How to Modify a Neural Network Gradually Without Changing Its Input-Output Functionality

Article *in* Neural Computation · October 2009

DOI: 10.1162/neco.2009.05-08-781 · Source: PubMed

Christopher Dimattina
Florida Gulf Coast University

How to Modify a Neural Network Gradually Without Changing Its Input-Output Functionality

Christopher DiMattina
chris_dimattina@yahoo.com
Department of Neuroscience, Johns Hopkins University School of Medicine, Baltimore, MD 21205, U.S.A.

Kechen Zhang
kzhang4@jhmi.edu
Department of Biomedical Engineering, Johns Hopkins University School of Medicine, Baltimore, MD 21205, U.S.A.

It is generally unknown when distinct neural networks having different synaptic weights and thresholds implement identical input-output transformations. Determining the exact conditions for structurally distinct yet functionally equivalent networks may shed light on the theoretical constraints on how diverse neural circuits might develop and be maintained to serve identical functions. Such consideration also imposes practical limits on our ability to uniquely infer the structure of underlying neural circuits from stimulus-response measurements. We introduce a biologically inspired mathematical method for determining when the structure of a neural network can be perturbed gradually while preserving functionality. We show that for common three-layer networks with convergent and nondegenerate connection weights, this is possible only when the hidden unit gains are power functions, exponentials, or logarithmic functions, which are known to approximate the gains seen in some biological neurons. For practical applications, our numerical simulations with finite and noisy data show that continuous confounding of parameters due to network functional equivalence tends to occur approximately even when the gain function is not one of the aforementioned three types, suggesting that our analytical results are applicable to more general situations and may help identify a common source of parameter variability in neural network modeling.

1 Introduction

An open problem in theoretical neuroscience is to determine when it is possible for distinct neural network models, such as those having different synaptic weights and thresholds, to give rise to identical transformations of inputs to outputs. Such networks are called *functionally equivalent*, and this concept is best illustrated by concrete examples of hierarchical or

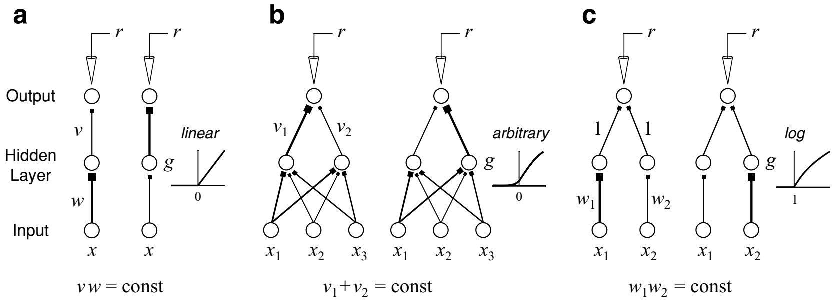


Figure 1: Functionally equivalent neural networks have diverse mechanisms by which the synaptic weights or other parameters can change gradually without affecting the overall input-output function. The strength of each weight is indicated by the line thickness. (**a**) In the simplest three-layer network with a hidden unit having threshold linear gain (inset), an increase of the output weight v can be compensated precisely by a decrease of the input weight w as long as their product (wv) is conserved, yielding an identical input-output function for all possible inputs. (**b**) The two hidden units with identical input weight vectors and an arbitrary gain behave as identical twins. As long as the sum of the output weights ($v_1 + v_2$) is conserved, an identical input-output function will result. (**c**) The two hidden units have gain functions that are logarithmic for inputs larger than 1. For this network, any values of the input weights with the same product ($w_1 w_2$) yield identical input-output relationship for all inputs above 1.

feedforward neural network models (see Figure 1). Here in each pair, the two networks differ in the strength of their synaptic connections, but they always respond identically to all input stimuli, making them functionally indistinguishable. The precise conditions under which the input-output transformation implemented by a neural network uniquely determines its structural parameters such as synaptic weights and thresholds is a fundamental theoretical problem that has not been solved completely. Previous studies have analyzed standard feedforward models or multilayer perceptrons (Rumelhart, Hinton, & McClelland, 1986) such as those in Figure 2, and have shown that under special assumptions on the hidden unit gain functions (input-output relationship for individual neurons), the overall input-output relationship of the network uniquely determines all network parameters, including synaptic weights and thresholds, up to permutation of neurons and regrouping of identical neurons (Albertini, Sontag, & Maillot, 1993; Chen, Lu, & Hecht-Nielsen, 1993; Fefferman, 1994; Kurkova & Kainen, 1994; Sussman, 1992). As a consequence, the entire structure of a neural network, including every parameter in the hidden layers, may in principle be recovered completely from stimulus-response data alone. However, these uniqueness results rely on highly restrictive assumptions

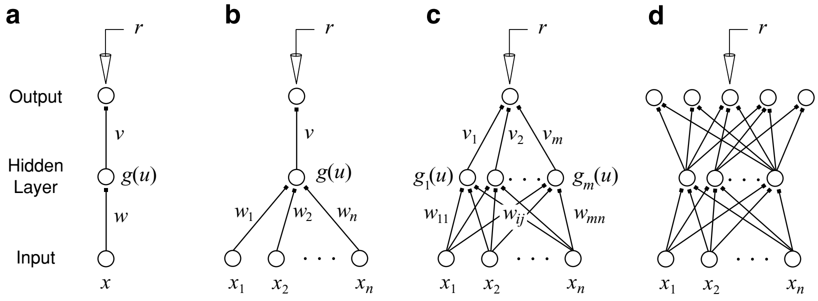


Figure 2: For the three-layer neural networks illustrated here, all possible mechanisms of functional equivalence can be identified under some general assumptions. (**a**) The simplest network has a single hidden unit with gain function g. (**b**) A network with a single hidden unit that has n independent inputs $x_1, \ldots, x_n$, weighted by the input weights $w_1, \ldots, w_n$. (**c**) A network with n inputs, m hidden units, and a single output unit. The gain functions of different hidden units may differ from one another. (**d**) In a network with multiple output units, a single output unit depends on the inputs in the same way as the network in **c**.

on the hidden unit gain functions, for instance, sigmoidal shape or asymptotic constancy, and hence do not accommodate all of the counterexamples shown in Figure 1. Furthermore, these uniqueness results assume that noiseless and potentially unlimited input-output data are available, thus limiting their applicability to modeling real neurophysiology data.

Determining when structurally distinct neural networks are functionally equivalent is a basic theoretical problem that has many practical implications. Hierarchical neural network models have been used in many previous studies to account for the stimulus-response properties of nonlinear neurons (Lau, Stanley, & Dan, 2002; Lehky, Sejnowski, & Desimone, 1992; Prenger, Wu, David, & Gallant, 2004; Wu, David, & Gallant, 2006; Zipser & Andersen, 1988). In principle, knowing the network architecture, gain functions of individual neurons, and the synaptic strengths and thresholds amounts to having a complete characterization of the stimulus-response properties of the neural network. The nonexistence of a unique relationship between a neural network's structure and its function would imply the impossibility of uniquely identifying the neural network from stimulus-response data alone. On the other hand, it might be biologically useful to have distinct networks that are capable of implementing an identical input-output function. For instance, if the function implemented by a neural network does not require a unique network structure, then when one synapse in a network is damaged, other synapses can be used to compensate for the damage and restore the original input-output function. Another related question is how diverse circuits in different brains could carry out identical functions, given that the synaptic connections in larger brains cannot be determined completely by genetic information so that the circuits in different individuals are unlikely to be identical. From this point of view, it is also of interest to understand the precise conditions for functionally equivalent networks.

In this article, we develop a biologically motivated approach to study the problem of neural network uniqueness. We ask when it is possible to slightly modify the parameters of a neural network while keeping its input-output relationship constant. This approach is sensible since changes in biological parameters like synaptic strength, resting potential, and threshold tend to be gradual and incremental. It allows us to derive a differential equation that specifies the precise conditions under which any neural model permits continuous modifications of its parameters while leaving functionality unchanged. While the equation holds true for all parametric models, we apply it to popular three-layer neural network models that have been widely used in various disciplines, including modeling neurophysiology data.

Our analysis leads to a complete classification of the solutions for admissible gain functions of three-layer networks with convergent and nondegenerate weights, given the constant input-output constraint. We show that one may continuously modify network parameters while preserving functionality only when the hidden unit gain functions are given by power functions, the exponentials, or the logarithmic functions (see Figures 1a and 1c for special cases and Figure 3 for more general cases). These special forms of gain functions may approximate the input-output properties of some biological neurons (Anderson, Lampl, Gillespie, & Ferster, 2000; Ermentrout, 1998; Gabbiani, Krapp, Koch, & Laurent, 2002; Smith, Nelson, & Du Lac, 2002; Stafstrom, Schwindt, & Crill, 1984). We will also extend the results to other types of gain functions, including the sigmoid and hyperbolic tangent commonly used in neural models, because they may approximate a power, exponential, or logarithmic function over limited ranges of inputs, especially for finite and noisy physiological data.

2 A Mathematical Condition for Functionally Equivalent Neural Networks

The output of a neural network is determined by both the input and the parameters that completely specify the network itself, including the input-output properties, or gain functions, of individual neurons and the synaptic connections between neurons. Following the biological motivation mentioned above, we derive a differential equation that tells us when it is possible to slightly modify the parameters of a neural network without altering its input-output relationship. The response r of an output neuron at the top of a feedforward network (see Figure 2) can be written as

$$r = f(x, \theta), \tag{2.1}$$

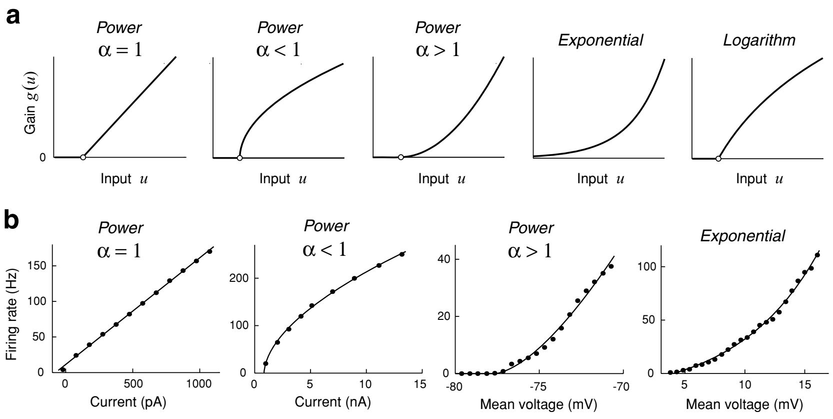


Figure 3: Only a few types of gain functions permit continuous functional equivalence in a generic three-layer network. (**a**) A three-layer network allows functional equivalence when the hidden unit gain functions are power, exponential, or logarithmic functions, with possible zero subthreshold regions. The three left-most panels show threshold power functions having linear ($\alpha = 1$), compressive ($\alpha < 1$), and expansive ($\alpha > 1$) shape. The two right-most panels show an exponential (left) and a threshold logarithmic function (right). (**b**) Experimental data from real neurons with input-output relationships that approximate the functional forms in panel **a**. Data were digitized from Smith et al. (2002) for the power law fit with $\alpha = 1$, Stafstrom et al. (1984) for $\alpha < 1$, and Anderson et al. (2000) for $\alpha > 1$, and from Gabbiani et al. (2002) for the exponential fit.

where $x = (x_1, \ldots, x_n)$ is the stimulus or the input to the n neurons at the bottom of the network, and the parameter set $\theta = (\theta_1, \ldots, \theta_k)$ includes the connection weights and the thresholds of all neurons in the network. The input-output relation specified by the function f depends implicitly on the choice of the gain functions for the individual neurons. Although only a single output neuron is considered here, there is no loss of generality because different output neurons in a feedforward network respond independently of one another (see Figure 2d).

Two networks with distinct parameter sets, θ and $\tilde{\theta}$, are called functionally equivalent if their outputs are always identical for all possible inputs, namely, $f(x, \theta) = f\left(x, \tilde{\theta}\right)$ for all input x. The case where the parameter sets θ and $\tilde{\theta}$ differ only slightly corresponds to biological networks that undergo parameter modifications in a gradual and incremental manner. If the parameters of a neural network can vary continuously without affecting the input-output relationship, we say these functionally equivalent networks form a *continuous equivalence class*. We also say that the network parameters that can vary together without affecting functionality are *continuously confounded.*

A necessary and sufficient condition for a neural network $f(x, \theta)$ to permit a continuous equivalence class of functionally equivalent networks is that there exists a set of coefficients $q_1(\theta), \ldots, q_k(\theta)$ such that

$$\sum_{i=1}^{k} q_i(\theta) \frac{\partial f(x, \theta)}{\partial \theta_i} = 0 \tag{2.2}$$

for all possible input x. The coefficients $q_i(\theta)$ may depend on the parameters θ but not on the input x, and they can be chosen freely, with the exception that the trivial choice where all $q_i(\theta) \equiv 0$ is not allowed. To see why the confounding equation 2.2 holds, fix the response r in equation 2.1, and consider an infinitesimal change of the parameters by taking the derivative with respect to a continuous variable t, which could be time or an arbitrary one-dimensional continuous index for equivalent networks. Since the response remains unchanged, we obtain by the chain rule that

$$\frac{\partial f(x, \theta)}{\partial t} = \sum_{i=1}^{k} \frac{\partial f(x, \theta)}{\partial \theta_i} \frac{d\theta_i}{dt} = 0, \tag{2.3}$$

which yields equation 2.2 by setting $q_i(\theta) = d\theta_i/dt$. Here each $q_i(\theta)$ is independent of x because $d\theta_i/dt$ describes how the parameter is changing and should not depend on the stimulus x. The condition in equation 2.2 is also sufficient because once equation 2.2 holds, we can solve the ordinary differential equations $d\theta_i/dt = q_i(\theta)$ $(i = 1, \ldots, k)$, and the solutions $\theta_1(t), \ldots, \theta_k(t)$ satisfy equation 2.3, meaning that these parameters can vary continuously with index t without affecting the output.

For a geometric interpretation of equation 2.2, rewrite it as a dot product:

$$q \cdot \nabla f = 0, \tag{2.4}$$

where $q = (q_1, \ldots, q_k)$ is a vector field in the network parameter space $\theta = (\theta_1, \ldots, \theta_k)$ and $\nabla f = (\partial f/\partial \theta_1, \cdots, \partial f/\partial \theta_k)$ is the gradient of the response with respect to these parameters. Suppose the vector field $q = (q_1, \ldots, q_k)$ satisfies equation 2.4 for all inputs x. Then we can perturb the parameters along the direction of vector q without affecting the value of the function f, because the perturbation is orthogonal to the gradient ∇f. The choice of q is not unique because there may exist multiple directions that are orthogonal to the gradient ∇f at each given location. Moreover, if $q = (q_1, \ldots, q_k)$ satisfies equation 2.4, so does $\phi q = (\phi q_1, \ldots, \phi q_k)$ for any scalar function ϕ of θ.

The condition for continuous confounding given by equation 2.2 is very general because it holds true for any smooth system that can be described by an input, an output, and a set of parameters. Although the basic equation 2.2 does not require any specific assumptions on the network architecture, in the rest of this article we will focus on the three-layer feedforward networks as in Figure 2 because this special case allows complete solutions with interesting biological interpretations.

3 The Simplest Neural Network That Requires a Power Gain Function for Parameter Confounding

We use the simplest network (see Figure 2a) to show how equation 2.2 can be applied to derive the form of the gain function. For simplicity, from now on we will ignore any nonlinear gain function for the output neuron, following a common practice in theoretical analysis, since the final nonlinearity can be readily added later.

We show that the simplest three-layer model as shown in Figure 2a allows continuous parameter confounding only when the gain function is a power function. Here the hidden unit has gain function g, input weight w, and output weight v. The response of the output neuron is

$$r = f\,(x, \theta) = vg\,(wx)\,, \tag{3.1}$$

where x is the input and $\theta = (\theta_1, \theta_2) = (v, w)$ is the parameter set. We assume $v \neq 0$ and $w \neq 0$ because otherwise the neuron would have either no input or no output. Equation 2.2 becomes

$$q_1 \frac{\partial f}{\partial v} + q_2 \frac{\partial f}{\partial w} = q_1 g\,(wx) + q_2 v x g'\,(wx) = 0, \tag{3.2}$$

where g' is the derivative of g, and coefficients q_1 and q_2 can be chosen freely as long as they do not vanish at the same time. Define a new variable $u = wx$ and rewrite 3.2 as

$$q_1 g\,(u) + q u g'\,(u) = 0, \tag{3.3}$$

with $q = q_2 v / w$. To solve equation 3.3, consider the following two cases, which are labeled as 3(i) and 3(ii), with "3" referring to section 3. Hereafter we will always include the section number for clear reference:

Case 3(i): $q = 0$. Since now $q_2 = q w / v = 0$, we must have $q_1 \neq 0$ because q_1 and q_2 are not allowed to vanish at the same time in the confounding equation, 3.3. Thus equation 3.3 is reduced to $q_1 g\,(u) = 0$, which yields the trivial solution $g\,(u) = 0$.

Case 3(ii): $q \neq 0$. To find the nontrivial solution $g(u) \neq 0$, rewrite equation 3.3 as $g'(u)/g(u) = \alpha/u$ with $\alpha = -q_1/q$ to obtain $\ln|g(u)| = \alpha \ln|u| + C$, or

$$g(u) = A|u|^{\alpha}, \tag{3.4}$$

where C and A are free parameters or integration constants. This solution includes the special case $\alpha = -q_1/q = 0$ or $g(u) = A$. The trivial solution $g(u) = 0$ may be accommodated formally as the special case of equation 3.4 with $A = 0$.

Thus, the general solution to differential equation 3.3 is a power function: $g(u) = A|u|^{\alpha}$, where A and α are free parameters. The special case $A = 0$ is the trivial solution $g(u) \equiv 0$, which is useful for accommodating zero firing rate for subthreshold stimuli. Since a biological gain function is typically monotonically increasing and the output firing rate cannot be negative, we require $\alpha > 0$, $A \geq 0$, and restrict the input to $u > 0$. Thus, the final solution is

$$g(u) = \begin{cases} Au^{\alpha} & u > 0 \\ 0 & u \leq 0 \end{cases}, \tag{3.5}$$

which is illustrated in Figure 3a (three left-most panels). The solution, equation 3.5, satisfies the original equation, 3.3, for all input u, provided that $\alpha > 1$. When $\alpha \leq 1$, however, the threshold $u = 0$ becomes a singular point at which $g(u)$ is not differentiable and the original equation, 3.3, breaks down. Because approximating biological gain functions by power functions sometimes requires an exponent $\alpha \leq 1$ (see Figure 3b, left-most two panels), we allow $\alpha \leq 1$ in solution 3.5, with the understanding that a singularity is present at the threshold.

In this simple example, the gain function must be a power function to allow continuous confounding of parameters. The linear example in Figure 1a is the special case with $\alpha = 1$. Intuitively, the confounding mechanism is that any increase (or decrease) in the input weight w can be compensated by a proper decrease (or increase) of the output weight v, so that output neuron would feel no difference. For a closer examination of the parameter confounding, we substitute solution 3.5 into equation 3.1 to obtain the final input-output relation: $r = A(vw^{\alpha})x^{\alpha}$ for $x > 0$, and $r = 0$ otherwise. Given a new set of parameters $(\tilde{v}, \tilde{w})$, as long as

$$\tilde{v}\tilde{w}^{\alpha} = vw^{\alpha}, \tag{3.6}$$

we have an identical input-output relationship. For an explicit demonstration using an index t, we set $\tilde{v} = vt^{-\alpha}$ and $\tilde{w} = wt$ so that equation 3.6 always holds as t varies continuously.

4 A Network with a Single Hidden Unit Requires Either Power or Exponential Gain Function

A new type of parameter confounding involving the exponential gain function occurs when a threshold or bias parameter w_0 is included for the hidden unit. In this section, we consider separately the case with a single input and the case with multiple inputs.

4.1 Single Hidden Unit with a Single Input and a Threshold Parameter. Consider a slightly more general model by adding a threshold parameter w_0 to the model considered in the previous section (see Figure 2a). We show that in this case, a new type of parameter confounding involving exponential gain functions occurs. Now the response to input x is

$$r = f(x, \theta) = vg(w_0 + wx), \tag{4.1}$$

where the parameter set is $\theta = (\theta_1, \theta_2, \theta_3) = (v, w_0, w)$, with $w \neq 0$ and $v \neq 0$ as before. Now equation 2.2 becomes

$$\begin{aligned} & q_1 \frac{\partial f}{\partial v} + q_2 \frac{\partial f}{\partial w_0} + q_3 \frac{\partial f}{\partial w} \\ & \quad = q_1 g(w_0 + wx) + (q_2 + q_3 x)\, vg'(w_0 + wx) = 0, \end{aligned} \tag{4.2}$$

which holds for all x for some fixed coefficients q_1, q_2, and q_3. Define a new variable $u = w_0 + wx$ and rewrite equation 4.2 as

$$q_1 g(u) + (a + bu)g'(u) = 0 \tag{4.3}$$

with

$$a = q_2 v - q_3 w_0 v/w, \quad b = q_3 v/w. \tag{4.4}$$

Equation 4.3 always has a trivial solution $g(u) = 0$. All other solutions are classified into the following three cases:

Case 4.1(i): $q_1 = 0$.

$$g(u) = A \tag{4.5}$$

is the solution, with A an arbitrary constant. In this case, equation 4.3 becomes $(a + bu)g'(u) = 0$, which implies either $g'(u) = 0$ or $a + bu = 0$. The former yields solution 4.5. The latter, $a + bu = 0$, holds for all u only if $a = b = 0$, which by equation 4.4 implies $q_1 = q_2 = q_3 = 0$, a situation that is not allowed in confounding equation 4.2.

Case 4.1(ii): $q_1 \neq 0$ and $b = 0$.

$$g(u) = Ae^{-(q_1/a)u} = Ae^{\alpha u} \tag{4.6}$$

is the general solution, with A a free parameter and $\alpha = -q_1/a$, assuming $a \neq 0$. If $a = 0$, equation 4.3 becomes $q_1 g(u) = 0$, which yields the trivial solution $g(u) = 0$.

Case 4.1(iii): $q_1 \neq 0$ and $b \neq 0$.

$$g(u) = A_1 |a + bu|^{-q_1/b} = A|u - B|^{\alpha} \tag{4.7}$$

is the general solution, where $\alpha = -q_1/b$, $B = -a/b$, and A_1 and A are free parameters.

Since the solutions in the three cases are obtained under logically mutually exclusive conditions for the coefficients, one cannot arbitrarily take pieces of different solutions and connect them together as a new solution. The only exception is that all three cases are compatible with the trivial solution $g(u) = 0$, which may be also regarded as a special case of the three solutions above with $A = 0$. The power solution in equation 4.8 is obtained with the additional requirements that the gain function should be nonnegative and monotonically increasing; that is, $u_1 < u_2$ should imply $0 \leq g(u_1) \leq g(u_2)$. Therefore, one can take only the increasing half of the power function and discard the decreasing half and replace it with the trivial solution. The exponential solution is always positive and cannot be connected continuously with the trivial solution.

In summary, the confounding equation 4.3 has two general solutions: one is a power function of the form

$$g(u) = \begin{cases} A(u - B)^{\alpha} & u > B \\ 0 & u \leq B \end{cases}, \tag{4.8}$$

and the other is an exponential function of the form

$$g(u) = Ae^{\alpha u}, \tag{4.9}$$

where A, α, and B are free parameters, and in both solutions we require $A \geq 0$ and $\alpha > 0$ in order to ensure that the gain function is monotonically increasing with the nonnegative firing rate.

The confounding mechanism for the exponential gain function in equation 4.9 is to compensate any increase (or decrease) of the output weight v by a decrease (or an increase) of the bias w_0 such that ve^{w_0} is conserved. For the power gain function in equation 4.8, the input weight w is confounded with the output weight v such that vw^{α} is conserved, just as in equation 3.6 and the bias w_0 also needs to be adjusted properly, as shown in section 6.

4.2 Single Hidden Unit with Multiple Inputs. The most general model with a single hidden unit is the one illustrated in Figure 2b with multiple inputs plus a threshold parameter. This model yields exactly the same three gain functions as in cases 4.1(i) to 4.1(iii) for a single input. To show this, consider the response by the neuron in Figure 2b to stimulus $x = (x_1, \ldots, x_n)$:

$$r = f(x, \theta) = vg\left(w_0 + \sum_{i=1}^{n} w_i x_i\right), \tag{4.10}$$

where the parameter set is $\theta = (v, w_0, w_1, \ldots, w_n)$ with weights $(w_1, \ldots, w_n)$ and threshold or bias w_0. We require $v \neq 0$ and the weights $w_1, \ldots, w_n$ not all equal to 0 to ensure that the hidden unit receives some input and has an output. Condition 2.2 becomes that

$$q\frac{\partial f}{\partial v} + \sum_{i=0}^{n} q_i \frac{\partial f}{\partial w_i} = 0 \tag{4.11}$$

holds for all input x for some fixed coefficients $q, q_0, q_1, \ldots, q_n$ that are independent of x. Substitution of equation 4.10 into 4.11 yields

$$qg(u) + \left(q_0 + \sum_{i=1}^{n} q_i x_i\right) vg'(u) = 0, \tag{4.12}$$

where $u = w_0 + \sum_{i=1}^{n} w_i x_i$ is the total input to the hidden unit.

To simplify equation 4.12, note that since it holds for arbitrary input $(x_1, \ldots, x_n)$, we can fix all inputs to 0 except for one, say, x_k, assuming $w_k \neq 0$. At least one nonzero weight $w_k \neq 0$ $(1 \leq k \leq n)$ exists, because otherwise the neuron would receive no input at all. We allow x_k to vary freely while fixing all other inputs to 0 ($x_i = 0$ for all $i \neq k$). Now we have $u = w_0 + w_k x_k$, and equation 4.12 is reduced to

$$qg(u) + (a + bu)g'(u) = 0, \tag{4.13}$$

the desired final equation, where $a = q_0 v - q_k w_0 v / w_k$ and

$$b = q_k v / w_k. \tag{4.14}$$

Since equation 4.13 is equivalent to equation 4.3 for neuron with a single input, we must have the same three general solutions as in equations 4.5 to 4.7.

Finally, we comment on how coefficients q_i should be chosen. If there is another nonzero weight $w_l \neq 0$ in addition to $w_k \neq 0$ $(l \neq k)$, then we can also vary x_l while setting all other $x_i = 0$ $(i \neq l)$. The same procedure above should lead to the same equation, 4.13, except that now parameters a and b have different expressions, with each subscript k replaced by the subscript l. Since changing the inputs should not affect the solution of the gain function, parameters a and b obtained in different ways should be identical. Thus, using equation 4.14, we should have $b = q_k v/w_k = q_l v/w_l$, which implies $q_k : q_l = w_k : w_l$. In general, for the original equation, 4.12, to hold for arbitrary inputs, the following two vectors should be proportional:

$$(q_1, q_2, \ldots, q_n) = D\,(w_1, w_2, \ldots, w_n)\,, \tag{4.15}$$

where $D = b/v$ is the proportionality constant. This relation holds even when some of the weights are zero because $w_i = 0$ implies $q_i = 0$. To see this, note that if $w_i = 0$, then input x_i would have no influence on the activity of the neuron, and thus in equation 4.12, we should have $q_i = 0$ to nullify the appearance of x_i.

5 A Network with Multiple Hidden Units Requires Power, Exponential, or Logarithmic Gain Functions

5.1 Overview of Main Results. In this section we consider the solutions for gain functions in the most general three-layer networks. Since different output neurons respond independently in a generic three-layer network (see Figure 2d), we need only focus on a single neuron in the output layer (see Figure 2c), with its output given by

$$r = f\,(x, \theta) = \sum_{i=1}^{m} v_i g_i \left(w_{i0} + \sum_{j=1}^{n} w_{ij} x_j \right), \tag{5.1}$$

where the gain functions g_i and the biases w_{i0} all may vary from neuron to neuron in the hidden layer. We assume each $v_i \neq 0$ because otherwise, the hidden unit would have no output. The input weights to each hidden unit are not allowed to vanish all at the same time because otherwise, the hidden unit would have no input. Applying the confounding equation 2.2 to this model yields

$$\sum_{i=1}^{m} q_i g_i\,(u_i) + \sum_{i=1}^{m} \left(q_{i0} + \sum_{j=1}^{n} q_{ij} x_j \right) v_i g_i'\,(u_i) = 0 \tag{5.2}$$

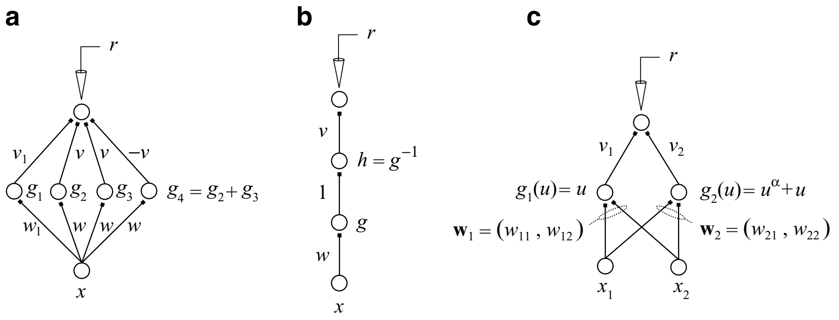


Figure 4: Examples of additional mechanisms of functional equivalence in neural networks. (**a**) Continuous parameter confounding occurs in a divergent network without involving the three gain functions in Table 1. Let g_1, g_2, and g_3 be arbitrary gain functions, and $g_4 = g_2 + g_3$ be the gain function for the rightmost hidden unit, whose contribution to the output unit is $-vg_4 = -v\,(g_2 + g_3)$. This is cancelled out by the contributions from the middle two hidden units: $v\,(g_2 + g_3)$. Thus, the total output $r = v_1 g_1\,(w_1 x)$ stays the same for any weights v and w. (**b**) Continuous parameter confounding occurs in a four-layer network with an arbitrary monotonically increasing gain function g in layer 2. The inverse function $h = g^{-1}$ is also monotonically increasing and serves as a gain function in layer 3. The output response $r = vh\,(1g\,(wx)) = vwx$ stays the same when vw is preserved. The confounded weights v and w are separated by a middle layer. (**c**) The gain functions in Table 1 were derived under the assumption that no other hidden units in the network have fully linear gain function. Relaxing this assumption leads to new solutions for continuous confounding, as shown here (see section 5.3).

with $u_i = w_{i0} + \sum_{j=1}^{n} w_{ij} x_j$, for some fixed coefficients q_i and q_{ij} that are independent of the input $(x_1, \ldots, x_n)$.

Solving the confounding equation 5.2 requires additional assumptions. First note that if the weight matrix w_{ij} is degenerate (see Figure 1b), or if the network is divergent $(m > n)$ with more hidden units than the number of inputs, then arbitrary gain functions can be used to generate continuous parameter confounding Figure 4a. That is, any given gain function can be assigned to a hidden unit as a part of a solution to equation 5.2.

In the following, we assume a convergent network with nondegenerate weight matrix and show next that only a few types of gain functions can permit continuous confounding. Here *convergent* means that the number of hidden units does not exceed the number of inputs $(m \le n)$. Under these assumptions, the activities of the hidden units can be controlled independently and arbitrarily by the stimulus input $(x_1, \ldots, x_n)$. In particular, we can use the input to alter only the activity of a single hidden unit i while keeping the activities of all other hidden units constant (DiMattina

Table 1: Classification of Gain Functions for Three-Layer Networks.

	$b = 0$	$b \neq 0$
$q = 0$	Linear solution: $g(u) = Au + C$	Logarithmic solution: $g(u) = A \ln \lvert u - B \rvert + C$
$q \neq 0$	Exponential solution: $g(u) = Ae^{\alpha u} + C$	Power solution: $g(u) = A \lvert u - B \rvert^{\alpha} + C$

Notes: The four types of gain functions are obtained by solving equation 5.3, with all subscripts omitted for simplicity. The classification is based on the coefficients b and q in the original equation.

& Zhang, 2008). In this way, equation 5.2 can be reduced to a form that explicitly involves only the gain function g_i of a single hidden unit i,

$$q_i g_i (u_i) + (a_i + b_i u_i) g_i' (u_i) + c_i = 0, \tag{5.3}$$

where the constants a_i, b_i, and c_i are given by equations 5.15 to 5.17. To obtain equation 5.3, we assume that no other hidden units in the network have a fully linear gain function, which is a trivial message pass-through. When this assumption is relaxed, the equation has an additional term linear in u_i and allows additional solutions, as will be discussed in section 5.3.

The solutions to differential equation 5.3 can be classified completely by Table 1, according to whether the parameters q_i and b_i vanish. In Table 1, the subscript i is omitted from parameters q_i and b_i for simplicity, and A, B, C, and α are free parameters that are described in section 5.2.2. Among the four types of gain functions (see Table 1 and Figure 3a), the power and exponential functions have been considered in the preceding sections, up to suitable scaling and shifting, whereas the linear function may be regarded as a special case of the power function. What is new is the logarithmic gain function, which allows confounding of the input weights of two separate neurons (see Figure 1c). Explicit mechanisms of parameter confounding for these gain functions are given in section 6.

5.2 Derivation of Confounding Equation 5.3 and Classification of Its Solutions. In this section we derive the confounding equation 5.3 for the generic three-layer network with multiple hidden units (see Figure 2c), and then obtain all four solutions as summarized in Table 1.

5.2.1 Derivation of Confounding Equation 5.3. For the model in equation 5.1, the parameter set is $\theta = (v_1, \ldots, v_m, w_{10}, \ldots, w_{m0}, w_{11}, \ldots, w_{mn})$. Condition 2.2 for the existence of continuous equivalence classes now becomes that for some fixed coefficients q_i and q_{ij} that are independent of the stimulus input, the equation

$$\sum_{i=1}^{m}\left(q_i\frac{\partial f}{\partial v_i}+\sum_{j=0}^{n}q_{ij}\frac{\partial f}{\partial w_{ij}}\right)=0 \tag{5.4}$$

holds for all stimulus input $(x_1,\ldots,x_n)$. Substitution of equation 5.1 into 5.4 gives

$$\sum_{i=1}^{m}\left(q_i g_i\left(u_i\right)+\left(q_{i0}+\sum_{j=1}^{n}q_{ij}x_j\right)v_i g_i'\left(u_i\right)\right)=0 \tag{5.5}$$

where

$$u_i=w_{i0}+\sum_{j=1}^{n}w_{ij}x_j \tag{5.6}$$

is the total input to hidden unit i. In vector matrix form, we can rewrite equation 5.6 as

$$\mathbf{u}=\mathbf{w}_0+\mathbf{W}\mathbf{x}, \tag{5.7}$$

where $\mathbf{u}=(u_1,\ldots,u_m)^{\mathrm{T}}$, $\mathbf{w}_0=(w_{10},\ldots,w_{m0})^{\mathrm{T}}$, $\mathbf{x}=(x_1,\ldots,x_n)^{\mathrm{T}}$, and $\mathbf{W}$ is the $m\times n$ matrix with entry w_{ij}.

We require that the activities of the hidden units can be independently controlled by the input. That is, given any desired hidden layer activity pattern $\mathbf{u}$, there is always a suitable input $\mathbf{x}$ that can generate it according to equation 5.7. This requires equation 5.7 to have a solution $\mathbf{x}$ for any given $\mathbf{u}$. The following two conditions ensure that the desired solution always exists: (1) the number of hidden units does not exceed the number of inputs, or $m\le n$, and (2) the weight matrix $\mathbf{W}$ is nondegenerate, or $\operatorname{rank}\mathbf{W}=m$ (DiMattina & Zhang, 2008). To obtain the three types of gain functions in Table 1 and Figure 3, one cannot in general relax the requirement for network convergence ($m\le n$), or the requirement for nondegenerate weight matrix $\mathbf{W}$. One may use arbitrary gain functions to achieve continuous confounding when the network is divergent with $m>n$ (see Figure 4a) or when $\mathbf{W}$ is degenerate (see Figure 1b).

Under the two assumptions discussed above, the desired input always exists:

$$\mathbf{x}=\mathbf{W}^{\dagger}\left(\mathbf{u}-\mathbf{w}_0\right), \tag{5.8}$$

where $\mathbf{W}^{\dagger}$ is Moore-Penrose generalized inverse, although the solution in general is not unique. Rewrite equation 5.8 as

$$x_j = \sum_{k=1}^{m} w_{jk}^{\dagger} \left(u_k - w_{k0}\right), \tag{5.9}$$

and then substitute it into equation 5.2 to obtain

$$\sum_{i=1}^{m} \left(q_i g_i\left(u_i\right) + \left(q_{i0} + \sum_{k=1}^{m} D_{ik}\left(u_k - w_{k0}\right) \right) v_i g_i'\left(u_i\right) \right) = 0, \tag{5.10}$$

where

$$D_{ik} = \sum_{j=1}^{n} q_{ij} w_{jk}^{\dagger} \tag{5.11}$$

is introduced for convenience. An equivalent matrix form of equation 5.11 reads

$$\mathbf{D} = \mathbf{Q}\mathbf{W}^{\dagger}, \tag{5.12}$$

where $\mathbf{D} = \left\{D_{ij}\right\}$ is an $m \times m$ matrix, and $\mathbf{Q} = \left\{q_{ij}\right\}$ is an $m \times n$ matrix to be solved. The general solution to equation 5.12 can be written as

$$\mathbf{Q} = \mathbf{D}\mathbf{W} + \mathbf{Z}\left(\mathbf{I}_n - \mathbf{W}^{\dagger}\mathbf{W}\right), \tag{5.13}$$

where $\mathbf{I}_n$ denotes the $n \times n$ identity matrix, and $\mathbf{Z}$ is an arbitrary $m \times n$ matrix (Ben-Israel & Greville, 2003). To see why this solution holds, first note that $\mathbf{Q} = \mathbf{D}\mathbf{W}$ is a special solution to equation 5.12 because $\mathbf{W}$ is a full rank $m \times n$ matrix with $m \leq n$ so that $\mathbf{W}\mathbf{W}^{\dagger} = \mathbf{I}_m$. A general solution to equation 5.12 should allow an arbitrary additional term in the null space of $\mathbf{W}^{\dagger}$. The second term in solution 5.13, $\mathbf{Z}\left(\mathbf{I}_n - \mathbf{W}^{\dagger}\mathbf{W}\right)$, is indeed in this null space because right-multiplying it by $\mathbf{W}^{\dagger}$ results in zero, by the identity $\mathbf{W}^{\dagger}\mathbf{W}\mathbf{W}^{\dagger} = \mathbf{W}^{\dagger}$. We can always set $\mathbf{Z} = \mathbf{0}$ to choose the special solution $\mathbf{Q} = \mathbf{D}\mathbf{W}$, which may be regarded as the multiple hidden units counterpart to the proportionality relation 4.15 for a single hidden unit.

To simplify equation 5.10, note that the activity of each hidden unit can be independently controlled by the stimulus. In particular, we can allow the activity of a single hidden unit, say, u_l, to vary freely while keeping the activities of all other hidden units constant, namely, $u_i \equiv U_i$ $(i \neq l)$. This trick reduces equation 5.10 to a form that is about the single hidden unit l:

$$q_l g_l(u_l) + (a_l + b_l u_l) g_l'(u_l) + c_l + d_l u_l = 0, \tag{5.14}$$

where all the coefficients as given below are independent of variable u_l:

$$a_l = \left(q_{l0} + \sum_{k=1}^{m} D_{lk} w_{k0} + \sum_{k \neq l} D_{lk} U_k \right) v_l, \tag{5.15}$$

$$b_l = D_{ll} v_l, \tag{5.16}$$

$$c_l = \sum_{i \neq l} q_i g_i(U_i) + \sum_{i \neq l} \left(q_{i0} - \sum_{k=1}^{m} D_{ik} w_{k0} + \sum_{k \neq l} D_{ik} U_k \right) v_i g_i'(U_i), \tag{5.17}$$

$$d_l = \sum_{i \neq l} D_{il} v_i g_i'(U_i). \tag{5.18}$$

Since equation 5.14 and the coefficient d_l in equation 5.18 should not depend on the level of the fixed activity U_k, which can be chosen arbitrarily, we must have $d_l = 0$ as long as none of the gain functions g_i is linear. To see this, we first show that in equation 5.18, we must have $D_{il} = 0$ for each $i \neq l$. Seeking a contradiction, suppose $D_{il} \neq 0$. Then we could vary the hold level U_i freely so that $g_i'(U_i)$ would also vary with U_i because g_i is nonlinear. Thus, d_l would depend on the hold level U_i since we always require $v_i \neq 0$. This contradiction implies that we must have $D_{il} = 0$ for all off-diagonal elements $i \neq l$. As a consequence, equation 5.18 is now reduced to $d_l = 0$. Thus we obtain the final confounding equation,

$$q_l g_l(u_l) + (a_l + b_l u_l) g_l'(u_l) + c_l = 0, \tag{5.19}$$

for any hidden unit $l = 1, \ldots, m$. This is the same as equation 5.3 in the preceding section. If linear gain function is allowed for some hidden units, we may have $d_l \neq 0$.

5.2.2 Classification of the Solutions to Confounding Equation 5.3. To solve equation 5.3 or 5.19, first omit all the subscripts l for simplicity, and then consider the following four mutually exclusive cases:

Case 5.2(i): $q = 0$ and $b = 0$. Equation 5.19 becomes $ag'(u) + c = 0$, which yields the linear solution

$$g(u) = -(c/a)u + C = Au + C, \tag{5.20}$$

where $A = -c/a$ and C is an arbitrary constant. Here $a \neq 0$ is assumed because otherwise the equation would be reduced to the degenerate form $c = 0$.

Case 5.2(ii): $q = 0$ and $b \neq 0$. Equation 5.19 becomes $(a + bu)\, g'(u) + c = 0$, which has as the general solution the logarithmic function

$$g(u) = -(c/b) \ln|a + bu| + C_1 = A \ln|u - B| + C, \tag{5.21}$$

where $A = -c/b$, $B = -a/b$, and C_1 and C are arbitrary constants.

Case 5.2(iii): $q \neq 0$ and $b = 0$. Equation 5.19 becomes $qg(u) + ag'(u) + c = 0$, which has as the general solution the exponential function

$$g(u) = Ae^{-(q/a)u} - c/q = Ae^{\alpha u} + C, \tag{5.22}$$

where $C = -c/q$ and A are arbitrary constants. Here $a \neq 0$ is assumed. If $a = 0$, equation 5.19 becomes $g(u) = -c/q$, which may be regarded formally as a special case of equation 5.22 with $A = 0$.

Case 5.2(iv): $q \neq 0$ and $b \neq 0$. The general solution to equation 5.19 is a power function,

$$g(u) = A_1 |a + bu|^{-q/b} - c/q = A|u - B|^{\alpha} + C, \tag{5.23}$$

where $\alpha = -q/b$, $B = -a/b$, $C = -c/q$, and A_1 and A are arbitrary constants. Although the linear solution in equation 5.20 is derived under a different condition, it may be regarded formally as a special case of the power function 5.23.

A classification of all the solutions is summarized in Table 1. Compared with the three solutions for a single hidden unit in section 4, the logarithmic gain function, equation 5.21, is a completely new solution, because its parameter confounding requires two or more hidden units (see Figure 1c). Another difference is the additional constant term, $-c/q$ in equations 5.22 and 5.23 for both the exponential and the power gain functions. Since c_l as defined by equation 5.17 involves the gain functions of other hidden units, it reflects the interaction between different hidden units. When all the confounded parameters involve a single hidden unit only and have nothing to do with the other hidden units, all terms involving c_l vanish, just as in the case for a single hidden unit.

5.3 The Special Situation with Fully Linear Hidden Units. In this section we have assumed that none of the hidden units has a fully linear gain function, or $g(u) = c_1 u + c_2$ for all u. If such a linear hidden unit is allowed, continuous confounding of parameters becomes possible for additional new types of gain functions. We first illustrate this situation with an explicit example and then derive the general solutions.

In the example in Figure 4c, the gain function $g_1(u) = u$ is fully linear, whereas the gain function $g_2(u) = u^\alpha + u$ differs from the three types considered above, where $\alpha > 0$ is a real number but not an integer. The response to input $\mathbf{x} = (x_1, x_2)^\mathrm{T}$ is

$$\begin{aligned} r &= v_1\left(\mathbf{w}_1^\mathrm{T}\mathbf{x}\right) + v_2\left(\left(\mathbf{w}_2^\mathrm{T}\mathbf{x}\right)^\alpha + \left(\mathbf{w}_2^\mathrm{T}\mathbf{x}\right)\right) \\ &= v_2\left(\mathbf{w}_2^\mathrm{T}\mathbf{x}\right)^\alpha + \left(v_1\mathbf{w}_1 + v_2\mathbf{w}_2\right)^\mathrm{T}\mathbf{x}. \end{aligned} \tag{5.24}$$

Let parameters v_i and $\mathbf{w}_i$ be replaced by the new values $\tilde{v}_i$ and $\tilde{\mathbf{w}}_i$ $(i = 1, 2)$. In order for the response to stay unchanged for all input $\mathbf{x}$, both the term with the exponent α and the linear term in equation 5.24 should stay unchanged:

$$\tilde{v}_2\left(\tilde{\mathbf{w}}_2^\mathrm{T}\mathbf{x}\right)^\alpha = v_2\left(\mathbf{w}_2^\mathrm{T}\mathbf{x}\right)^\alpha, \tag{5.25}$$

$$\tilde{v}_1\tilde{\mathbf{w}}_1 + \tilde{v}_2\tilde{\mathbf{w}}_2 = v_1\mathbf{w}_1 + v_2\mathbf{w}_2. \tag{5.26}$$

These equations are satisfied by the following construction with a continuous index t:

$$\tilde{\mathbf{w}}_2 = t\mathbf{w}_2, \tag{5.27}$$

$$\tilde{v}_2 = t^{-\alpha}v_2, \tag{5.28}$$

$$\tilde{v}_1 = v_1, \tag{5.29}$$

$$\tilde{\mathbf{w}}_1 = \mathbf{w}_1 + \left(1 - t^{1-\alpha}\right)(v_2/v_1)\,\mathbf{w}_2. \tag{5.30}$$

Equations 5.27 and 5.28 ensure that equation 5.25 holds for all inputs $\mathbf{x}$. The new parameter $\tilde{v}_1$ can be picked arbitrarily, and equation 5.29 is a simple choice. Equation 5.30 is obtained by solving for $\tilde{\mathbf{w}}_1$ from equation 5.26 using equations 5.27 to 5.29. Thus both equations 5.25 and 5.26 are satisfied, creating continuous parameters confounding, with $t = 1$ corresponding to the original parameters.

When linear hidden units are allowed, we need to consider the confounding equation 5.14 with $d_l \neq 0$. We omit the subscript l for simplicity and write equation 5.14 as

$$qg(u) + (a + bu)\,g'(u) + c + du = 0. \tag{5.31}$$

The solutions to this equation are classified into five mutually exclusive cases according to the values of the parameters. In all following solutions, C_0 is a free parameter:

Case 5.3(i): $q = 0$ and $b = 0$. The solution is

$$g(u) = -(d/2a)\,u^2 - (c/a)\,u + C_0, \tag{5.32}$$

where $a \neq 0$ is assumed because if $a = 0$, equation 5.31 becomes degenerate: $c + du = 0$.

Case 5.3(ii): $q = 0$ and $b \neq 0$. The solution is

$$g(u) = \left((ad - bc)/b^2\right)\ln(a + bu) - (d/b)\,u + C_0. \tag{5.33}$$

Case 5.3(iii): $q \neq 0$ and $b = 0$. The solution is

$$g(u) = C_0 e^{-(q/a)u} - (d/q)\,u + (ad - qc)/q^2. \tag{5.34}$$

Case 5.3(iv): $q \neq 0$, $b \neq 0$ and $q + b \neq 0$. The solution is

$$g(u) = C_0\,|a + bu|^{-q/b} + d\,(u - a)/q\,(q + b) + c. \tag{5.35}$$

Case 5.3(v): $q \neq 0$, $b \neq 0$ and $q + b = 0$. The solution is

$$g(u) = -\left(d/b^2\right)(a + bu)\left(C_0 + \ln(a + bu)\right) + (bc - ad)/b^2. \tag{5.36}$$

The first four cases correspond to the four cases considered in section 5.2.2, and the only difference is the additional linear term in u, which disappears when $d = 0$. The power solution with a linear term explains why the example in Figure 4c considered earlier in this section works. The case 5.3(v) is a new form of gain function: a product of logarithm with a linear function. This new form occurs only when $d \neq 0$.

5.4 Comparison with Biological Gain Functions. Several experimental studies investigating the relationship between the output firing rate and either the current injected into a neuron or the mean membrane voltage found neurons with gain functions that can be approximated by power law or exponential functions (Anderson et al., 2000; Ermentrout, 1998; Gabbiani et al., 2002; Smith et al., 2002; Stafstrom et al., 1984). A few examples of such gain functions taken from real neuronal data are shown in Figure 3b. There is also evidence from studies in locust visual neurons suggesting logarithmic transformation of sensory variables in dendrites, as part of a circuit that implements multiplication by adding logarithmically scaled inputs in the dendrites of a neuron, followed by an exponential transformation into spiking rate (Gabbiani et al., 2002).

By visual appearance (see Figure 3), it may seem that the exponential function somewhat resembles the power function with exponent $\alpha > 1$ while the logarithmic function resembles the power function with $\alpha < 1$.

This is not an accident because both the exponential and log functions can be derived as limiting cases of power functions (see appendix A).

Although the data shown in Figure 3 are generally consistent with the theory of continuous parameter confounding, the agreement does not prove that the confounded parameters can indeed compensate each other in a biological network as predicted by the theory. This is because the data in Figure 3 were obtained from a single neuron, while our confounding theory always involves structural parameters such as weights and thresholds coming from two or more neurons that form a network. To further examine this issue, one needs to measure network parameters simultaneously from multiple neurons within the same network, under at least two sets of conditions where the parameter values are allowed to vary while the input-output function of the network is kept the same.

6 Explicit Indexing of Continuous Confounding of Parameters

For simple power law confounding between input and output weights (see equation 3.6 and Figures 1a and 5), it is intuitively clear how a change of one parameter can be compensated precisely by suitable change of another parameter. For more complex forms of parameter confounding, it is not always obvious how different parameters can change simultaneously while preserving network functionality. In this section we demonstrate explicitly how different parameters are confounded for various gain functions derived in preceding sections. We use an arbitrary one-dimensional continuous variable t as an explicit index, with the understanding that the choice is generally not unique.

6.1 Index for Continuous Parameter Confounding for a Single Hidden Unit. The most general situation here is a single hidden unit receiving multiple inputs (see Figure 2b), and the gain function has three types of solutions given by equations 4.5 to 4.7 as described in cases 4.1(i), 4.1(ii), and 4.1(iii). The response in equation 4.10 can be written as

$$r = vg\left(w_0 + \sum_{i=1}^{n} w_i x_i\right) = vg\left(w_0 + \mathbf{w}^{\mathrm{T}}\mathbf{x}\right), \tag{6.1}$$

where $\mathbf{w} = (w_1, \ldots, w_n)^{\mathrm{T}}$ is the weight vector and $\mathbf{x} = (x_1, \ldots, x_n)^{\mathrm{T}}$ is the input vector.

In case 4.1(i), the gain function in equation 4.5 is a constant, so that the weight vector $\mathbf{w}$ and the threshold w_0 can be changed arbitrarily without affecting the output.

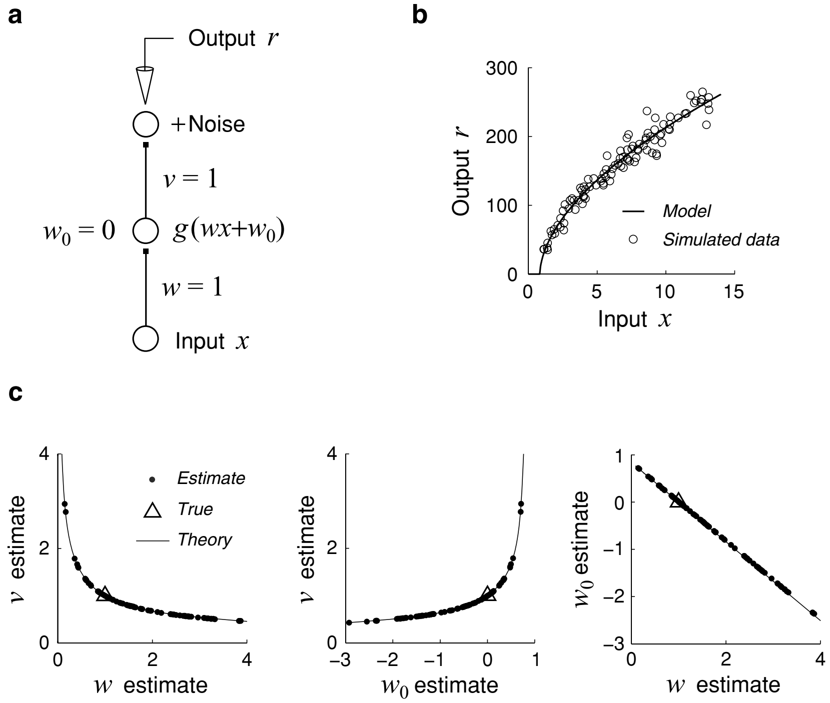


Figure 5: Continuous parameter confounding in a simple network with power gain function. (**a**) The network has input weight w, output weight v, and bias w_0. The hidden unit gain g is the same power function as illustrated in Figure 3b ($\alpha < 1$) and is given by equation 4.8 with the parameters $A = 61.1$, $B = 0.837$, and $\alpha = 0.564$. (**b**) The input-output data (circles) were generated by adding Poisson noise to the expected responses of the model (black curve) to random inputs. (**c**) Parameter estimates (dots) attained from the data set in **b** via least squares. Different starting points for the optimization led to different estimates, all of which lie along the power law confounding curves containing the true parameters (triangles).

In case 4.1(ii), the gain function is the exponential function in equation 4.6, and the output in equation 6.1 becomes

$$r = vAe^{\alpha\left(w_0+\mathbf{w}^{\mathrm{T}}\mathbf{x}\right)} = A\left(ve^{\alpha w_0}\right)e^{\alpha\mathbf{w}^{\mathrm{T}}\mathbf{x}}. \tag{6.2}$$

To maintain the same input-output relation with a new set of parameters $\tilde{\mathbf{w}}$, $\tilde{v}$, and $\tilde{w}_0$, we need to have $\left(\tilde{v}e^{\alpha\tilde{w}_0}\right)e^{\alpha\tilde{\mathbf{w}}^{\mathrm{T}}\mathbf{x}} = \left(ve^{\alpha w_0}\right)e^{\alpha\mathbf{w}^{\mathrm{T}}\mathbf{x}}$ for all $\mathbf{x}$. This requires $\tilde{\mathbf{w}} = \mathbf{w}$ and

$$\tilde{v}e^{\alpha\tilde{w}_0} = ve^{\alpha w_0}. \tag{6.3}$$

An explicit construction that satisfies equation 6.3 with a one-dimensional index t is

$$\tilde{w}_0 = w_0 + t, \tag{6.4}$$

$$\tilde{v} = ve^{-\alpha t}. \tag{6.5}$$

Now equation 6.3 holds for arbitrary t, which can change continuously, with $t = 0$ corresponding to the original values of the parameters. The way to introduce the index is not unique, and another example that will become convenient later is

$$\tilde{v} = v + t, \tag{6.6}$$

$$\tilde{w}_0 = w_0 + \alpha^{-1} \ln\left(v/(v+t)\right). \tag{6.7}$$

An intuitive explanation of the parameter confounding for exponential function is that an increase (or decrease) of the output weight v can be compensated by a proper decrease (or increase) of the threshold parameter w_0.

In case 4.1(iii), the gain function is the power function in equation 4.7, and the output in equation 6.1 becomes

$$r = vA\left|\mathbf{w}^{\mathrm{T}}\mathbf{x} + w_0 - B\right|^{\alpha}. \tag{6.8}$$

Keeping the same input-output relation requires new parameters $\tilde{v}$, $\tilde{\mathbf{w}}$, and $\tilde{w}_0$ such that

$$\tilde{v}\left|\tilde{\mathbf{w}}^{\mathrm{T}}\mathbf{x} + \tilde{w}_0 - B\right|^{\alpha} = v\left|\mathbf{w}^{\mathrm{T}}\mathbf{x} + w_0 - B\right|^{\alpha} \tag{6.9}$$

holds for all input $\mathbf{x}$. The following construction with index t satisfies equation 6.9:

$$\tilde{\mathbf{w}} = \mathbf{w}t, \tag{6.10}$$

$$\tilde{w}_0 = B + (w_0 - B)\,t, \tag{6.11}$$

$$\tilde{v} = vt^{-\alpha}. \tag{6.12}$$

When $t = 1$, the new parameters become identical to their original values. It follows from equations 6.10 to 6.12 that we always have

$$\tilde{\mathbf{w}}/(\tilde{w}_0 - B) = \mathbf{w}/(w_0 - B), \tag{6.13}$$

$$\tilde{v}\,\|\tilde{\mathbf{w}}\|^{\alpha} = v\,\|\mathbf{w}\|^{\alpha}, \tag{6.14}$$

$$\tilde{v}\,(\tilde{w}_0 - B)^{\alpha} = v\,(w_0 - B)^{\alpha}. \tag{6.15}$$

Here equation 6.14 becomes $\tilde{v}\tilde{w}^{\alpha} = vw^{\alpha}$, the same as equation 3.6, if we write the Euclidean norm of the weight vector $\mathbf{w} = (w_1, \ldots, w_n)^{\mathrm{T}}$ as $w = \|\mathbf{w}\| = \sqrt{w_1^2 + \cdots + w_n^2}$, and similarly, $\tilde{w} = \|\tilde{\mathbf{w}}\|$. Another equivalent index formulation that will be used later is

$$\tilde{v} = v + t, \tag{6.16}$$

$$\tilde{\mathbf{w}} = \big(v/(v + t)\big)^{1/\alpha}\, \mathbf{w}, \tag{6.17}$$

$$\tilde{w}_0 = B + \big(v/(v + t)\big)^{1/\alpha} (w_0 - B)\,. \tag{6.18}$$

An intuitive explanation of the parameter confounding for power function is that an increase (or decrease) of the output weight v can be compensated by a proper decrease (or increase) of the norm of the input weight vector $\mathbf{w}$. At the same time, the threshold parameter w_0 also needs to be adjusted accordingly.

6.2 Index for Continuous Parameter Confounding with Multiple Hidden Units. In this section, we demonstrate explicitly how the four types of gain functions in Table 1 can give rise to continuous confounding of parameters through interaction of different hidden units. The output response in equation 5.1 can be written as

$$r = \sum_{i=1}^{m} v_i g_i \left(w_{i0} + \sum_{j=1}^{n} w_{ij} x_j \right) = \sum_{i=1}^{m} v_i g_i \left(w_{i0} + \mathbf{w}_i^{\mathrm{T}} \mathbf{x} \right), \tag{6.19}$$

where $\mathbf{x} = (x_1, \ldots, x_n)^{\mathrm{T}}$ is the input vector and $\mathbf{w}_i = (w_{i1}, \ldots, w_{in})^{\mathrm{T}}$ is the weight vector for hidden unit i. We consider separately the solutions given by equations 5.20 and 5.21.

In case 5.2(i), the linear function gain function given by equation 5.20 may be regarded formally as a special case of the power function in case 5.2(iv) below.

In case 5.2(ii), consider a network with two hidden units 1 and 2, each with a logarithmic gain function of the form in equation 5.21 with potentially distinct parameter values as indicated by subscripts 1 and 2. The response in equation 6.19 now reads

$$r = \sum_{i=1}^{2} \left(v_i A_i \ln \left| \mathbf{w}_i^{\mathrm{T}} \mathbf{x} + w_{i0} - B_i \right| + v_i C_i \right). \tag{6.20}$$

To keep the same the input-output relation for all input $\mathbf{x}$, the new parameters (with a tilde) should be related to the original parameters by

$$\prod_{i=1}^{2} \left|\tilde{\mathbf{w}}_i^{\mathrm{T}}\mathbf{x} + \tilde{w}_{i0} - B_i\right|^{v_i A_i} = \prod_{i=1}^{2} \left|\mathbf{w}_i^{\mathrm{T}}\mathbf{x} + w_{i0} - B_i\right|^{v_i A_i}. \tag{6.21}$$

An explicit construction of parameter confounding with a continuous parameter t is

$$\tilde{\mathbf{w}}_1 = e^{v_2 A_2 t}\mathbf{w}_1, \tag{6.22}$$

$$\tilde{\mathbf{w}}_2 = e^{-v_1 A_1 t}\mathbf{w}_2, \tag{6.23}$$

$$\tilde{w}_{10} = B_1 + e^{v_2 A_2 t}\left(w_{10} - B_1\right), \tag{6.24}$$

$$\tilde{w}_{20} = B_2 + e^{-v_1 A_1 t}\left(w_{20} - B_2\right), \tag{6.25}$$

which satisfy equation 6.21 for arbitrary t, with $t = 0$ corresponding to the original parameter values. It follows from equations 6.22 and 6.23 that

$$\tilde{w}_1^{v_1 A_1}\tilde{w}_2^{v_2 A_2} = w_1^{v_1 A_1} w_2^{v_2 A_2}, \tag{6.26}$$

where $w_i = \|\mathbf{w}_i\|$ and $\tilde{w}_i = \|\tilde{\mathbf{w}}_i\|$ are the Euclidean vector norms ($i = 1, 2$). This equation generalizes the invariance of the product of weights as shown in Figure 1c.

In case 5.2(iii), consider a network with two hidden units 1 and 2, each having an exponential gain function of the form in equation 5.22 with potentially distinct parameter values as indicated by subscripts 1 and 2. The response in equation 6.19 now reads

$$r = \sum_{i=1}^{2} \left(v_i A_i e^{\alpha_i\left(w_{i0} + \mathbf{w}_i^{\mathrm{T}}\mathbf{x}\right)} + v_i C_i\right). \tag{6.27}$$

To maintain the same input-output relation for all input $\mathbf{x}$, the new parameters (with tilde) should be related to the original parameters by

$$\tilde{v}_1 C_1 + \tilde{v}_2 C_2 = v_1 C_1 + v_2 C_2, \tag{6.28}$$

$$\tilde{v}_1 e^{\alpha_1 \tilde{w}_{10}} = v_1 e^{\alpha_1 w_{10}}, \tag{6.29}$$

$$\tilde{v}_2 e^{\alpha_2 \tilde{w}_{20}} = v_2 e^{\alpha_2 w_{20}}, \tag{6.30}$$

together with $\tilde{\mathbf{w}}_1 = \mathbf{w}_1$ and $\tilde{\mathbf{w}}_2 = \mathbf{w}_2$. Note that equation 6.29 or 6.30 for each hidden unit is the same as equation 6.3 for a single neuron, so that the output weight v_i and threshold w_{i0} for each unit are confounded as before. What is new now is that the output weights of the two hidden units need to be coordinated by equation 6.28. One explicit construction with a continuous index t is as follows:

$$\tilde{v}_1 = v_1 + C_2 t, \tag{6.31}$$

$$\tilde{v}_2 = v_2 - C_1 t, \tag{6.32}$$

$$\tilde{w}_{10} = w_{10} + \alpha_1^{-1} \ln \left(v_1 / (v_1 + C_2 t) \right), \tag{6.33}$$

$$\tilde{w}_{20} = w_{20} + \alpha_2^{-1} \ln \left(v_2 / (v_2 - C_1 t) \right). \tag{6.34}$$

These new parameters satisfy equations 6.28 to 6.30 for arbitrary t, with $t = 0$ corresponding to the original parameters.

In case 5.2(iv), consider a network with two hidden units with subscripts 1 and 2, each having a power gain function of the form in equation 5.23. The response in equation 6.19 becomes

$$r = \sum_{i=1}^{2} \left(v_i A_i \left| \mathbf{w}_i^{\mathrm{T}} \mathbf{x} + w_{i0} - B_i \right|^{\alpha_i} + v_i C_i \right). \tag{6.35}$$

To maintain the same the input-output relation for all input $\mathbf{x}$, the new parameters (with a tilde) should be related to the original parameters by

$$\tilde{v}_1 C_1 + \tilde{v}_2 C_2 = v_1 C_1 + v_2 C_2, \tag{6.36}$$

$$\tilde{v}_1 \left| \tilde{\mathbf{w}}_1^{\mathrm{T}} \mathbf{x} + \tilde{w}_{10} - B_1 \right|^{\alpha_1} = v_1 \left| \mathbf{w}_1^{\mathrm{T}} \mathbf{x} + w_{10} - B_1 \right|^{\alpha_1}, \tag{6.37}$$

$$\tilde{v}_2 \left| \tilde{\mathbf{w}}_2^{\mathrm{T}} \mathbf{x} + \tilde{w}_{20} - B_2 \right|^{\alpha_2} = v_2 \left| \mathbf{w}_2^{\mathrm{T}} \mathbf{x} + w_{20} - B_2 \right|^{\alpha_2}. \tag{6.38}$$

Since equations 6.37 and 6.38 are equivalent to equation 6.9 for a single neuron, the input weights and the output weights should be confounded as before. What is new is the coordination of the input weights of the two hidden units according to equation 6.36. One explicit construction with a continuous parameter t is as follows:

$$\tilde{v}_1 = v_1 + C_2 t, \tag{6.39}$$

$$\tilde{v}_2 = v_2 - C_1 t, \tag{6.40}$$

$$\tilde{\mathbf{w}}_1 = \left(v_1 / (v_1 + C_2 t) \right)^{1/\alpha_1} \mathbf{w}_1, \tag{6.41}$$

$$\tilde{\mathbf{w}}_2 = \left(v_2 / (v_2 - C_1 t) \right)^{1/\alpha_2} \mathbf{w}_2, \tag{6.42}$$

$$\tilde{w}_{10} = B_1 + \left(v_1 / (v_1 + C_2 t) \right)^{1/\alpha_1} (w_{10} - B_1), \tag{6.43}$$

$$\tilde{w}_{20} = B_2 + \left(v_2 / (v_2 - C_1 t) \right)^{1/\alpha_2} (w_{20} - B_2), \tag{6.44}$$

which satisfy equations 6.36 to 6.38 for any t. In particular, the new parameters reduce to the old ones when $t = 0$. It follows from equations 6.39 to 6.42 that we always have $\tilde{v}_i \tilde{w}_i^{\alpha} = v_i w_i^{\alpha}$ for each hidden unit i despite their interaction, with w_i being the norm of the weight vector $\mathbf{w}_i$.

As shown above, both the exponential and the power gain functions can generate parameter confounding when multiple hidden units are involved. The examples with two hidden units considered above can be readily generalized to networks with an arbitrary number of hidden units. The mechanism of parameter confounding for multiple hidden units is that the output weights of different units should be properly coordinated, while the parameter confounding for individual hidden units works in the same way as before. It is also possible to generate parameter confounding with multiple hidden units of distinct types, some having the exponential gain function, and some having the power gain function. Since the idea is similar to the coordination of the output weights as described above, further discussion will be omitted.

7 Parameter Confounding Occurs when Fitting Neural Networks to Noisy Data

At the first glance, it may seem that the problem of continuous equivalence classes should disappear for commonly used gain functions or biological gain functions that are not exactly power, exponential, or log functions. This is not the case, however, as many gain functions can be well approximated over some range of their inputs by one of these forms. If a set of training data drives a hidden unit only into a range of its gain function over which a power law, exponential or logarithm approximation holds, then we may *de facto* replace this unit with a new unit having one of these gain functions, which according to our theory will produce a continuum of networks functionally identical to the original network. In this section, we show numerically that continuous parameter confounding can occur for commonly used gain functions when the data contain noise, which is always present in practical applications.

7.1 Exact Continuous Parameter Confounding in a Simple Model. Before turning to more general gain functions, first consider a simple network with a gain function that is precisely a power function (see Figure 5a). By our confounding theory, an identical input-output relationship of the network can be defined by a continuum of parameter values. Although this continuum includes the true model parameters, it is impossible to recover the true parameters uniquely from any given input-output data, because the model output is identical for other parameters as well. Therefore, we expect that the parameter estimates obtained from data are highly variable, yet all lying along this continuum.

We confirmed this argument with the network in Figure 5a, where the gain function $g(u)$ is the threshold power function in equation 4.8, and the parameters for this gain function were obtained by least-squares fitting to the biological data in Figure 3b (second panel from left, $\alpha < 1$). We generated a data set of input-output pairs (see Figure 5b, circles) from random inputs by assuming that the output r obeyed Poisson probability distribution $P(r\,|x, \theta) = f(x, \theta)^r \exp(-f(x, \theta))/r!$, where the mean is given by the deterministic network output $f(x, \theta) = vg(wx + w_0)$, with the parameter values $\theta = (w, w_0, v) = (1, 0, 1)$. Our task was to estimate the three parameters (w, w_0, v) from the input-output data by least squares. We used simplex optimization (Nelder & Mead, 1965) as implemented by Matlab *fminsearch*, starting from random initial values in the following ranges: $-3 \le w_0 \le 3$, $0.1 \le w, v \le 4$. The final estimates of the parameters varied from trial to trial, depending on the initial values (see Figure 5c). The scatter of the final estimates followed precisely the confounding curves predicted by the theory, as given by equations 6.13 to 6.15.

In this example, the variability of the parameter estimates arises simply because the power gain function in equation 4.8 allows different parameter values to account for the same input-output relationship. The algorithm used for parameter estimation and the composition of the data set can affect the exact distribution of the scatter, but not the existence of the parameter confounding phenomenon itself.

7.2 Continuous Confounding in a Network with Hyperbolic Tangent Gain. In all of the remaining examples, the gain functions are not power, exponential, or logarithmic functions. Yet parameter confounding occurs approximately as predicted by theory because for a given noisy data set, the gain function may be well approximated by one of the three functions.

This section focuses on the simple network in Figure 6a, where the hidden unit gain is a threshold hyperbolic tangent function. The output r of the network has Poisson probability distribution $P(r\,|\,x, \theta)$, with the mean given by $f(x, \theta) = vg(wx)$, where $g(u) = [\tanh u]_+ = \max(0, \tanh u)$ is the gain function and $\theta = (w, v) = (1, 50)$ are the true parameter values. We first used the network to generate a data set comprising the input-output pairs $(x_1, r_1), \ldots, (x_N, r_N)$, and then estimated the two parameters (w, v) from the data set by maximizing the likelihood function $L(\theta) = \prod_{k=1}^{N} P(r_k\,|\,x_k, \theta)$. Besides maximum likelihood, least squares can also be used to obtain similar final results.

Given a data set, the maximum likelihood method yielded unique parameter estimates regardless of the starting point in the optimization procedure. As shown in Figure 6d for one typical data set, the error surface ($-\ln L(\theta)$, negative log likelihood) had a shallow valley whose minimum (square) deviated from the true parameters (triangle). The optimization procedure found the same minimum from various starting points ($\times$).

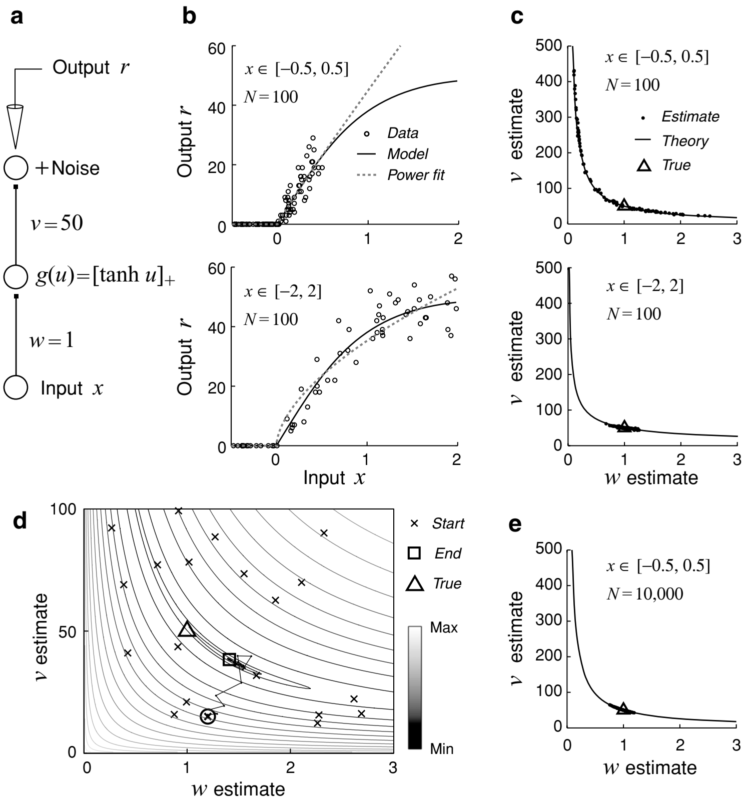


Figure 6: Continuous parameter confounding occurs when the gain is well approximated by a power function. (**a**) The simple network has input weight w, output weight v, and rectified tanh gain. (**b**) The mean output (solid line) is better approximated over the smaller input range $[-0.5, 0.5]$ by a power function (dashed line, top) than over the larger range $[-2, 2]$ (bottom). Circles show a typical data set having $N = 100$ random inputs and the elicited Poisson outputs. Some subthreshold data are clipped in the bottom panel. (**c**) Each point corresponds to maximum likelihood estimates from a single data set like the ones shown in **b**. Different random data sets yielded estimates scattered along the equivalence curves predicted by the power law approximations in **b**. The estimates for the larger input range (bottom panel) stayed closer to the truth (triangle) than those for the smaller input range (top panel). (**d**) For a single data set, maximum likelihood yielded the same estimates (square) at the minimum of the error surface in the contour plot (negative log likelihood) regardless of the optimization starting point (crosses). But the result differed from the true parameter values (triangle). The zigzag line shows one optimization trajectory, starting from the circled cross. (**e**) By increasing the data points in each data set from 100 to 10,000, the estimates became closer to the truth (triangle), even for the smaller input range.

For different data sets, the maximum likelihood estimates were different from one another, and different from the truth as well, but all the results were scattered approximately along the curve predicted by the power law confounding theory (see Figures 6c and 6e). Here each point is the maximum likelihood estimate from a single data set (by Matlab *fminsearch* with random initial values $0.1 \leq w \leq 4$, $5 \leq v \leq 200$). The variability along the curve is not due to different choices of optimization starting point as in Figure 5 or some artifact of the optimization procedure, but rather is due to random differences in the data sets.

The confounding curves in Figures 6c and 6e are given by equation 3.6, where parameter α was obtained by approximating the gain function by a power function $g(u) \approx Au^{\alpha}$, so that the mean network output became $f(x, \theta) \approx vAx^{\alpha}$. Over the smaller input interval $[-0.5, 0.5]$ (see Figure 6b, top), power law fitting to the gain function by least squares yielded the parameters $A = 0.899$ and $\alpha = 0.944$. Over the larger input interval $[-2, 2]$ (see Figure 6b, bottom), the results were $A = 0.704$ and $\alpha = 0.586$.

Over the larger interval, the gain function can no longer be well approximated by a power function (see Figure 6b, bottom), and the scattering of the estimates was greatly reduced (see Figure 6c, bottom). Similar reduction was achieved for the smaller input interval by increasing the number of data points (see Figure 6e), consistent with the theory that, in the limit of infinite and noiseless data, all parameters of a network with tanh gain function should be recovered uniquely (Fefferman, 1994).

7.3 The Same Network Can Exhibit Multiple Types of Confounding Simultaneously. A gain function may be approximated in different ways. The standard sigmoidal gain function, for example, can be approximated over a limited input range by both the exponential and the power functions. Because exponential and power functions imply different parameter confounding curves, which curve will the results actually follow when the network parameters are estimated from noisy data?

To examine this issue, consider the simple network in Figure 7a with the standard sigmoidal gain function $g(u) = (1 + e^{-u})^{-1}$, which can be approximated over the interval $[-3, -1]$ by the power function $g(u) \approx A(u - B)^{\alpha}$ with $A = 0.00434$, $B = -5.21$ and $\alpha = 2.86$, as well as by the exponential function $g(u) \approx Ae^{\alpha u}$ with $A = 0.641$ and $\alpha = 0.864$, where the parameters were found by least squares. The output of the network is Poisson with the mean $f(x, \theta) = vg(wx + w_0)$, where $\theta = (w, w_0, v) = (1, -2, 300)$. Based on the two approximations, the mean output of the network $f(x, \theta)$ (see Figure 7b, black line) can be approximated over the input interval $[-1, 1]$ also by a power function (see Figure 7b, dashed line) as well as an exponential function (see Figure 7b, gray line).

We estimated the three parameters (w, w_0, v) from 100 random data sets by least squares by sequential quadratic programming (Powell, 1978)

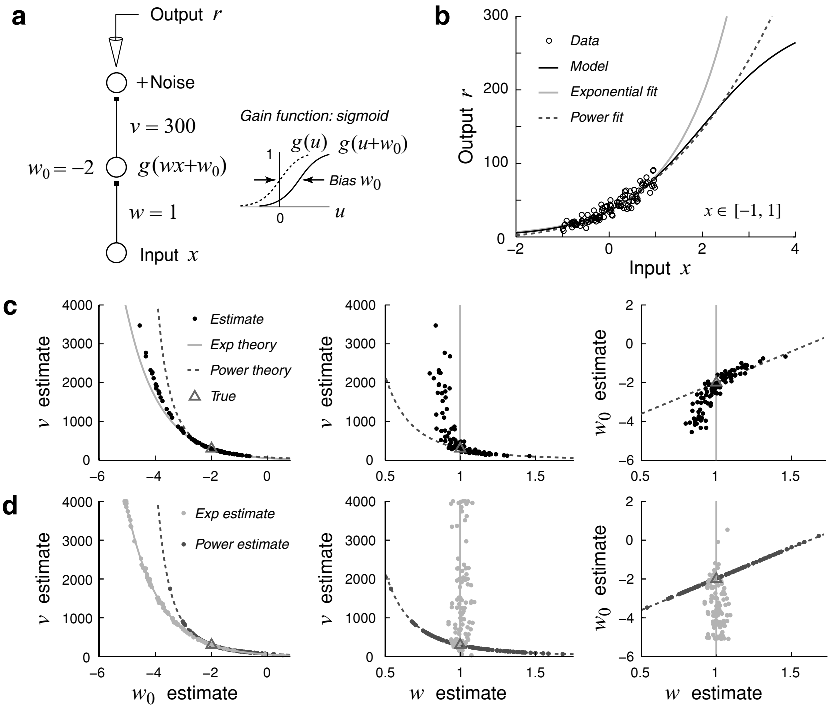


Figure 7: Parameter confounding for multiple forms of gain function approximations. (**a**) The simple network has a hidden unit with the standard sigmoidal gain function g. (**b**) The mean output (black line) is well approximated by both an exponential function (gray line) and a power function (dashed line) for inputs in $[-1, 1]$. The circles show a typical data set with 100 inputs and the elicited Poisson responses. (**c**) The scattering of the parameter estimates (dots) obtained from different random data sets cannot be entirely accounted for by either the power law theory (dashed lines) or the exponential theory (solid lines). (**d**) For comparison, when the gain function was replaced by either the power law or the exponential approximation, the new estimates followed either the power law (dark gray dots) or the exponential theory predictions (light gray dots).

using Matlab *fmincon* with random initial values: $0.1 \le w \le 10$, $-8 \le w_0 \le 8$, $100 \le v \le 1000$. The estimates seemed to spread approximately along a continuum in the three-dimensional parameter space (see Figure 7c), falling between the two extreme cases where the gain function is truly either an exponential or a power function (see Figure 7d). The power law theory curves (dashed) are given by equations 6.13 to 6.15, and the exponential theory curves (gray) by equation 6.3. The scattering of the estimates (black dots, in $w-v$ and $w-w_0$ planes in Figure 7c) conforms better to the power law theory for larger values of w and better to the exponential theory for smaller values of w. This behavior is largely consistent with whether the exponential or the power functions approximates better.

7.4 Confounds Between Input Weights to Different Hidden Units. The logarithm is the only gain function that permits exact confounds between the input weights from different units. Here we show that similar confounding occurs for a power gain function that approximates a logarithm (see Figure 8). The power function has a threshold and is given by $g(u) = z^{-1}(u^z - 1)$ with $z = 0.001$ for $u \geq 1$, and $g(u) = 0$ otherwise (see appendix A). Assuming that the output weights ($v_1 = v_2 = 100$) are known, we want to estimate the input weights w_1 and w_2 (with true values $w_1 = w_2 = 1$) from noisy input-output data by least squares (using Matlab *fminsearch* with random initial values $0.5 \leq w_1, w_2 \leq 2$). As shown in Figure 8c, the estimates (black dots) from 100 random datasets are scattered along the confounding curve $w_1 w_2 = 1$ (black curve), as given by equation 6.26 (see also Figure 1c), with each data set comprising 100 uniformly distributed random inputs $x_1, x_2 \in [2, 4]$ and the elicited Poisson outputs. Similar results were obtained when the optimization was initialized at the true model parameters instead of randomly (see Figure 8d).

7.5 Confounding in a Center-Surround Neural Network Model. Parameter confounding can also occur in more complex networks like the one in Figure 9, which is inspired by a hypothetical model of an auditory neuron. We demonstrate that due to confounding, it may not be possible to accurately recover the parameters of this network with the kinds of stimuli typically used in experiments, and demonstrate a bootstrapping method to test for confounding when fitting real data.

The network in Figure 9a has a generic center-surround organization that could exist in various brain areas, such as in the dorsal cochlear nucleus (Hancock, Davis, & Voigt, 1997; Nelken & Young, 1994; Young & Davis, 2002). The network can be rearranged into the layered form analyzed in this article by the addition of virtual pass-through or "ghost" units (dotted circles in Figure 9a, bottom panel, see also DiMattina & Zhang, 2008). For input $\mathbf{x} = (x_1, \ldots, x_{21})^{\mathrm{T}}$, the response r has Poisson distribution with mean $f(\mathbf{x}, \theta) = v_{\mathrm{E}} g(\mathbf{w}_{\mathrm{E}}^{\mathrm{T}}\mathbf{x} + w_{0\mathrm{E}} + v_{\mathrm{I}} g(\mathbf{w}_{\mathrm{I}}^{\mathrm{T}}\mathbf{x} + w_{0\mathrm{I}}))$, where the gain function g (left inset) is rectified hyperbolic tangent, $\mathbf{w}_{\mathrm{E}}$ is a weight vector of narrow gaussian connections (center μ_{E}, spread σ_{E} and amplitude A_{E}) of the excitatory unit (E-unit) with the input layer, and $\mathbf{w}_{\mathrm{I}}$ specifies the broad gaussian connections (center μ_{I}, spread σ_{I} and amplitude A_{I}) of the inhibitory unit (I-unit) with the input layer (right insets). All model parameters are summarized in Table 2.

Two types of stimuli were used to generate the response data. The first type was diffuse random stimuli (see Figure 9b, top) where each input was

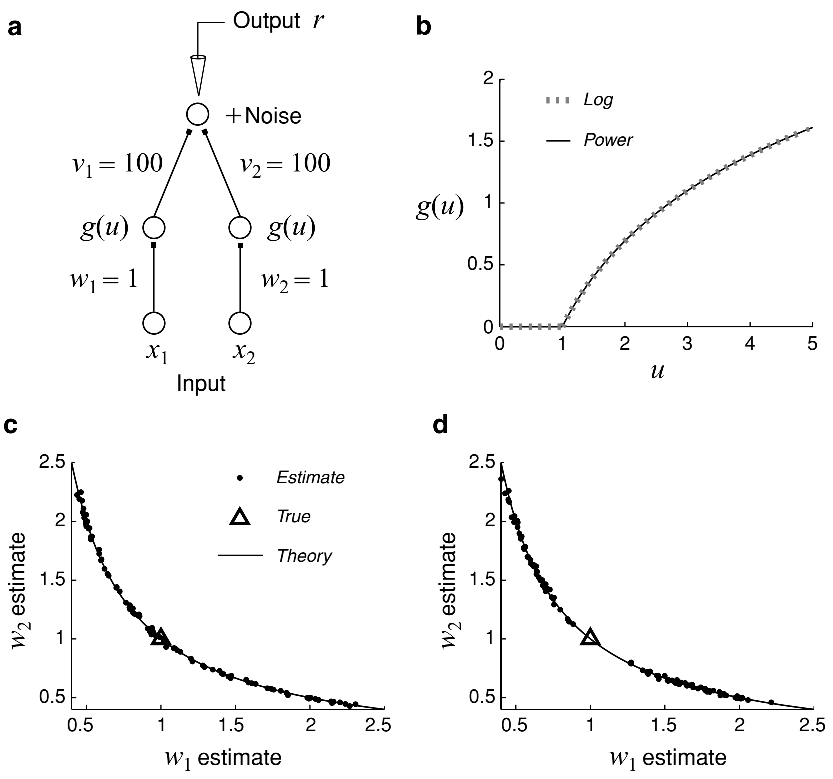


Figure 8: Confounding in a network with power gain functions that resemble a logarithm. (**a**) In this neural network, the output weights (v_1 and v_2) are known, and we want to estimate the two input weights w_1 and w_2 from input-output data with Poisson noise. (**b**) The gain function g for both hidden units is a threshold power function (solid line) that closely approximates the rectified logarithm (dashed line). (**c**) The weight estimates were scattered along the curve predicted by the logarithmic theory. One dot corresponds to the estimate from one data set. (**d**) Same as **c**, but with the minimization procedure always initialized from the truth (triangle) rather than from a random point, showing again that the choice of the starting point had no effect on the existence of parameter confounding.

drawn randomly with uniform distribution from 0 to a maximum value, which itself was selected randomly from [0, 1] for each stimulus. The inputs x_i were always restricted to the range [0, 1], which may be interpreted as the range of responses of sensory transducers. The second type was spot stimuli (see Figure 9b, bottom), each of which was three receptors wide, located at 19 possible center locations (2, . . . , 20), and had five possible amplitudes (0.2, 0.4, . . . , 1).

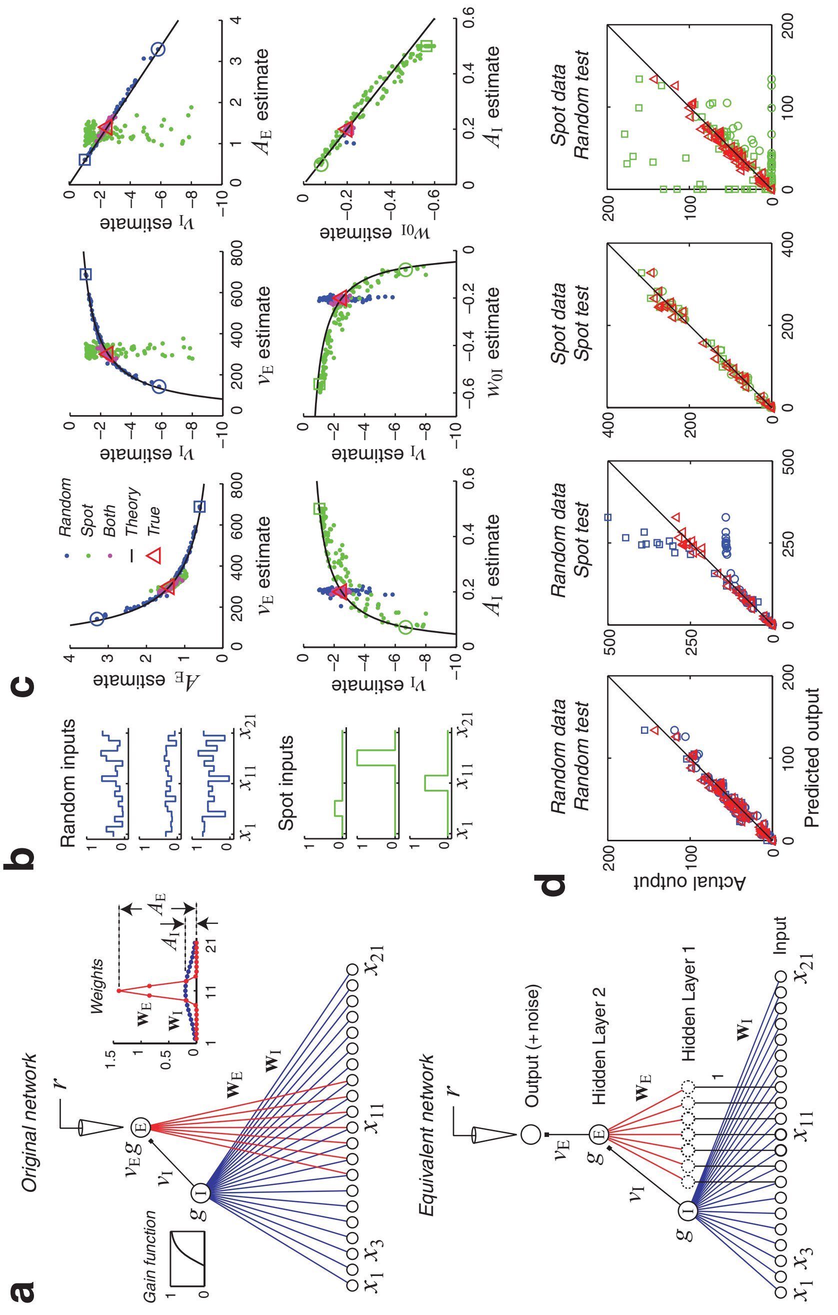

a
Original network
Gain function
Weights
$\mathbf{w}_E$
$\mathbf{w}_I$
A_E
A_I
$v_E g$
v_I
g
r
x_1 x_3 x_{11} x_{21}
Equivalent network
Output (+noise)
Hidden Layer 2
Hidden Layer 1
Input
b
Random inputs
Spot inputs
c
Random
Spot
Both
Theory
True
A_E estimate
v_E estimate
v_I estimate
A_I estimate
w_{0I} estimate
d
Random data Random test
Random data Spot test
Spot data Spot test
Spot data Random test
Actual output
Predicted output

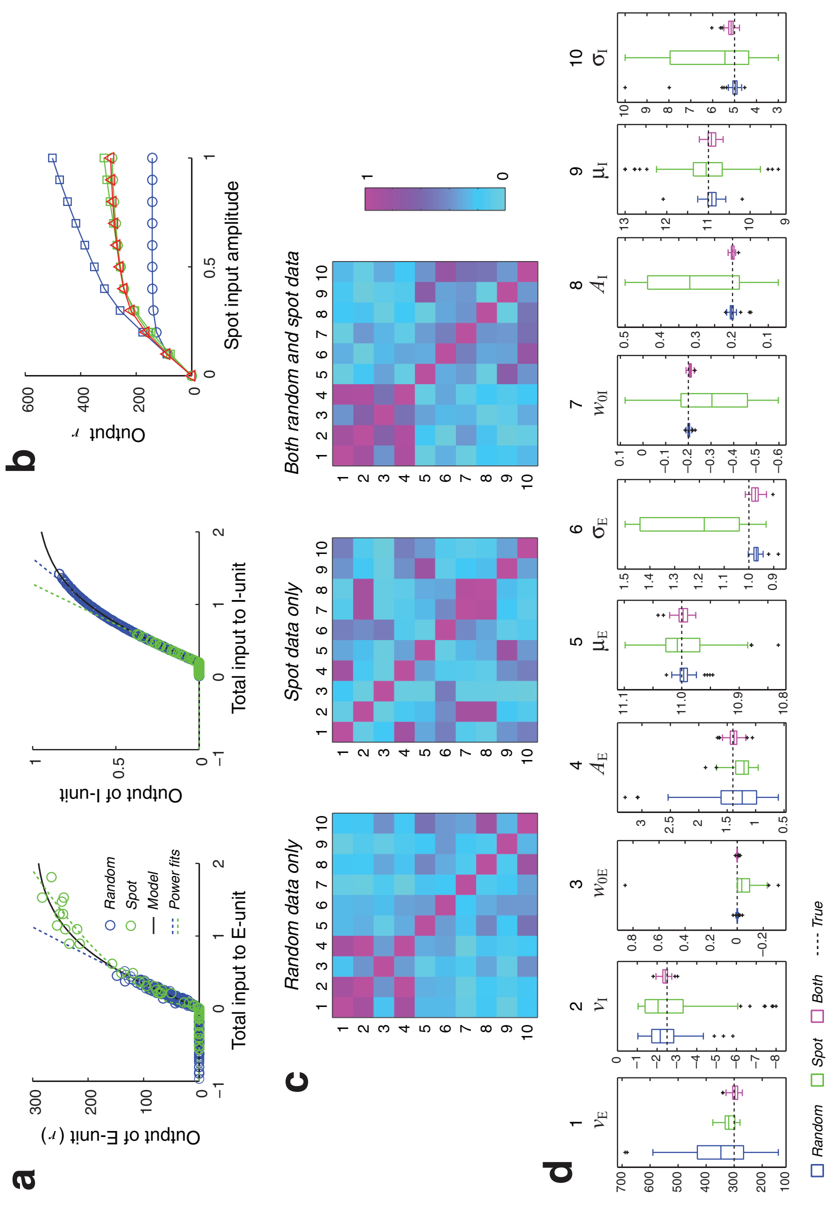
Table 2: Parameters for the Center-Surround Network in Figure 9a.

Parameter	*Symbol*	*Description*	*Value*	*Bounds*
θ_1	v_E	E-unit saturation	300	[75, 1200]
θ_2	v_I	I-unit strength	–2.5	[–8, –0.5]
θ_3	w_{0E}	E-unit bias	0	[–2, 2]
θ_4	A_E	E-unit weight amplitude	1.4	[0.5, 4]
θ_5	μ_E	E-unit weight center	11	[9, 13]
θ_6	σ_E	E-unit weight spread	1	[0.5, 1.5]
θ_7	w_{0I}	I-unit bias	–0.2	[–2, 2]
θ_8	A_I	I-unit weight amplitude	0.2	[0, 0.5]
θ_9	μ_I	I-unit weight center	11	[9, 13]
θ_{10}	σ_I	I-unit weight spread	5	[3, 10]

Note: The assumed true values of the underlying model and the search bounds used by the optimization procedure are given in the last two columns.

Random stimuli drive the I-unit (inhibitory) better than the E-unit (excitatory) because the I-unit weight distribution is wider. In contrast, the localized spot stimuli drive the E-unit better than the I-unit. Consistent with these observations, the E-unit gain function is well approximated by

Figure 9: Continuous parameter confounding in a more complex neural network. (**a**) Top panel: The center-surround network has an excitatory unit (E) with narrow gaussian connections to the input layer and an inhibitory hidden unit (I) with broad gaussian connections (right inset). Both units have rectified tanh gain (left inset), and the maximum firing rate of the E-unit is scaled by parameter v_E. See Table 2 for detail. Bottom panel: The layered neural network is equivalent to the model at the top. (**b**) Examples of diffuse random inputs (top) and localized spot inputs (bottom). (**c**) Parameter estimates attained from data with random inputs only (blue dots), spot inputs only (green dots), or a mixture of both (magenta dots). Top panels: Estimates from random input data (blue dots) showed confounding among parameters v_E, v_I, and A_E, as predicted by theory (black curves). Bottom panels: Similarly, estimates from spot input data (green dots) showed confounding among parameters v_I, A_I, and w_{0I}, as predicted by theory (black curves). Combining both spot and random inputs yielded estimates (magenta dots) lying closer to the truth (red triangle). (**d**) Predicting the network responses across all types of stimuli requires knowing the true parameters (red triangles). Two left panels: Diverse parameter values obtained from the random data, as indicated by the blue circle or square in panel **c** (top) for the smallest or largest values of v_E, can account for the responses to random inputs, but not to spot inputs. Two right panels: Parameters obtained from the spot data, as indicated by the green circle or square in panel **c** (bottom) for the smallest or largest values of A_I, can account for the responses to spot inputs but not to random inputs.


a
Output of E-unit (r)
Total input to E-unit
Random
Spot
Model
Power fits
Output of I-unit
Total input to I-unit
b
Output r
Spot input amplitude
c
Random data only
Spot data only
Both random and spot data
1
0
d
1 ν_E
2 ν_I
3 w_{0E}
4 A_E
5 μ_E
6 σ_E
7 w_{0I}
8 A_I
9 μ_I
10 σ_I
Random
Spot
Both
True

a power function Au^{α} ($\alpha = 0.941$, $A = 0.896$) over the smaller response range elicited by the random stimuli, but less well approximated by the best-fitting power function ($\alpha = 0.555$, $A = 0.699$) over the larger response range elicited by the spot stimuli (see Figure 10a, left). Conversely, the I-unit gain is better approximated by a power function ($\alpha = 0.964$, $A = 0.926$) for the spot stimuli than by a power function ($\alpha = 0.760$, $A = 0.761$) for the random stimuli (see Figure 10a, right).

The power law approximations allowed us to predict the parameter confounding observed when the network parameters were recovered from the stimulus-response data. A set of 300 random stimuli and their associated Poisson responses was generated, and 100 bootstrapped training sets were created by randomly drawing 300 stimuli with replacement from the original set. For each bootstrapped training set, an estimate of the model parameters was obtained by least-squares optimization (Matlab *fmincon*, with search ranges for each parameter given in Table 2). As shown in Figure 9c (top), the estimates for the E-unit parameters (blue dots) attained from these bootstrapped data sets were scattered along the confounding curves (black lines) as predicted by the power law theory. Similarly, as shown in Figure 9c (bottom), estimates from the spot stimuli (100 bootstrapped data sets drawn from an original set of 300 stimuli selected randomly from 95 possible spot stimuli) for the I-unit parameters (green dots) also were scattered along predicted confounding curves (black lines).

From these results, one would predict that continuous confounding can be avoided by using a stimulus set that contains both spot and random stimuli, since with this combined stimulus set, the power law approximation

Figure 10: Additional results for the center-surround network in Figure 9. (**a**) The spot inputs (green circles) drove the E-unit over a wider dynamic range of its output (black curve) than the random inputs (blue circles). The opposite was true for the I-unit. A power law approximated better over a smaller range than a wider range (dashed lines with the same colors as the data). (**b**) Networks with diverse parameter values as indicated by the blue and green squares and circles in Figure 9c responded differently to spot stimuli presented at the receptive field center with varying amplitude. Parameters obtained from the spot data yielded responses (green circles and squares) closer to the true model (red triangles) than that obtained from the random data (blue circles and squares). Noise was excluded. (**c**) The correlation coefficient matrices of the estimates may help detect confounded parameters. In the two left-most panels, all pairs of parameters that were continuously confounded (see Figure 9c) exhibited high correlations. Conversely, high correlations such as those in the right-most panel do not necessarily mean confounding. (**d**). Box plot of all ten parameters estimated from three data sets. All parameters were accurately recovered from the combined data set with both random and spot inputs, but not from either one type of inputs alone.

will not be valid for either unit. To verify this, we generated 100 bootstrapped data sets of 300 stimuli taken from a set of 205 random stimuli and 95 spot stimuli and confirmed that the final parameter estimates attained with these data sets (see Figure 9c, magenta dots: 'Both') indeed had little spread and the true model parameters (red triangles) were accurately recovered. The estimation errors for all three bootstrapped training sets are summarized in Figure 10d, and we see that only this combined training set allowed accurate recovery of all parameters.

The bootstrapping test used here is a practical way to determine if there is continuous confounding when one fits a neural network to real data. Since confounded parameters tend to have correlated estimates, one possible method to identify potential sets of confounded parameters is to look for correlations between different bootstrapped estimates by computing a correlation matrix as shown in Figure 10c. This method most likely will detect parameter confounding when it exists, but it is also subject to false alarm because correlation may arise also from statistical regularities in the stimuli and does not always imply parameter confounding.

When a neural network model exhibits continuous parameter confounding, the disparate parameter estimates attained by training with different data sets can make drastically different predictions of the model's responses to stimuli not in the training set. For example, we used three sets of parameter values, as denoted by the circle, square, and triangle in Figure 9c, to predict responses to various stimuli (see Figure 9d). Only the model with the correct parameters made reliable predictions across all types of stimuli (also see Figure 10b). Thus, the existence of continuous equivalence classes is relevant for recovering the parameters of a model, as well as for making accurate predictions using the estimated model.

8 Discussion

We have introduced a biologically motivated mathematical method that allows one to determine when a neural network can be gradually modified while keeping the input-output relationship unchanged. Applying this method to a standard model of a three-layer neural network (Rumelhart et al., 1986) with convergent and nondegenerate weights, we show that this is possible only when the hidden unit gain functions are given by power, exponential, and logarithmic functions, possibly with zero sub-threshold regions (see Table 1 and Figure 3). These three gain function forms are mathematically related, with the exponential and logarithm being limiting cases of power law functions (see appendix A).

Our result has direct implications for the network uniqueness problem: whether the input-output function of a neural network uniquely determines its structure. For the three-layer networks as we have discussed (see Figure 2), uniqueness is guaranteed as long as none of the hidden units has gain function that belongs to the three types noted. One caveat is that

Table 3: Gain Functions Required for Additive and Multiplicative Computations.

	Additive Inputs	*Multiplicative Inputs*
Additive outputs	Equation: $g(x+y) = g(x) + g(y)$ Linear solution: $g(u) = Au$	Equation: $g(xy) = g(x) + g(y)$ Logarithmic solution: $g(u) = A \ln \lvert u \rvert$
Multiplicative outputs	Equation: $g(x+y) = g(x)\, g(y)$ Exponential solution: $g(u) = e^{\alpha u}$	Equation: $g(xy) = g(x)\, g(y)$ Power solution: $g(u) = \lvert u \rvert^{\alpha}$

Notes: Each of the four types of gain functions is uniquely required by one of the four computational tasks of transforming additive or multiplicative inputs into additive or multiplicative outputs (see appendix B). The four types coincide with those in Table 1.

because the modification of parameters has been assumed to be continuous in this article, our result does not automatically rule out the possibility of functionally equivalent networks with distinct discrete parameter sets that cannot be linked by a continuous equivalence class of networks. We have focused on the continuous case because it corresponds to biological learning processes that tend to be gradual and incremental. It will be of interest for future research to determine if the solutions obtained using our continuous perturbation method exhaust all of the mathematical possibilities. Our theory does not contradict existing theoretical results about network uniqueness (Albertini et al., 1993; Chen et al., 1993; Fefferman, 1994; Kurkova & Kainen, 1994; Sussman, 1992) because these results assume gain functions that are different from the solutions obtained in this article.

Previous studies have suggested that many biological neurons have gain functions that can be well described by power and exponential functions (Anderson et al., 2000; Ermentrout, 1998; Gabbiani et al., 2002; Smith et al., 2002; Stafstrom et al., 1984). Here gain function is loosely interpreted as the dependence of firing rate on input current or membrane potential as in Figure 3b. Furthermore, many computational models of sensory processing have made use of power law, exponential, or logarithmic transformations of sensory variables (Adelson & Bergen, 1985; Colburn, 1973; Heeger, 1992; Pollen & Ronner, 1983). The usefulness of these transformations is also underscored by the fact they are the only possibilities for transformations between additive or multiplicative inputs or outputs (see Table 3; see appendix B for derivations). Previous theoretical work has shown that the power law function is unique in transforming multiplicative inputs into multiplicative outputs (Hansel & Van Vreeswijk, 2002; Miller & Troyer, 2002). This table extends these observations by including transformations involving both additive and multiplicative inputs and outputs. Another interesting observation is that the classification of gain functions in Table 3 coincides with that in Table 1. The gain functions in the two tables can actually be made identical with proper scaling and shifting (see appendix B).

Our results also imply that different neural networks with disparate values of weight and threshold parameters can perform identical input-output transformations. This possibility is biologically plausible, as previous studies of more biophysically realistic neural network models have suggested that networks having disparate parameter values can exhibit functionally nearly identical behavior (Achard & De Schutter, 2006; Marder & Goaillard, 2006; Olypher & Calabrese, 2007; Prinz, Bucher, & Marder, 2004; Weaver & Wearne, 2008). It may be actually beneficial for a neuronal network to be able to maintain functional homeostasis by modifying suitable synaptic weights or threshold parameters in order to offset or compensate for unexpected changes due to injury or nonstationarity in different parameters elsewhere in the network. Our analysis suggests that how different parameters may compensate should depend on the input-output relation or the gain function of individual neurons. For power gain function, for instance, the condition for preserving functionality such as $vw^{\alpha} = \text{const}$ (see equation 3.6) implies that a small increment Δw of the input weight should be related to the increment Δv of the output weight by

$$\Delta v/v = -\alpha \Delta w/w \tag{8.1}$$

or $\Delta \ln v = -\alpha \Delta \ln w$, where Δ indicates a small change. Such quantitative relationships might help constrain further studies of network functional homeostasis.

The basic confounding equation, 2.2, applies to arbitrary feedforward networks because it assumes only the differentiability of a generic input-output relation. We have focused on the three-layer perceptrons (Rumelhart et al., 1986) because they allow complete analytical solutions to equation 2.2 under the assumption of convergent and nondegenerate weight matrices, as shown in this article. The three-layer networks are useful as tools of nonlinear regression in a wide variety of applications, including modeling the responses of nonlinear sensory neurons (Lau et al., 2002; Lehky et al., 1992; Prenger et al., 2004; Wu et al., 2006; Zipser & Andersen, 1988). These networks may also be implemented by the dendrites of a single neuron instead of a network of neurons (Poirazi, Brannon, & Mel, 2003). Although three-layer neural networks also enjoy universal function approximation capability (Cybenko, 1989; Funahashi, 1989; Hornik, Stinchcombe, & White, 1989), a sufficiently large number of hidden units or a divergent connection pattern may be required. In a divergent three-layer network, the gain functions identified in Table 1 can still induce parameter confounding, but no longer exhaust all possibilities of confounding mechanisms (see Figure 4a).

It is of interest to see how our methods generalize to more complex neural models having more layers, as well as models having recurrent connections and temporal dynamics. We emphasize that the parameter confounding mechanisms identified in this article remain valid as a subnetwork embedded in a larger and more complex network. If a subnetwork contains confounded parameters, so does the whole network. More complex networks, however, may allow additional parameter confounding mechanisms that have no counterpart in three-layer networks (see Figure 4b).

The sigmoid and hyperbolic tangent gain functions typically used in multilayer perceptrons do not in theory permit continuous equivalence classes (Albertini et al., 1993; Chen et al., 1993; Fefferman, 1994; Kurkova & Kainen, 1994; Sussman, 1992), but these functions can be well approximated over restricted regions of their inputs by power law or exponential functions. We have shown that for training sets with stimuli that drive the hidden units only into ranges where the approximation holds, one finds a flat ridge in the error surface along the continuum of networks that are functionally equivalent to the true network (see Figure 6d), as predicted by the approximating function. It can be impossible to use these restricted training sets to uniquely recover the true model parameters, which are needed for correctly predicting the responses across all types of stimuli (see Figure 9). As a related topic, it is shown in appendix C that continuous parameter confounding, which can cause a ridge in the error surface, is related to the degeneracy of the Fisher information matrix (Wei, Zhang, Cousseau, Ozeki, & Amari, 2008).

For practical applications to neural network training, our results suggest that it may be useful to monitor the activities of the hidden units in order to identify parameter variability due to continuous parameter confounding, which can be predicted based on approximations of the used range of a gain function by the three types of functions discussed in this article. One may reduce parameter confounding by increasing either the amount of data or the dynamic range of the net inputs to the gain functions (see Figure 6). To improve neural network modeling of neurophysiological stimulus-response data, one should try to include stimuli that can drive hidden units activities over a wide dynamic range. One possibility for choosing proper stimuli is to use optimal experimental design methods (Atkinson & Donev, 1992) to adaptively generate stimuli that extract the most information about model parameters (Cohn, 1996; MacKay, 1992; Paninski, 2005). However, for some neural network structures, it is possible that no sensory stimuli can effectively drive one or more hidden units over a wide enough dynamic range, thus making unique identification of network structure from input-output measurements intractable. In such cases, this inherent limitation can be overcome only by new experimental methodology that can directly collect information from neurons in the hidden layers or directly measure network connectivity.

Appendix A: Exponential and Logarithm as the Limits of Power Function

An exponential gain function can be approximated by a power function because

$$(1 + u/z)^z \to e^u \tag{A.1}$$

in the limit of large z as $z \to \infty$, whereas a logarithmic gain function can be approximated by a power function because

$$(u^z - 1)/z \to \ln u \tag{A.2}$$

in the limit of small z as $z \to 0$. When a power function closely approximates the logarithm, the logarithmic confounding as in Figure 1c occurs approximately (see Figure 8).

Now we examine how the three main types of gain functions are related in the light of the limits. We start with the power gain function solution in equation 5.23,

$$g(u) = -c/q + A\,|a + bu|^{-q/b}, \tag{A.3}$$

which has been derived under the assumption that $q \neq 0$ and $b \neq 0$. Since the exponential solution in equation 5.22 assumes $b = 0$, we take the limit of $g(u)$ in equation A.3 as $b \to 0$ and indeed obtain the same exponential solution as in equation 5.22:

$$\lim_{b\to 0} g(u) = \lim_{b\to 0}\left(-c/q + A\,|a + bu|^{-q/b}\right) = -c/q + Ae^{-(q/a)u}. \tag{A.4}$$

This is because

$$|a + bu|^{-q/b} = |a|^{-q/b}\left(|1 + u/z|^z\right)^{-q/a} \to 1\,(e^u)^{-q/a} = e^{-(q/a)u}, \tag{A.5}$$

according to equation A.1, as $z \to \infty$ or $b \to 0$, assuming $z = a/b$ and $q/b > 0$.

The logarithmic solution in equation 5.21 assumes $q = 0$. Simply taking the limit of equation A.3 as $q \to 0$ yields the indefinite result $\lim_{q\to 0} g(u) = \infty$. To obtain a proper limit compatible with solution 5.21, we may put $c = -qA_0 + c_0$ and $A = c_0/q$ in equation A.3, where A_0 and c_0 are new constants corresponding to A and c in equation 5.21. Then we have

$$g(u) = A_0 + (c_0/q)\left(\left(|a + bu|^{-1/b}\right)^q - 1\right) \to A_0 + c_0 \ln |a + bu|^{-1/b} \tag{A.6}$$

as $q \to 0$, where the limit follows from equation A.2 with $z = q$. Taken together, now we have

$$\lim_{q\to 0} g(u) = \lim_{q\to 0}\left(-c/q + A\,|a + bu|^{-q/b}\right) = A_0 - (c_0/b)\ln|a + bu|. \tag{A.7}$$

Thus, the logarithmic solution can be taken formally as a limit of the power solution only after suitable adjustment of parameters, as shown above.

Appendix B: Additive and Multiplicative Inputs and Outputs

Given a gain function g and two inputs x and y, the outputs are $g(x)$ and $g(y)$, respectively. When the sum $x + y$ or the product xy is presented as the new input, the new output may be equal to the sum $g(x) + g(y)$ or the product $g(x)g(y)$ of the old outputs, depending on the exact form of the gain function. Below we consider all four possibilities in detail by solving functional equations (Aczél & Dhombres, 1989). We assume differential gain functions and ignore the trivial solution $g(x) = 0$.

Case B(i): *Additive inputs are mapped to additive outputs.* The gain function is characterized by the equation

$$g(x + y) = g(x) + g(y). \tag{B.1}$$

To solve this functional equation, first take the partial derivative $\partial/\partial y$ to obtain $g'(x + y) = g'(y)$. Setting $y = 0$ gives $g'(x) = g'(0) \equiv A$, so that $g(x) = Ax + C$. Substitution back into equation B.1 gives $C = 0$. Thus,

$$g(x) = Ax. \tag{B.2}$$

Case B(ii): *Multiplicative inputs are mapped to additive outputs.* The gain function is characterized by

$$g(xy) = g(x) + g(y). \tag{B.3}$$

Take the partial derivative $\partial/\partial y$, and then set $y = 1$ to obtain $xg'(x) = g'(1) \equiv A$. Thus,

$$g(x) = A \ln |x|. \tag{B.4}$$

Case B(iii): *Additive inputs are mapped to multiplicative outputs.* The gain function is characterized by

$$g(x + y) = g(x)g(y). \tag{B.5}$$

Take the partial derivative $\partial/\partial y$ and then set $y = 0$ to obtain $g'(x) = g(x)\alpha$ with $\alpha = g'(0)$. The general solution is of the form

$$g(x) = e^{\alpha x}. \tag{B.6}$$

Case B(iv): *Multiplicative inputs are mapped to multiplicative outputs.* The gain function is characterized by

$$g(xy) = g(x)\,g(y)\,. \tag{B.7}$$

Take the partial derivative $\partial/\partial y$ and then set $y = 1$ to obtain $xg'(x) = g(x)\,g'(1)$, whose general solution has the form $g(x) = c\,|x|^{\alpha}$ with $\alpha = g'(1)$. Substituting this solution back to equation B.7 and then setting $x = y = 1$, we obtain $c = c^2$, which means $c = 1$ or 0. Thus, the general solution to equation B.7 is

$$g(x) = |x|^{\alpha}\,. \tag{B.8}$$

The argument for the four cases can be readily generalized to allow additional scaling and shifting for both the inputs and outputs. The generalized solutions can be made identical to the gain functions shown in Table 1. For example, we can replace equation B.7 by the equation $Ag(xy) = g(x)\,g(y)$ to obtain the solution $g(x) = A\,|x|^{\alpha}$ instead of equation B.8. More generally,

$$A(g(xy + B) - C) = (g(x + B) - C)\,(g(y + B) - C) \tag{B.9}$$

yields the solution

$$g(x) = A|x - B|^{\alpha} + C, \tag{B.10}$$

which is of the same form as that in Table 1. The other three cases can be generalized similarly.

Appendix C: A Relation Between Continuous Parameter Confounding and Fisher Information Degeneracy

In this appendix we show that a network with either Poisson or gaussian noise permits continuous parameter confounding only if its Fisher information matrix is degenerate for all inputs. Since a necessary and sufficient condition for continuous parameter confounding is equation 2.2 or 2.4, we need only to show that equation 2.4 implies Fisher information degeneracy. Given the mean output $f(x, \theta)$ for input x and network parameter set $\theta = (\theta_1, \ldots, \theta_k)$, we rewrite equation 2.4 as

$$\mathbf{q}^{\mathrm{T}} \nabla f = 0, \tag{C.1}$$

where both coefficient set $\mathbf{q} = (q_1, \ldots, q_k)^{\mathrm{T}}$ and gradient $\nabla f = (\partial f/\partial\theta_1, \ldots, \partial f/\partial\theta_k)^{\mathrm{T}}$ are taken as column vectors, with T indicating transpose.

When Poisson or gaussian noise is added to the mean output, the Fisher information matrix with respect to the parameter set is

$$\mathbf{J} = \frac{1}{V} \nabla f \nabla f^{\mathrm{T}}, \tag{C.2}$$

where the variance $V = f$ for Poisson noise, and $V = \sigma^2$ for gaussian noise (Seung & Sompolinsky, 1993).

Now suppose equation C.1 holds for all inputs x for some fixed nonzero vector $\mathbf{q}$ that is independent of x; then we have a vanishing quadratic form:

$$\mathbf{q}^{\mathrm{T}}\mathbf{J}\mathbf{q} = \frac{1}{V}\left(\mathbf{q}^{\mathrm{T}}\nabla f\right)\left(\nabla f^{\mathrm{T}}\mathbf{q}\right) = \frac{1}{V}\left(\mathbf{q}^{\mathrm{T}}\nabla f\right)^2 = 0, \tag{C.3}$$

where the first step follows from equation C.2 and the last step follows from equation C.1. Since $\mathbf{q}$ is nonzero, the vanishing quadratic form $\mathbf{q}^{\mathrm{T}}\mathbf{J}\mathbf{q} = 0$ means that the Fisher information matrix $\mathbf{J}$ must be degenerate for all inputs, or

$$\operatorname{rank}\mathbf{J} < \dim\theta = k. \tag{C.4}$$

This proves the statement at the beginning of this appendix. The converse statement is not necessarily true. Although a degenerate Fisher information matrix $\mathbf{J}$ implies $\mathbf{q}^{\mathrm{T}}\mathbf{J}\mathbf{q} = 0$ for some vector $\mathbf{q}$, which in turn implies equation C.1, there is no guarantee that $\mathbf{q}$ is independent of input x, as required in confounding equation, C.1.

Acknowledgments

Supported partially by grant NSF IIS-0827695.

References

Achard, P., & De Schutter, E. (2006). Complex parameter landscape for a complex neuron model. *PLoS Comput Biol., 2*(7), e94.

Aczél, J., & Dhombres, J. G. (1989). *Functional equations in several variables*. Cambridge: Cambridge University Press.

Adelson, E. H., & Bergen, J. R. (1985). Spatiotemporal energy models for the perception of motion. *J. Opt. Soc. Am. A, 2*(2), 284–299.

Albertini, F., Sontag, E. D., & Maillot, V. (1993). Uniqueness of weights for neural networks. In R. J. Mammone (Ed.), *Artificial neural networks for speech and vision*. London: Chapman & Hall.

Anderson, J. S., Lampl, I., Gillespie, D. C., & Ferster, D. (2000). The contribution of noise to contrast invariance of orientation tuning in cat visual cortex. *Science, 290*(5498), 1968–1972.

Atkinson, A. C., & Donev, A. N. (1992). *Optimum experimental designs*. Oxford: Clarendon Press.

Ben-Israel, A., & Greville, T. N. E. (2003). *Generalized inverses: Theory and applications* (2nd ed.). New York: Springer-Verlag.

Chen, A. M., Lu, H., & Hecht-Nielsen, R. (1993). On the geometry of feedforward neural network error surfaces. *Neural Comput., 5*, 910–927.

Cohn, D. A. (1996). Neural network exploration using optimal experimental design. *Neural Networks, 9*(6), 1071–1083.

Colburn, H. S. (1973). Theory of binaural interaction based on auditory-nerve data. I. General strategy and preliminary results on interaural discrimination. *J. Acoust. Soc. Am., 54*(6), 1458–1470.

Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems, 2*, 303–314.

DiMattina, C., & Zhang, K. (2008). How optimal stimuli for sensory neurons are constrained by network architecture. *Neural Comput., 20*(3), 668–708.

Ermentrout, B. (1998). Linearization of F-I curves by adaptation. *Neural Comput., 10*(7), 1721–1729.

Fefferman, C. (1994). Reconstructing a neural net from its output. *Revista Mathematica Iberoamerica, 10*(3), 507–555.

Funahashi, K. (1989). On the approximate realization of continuous mappings by neural networks. *Neural Networks, 2*, 183–192.

Gabbiani, F., Krapp, H. G., Koch, C., & Laurent, G. (2002). Multiplicative computation in a visual neuron sensitive to looming. *Nature, 420*(6913), 320–324.

Hancock, K. E., Davis, K. A., & Voigt, H. F. (1997). Modeling inhibition of type II units in the dorsal cochlear nucleus. *Biol. Cybern., 76*(6), 419–428.

Hansel, D., & Van Vreeswijk, C. (2002). How noise contributes to contrast invariance of orientation tuning in cat visual cortex. *J. Neurosci., 22*(12), 5118–5128.

Heeger, D. J. (1992). Half-squaring in responses of cat striate cells. *Vis. Neurosci., 9*(5), 427–443.

Hornik, K., Stinchcombe, M., & White, H. (1989). Multi-layer feed-forward neural networks are universal approximators. *Neural Netw., 2*, 359–366.

Kurkova, V., & Kainen, P. C. (1994). Functionally equivalent feedforward neural networks. *Neural Comput., 6*, 543–558.

Lau, B., Stanley, G. B., & Dan, Y. (2002). Computational subunits of visual cortical neurons revealed by artificial neural networks. *Proc. Natl. Acad. Sci. U.S.A., 99*(13), 8974–8979.

Lehky, S. R., Sejnowski, T. J., & Desimone, R. (1992). Predicting responses of nonlinear neurons in monkey striate cortex to complex patterns. *J. Neurosci., 12*(9), 3568–3581.

MacKay, D. J. C. (1992). Information-based objective functions for active data selection. *Neural Comput., 4*, 590–604.

Marder, E., & Goaillard, J. M. (2006). Variability, compensation and homeostasis in neuron and network function. *Nat. Rev. Neurosci., 7*(7), 563–574.

Miller, K. D., & Troyer, T. W. (2002). Neural noise can explain expansive, power-law nonlinearities in neural response functions. *J. Neurophysiol., 87*(2), 653–659.

Nelder, J. A., & Mead, R. (1965). A simplex method for function minimization. *Computer Journal, 7*, 308–313.

Nelken, I., & Young, E. D. (1994). Two separate inhibitory mechanisms shape the responses of dorsal cochlear nucleus type IV units to narrowband and wideband stimuli. *J. Neurophysiol., 71*(6), 2446–2462.

Olypher, A. V., & Calabrese, R. L. (2007). Using constraints on neuronal activity to reveal compensatory changes in neuronal parameters. *J. Neurophysiol., 98*, 3749–3758.

Paninski, L. (2005). Asymptotic theory of information-theoretic experimental design. *Neural Comput., 17*(7), 1480–1507.

Poirazi, P., Brannon, T., & Mel, B. W. (2003). Pyramidal neuron as two-layer neural network. *Neuron, 37*(6), 989–999.

Pollen, D. A., & Ronner, S. F. (1983). Visual cortical neurons as localized spatial frequency analyzers. *IEEE Transactions on Systems, Man and Cybernetics, 13*(5), 907–916.

Powell, M. J. D. (1978). A fast algorithm for nonlinear constrained optimization calculations. In G. A. Watson (Ed.), *Numerical analysis*. Berlin: Springer-Verlag.

Prenger, R., Wu, M. C., David, S. V., & Gallant, J. L. (2004). Nonlinear V1 responses to natural scenes revealed by neural network analysis. *Neural Netw., 17*(5–6), 663–679.

Prinz, A. A., Bucher, D., & Marder, E. (2004). Similar network activity from disparate circuit parameters. *Nat. Neurosci., 7*(12), 1345–1352.

Rumelhart, D. E., Hinton, G. E., & McClelland, J. L. (1986). Learning internal representations by error propagation. In D. E. Rumelhart & J. L. McClelland (Eds.), *Parallel distributed processing: Explorations in the microstructure of cognition* (Vol. 1). Cambridge, MA: MIT Press.

Seung, H. S., & Sompolinsky, H. (1993). Simple models for reading neuronal population codes. *Proc. Natl. Acad. Sci. USA, 90*(22), 10749–10753.

Smith, M. R., Nelson, A. B., & Du Lac, S. (2002). Regulation of firing response gain by calcium-dependent mechanisms in vestibular nucleus neurons. *J. Neurophysiol., 87*(4), 2031–2042.

Stafstrom, C. E., Schwindt, P. C., & Crill, W. E. (1984). Repetitive firing in layer V neurons from cat neocortex in vitro. *J. Neurophysiol., 52*(2), 264–277.

Sussman, H. J. (1992). Uniqueness of the weights for minimal feedforward nets with a given input-output map. *Neural Networks, 5*, 589–593.

Weaver, C. M., & Wearne, S. L. (2008). Neuronal firing sensitivity to morphologic and active membrane parameters. *PLoS Comput Biol, 4*(1), e11.

Wei, H., Zhang, J., Cousseau, F., Ozeki, T., & Amari, S. (2008). Dynamics of learning near singularities in layered networks. *Neural Comput., 20*(3), 813–843.

Wu, M. C., David, S. V., & Gallant, J. L. (2006). Complete functional characterization of sensory neurons by system identification. *Annu. Rev. Neurosci., 29*, 477–505.

Young, E. D., & Davis, K. A. (2002). Circuitry and function of the dorsal cochlear nucleus. In D. Oertel, R. R. Fay, & A. N. Popper (Eds.), *Integrative functions in the mammalian auditory pathway* (pp. 160–206). New York: Springer.

Zipser, D., & Andersen, R. A. (1988). A back-propagation programmed network that simulates response properties of a subset of posterior parietal neurons. *Nature, 331*(6158), 679–684.

Received May 7, 2008; accepted May 14, 2009.