements. *J. Neurophysiol.*, *84*(2), 853–862.

Shadmehr, R. Moussavi, Z. M. (2000). Spatial generalization from learning dynamics of reaching movements. *J. Neurosci.*, *20*(20), 7807–7815.

Thoroughman, K. A. Shadmehr, R. (1999). Electromyographic correlates of learning an internal model of reaching movements. *J. Neurosci.*, *19*(19), 8573–8588.

Thoroughman, K. A. Shadmehr, R. (2000). Learning of action through adaptive combination of motor primitives. *Nature*, *407*(6805), 742–747.
Wang, T., Dordevic, G. S. Shadmehr, R. (2001). Learning the dynamics of reaching movements results in the modification of arm impedance and long-latency perturbation responses. *Biol. Cybern*, *85*(6), 437–448.

Received June 9, 2003; accepted February 24, 2004.