frontiers in Computational Neuroscience

ORIGINAL RESEARCH
published: 29 February 2016
doi: 10.3389/fncom.2016.00015

Obtaining Arbitrary Prescribed Mean Field Dynamics for Recurrently Coupled Networks of Type-I Spiking Neurons with Analytically Determined Weights

Wilten Nicola[1]*, *Bryan Tripp*[2,3] and *Matthew Scott*[1]

[1] Department of Applied Mathematics, University of Waterloo, Waterloo, ON, Canada, [2] Department of Systems Design Engineering, University of Waterloo, Waterloo, ON, Canada, [3] Center for Theoretical Neuroscience, University of Waterloo, Waterloo, ON, Canada

A fundamental question in computational neuroscience is how to connect a network of spiking neurons to produce desired macroscopic or mean field dynamics. One possible approach is through the Neural Engineering Framework (NEF). The NEF approach requires quantities called decoders which are solved through an optimization problem requiring large matrix inversion. Here, we show how a decoder can be obtained analytically for type I and certain type II firing rates as a function of the heterogeneity of its associated neuron. These decoders generate approximants for functions that converge to the desired function in mean-squared error like 1/N, where N is the number of neurons in the network. We refer to these decoders as scale-invariant decoders due to their structure. These decoders generate weights for a network of neurons through the NEF formula for weights. These weights force the spiking network to have arbitrary and prescribed mean field dynamics. The weights generated with scale-invariant decoders all lie on low dimensional hypersurfaces asymptotically. We demonstrate the applicability of these scale-invariant decoders and weight surfaces by constructing networks of spiking theta neurons that replicate the dynamics of various well known dynamical systems such as the neural integrator, Van der Pol system and the Lorenz system. As these decoders are analytically determined and non-unique, the weights are also analytically determined and non-unique. We discuss the implications for measured weights of neuronal networks.

Keywords: mean field analysis, neural engineering framework, neuronal heterogeneity, integrate-and-fire neurons, recurrently coupled networks, synaptic weights

OPEN ACCESS

Edited by:
Boris Gutkin,
École Normale Supérieure, France

Reviewed by:
Alex Roxin,
Centre de Recerca Matemàtica, Spain
Brian J. Fischer,
Seattle University, USA

***Correspondence:**
Wilten Nicola
wnicola@uwaterloo.ca

Received: *17 November 2015*
Accepted: *05 February 2016*
Published: *29 February 2016*

Citation:
Nicola W, Tripp B and Scott M (2016) Obtaining Arbitrary Prescribed Mean Field Dynamics for Recurrently Coupled Networks of Type-I Spiking Neurons with Analytically Determined Weights. Front. Comput. Neurosci. 10:15. doi: 10.3389/fncom.2016.00015

1. INTRODUCTION

There are many spiking models that exist in the literature that can be fit to reproduce the membrane potential and the firing rates of real neurons. Examples include the leaky integrate and fire neuron, the Izhikevich model (Izhikevich, 2003, 2007), the theta model (Ermentrout and Kopell, 1986), the quartic integrate and fire model (Touboul, 2008) and the adaptive exponential integrate and fire model (Brette and Gerstner, 2005; Naud et al., 2008). When these models are coupled together to form networks, one can predict the the macroscopic or mean field behavior of a network of

these neurons via a suitably derived mean field system (Nicola and Campbell, 2013a,b; Nesse et al., 2008). This can even be done when one considers the effects of heterogeneity in the neurons (Nicola and Campbell, 2013b).

While the mean field system for networks of neurons with prescribed sources of heterogeneity is important for predicting the behavior of the network of equal importance is the inverse problem: given a particular macroscopic behavior or mean field system, what distributions of heterogeneity, either in the neuronal parameters themselves, or the synaptic weights are required to produce said behavior?

One possible numerical solution to the inverse problem is through the Neural Engineering Framework (NEF) (Eliasmith and Anderson, 2004). In the NEF approach, one can specify the macroscopic dynamics or mean field for a spiking neuronal network. Given a network of neurons with a source of heterogeneity, one can find a set of optimal linear weights, referred to as linear decoders, for their firing rates in such a way that the weighted linear sum of the firing rates optimally approximates any function of choice. This allows for specifying the network connectivity in such a way as to obtain arbitrary dynamics from the network(s) of neurons (Eliasmith, 2005). For example, the NEF has been used to develop a wide variety of models, including the most behaviorally sophisticated spiking neural model to date (Eliasmith et al., 2012) as well as more specialized models of path integration (Conklin and Eliasmith, 2005), working memory (Singh and Eliasmith, 2006), visual attention (Bobier et al., 2014), motor control (DeWolf and Eliasmith, 2011), various cognitive functions (Bekolay et al., 2014; Rasmussen and Eliasmith, 2014), and many others.

However, the optimality requirement in the linear decoders introduces complications in the NEF approach. The optimal decoders are computed via least-squares optimization which is a computationally-intensive process; and yet very little information about the network can be determined once the optimal decoders have been obtained. Additionally, one cannot determine how the distribution of heterogeneity in the tuning curves of the neurons is related to the other distributions across the network, such as the distribution of connection weights.

Here, we will show that if one loosens the optimality requirement in the linear decoders, it is possible to obtain linear decoders that converge to any function of choice in the large network limit. Due to their form, we will refer to these decoders as scale-invariant linear decoders. These scale-invariant decoders have several advantages over optimal decoders, at the primary cost of a slower convergence rate in network size. However, using any gradient descent algorithm that does not directly compute the Hessian, one can decrease the error of the scale-invariant decoders very rapidly with very few iterations for any finite network size.

In Section 2.1, we will quickly introduce the NEF. A more thorough introduction can be found in (Eliasmith and Anderson, 2004). This will be followed by Section 2.2 where we will demonstrate that as the networks become arbitrarily large, the optimal decoders tend to an asymptotic limit. This will be our motivation in defining a scale-invariant decoder. In Section 2 we will determine what this asymptotic limit is for the scalar case and for multivariable functions in Section 2.4. In Section 3 we will demonstrate how the decoders can yield the weights to couple neurons together and simulate spiking networks with the specified dynamics by using these weights.

2. METHODS: DETERMINING THE DECODER SURFACE

2.1. The Neural Engineering Framework

Suppose we knew the firing rate of a class of neurons, $f(I)$ as a function of the input current I. Then we can take any input variable x and linearly transform it into a current via $I = \alpha x + \beta$. If we allow α and β to be drawn from a random distribution, then we can generate a network of neurons with firing rates $f(\alpha_i x+\beta_i)$ where α_i, β_i are drawn from some specified probability distribution $\rho_{\alpha,\beta}(\alpha, \beta)$. As a function of x the curve $f(\alpha_i x + \beta_i)$ is typically referred to as the tuning curve of neuron i. The output of these neurons is the sum of their weighted firing rates:

$$\hat{g}_N(x) = \sum_{i=1}^{N} \phi_i f(\alpha_i x + \beta_i). \tag{1}$$

Thus, the network takes any input x belonging to the appropriate space, and transforms it into some function $\hat{g}_N(x)$. If for example we wanted to compute the function $g(x)$, we would need to pick ϕ_i such that $\hat{g}_N(x) \approx g(x)$. The ϕ_i are referred to as the linear decoders in the NEF approach (Eliasmith and Anderson, 2004). They can be determined by minimizing the the following functional with respect to ϕ over some region X in x (Salinas and Abbott, 1995; Eliasmith and Anderson, 2004):

$$\begin{aligned} C(\phi) &= \int_X (\hat{g}_N(x) - g(x))^2 \, dx + \lambda \sum_{i=1}^{N} \phi_i^2 \\ &= \int_X \left(\sum_{i=1}^{N} \phi_i f(\alpha_i x + \beta_i) - g(x) \right)^2 dx + \lambda \sum_{i=1}^{N} \phi_i^2 \end{aligned} \tag{2}$$

where the first term in 2 corresponds to the error in the approximation and the second term penalizes large ϕ_i. Minimizing $C(\phi)$ for ϕ yields the following linear system of equations:

$$\Phi = A^{-1}\Gamma \tag{3}$$

$$A_{ij} = \int_X f(\alpha_i x + \beta_i) f(\alpha_j x + \beta_j) \, dx + \delta_{ij}\lambda \tag{4}$$

$$\Gamma_j = \int_X f(\alpha_j x + \beta_j) g(x) \, dx. \tag{5}$$

Equations (3–5) correspond to standard function approximation (Bishop, 1995), although the basis functions f are randomly drawn. We will refer to the optimal decoders as Φ and any other decoder as ϕ. There are various functions f that have appeared in the literature. These are derived from complicated neural models using topological normal form theory (Ermentrout and Kopell, 1986; Izhikevich, 2007), are fits to experimental data from real neurons (Shriki et al., 2006), or are analytically derived from

integrate and fire neurons. The general form the integrate-and-fire models we will consider is given by

$$\dot{v} = F(v) + I \tag{6}$$

$$v(t^-) = v_{peak}, \quad \rightarrow v(t^+) = v_{reset} \tag{7}$$

$$f(I) = \begin{cases} \left(\int_{v_{reset}}^{v_{peak}} \frac{dv}{F(v)+I} \right)^{-1} & I > 0 \\ 0 & I < 0 \end{cases}. \tag{8}$$

Specific examples include:

$$F(v) = -\frac{v}{\tau_v} \quad \text{(Leaky Integrate-and-Fire Model (Lapicque, 1907 Abbott, 1999; Brunel and Van Rossum, 2007)} \tag{9}$$

$$F(v) = v^2 \quad \text{(Quadratic Integrate-and-Fire Model) (Izhikevich, 2003, 2007)} \tag{10}$$

$$F(v) = v^2, \quad v_{reset} = -\infty, \quad v_{peak} = \infty \quad \text{(Theta Model) (Ermentrout and Kopell, 1986)} \tag{11}$$

$$F(v) = \exp(v) - v \quad \text{(Exponential-Integrate-and-Fire Model) (Brette and Gerstner, 2005; Naud et al., 2008).} \tag{12}$$

Other FI curves are fits to the measured FI curves of more sophisticated conductance based models or experimental measurements. For example, the function

$$f(I) = \begin{cases} I + c & I > 0 \\ 0 & I < 0 \end{cases} \tag{13}$$

can be fit to type-II firing rates when $c > 0$, and can be shown to be the steady state firing rate for neurons that display spike frequency adaptation when $c = 0$ (Ermentrout, 2006). Equation (13) has also been fit to conductance based models (Shriki et al., 2006) (with $x = 0$) and adequately describes the FI curves for many real cortical neurons (Stafstrom et al., 1984; Azouz et al., 1997; Ahmed et al., 1998).

As x is often thought of as a real world input variable in the NEF approach, the α_i, β_i distribution can only be known once one specifies a distribution of maximal firing rates, r_i and x-intercepts, a_i for the tuning curves. For the time being, we will restrict the variable x to the interval $[-1, 1]$. It can be rescaled to an arbitrary interval, so this is no loss of generality. We will show later how x can also be extended to a vector as in the original NEF framework (Eliasmith and Anderson, 2004; Eliasmith, 2005). Once one specifies the distribution of (r_i, a_i), one can obtain a transformation of random variables. First, let us consider a population of neurons such that the maximal firing rate is achieved at $x = 1$:

$$r_i = f(\alpha_i + \beta_i)$$
$$0 = \alpha_i a_i + \beta_i$$

We will refer to these neurons as ON neurons as the neurons can either increase in firing rate with respect to x (ON neurons) or decrease (OFF neurons). The maximal firing rate for the ON population is reached at $x = 1$ (Eliasmith and Anderson, 2004) while the maximal firing rate for the OFF population is reached at $x = -1$. To generate a population of OFF neurons, we can reflect the tuning curves in the $x = 0$ axis by multiplying α_i by -1, which yields the following pair of transformations:

$$\alpha_i = \pm \frac{f^{-1}(r_i)}{1 - a_i} \tag{14}$$

$$\beta_i = -\frac{a_i f^{-1}(r_i)}{1 - a_i} \tag{15}$$

where the $\pm$ indicates ON/OFF, respectively and by $f^{-1}(x)$. Note that one can generate a population of ON and OFF neurons in different ways, for example by reflecting about the $x = a_i$ axis for each neuron. We will treat a_i and r_i as our primary sources of heterogeneity in the case of approximating a function of a single variable and we will assume that the marginal densities are given by $\rho_a(a)$ and $\rho_r(r)$. Furthermore, we will write $a_i^{\pm}$ and $r_i^{\pm}$ to distinguish between the heterogeneous parameters for the ON(+) and OFF(−) populations. In this case, we can rewrite the sum (1) as

$$\hat{g}_N(x) = \sum_{i=1}^{N/2} \phi_i^+ f\left(f^{-1}(r_i^+) \left(\frac{x - a_i^+}{1 - a_i^+} \right) \right) + \sum_{i=1}^{N/2} \phi_i^- f\left(f^{-1}(r_i^-) \left(\frac{-x - a_i^-}{1 - a_i^-} \right) \right) \tag{16}$$

where the first half represents the population of ON neurons and the second half of the sum represents the population of OFF neurons. Note that when we refer to the heterogeneous parameters r_i^{max} and a_i for the ON and OFF populations, we do not imply that they have the same value for both populations for $i = 1, 2, \ldots N$ and they should have a superscript $\pm$ to denote which population the parameter belongs to. We do not include this superscript and note the abuse of notation for readers.

Suppose, for example, we wanted to approximate the function $g(x) = x$ using a population of 50 quadratic integrate and fire tuning curves with 25 ON and 25 OFF neurons. This is shown in **Figure 1** where the decoders are given by Equation (5). Note that reasonable accuracy is achieved despite the small population of neurons.

So far this has been fairly standard function approximation with a non-orthogonal basis (Bishop, 1995). The difference in the NEF approach is that one uses these linear decoders obtained from the firing rate curves to design a network of spiking neurons and the function is represented in the output of the network simulation. For example, the differential equation for the quadratic integrate and fire model is given by

$$\dot{v}_i = v_i^2 + \alpha_i x + \beta_i \tag{17}$$

where if $v(t^-) = \infty$, $v(t^+) = -\infty$. This can be written as the equivalent θ model with the transformation $v = \tan(\theta/2)$ yielding:

$$\dot{\theta}_i = 1 - \cos(\theta_i) + (1 + \cos(\theta_i))(\alpha_i x + \beta_i) \tag{18}$$

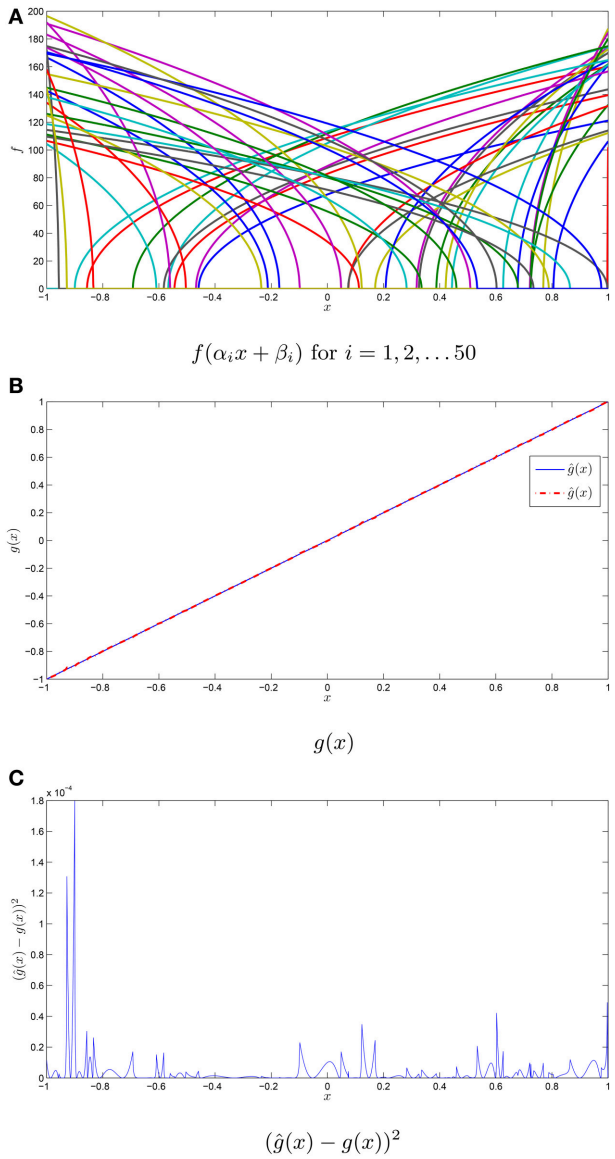


FIGURE 1 | Function approximation with neuronal tuning curves. (A) The tuning curves for a population of 50 quadratic integrate and fire neurons with their intercepts and maximum firing rates drawn from independent uniform random variables on [−1, 1] and [100, 200]. The maximum firing rate is in Hz. **(B)** The function $g(x) = x$, in addition to the approximation $\hat{g}(x)$ using the tuning curves from **(A)**. **(C)** The squared error in the approximation $\hat{g}(x)$.

which produces a spike when $\theta(t^-) = \pi$ and is reset to $\theta(t^+) = -\pi$.

Each of these differential equations generates a sequence of action potentials at specific spike times, t_{jk} where t_{jk} is the time of the kth spike fired by the jth neuron. These spike times are then fed into a post-synaptic filter $s(t)$;

$$\dot{s}(t) = -\frac{s(t)}{\tau_s} + \frac{1}{\tau_s}\sum_{j=1}^{N}\sum_{t_{jk}<t}\phi_j\delta(t - t_{jk}). \tag{19}$$

The linear decoders, ϕ_j are used to weight the spikes for their respective neuron. This post-synaptic filter equation can be explicitly integrated to yield:

$$s(t) = \sum_{j=1}^{N}\sum_{t_{jk}<t}\phi_j \exp\left(\frac{t_{jk} - t}{\tau_s}\right) = \sum_{j=1}^{N}\sum_{t_{jk}<t}\phi_j E(t - t_{jk}) \tag{20}$$

where $E(t) = \exp(-t/\tau_s)$. The integrated spike train for the jth neuron is approximately equal to its tuning curve, $f(\alpha_j x + \beta_j)$:

$$\int_0^t \sum_{t_{jk}<t}\delta(t - t_{jk})\,dt \approx \int_0^t f(\alpha_j x + \beta_j)\,dt,$$

provided that x varies on a suitably slow time scale (Dayan and Abbott, 2001; Eliasmith and Anderson, 2004). In this case, the dynamics in Equation (19) are approximately given by

$$s' = -\frac{s}{\tau_s} + \frac{1}{\tau_s}\sum_{j=1}^{N}\phi_j f(\alpha_j x + \beta_j) \tag{21}$$

This allows one to approximate an arbitrary dynamical system (Eliasmith, 2005). For example, if we consider a recurrent network ($x = s$), then to approximate the dynamics $s' = F(s)$ we merely require

$$\sum_{i=1}^{N}\phi_i f(\alpha_i s + \beta_i) \approx s + \tau_s F(s) = \hat{g}_N(s) \tag{22}$$

and where the ϕ_i are given by Equation (3). Returning to the neural equations, if we take $x = s$ and consider a recurrently coupled network of neurons then we have the following:

$$\dot{v}_i = F(v_i) + \alpha_i s + \beta_i \tag{23}$$

$$= F(v_i) + \alpha_i \sum_{j=1}^{N}\sum_{t_{jk}<t}\phi_j E(t - t_{jk}) + \beta_i \tag{24}$$

$$= F(v_i) + \sum_{j=1}^{N}\sum_{t_{jk}<t}\omega_{ij} E(t - t_{jk}) + \beta_i \tag{25}$$

where $\omega_{ij} = \alpha_i\phi_j$ is the NEF equation for the weight coupling neuron j to neuron i (Eliasmith and Anderson, 2004; Eliasmith, 2005) and the quantity

$$I_{syn,i} = \sum_{i=1}^{N}\sum_{t_{jk}<t}\omega_{ij} E(t - t_{jk}) + \beta_i$$

is the post-synaptic current going to the ith neuron.

For example, if we wanted the macroscopic dynamics to be exponential decay, $F(x) = ks$, then we require $\hat{g}(s) = s(1 + \tau_s k)$. We would obtain the ϕ_i by using Equations (3–5) which yields the optimal decoders Φ_i for $\hat{g}(x) = x(1 + \tau_s k)$ and simulate our spiking network using the weights $\omega_{ij} = \alpha_i\Phi_j$. This yields

a recurrently coupled spiking neural network with macroscopic dynamics $s' = ks$.

In addition to recurrent networks, one can also construct feedforward networks with the NEF approach. For example, we can also treat x as an input variable. This allows a network to represent an input variable x in terms of its spiking. If τ_s is not large, then one can represent the input variable x as a postsynaptic current s:

$$s \approx \sum_{i=1}^{N} \phi_i f(\alpha_i x + \beta_i) = \hat{g}_N(x) \approx x \tag{26}$$

assuming that x varies on a suitably slow time scale (slower than τ_s). This is shown for example with networks of various sizes in **Figure 2**, with a synaptic time constant of $\tau_s = 5$ ms approximating the function $g(x) = x$. The network of differential equations for the neurons is simulated using Equation (17). These neurons then generate a spike train which is weighted by the decoders. The weighted spike train is fed into the post-synaptic current variable $s(t)$, which acts as the approximation for $g(x) = x$. A time varying $x(t)$ is used that varies on a suitably slow time scale.

It is clear that given the fact that arbitrary functions or dynamics (via recurrent networks) can be computed, then one can generate multiple networks that perform different functions, and feed into one another. In this way, one could create large networks composed of interconnected subnetworks that perform functions such as controlling limbs, detecting objects, and performing tasks by using the mathematical approaches that already exist for accomplishing these feats and translating them into an equivalent neural network representation. This is the core idea in the NEF (Eliasmith and Anderson, 2004; Eliasmith et al., 2012).

Although a network of $N = 100$ neural tuning curves $f(\alpha_i x + \beta_i)$ is sufficient for a good approximation of many functions, depending on the dynamics being computed, significantly more neurons are needed in spiking simulations, as shown in **Figure 2**. Hundreds, if not thousands, of neurons are necessary for adequate approximation when spikes are used. The network size becomes even larger when we want to perform complicated functions involving more then one variable x or functions with higher frequency oscillations present. As the decoders are determined by large matrix inversion (Equation 3), this can take quite a while when dealing with more then 5000 neurons on a conventional computer. Furthermore, the smaller the synaptic time constant τ_s, the more neurons are required. This is due to the fact that Equation (21) is effectively a kernel density estimator of the firing rate and when the bandwidth is too small, the resulting estimate is under-smoothed, thus requiring more neurons for a comparable degree of accuracy

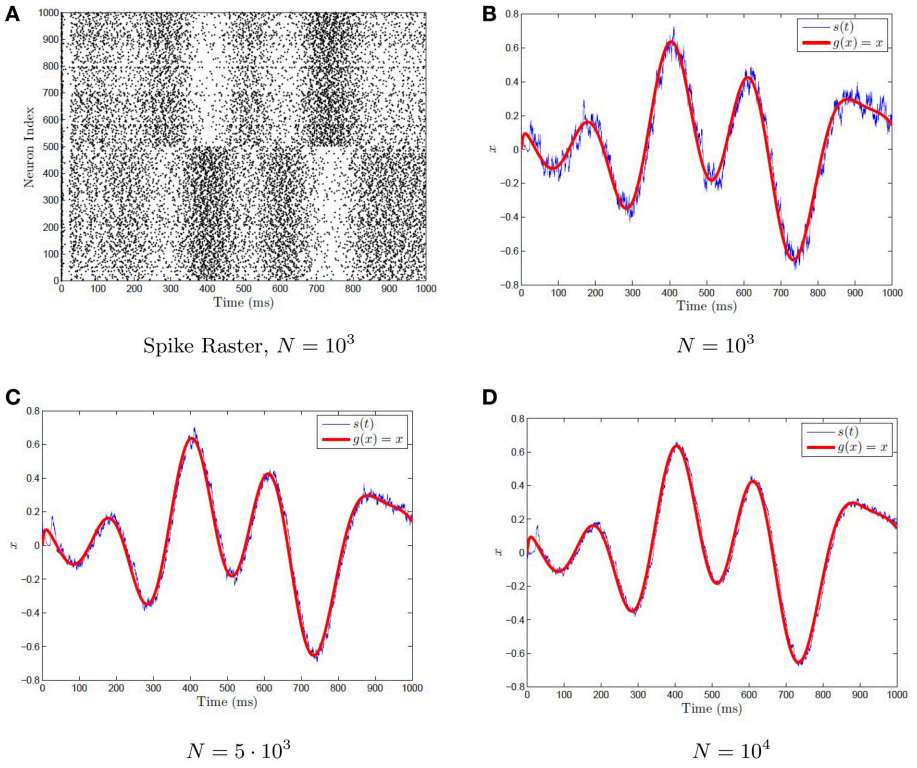


FIGURE 2 | Representation with a spiking neuronal network. (A) A sample raster plot of a neural network with 1000 neurons performing representation. **(B–D)** The representation problem is approximated by networks of various sizes by imposing the condition (26). The neurons are quadratic integrate-and-fire neurons given by Equation (17). The spike train for **(B)** is plotted in **(A)**. A time varying randomly generated signal (red) is fed into the network, and is computed via the synaptic current variable $s(t)$ using Equation 19 (blue). As the network size increases, the approximation becomes better.

as that of a network with larger τ_s. Furthermore, as the NEF weights are numerically determined (via the NEF decoders), the possible analysis is very limited. Thus, an analytical solution to the NEF decoders (and thus weights) would allow a greater insight into large networks and may also facilitate faster numerics.

2.2. Decoder Asymptotics as $N \to \infty$

In order to proceed analytically, we will first look at the behavior of the optimal decoders Φ for large networks ($N \to \infty$). To facilitate plotting, let us consider the case where for an arbitrary neural model, $f^{-1}(r_i^{\pm}) = \pm 1 - a_i^{\pm}$, that is the maximim firing rate is given by $r_i^{\pm} = f(\pm 1 - a_i^{\pm})$, which reduces the random variables associated with the heterogeneity to the intercept variable, $a_i^{\pm}$. Additionally, note that for Type-I neurons:

$$f\left(\frac{f^{-1}(r_i^{\pm})}{1-a_i^{\pm}}(\pm x - a_i^{\pm})\right) = \sqrt{\frac{r_i^{\pm 2}}{1-a_i^{\pm}}(\pm x - a_i^{\pm})} = \frac{r_i^{\pm}}{\sqrt{1-a_i^{\pm}}}\sqrt{\pm x - a_i^{\pm}} \quad (27)$$

which immediately implies that we can absorb the quantitiy $\frac{r_i^{\pm}}{\sqrt{1-a_i^{\pm}}}$ into the decoder $\phi_i^{\pm}$ and rescale any solution we obtain by this quantity at the end. With Equation (27), the sum in Equation (16) becomes:

$$\hat{g}_N(x) = \sum_{i=1}^{N/2} \Phi_i^{+} f(x - a_i^{+}) + \sum_{i=1}^{N/2} \Phi_i^{-} f(-x - a_i^{-}) \quad (28)$$

where the Φ_i are determined by Equation (5). One should note that there is an abuse of notation here, as the optimal decoders differ for the ON/OFF subpopulations for $i = 1, 2, \ldots N$ however we have used the same symbol to denote the optimal decoders, Φ_i for both populations. Additionally, the value a_i^{+} is the threshold to firing for the ith ON neuron while the quantity $-a_i^{-}$ is the threshold to firing for the ith off neuron. If one were to plot the decoders for large N, then one can easily see that in the limit of large network size ($N \to \infty$), the individual decoders vanish ($\Phi_i \to 0$, not shown here). However, for increasing N, it seems that the quantity $\gamma_i = N\Phi_i/2$ converges to some non-zero value $\gamma(a_i)$ and thus it appears that γ_i converges to some function of the x-intercept, a_i, the source of heterogeneity for the neurons. This is shown in **Figure 3** for increasingly large networks. The quantity

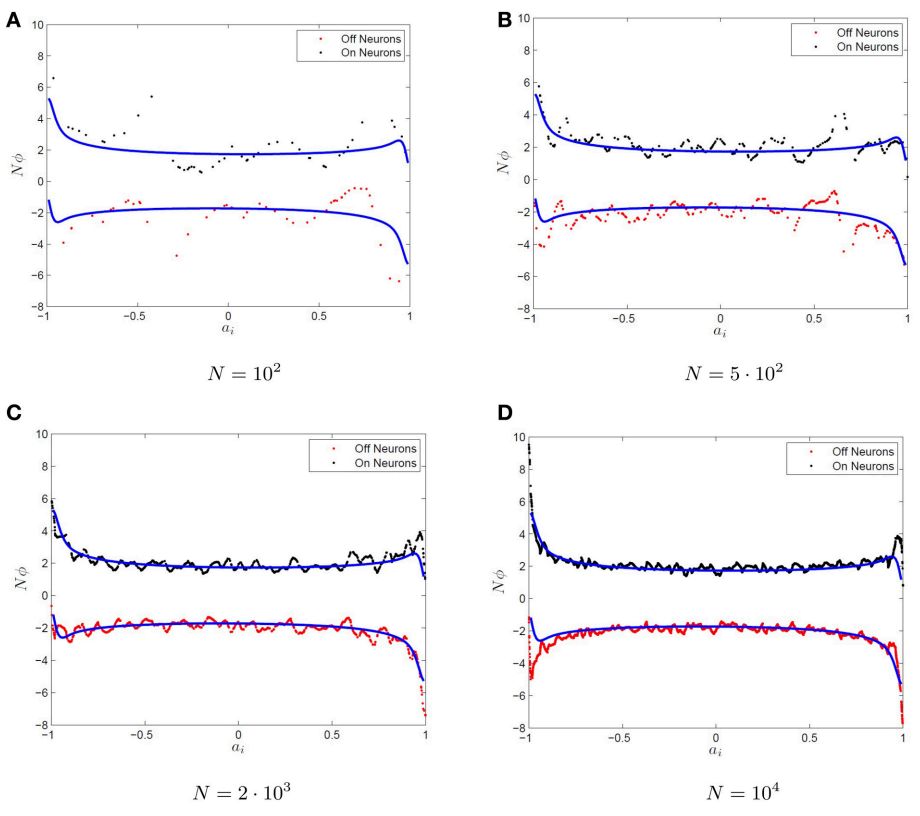


FIGURE 3 | Convergence of the optimal decoders to an invariant surface. (A–D) The function $g(x) = x$ is approximated by networks of various sizes using Equation (17) with firing rate curves. Plotted are the optimal decoders Φ_i scaled up by the network size N for ON (black dots) and OFF (red dots) neurons as a function of the intercept, $I(0)$ for QIF firing rate functions. The quantity $N\Phi_i$ appears to converge as $N \to \infty$ to the blue curve. The blue curve is determined by using a uniform grid of neurons over the heterogeneous parameter a_i and optimizing for the resulting decoders.

$N\Phi_i/2$ is plotted vs. a_i. The predicted surface for convergence, $\gamma(a)$, is also plotted which is determined by optimizing over a uniform mesh in the parameter space. We will refer to any γ_i that satisfies $\gamma_i = \phi_i N/2$ for some decoder ϕ_i as *scale invariant decoders* and $\gamma^{\pm}(a)$ as the *decoder surface*. We will not necessarily use the same decoders for the ON and OFF neurons hence the superscipt on $\gamma(a)$. We will show in the subsequent sections how to determine the decoder surfaces for the type-I and type-II (approximate) firing rates for single variable and multivariable functions.

In order to determine the decoder surface analytically, we need to understand the behavior of the network as $N \to \infty$. Using the scale invariant decoders from Equation (28):

$$\hat{g}_N(x) = \sum_{i=1}^{N/2} \phi_i^+ f(x - a_i^+) + \sum_{i=1}^{N/2} \phi_i^- f(-x - a_i^-) \quad (29)$$

$$= \frac{2}{N} \sum_{i=1}^{N/2} \gamma^+(a_i^+) f(x - a_i^+) + \frac{2}{N} \sum_{i=1}^{N/2} \gamma^-(a_i^-) f(-x - a_i^-) \quad (30)$$

$$= \overline{\gamma_i^+ f(x - a_i^+)} + \overline{\gamma_i^- f(-x - a_i^-)} \quad (31)$$

where the overline denotes the finite average over the inhomogeneity in the intercepts. We should expect that as $N \to \infty$, the finite network averaging turns into an expectation:

$$\overline{\gamma_i f(x - a_i^+)} \to E(\gamma^+(a_i^+) f(x - a_i^+)) = \int_{-1}^{x} \gamma^+(a) \rho_a(a) f(x - a)\, da \quad (32)$$

$$\overline{\gamma_i f(-x - a_i^-)} \to E(\gamma^-(a_i^-) f(-x - a_i^-)) = \int_{-1}^{-x} \gamma^-(a) \rho_a(a) f(-x - a)\, da \quad (33)$$

where $\rho_a(a)$ is the probability density describing the heterogeneity variable a_i for the neurons. For the Equations (32, 33) to be valid for the optimal decodoers, we would need to formally show that $\Phi_i \to 2\gamma(a_i)/N$, where Φ_i are the optimal decoders for some particular scale-invariant decoder $\gamma(a_i)$. However, this is unnecessary as we can regard $\phi_i = 2\gamma(a_i)/N$ as a suboptimal decoder and independent of the optimal decoders which are generated by minimizing the integral Equation (2). In which case, the limit exists by the law of large numbers. Furthermore, as we shall show later, Φ_i will not necessarily converge to $\gamma(a_i)$ as $\gamma(a_i)$ is non-unique, where as Φ_i is the optimal decoder which is unique due to the quadratic error surface in $C(\phi)$.

Note that $\gamma(a)\rho_a(a)$ appears as a product in the integral. These terms can be collapsed into a single function $\hat{P}^{\pm}(a) = \gamma^{\pm}(a)\rho_a(a)$. We will refer to these quantities as the *weighted decoders* of the ON/OFF neurons and use the weighted decoders to define the linear operators;

$$L^+(\hat{P}^+) = \int_{-1}^{x} \hat{P}^+(a) f(x - a)\, da = \hat{g}^+(x) \quad (34)$$

$$L^-(\hat{P}^-) = \int_{-1}^{-x} \hat{P}^-(a) f(-x - a)\, da = \hat{g}^-(x) \quad (35)$$

$$M(\hat{P}^+, \hat{P}^-) = L^+(\hat{P}^+) + L^-(\hat{P}^-) = \hat{g}(x). \quad (36)$$

which we will refer to as the *tuning curve transforms* (TCT). The TCTs map functions from the space of the variable(s) assigned to the heterogeneous parameters to the space of functions we are trying to approximate. Note that these operators are actually applied to different weighted decoders as the decoder surfaces are different for ON and OFF neurons. Furthermore, the density $\rho_a(a)$ need not be identical for both ON and OFF neurons. However, in all numerical implementations, $\rho_a(a)$ will be identical for the sake of simplicity.

Suppose we could determine $\hat{P}(a)$ analytically. In this case, as $\gamma(a)\rho_a(a) = \hat{P}(a)$, whenever $\rho_a(a) \neq 0$, we can compute $\gamma(a) = \hat{P}(a)/\rho_a(a)$ and leave $\gamma(a)$ undefined otherwise (as there is no neuron that has parameter(s) in this region). Now, given the fact that we obtain a linear operator as $N \to \infty$ case, the real problem becomes in finding the $(\hat{P}^+, \hat{P}^-)$ such that $M(\hat{P}^+, \hat{P}^-)$ maps to $g(x)$, the function we want to approximate. That is, we have to invert the operator M for these $\hat{P}$. If we know these $\hat{P}$, then as we presumably know the distribution of tuning curve intercepts, we can determine the decoders ϕ_i with:

$$\phi^{\pm}(a_i^{\pm}) = \frac{2\gamma^{\pm}(a_i^{\pm})}{N} = \frac{2}{N} \frac{\hat{P}^{\pm}(a_i^{\pm})}{\rho_a(a_i^{\pm})} \quad (37)$$

and thus the analytically determined scale-invariant decoders $\gamma(a_i)$ are effectively weights for a Monte Carlo estimate of the integral operator TCTs.

Here, we will explicitly invert the tuning curve transforms for single variable functions in Section 2.3. The resulting equation for the weighted decoders is a convolution integral. In Section 2.4 we will show that with a basis to basis mapping, one can also invert the tuning curve transforms for multi-variable functions.

2.3. Single Variable Functions

If we work with the operators L^+ and L^- separately, the problem becomes entirely tractable. One of the surprising things about the operators L^+ and L^- is that provided that the functions we are considering are constrained to a subset where $g^+(x)$ vanishes to first order at $x = -1$ and $g^-(x)$ vanishes to first order at $x = 1$, and are both smooth, then the operators are invertible analytically on this constrained subspace of functions using Laplace transforms (see Appendix). Additionally, by using both $\hat{P}^+(a)$ and $\hat{P}^-(a)$, one can compute any smooth function irrespective of the conditions at $x = \pm 1$. Furthermore, piecewise smooth continuous functions can also be computed (see Supplementary Materials). Closed form solutions do not exist for all neuronal firing rates as the Laplace transform cannot always be inverted explicitly. However, the type-I and type-II firing rate models do have analytically determined decoders. For the type-I/theta neuron firing rate, we have (see Appendix for derivation):

$$\hat{P}^+(a) = \frac{2}{\pi} \left(\frac{g(-1) + g(1)}{2} + g'(-1) \right) \frac{1}{\sqrt{1 + a}}$$

$$+\int_0^{a+1} g''(a-t)\frac{2}{\pi\sqrt{t}}\,dt \tag{38}$$

$$\hat{P}^-(a) = \frac{2}{\pi}\left(\frac{g(-1)+g(1)}{2} - g'(1)\right)\frac{1}{\sqrt{1+a}} + \int_0^{a+1} g''(t-a)\frac{2}{\pi\sqrt{t}}\,dt \tag{39}$$

The weighted decoder solutions for the type-II firing rate are also contained in the Appendix. For example, we have approximated the function $g(x) = \sin(2\pi x)$ using 10,000 type I tuning curves, as shown in **Figure 4**. An important thing to notice is the linearity in Equations (38,39) and (36) in the target function g and thus linearity for the scale-invariant decoders $\gamma(a)$. Furthermore, due to the fact we have considered $\hat{P}^{\pm}(a)$ to be separate for the ON and OFF populations, our operator for determining $g(x)$ is $\hat{g}(x) = M(\hat{P}^+, \hat{P}^-) = L^+(\hat{P}^+) + L^-(\hat{P}^-)$. However, while our range in L^+ and L^- was constrained, it was not constrained enough to provide a unique solution to $M(P^+, P^-) = g(x)$. In particular, if we consider any function $\epsilon(x)$ that lies in both admissable spaces (vanishes to first orders at $x = 1$ and $x = -1$), then $\epsilon(x)$ can be represented by both populations with $\hat{P}^{\pm}_{\epsilon}(a)$, respectively and thus

$$\tilde{P}^+_{g(x)+\epsilon(x)}(a) = \hat{P}^+_{g(x)}(a) + \hat{P}^+_{\epsilon(x)}(a) \tag{40}$$

$$\tilde{P}^-_{g(x)-\epsilon(x)}(a) = \hat{P}^-_{g(x)}(a) - \hat{P}^+_{\epsilon(x)}(a) \tag{41}$$

are also valid solutions to $M(P^+, P^-) = g(x)$. One can interpret this as a degree of freedom in terms of the decoders (and thus the synaptic weights). For example, we can use $e(x)$ to minimize the expected squared error or other criterion. Thus, in the following we will strictly assume that $e(x) = 0$.

We should note that it is possible to numerically invert the Laplace transforms resulting from the derivation process for the other firing rate curves. However, for now we will primarily work with the type I and type II curves. Our numerics will also primarily consist of networks of theta neurons (type-I).

2.3.1. Convergence Rate for Single-Variable Functions

With our decoder surfaces in hand, we can now proceed to determine the various convergence properties in the limit as $N \to \infty$. In particular, we have the following:

$$E_a((g(x) - \hat{g}_N(x))^2) = \frac{2}{N}\left(\int_{-1}^{x} \gamma^+(a)^2\rho_a(a)f(x-a)^2\,da - g^+(x)^2\right)$$

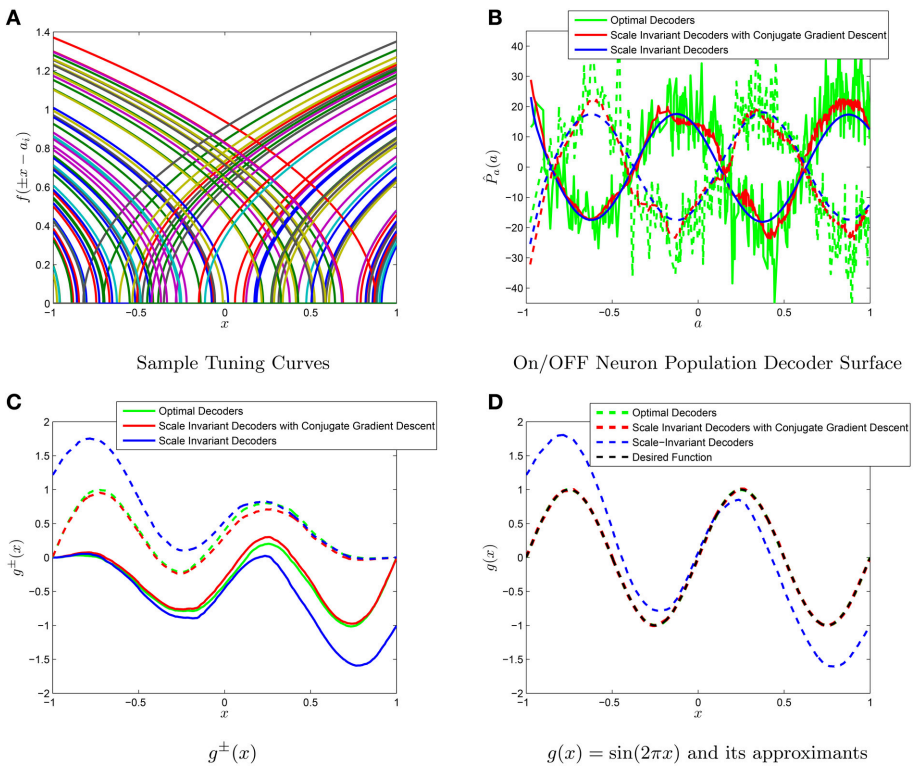


FIGURE 4 | Comparison of the scale-invariant decoders and the NEF decoders. (A) A sample subset of tuning curves from a population of 1000 neurons. The function $g(x) = \sin(2\pi x)$ is approximated using populations of $N = 10^3$ ON and OFF neurons. **(B)** The scale invariant decoders (blue), the scale invariant decoders with conjugate gradient descent fine-tuning (red) and the optimal decoders multiplied by N (green) for both the ON (solid) and OFF (dashed) groups of neurons. **(C)** The different decoders correspond to different $g^{\pm}(x)$. **(D)** Summing the $g^{\pm}(x)$ yields approximations to $g(x)$. The conjugate gradient descent improves the approximation (from $6{*}10^{-1}$ to $7{*}10^{-5}$) of magnitude while still maintaining a tight correlation with the scale-invariant decoders ($p = 0.9811$). The optimal decoders have a mean-squared error of $9{*}10^{-7}$ for comparison purposes.

$$+\frac{2}{N}\begin{pmatrix}\int_{-1}^{-x}\gamma^{-}(a)^2\rho_a(a)f(-x-a)^2\,da\\ -g^{-}(x)^2\end{pmatrix} \tag{42}$$

which immediately implies that our approximant $\hat{g}_N(x)$ converges in mean-square to $g(x)$ pointwise in x provided that:

$$\int_{-1}^{1}\gamma^{\pm}(a)^2\rho_a(a)\,da=\int_{-1}^{1}\frac{\hat{P}^{\pm}(a)^2}{\rho_a(a)}\,da<\infty. \tag{43}$$

A derivation of Equation (42) can be found in the Supplementary Materials. Letting $\gamma_N = (\gamma(a_i)/N,\ldots\gamma(a_N)/N)$, then we can also consider how the distribution of the quadratic cost function $C(\gamma_N)$ scales as $N\to\infty$; from Equation (2):

$$C(\gamma_N)=\int_X\left(\frac{1}{N}\sum_{i=1}^{N}\gamma^{e_i}(a_i)f(e_ix-a_i)-g(x)\right)^2dx+\frac{\lambda}{N^2}\sum_{i=1}^{N}\gamma(a_i)^2 \tag{44}$$

where e_i is the symbolic variable $\pm$ denoting the identify of a neuron as OFF/ON. One can show that provided that Equation (43) holds, and $\gamma^{\pm}(a)$ is bounded on $[-1, 1]$, then we have the following

$$E(C(\gamma_N))\le O(N^{-1}),\quad E(\left(C(\gamma_N)-E(C(\gamma_N))\right)^2)\le O(N^{-2}) \tag{45}$$

which implies that as $N\to\infty$, then $C(\gamma_N)\to 0$ in a mean-square sense. This is proven in the Supplementary Materials. As the cost function is strictly positive, then we can interpret this as γ_N minimizing the cost asymptotically.

For other neuronal models, one can merely use the maximum firing rates and intercepts to approximate their tuning curves with the type I standard form or the linear firing rate tuning curves. This will yield scale-invariant decoders that can be used on the tuning curves for the actual neuronal model with some degree of error. Additionally, gradient descent algorithms can be used to refine the scale-invariant decoders and that take into account the systematic error in using the type-I/type-II tuning curve approximation.

2.4. Multivariable Functions

It is clear that in the preceding section, one could approximate any arbitrary single variable function using scale invariant decoders. The same can be said about multi-variable functions. We will first introduce linear encoding for multi-variable inputs. In the NEF, it is assumed that the current input into each neuron takes the form:

$$I_i=\alpha_i\langle\boldsymbol{e},\boldsymbol{x}\rangle+\beta_i$$

where $\boldsymbol{e}$ is the encoding vector that lies on the n-dimensional unit sphere and $\boldsymbol{x}$ lies in the interior; $\langle\boldsymbol{e},\boldsymbol{x}\rangle$ is the standard dot-product. The maximum firing rate occurs when $\boldsymbol{x}=\boldsymbol{e}$, and thus $\langle\boldsymbol{e},\boldsymbol{x}\rangle = 1$ is the maximum. The vector $\boldsymbol{e}$ is also referred to as the preferred direction vector. In this case, there are no ON and OFF neurons as they are effectively taken care of by the angle in between $\boldsymbol{x}$ and $\boldsymbol{e}$. If $\boldsymbol{x}$ and $\boldsymbol{e}$ are colinear, then the maximum firing rate occurs when $\boldsymbol{e}=\boldsymbol{x}$, and the firing rate is zero when $\langle\boldsymbol{e},\boldsymbol{x}\rangle = a$, the equation for the hyperplane with normal vector $\boldsymbol{e}$. Note that because the unit sphere in one-dimension is merely ±1, we have a direct correspondence with the e_i from the single variable analysis in the previous section.

Once again, we can non-dimensionalize:

$$f(\alpha_i\langle\boldsymbol{e}_i,\boldsymbol{x}\rangle+\beta_i)=f\left(\frac{f^{-1}(r_i)}{1-a_i}(\langle\boldsymbol{e},\boldsymbol{x}\rangle-a_i)\right).$$

To simplify the situation, we will again assume that $f^{-1}(r_i) = 1-a_i$, to remove this term. As before, for type-I firing rates this occurs without any loss of generality.

While it may seem like this setup complicated matters somewhat, in the limit that $N\to\infty$, the end result is simpler then the single variable case as we can make use of the orthogonality of the trigonometric functions to derive an appropriate basis to basis mapping. Consider suboptimal decoders of the form $\phi=\frac{\gamma(\boldsymbol{e},a)}{N}=\frac{\gamma_e(\boldsymbol{e})\gamma_a(a)}{N}$. For a separable decoder we have:

$$\hat{g}_N(\boldsymbol{x})=\frac{1}{N}\sum_{i=1}^{N}\gamma_e(\boldsymbol{e}_i)\gamma_a(a_i)f(\langle\boldsymbol{e}_i,\boldsymbol{x}\rangle-a_i)$$

which in the large network limit becomes:

$$\hat{g}_N(\boldsymbol{x})=\int_{\|\boldsymbol{e}\|=1}\int_{-1}^{\langle\boldsymbol{e},\boldsymbol{x}\rangle}\gamma_e(\boldsymbol{e})\gamma_a(a)f(\langle\boldsymbol{e},\boldsymbol{x}\rangle-a)\rho_e(\boldsymbol{e})\rho_a(\boldsymbol{a})\,da\,d\boldsymbol{e} \tag{46}$$

$$=\int_{\|\boldsymbol{e}\|=1}\int_{-1}^{\langle\boldsymbol{e},\boldsymbol{x}\rangle}\hat{P}_e(\boldsymbol{e})\hat{P}_a(a)f(\langle\boldsymbol{e},\boldsymbol{x}\rangle-a)\,da\,d\boldsymbol{e} \tag{47}$$

From our previous work, we know that the weighted decoder $\hat{P}_a(a)$ can be chosen such that:

$$\int_{-1}^{\langle\boldsymbol{e},\boldsymbol{x}\rangle}\hat{P}_a(a)f(\langle\boldsymbol{e},\boldsymbol{x}\rangle-a)\,da=(\langle\boldsymbol{e},\boldsymbol{x}\rangle+1)^n \tag{48}$$

by treating $z=\langle\boldsymbol{e},\boldsymbol{x}\rangle$ and using Equation (38) to determine $\hat{P}(a)$. The specific form of $\hat{P}_a(a)$ that performs this transformation varies from neural model to neural model. For the type I/type II firing rate, it is given by a recurrence relationship in terms of the binomial exponent n and is included in the Supplementary Materials. With $\hat{P}_a(a)$ determined, the decoders $\hat{P}_e(\boldsymbol{e})$ are characterized by the integral equation:

$$\hat{g}_N(x)=\int_{\|\boldsymbol{e}\|=1}\hat{P}_e(\boldsymbol{e})(\langle\boldsymbol{e},\boldsymbol{x}\rangle+1)^n\,d\boldsymbol{e}$$

To proceed further, we will exploit the orthogonoality of the Fourier series in a hyper-spherical coordinate system. For example, in two-dimensions we have:

$$\hat{g}_N(x)=\int_0^{2\pi}\hat{P}_\theta(\theta)(\cos(\theta)x+\sin(\theta)y+1)^n\,d\theta \tag{49}$$

The second term in the integrand is a polynomial in $\cos(\theta)$ and $\sin(\theta)$. By DeMoivres formula, this can be expressed as a Fourier series with coefficients that depend on x and y where the series contains no $\cos(m\theta)$ or $\sin(m\theta)$ for $m > n$. Thus, we can extract out polynomial basis functions in the x and y using $\hat{P}(\theta) = \cos(m\theta)\sin(k\theta)$ for $m, k < n$. For example, the first few $\hat{P}(\theta)$ and the corresponding $\hat{g}_N(x)$ are shown in **Table 1**.

As we can obtain a polynomial basis where the maximum polynomial power is arbitrary, we can approximate any arbitrary integrable function. In general, one uses a sequence of trigonometric basis in the heterogeneous space to yield a polynomial basis in the function approximation space. One may wonder if the non-uniqueness in the scalar case was due to the peculiarities of the unit sphere in 1-dimension (an isolated pair of points). This turns out not be the case (see Supplementary Materials).

2.4.1. Convergence Rate for Multivariable Functions

One can again determine the order of convergence for the scale-invariant decoders for a multi-variable funciton. An application of the law of large numbers yields:

$$\begin{aligned} E(\hat{g}_N(\boldsymbol{x})) &= E\left(\gamma(\boldsymbol{e}_i)\gamma(a_i)f(\langle \boldsymbol{e}_i, \boldsymbol{x}\rangle - a_i)\right) \\ &= g(\boldsymbol{x}) \end{aligned} \tag{50}$$

$$\begin{aligned} E((\hat{g}_N(\boldsymbol{x}) - E(\hat{g}_N(\boldsymbol{x})))^2) &= E((\hat{g}_N(\boldsymbol{x}) - g(\boldsymbol{x}))^2) \\ &= \frac{1}{N}\left[E\left(\gamma(\boldsymbol{e}_i)^2\gamma(a_i)^2 f^2(\langle \boldsymbol{e}_i, \boldsymbol{x}\rangle - a_i)\right) - g(x)^2\right] \end{aligned} \tag{51}$$

and thus the expected square error converges like $1/N$ implying that $\hat{g}_N(x)$ converges to $g(x)$ in mean-squared. The expectation is taken over the random variables $\boldsymbol{e}_i$ and a_i. As the convergence rate is somewhat slow, it is natural to ask whether or not it is possible to improve the the expected squared error.

While there are many analytical paths one may take, we leave these approaches for future work. We will primarily use gradient descent variants that do not require computing the Hessian. If we knew the Hessian, then for a finite network we could immediately solve the system of Equations (3–5) as the problem is entirely quadratic and can be resolved numerically with the Hessian matrix. However, solving the quadratic problem with the Hessian requires large matrix inversion, and this is simply not feasible for large networks. Thus, we can use Hessian-free gradient descent methods. For example, one can use various conjugate gradient type algorithms to improve the expected squared error significantly with only a few iterations, and no large matrix inversion. Additionally, one can use the methods of stochastic gradient descent, such as weight perturbation, and node perturbation (Werfel et al., 2005). We will primarily use conjugate-gradient descent implemented with the PCG function in MATLAB (MATLAB, 2014). The crucial thing about these approaches is that we can obtain substantial improvements to the expected squared error with only slight perturbations to the scale-invariant decoders, as we shall see when we look at specific examples.

TABLE 1 | The basis-to-basis mapping for a polar coordinate system for the first few *n*.

$\hat{g}_N(x)$	$\hat{P}(\theta)$	n
1	$\frac{1}{2\pi}$	1
x	$\frac{\cos(\theta)}{\pi}$	1
y	$\frac{\sin(\theta)}{\pi}$	1
$x^2 + y^2 + 2$	$\frac{1}{\pi}$	2
$\frac{1}{2}(x^2 - y^2)$	$\frac{sin(2\theta)}{\pi}$	2
xy	$\frac{\cos(2\theta)}{\pi}$	2

3. RESULTS

To simulate networks with arbitrary dynamics, we can use the decoders derived in the previous sections along with neurons that correspond to the appropriate firing rates (Eliasmith and Anderson, 2004). Suppose the variable $\boldsymbol{s}(t)$ represents a vector of decoded firing rates given by the following equation:

$$s_i(t)' = -\frac{s_i}{\tau_s} + \frac{1}{\tau_s}\sum_{j=1}^{N}\phi_{ji}\sum_{t_{j,k}<t}\delta(t - t_{j,k}) \tag{52}$$

$$\boldsymbol{s} = -\frac{\boldsymbol{s}}{\tau_s} + \frac{1}{\tau_s}\sum_{j=1}^{N}\boldsymbol{\phi}_j\sum_{t_{j,k}<t}\delta(t - t_{j,k}) \tag{53}$$

$$\approx -\frac{\boldsymbol{s}}{\tau_s} + \frac{1}{\tau_s}\sum_{j=1}^{N}\boldsymbol{\phi}_j f(\langle \boldsymbol{e}, \boldsymbol{s}\rangle - a_i) \tag{54}$$

Where $\boldsymbol{\phi}_j$ is the decoder for the jth neuron. Equation (54) is referred to as the rate equation while Equation (52) is the equation for s_i under neuronal spiking with a simulated spiking neuronal network. The time constant used will be 50 ms unless otherwise stated.

Using the same procedure as before, by integrating the spiking equation for $\boldsymbol{s}(t)$ explicitly, one can derive the NEF equation for the synaptic weights:

$$\omega_{ij} = \alpha_i\langle \boldsymbol{e}_i, \boldsymbol{\phi}_j\rangle \tag{55}$$

where ω_{ij} is the synaptic weight for the post-synaptic neuron i and the presynaptic neuron j in a recurrent neuronal network and $\boldsymbol{\phi}_j$ is the scale invariant decoder for the function

$$F(\boldsymbol{s})\tau_s + \boldsymbol{s}$$

For a scale invariant decoder, this yields the following synaptic weight:

$$\omega_{ij} = \frac{1}{N\rho(a_j)\rho_{\boldsymbol{e}}(\boldsymbol{e}_j)}\frac{f^{-1}(r_i)}{1 - a_i}\left\langle \boldsymbol{e}_i, \hat{P}_{\boldsymbol{e}}(\boldsymbol{e}_j)\hat{P}_a(a_j)\right\rangle \tag{56}$$

Note that $\omega_{ij} = F(a_i, a_j, r_i, r_j, \boldsymbol{e}_i, \boldsymbol{e}_j)$, is a function of random variables for the presynaptic and post-synaptic neurons. Thus, instead of thinking the weights as a matrix of numerical values, or as a direct graph, one may think of the weights as defining a hypersurface in a higher dimensional space. For example, the

formula (Equation 56) describes a hypersurface with $2m + 4$ dimesions where m is the dimension of the dynamics the network simulates.

In the following examples, we will generate networks with these analytically determined weights using scale-invariant decoders that display the prescribed dynamics. Additionally, we will assume that

$$r_i^{max} = M\sqrt{1-a_i} \tag{57}$$

$$\rho_a(a) = \frac{1}{2\sqrt{2}\sqrt{1+a}} \tag{58}$$

The variable M controls the maximum firing rate of the neurons, with the range of maximum firing rates being between $[0, \sqrt{2}M]$. We take M to be 60 Hz for all subsequent numerics, unless otherwise specified. Note that we need α_i to compute the weights. The α_i differs depending on whether or not we are simulating a scalar system or a multi-variable system. For a scalar system, $\alpha_i = M^2 e_i$ where $e_i = 1$ for ON neurons and -1 for OFF neurons. For a vector, $\alpha_i = M^2$. For multi-variable dynamics we will assume uniform distributions in the hyperspherical coordinate systems. With the former assumption, the tuning curves for the neurons simply become $M\sqrt{\langle \boldsymbol{e}, \boldsymbol{x}\rangle - a}$ where the M can be absorbed into the decoder. We will generate networks of spiking theta neurons that simulate a neural integrator, a Van der Pol Oscillator, and the Lorenz system.

3.1. Example 1: Neural Integrator

A neural integrator is a recursively coupled neural network that integrates an incoming signal, $u(t)$. The coupling variable $s(t)$, will have dynamics given by

$$s'(t) = u(t) = -\frac{s}{\tau_s} + \frac{1}{\tau_s}\sum_{i=1}^{N}\phi_i\sqrt{e_i(s(t)+\tau_s u(t)) - a_i} \tag{59}$$

where $e_i = 1$ if neuron i is an ON neuron and -1 for OFF neurons. Note that we have scaled $u(t)$ by τ_s as this allows us to write:

$$\tau_s u + s = \sum_{i=1}^{N}\phi_i\sqrt{e_i(s+\tau_s u) - a_i}. \tag{60}$$

We then require the scale invariant decoders such that $\hat{g}(z) = \sum_{i=1}^{N}\phi_i\sqrt{e_i z - a_i} \approx z$. A set of $g^{\pm}(x)$ and the corresponding scale-invariant decoders is given by:

$$g^{+}(x) = \frac{1}{2}(1+x), \quad g^{-}(x) = -\frac{1}{2}(1-x) \tag{61}$$

$$\hat{P}^{+}(a) = \frac{2}{\pi}\frac{1}{\sqrt{1+aM}}, \quad \hat{P}^{-}(a) = -\frac{2}{\pi}\frac{1}{\sqrt{1+aM}} \tag{62}$$

$$\phi_i^{\pm} = \frac{\gamma^{\pm}(a)}{N} = e_i\frac{4\sqrt{2}}{NM\pi} \tag{63}$$

which yields $\hat{g}(z) = z$. From formula (14) for α_i and (58) for the density of a_i, and we have $\alpha_i = e_i M^2$ and the neuronal weight

$$\omega_{ij} = \alpha_i\phi_j = e_i e_j\frac{4\sqrt{2}M}{N\pi}. \tag{64}$$

All the synaptic weights here are given by $4\sqrt{2}M/(N\pi)$ for ON/ON and OFF/OFF connections and $-4\sqrt{2}M/(N\pi)$ for ON/OFF and OFF/ON connections. Now, we will exploit symmetry and non-uniqueness to generate two more neuronal integrators with the same initial distributions of heterogeneity $\rho_a(a)$. In particular, consider the function $\epsilon(x) = (1-x^2)^2$, this function lies in both function spaces for $g^{\pm}(x)$ as it vanishes to second order at both boundaries. Additionally, it can be computed using the following scale-invariant decoders:

$$\hat{P}^{\pm}_{\epsilon(x)}(a) = \frac{32\pi}{5M}(4a^2 - 2a - 1)\sqrt{a+1} \tag{65}$$

$$\phi_i = \frac{\gamma(a_i^{\pm})}{N} = e_i\frac{64\sqrt{2}}{5M\pi N}(1+a_i^{\pm})(4a_i^{\pm 2} - 2a_i^{\pm} - 1) \tag{66}$$

which implies the following decoders for the ON/OFF population still give us $g(x) = x$

$$\phi_i^{\pm} = e_i\frac{4\sqrt{2}}{NM\pi} + e_i\frac{64\sqrt{2}}{5M\pi N}(1+a_i^{\pm})(4a_i^{\pm 2} - 2a_i^{\pm} - 1) \tag{67}$$

which yields the weight matrix

$$\omega_{ij} = \alpha_i\phi_j = e_i e_j\frac{4\sqrt{2}M}{N\pi} + e_i e_j\frac{64M\sqrt{2}}{5\pi N}(1+a_j)(4a_j^2 - 2a_j - 1) \tag{68}$$

Additionally, we can exploit symmetry to derive yet another weight matrix with precisely the same network of neurons:

$$g^{+}(x) = \frac{1}{4}(1+x)^2, \quad g^{-}(x) = -\frac{1}{4}(1-x)^2 \tag{69}$$

$$\hat{P}^{+}(a) = \frac{1}{M\pi}\sqrt{1+a}, \quad \hat{P}^{-}(a) = -\frac{1}{M\pi}\sqrt{1+a} \tag{70}$$

$$\phi_i^{\pm} = \frac{\gamma(a_i^{\pm})}{N} = e_i\frac{2\sqrt{2}(1+a_i^{\pm})}{MN\pi} \tag{71}$$

$$\omega_{ij} = e_i e_j\frac{2\sqrt{2}(1+a_i^{\pm})M}{N\pi} \tag{72}$$

and thus, we have the following three separate weight matrices

$$\omega_{ij} = \alpha_i\phi_j = e_i e_j\frac{4\sqrt{2}M}{N\pi} \tag{73}$$

$$\omega_{ij} = \alpha_i\phi_j = e_i e_j\frac{4\sqrt{2}M}{N\pi} + e_i e_j\frac{64M\sqrt{2}}{5\pi N}(1+a_j)(4a_j^2 - 2a_j - 1) \tag{74}$$

$$\omega_{ij} = e_i e_j\frac{2\sqrt{2}(1+a_j)M}{N\pi} \tag{75}$$

for $i, j = 1, 2, \ldots N$ that yield identical macroscopic dynamics from the same network of neurons as $N \to \infty$. Furthermore, while all the weights converge to 0 as $N \to \infty$, the scaled weights $N\omega_{ij}$ do not converge toward one another in the same limit. One important point is that none of the weights necessarily satisfy the constraint that the action of neuron j on all its downstream targets is the same, either excitatory or inhibitory. Or more precisely that

$$\text{sign}\ \omega_{ij} = \text{sign}\ \omega_{i'j}$$

for all $i, i', j = 1, 2, \ldots N$. Fortunately however, this issue has already been dealt with in the existing literature (Parisien et al., 2008). To summarize, one is able to take the weights generated by the NEF solution, and linearly transform them to yield a new network consisting of excitatory and inhibitory neurons (instead of ON/OFF) with weights that respect this constraint, which is related to Dale's principle.

We have simualted four neural networks with 5000 neurons each all generated with the same random sample drawn from the distribution (Equation 58) using the weights given by Equations (73–75), in addition to the weights generated by determining the optimal decoders. The scale-invariant decoders that correspond to Equations (73–75) were put through conjugate gradient fine-tuning with the final decoders being correlated to the initial decoders with a correlation coefficient greather than 0.98 in all cases. This lowered the root-mean-squared-error by 2–3 orders of magnitude with only slight perturbations off the scale-invariant decoding surface in each case. A sample set of tuning curves is shown in **Figure 5A** with the $\hat{g}^{\pm}(x)$ that correspond to the four different weight structures in **Figure 5B**. The neural integrators are shown in **Figures 5C–F**. The synaptic weights that correspond to the integrators are shown in **Figure 6**. The neurons have been sorted into ON/OFF populations and increasing a within a sub-population prior to plotting the weight matrix in the left column of **Figure 6**. A sub-sample of 20 neurons is also selected (which is identical across the four integrator networks) and their weights are plotted in the right column of **Figure 6**. For the optimal decoders, $g^{\pm}(x) \approx \pm(1 \pm x)/2$, which results in a weight structure similar to Equation (73). The weights differ substantially however in comparison to Equations (74) and (75) as the $\hat{g}^{\pm}$ differ substantially from $\pm(1 \pm x)/2$

This example illustrates that identical networks of neurons can have identical macroscopic dynamics with vastly different weight matrices. While this is not particularly surprising as going from a microscopic description (the individual weights) to a macroscopic description (the dynamics) of a dynamical system is seldom a unique process, the surprising thing is one can explore this issue analytically. An important point to note is that even though the weight matrices are non-unique, they all have the form

$$\omega_{ij} = f(e_i, e_j, a_i) \tag{76}$$

and the weight matrices are nothing more than sample points drawn from different surfaces.

3.2. Exampe 2: Van der Pol Oscillator

The Van der Pol oscillator (Van der Pol, 1926) is given by the dynamical system:

$$\dot{x} = \mu\left(x - \frac{1}{3}x^3 - y\right) = F(x, y) \tag{77}$$

$$\dot{y} = \frac{x}{\mu} = G(x) \tag{78}$$

Here, we will simulate the oscillator with large networks of neurons using the scale-invariant decoders with conjugate gradient descent fine tuning. As the decoding is linear, then from the above equations we only require the decoders for the functions $f(x, y) = x$, $f(x, y) = y$ and $f(x, y) = x^3$. We use a two-dimensional spherical coordinate system for the encoding $\boldsymbol{e}_i = (\cos(\theta_i), \sin(\theta_i))$ and assume uniform distributions in the a_i and the θ_i. Note that:

$$x = \int_0^{2\pi}\int_{-1}^{\cos(\theta)x+\sin(\theta)y} \frac{\cos(\theta)}{\pi}\frac{2}{\pi\sqrt{1+a}} \frac{}{\sqrt{\cos(\theta)x + \sin(\theta)y - a}}\, da d\theta$$

$$y = \int_0^{2\pi}\int_{-1}^{\cos(\theta)x+\sin(\theta)y} \frac{\sin(\theta)}{\pi}\frac{2}{\pi\sqrt{1+a}} \frac{}{\sqrt{\cos(\theta)x + \sin(\theta)y - a}}\, da d\theta$$

$$x^3 + 3x = \int_0^{2\pi}\int_{-1}^{\cos(\theta)x+\sin(\theta)y} \frac{\cos(3\theta) + \cos(\theta)}{\pi}\frac{16(\sqrt{1+a})^3}{\pi} \frac{}{\sqrt{\cos(\theta)x + \sin(\theta)y - a}}\, da d\theta$$

which, immediately allows us to use the following scale-invariant decoders for the sub-functions x, y, and x^3:

$$\phi_i^x = \frac{4\sqrt{2}\cos\theta_i}{MN\pi} \tag{79}$$

$$\phi_i^y = \frac{4\sqrt{2}\sin\theta_i}{MN\pi} \tag{80}$$

$$\phi_i^{x^3} = 32\sqrt{2}\left(\cos(3\theta_i) + \cos(\theta_i)\right)\frac{(1+a_i)^2}{\pi MN} - 3\phi_i^x \tag{81}$$

which yields the decoders for $F(x, y)$ and $G(x, y)$:

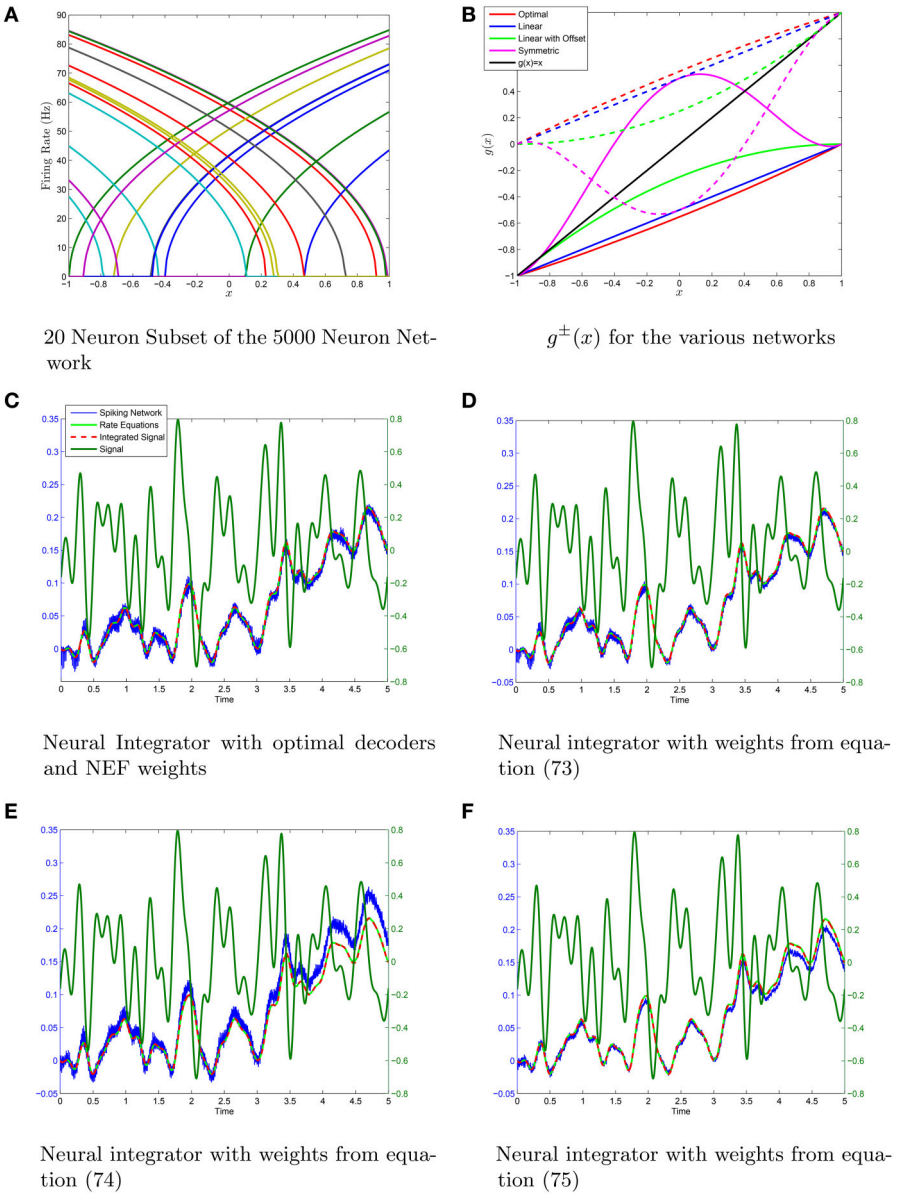
$$\phi_i^F = \phi_i^x + \tau_s\mu\left(\phi^x - \frac{1}{3}\phi^{x^3} - \phi^y\right) \tag{82}$$

$$\phi_i^G = \phi_i^y + \tau_s\frac{\phi^x}{\mu} \tag{83}$$

Thus, the weights are given by

$$\omega_{ij} = \omega(\theta_i, \theta_j, a_j) = M^2\cos(\theta_i)\phi_j^F + M^2\sin(\theta_i)\phi_j^G \tag{84}$$

Both the rate equations and the spiking neural network are simulated using the scale invariant decoders to weight the firing rates/spikes. To more explicitly show the effects of the conjugate-gradient descent fine tuning, we have plotted the scale-invariant decoder surfaces $N\phi^F$ and $N\phi^G$ in **Figure 7**, in addition to the decoders after conjugate gradient descent fine tuning. The scale-invariant decoders and the conjugate gradient descent fine-tuned decoders are again very tightly correlated with $r > 0.95$. We have simulated the Van der Pol Oscillator, as shown in **Figure 8** in both the relaxation ($\mu = 5$) and harmonic ($\mu = 0.7$) oscillator regimes. In both cases, we have excellent correspondence with the network and the actual oscillator. The synaptic weight matrices are also shown in **Figure 8** for a subset of neurons. Like the neural integrator, the weights again lie on a surface, only the surface is 3-dimensional.



20 Neuron Subset of the 5000 Neuron Network

$g^{\pm}(x)$ for the various networks

Neural Integrator with optimal decoders and NEF weights

Neural integrator with weights from equation (73)

Neural integrator with weights from equation (74)

Neural integrator with weights from equation (75)

FIGURE 5 | Neural integrators generated using the same initial heterogeneous network of $N = 5 \cdot 10^3$ Theta neurons. (A) A subset of 20 tuning curves from the network. **(B)** The different neural integrators are generated by using different $g^{\pm}(x)$ to form $g(x)$. **(C–F)** The integrators generated using optimal decoders and decoders from Equations (63),(66), and (71) which results in weight matrcies Equations (73),(74), and (75). The scale-invariant decoders are fine-tuned with conjugate gradient descent. In all cases the fine-tuned decoders are very highly correlated ($\rho > 0.98$) with the scale-invariant decoders indicating only small perturbations off the scale-invariant decoder surface with substantial improvements in the root mean-squared error (RMSE) in computing $\hat{g}(z) = z$. The RMSE was $O(10^{-5})$ with conjugate-gradient descent vs $O(10^{-2})$ without. The variable $\lambda = 0.01$ was taken in the conjugate gradient descent fine-tuning.

3.3. Example 3: Lorenz Attractor

The Lorenz system is given by the equations

$$\dot{x} = \sigma(y - x) = F(x, y) \tag{85}$$

$$\dot{y} = x(\rho - z) - y = G(x, y, z) \tag{86}$$

$$\dot{z} = xy - \beta z = H(x, y, z) \tag{87}$$

and is known to exhibit chaotic behavior for specific values of σ, ρ and β (Lorenz, 1963). In order to approximate the Lorenz system, we require the decoders for the functions x, y, z, xz and xy. With a 3-dimensional spherical coordinate system, the encoding vectors $\boldsymbol{e}$ are given by $\boldsymbol{e} = (\sin(\theta)\cos(\psi), \sin(\theta)\sin(\psi), \cos(\theta))$ where $\theta \in [0, \pi]$ $\psi \in [0, 2\pi]$. The decoders as a function of (ψ, θ, a) are given by:

$$\phi_i^x = \frac{4\sqrt{2}\cos(\psi_i)}{NM\pi}$$

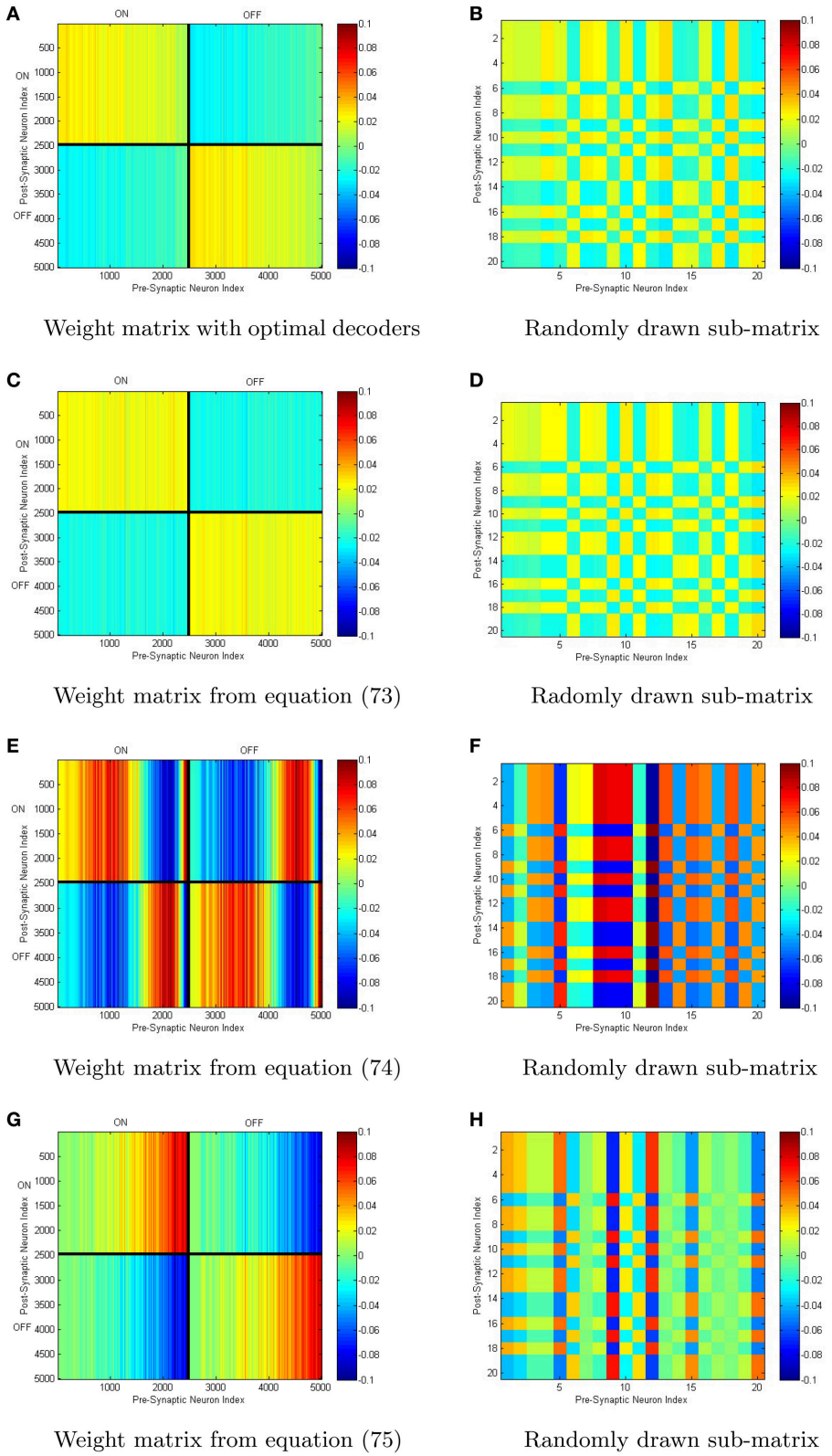
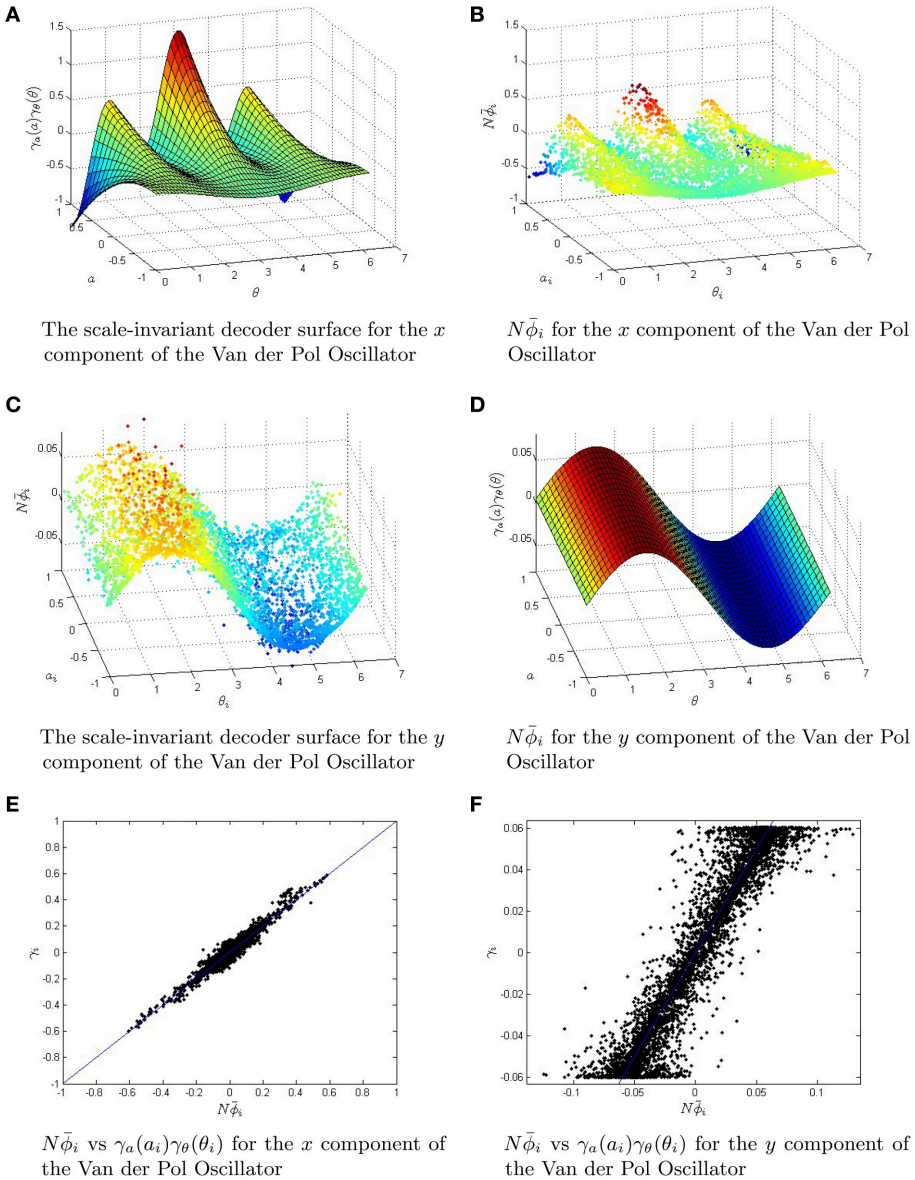


FIGURE 6 | (A–H) Different connectivity weights can generate identical macroscopic dynamics from identical neuronal populations. Shown above are the weight matrices generated for the neural integrators in **Figure 5**. On the right are smaller sub-matrices of weights between 20 randomly selected neurons (the same 20 in each case).



The scale-invariant decoder surface for the x component of the Van der Pol Oscillator

$N\bar{\phi}_i$ for the x component of the Van der Pol Oscillator

The scale-invariant decoder surface for the y component of the Van der Pol Oscillator

$N\bar{\phi}_i$ for the y component of the Van der Pol Oscillator

$N\bar{\phi}_i$ vs $\gamma_a(a_i)\gamma_\theta(\theta_i)$ for the x component of the Van der Pol Oscillator

$N\bar{\phi}_i$ vs $\gamma_a(a_i)\gamma_\theta(\theta_i)$ for the y component of the Van der Pol Oscillator

FIGURE 7 | The Van der Pol Oscillator is approximated by using a scale-invariant decoder surface. The two functions $F(x, y)$ and $G(x, y)$ that are responsible for the dynamics of the Van der Pol Oscillator have scale-invariant decoder surfaces given by Equations (82,83). **(A,B)** The equations for the scale-invariant decoder surfaces are plotted in 3D. These surfaces are used to initialize conjugate gradient descent fine-tuning for a network of 5000 neurons. **(C,D)** The fine-tuned decoders $N\bar{\phi}_i$. They are slightly perturbed off of the surfaces in **(A,B)**. **(E,F)** The strong linear relationship in the scale-invariant surface and the conjugate gradient descent optimized decoders $N\bar{\phi}_i$. The correlation coefficient $r \geq 0.95$ in both cases, while the root mean squared error was reduced from $O(10^{-2})$ to $O(10^{-5})$. The parameter μ for the Van der Pol Oscillator was taken to be 0.7.

$$\phi_i^y = \frac{4\sqrt{2}\sin(\psi_i)}{NM\pi}$$
$$\phi_i^z = \frac{8\sqrt{2}\cos(\theta_i)}{NM\pi}$$
$$\phi_i^{xz} = \frac{24\sqrt{2}\cos(\theta_i - \psi_i)(1 + a_i)}{2NM} - \frac{3\pi}{4}\phi_i^y$$
$$\phi_i^{xy} = \frac{64\sqrt{2}\sin(2\psi)(1 + a_i)}{NM\pi}$$

which yields the decoders for F, G, H as:

$$\phi_i^F = \sigma(\phi_i^y - \phi_i^x) \quad (88)$$
$$\phi_i^G = \phi_i^x\rho - \phi_i^{xz} - \phi_i^y \quad (89)$$
$$\phi_i^H = \phi_i^{xy} - \beta\phi_i^z \quad (90)$$

The strange attractor generated by the Lorenz system and the neural rate equations using the decoders from Equations (88–90) are shown in **Figure 9**. The chaotic behavior and the strange

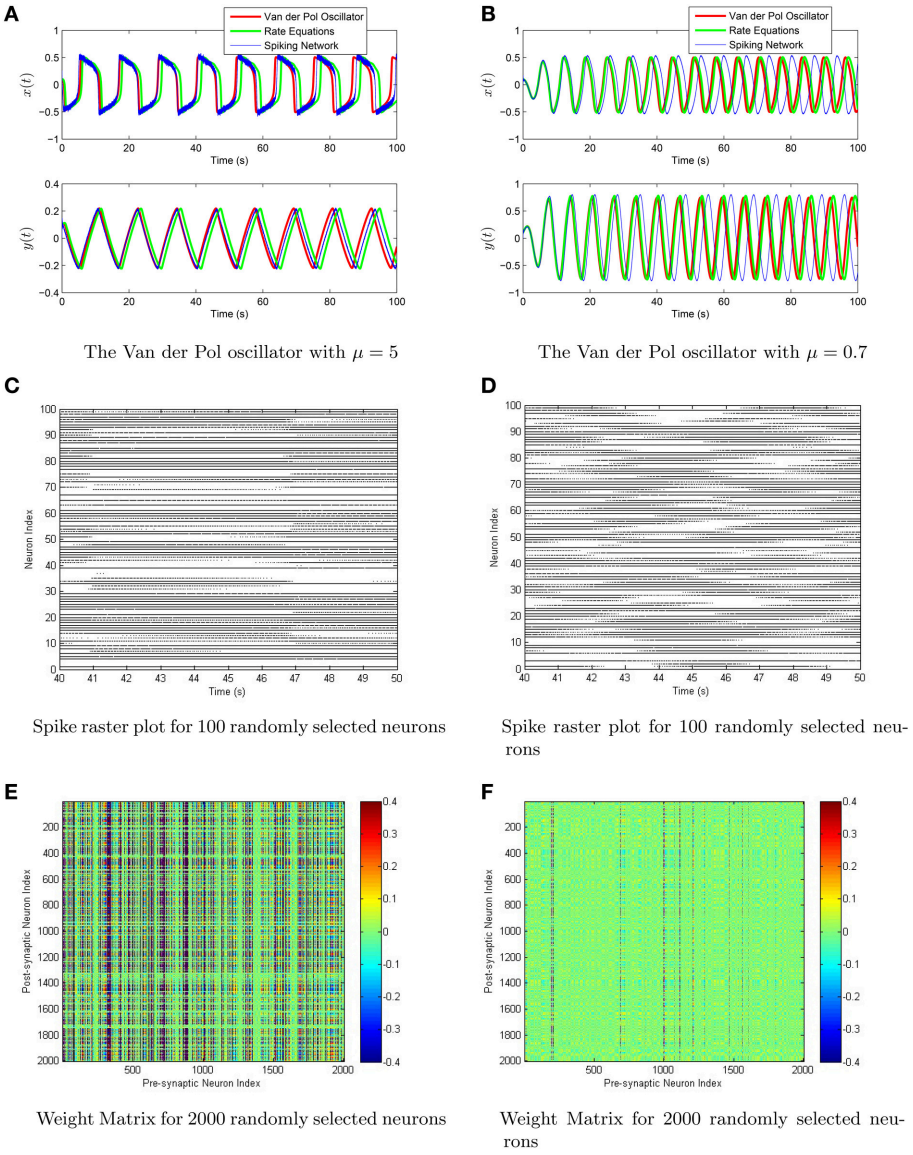


FIGURE 8 | (A,B) The Van der Pol oscillator is simulated using a network of 10^5 theta neurons with scale-invariant decoders after conjugate-gradient descent fine tuning in the relaxation oscillator regime ($\mu = 5$, left column) and in the harmonic oscillator regime ($\mu = 0.7$, right column). Shown in the top row is the comparison between the oscillator (red), the rate equations (green), and the spiking network (blue). **(C,D)** Shown in the middle is the spike raster plot for a 10 s interval of both networks. **(E,F)** The last row consists of a computed weight matrix for 2000 randomly selected neurons in the network.

attractor is also preserved when one uses a spiking neuronal network with the decoder weights on the spikes. Note that a great many neurons are required to adequately visualize the strange attractor, however the chaotic behavior is present even for smaller networks. We have also plotted the location of neural spiking with regards to the strange attractor. The neurons tend to spike more in specific regions of the strange attractor in accordance with their preferred orientation vectors and their a_i parameters. The weights are again given by the NEF formula:

$$\omega_{ij} = \omega(\theta_i, \theta_j, \psi_i, \psi_j, a_j) = M^2 \sin(\theta_i)\cos(\psi_i)\phi_j^F + M^2 \sin(\theta_i)\sin(\psi_i)\phi_j^G + M^2 \cos(\theta_i)\phi_j^H \quad (91)$$

which defines a five dimensional surface.

4. DISCUSSION

The NEF has been used to develop a wide variety of neural circuit models. The spiking networks generated from the NEF approach are spiking neural networks that are capable of functionally reproducing very sophisticated behaviors (Eliasmith, 2005). In the NEF approach, a synaptic weight between two neurons is a dot product of the post-synaptic neuron's preferred direction vector and the presynaptic neuron's linear decoding vector or "decoder." The decoders weight the tuning curves for the

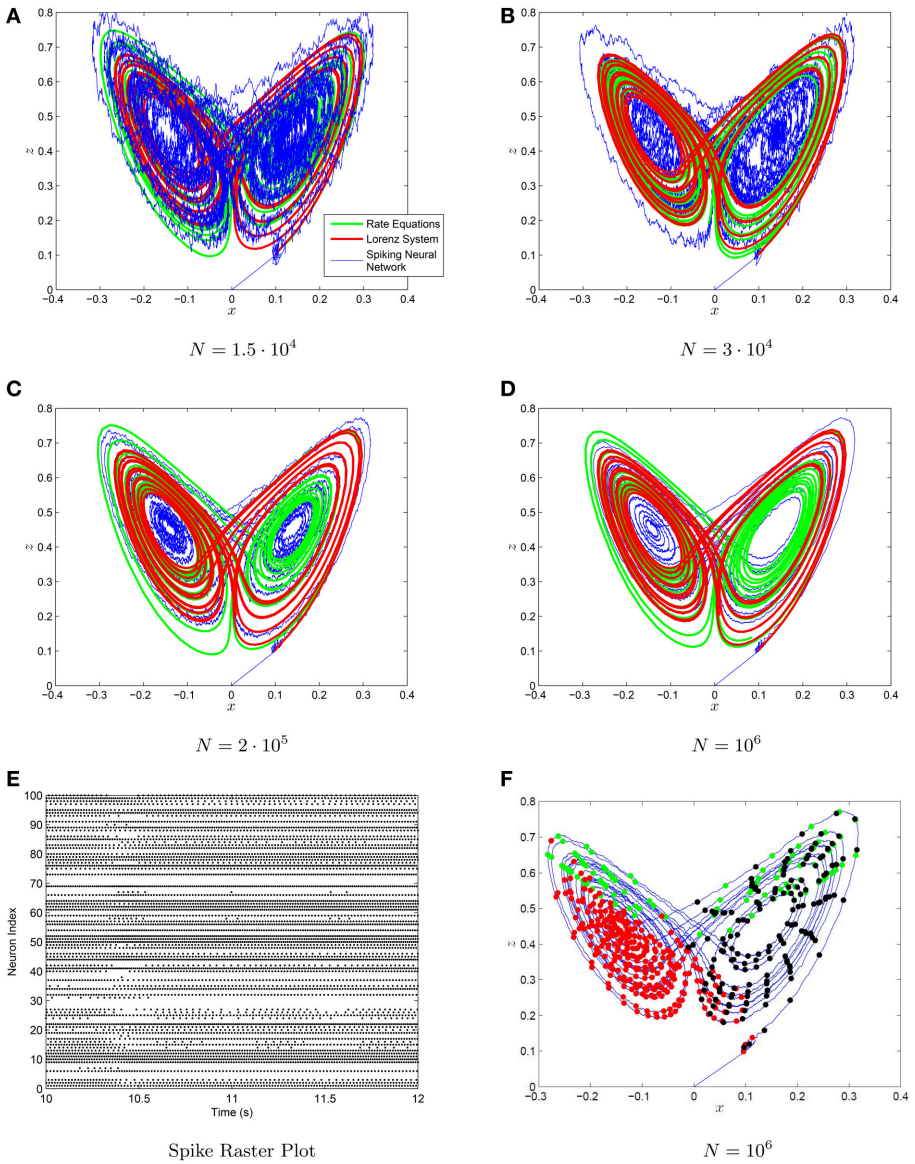


FIGURE 9 | (A–D) The Lorenz attractor generated with a spiking neuronal network (blue) the neural rate equations (green) and integrating the Lorenz system (red) with neural networks of increasing size in figures. The decoders used are scale-invariant with conjugate gradient descent fine-tuning. The synaptic time constant was $\tau_S = 50$ ms. **(E)** The spike raster plot for 100 neurons chosen at random from the $N = 10^6$ neuron simulation. **(F)** The location of the spikes with regards to the strange attractor for three randomly selected neurons are shown in **(F)**.

neurons and are determined by an optimization criterion that minimizes the L_2 error in the linear combination of tuning curves and the target dynamics of the network in addition to a factor that punishes the size of the decoders. The optimal decoders are unique, as they are determined by a convex optimization problem.

The first main point of this study is that in the large network limit, one can define a scale-invariant decoder that scales in inverse proportion to the network size N such that the scale-invariant decoders zero the cost function used to define the optimal decoders asymptotically in the mean-squared error. In the asymptotic limit, the scale-invariant decoders multiplied by the probability density governing the source(s) of heterogeneity converge to a constant function. The resulting linear combination of weighted firing rates converges to the tuning curve transforms asymptotically. We have determined this to also be the case with optimal decoders given by large matrix inversion, where the product of the optimal decoders and the heterogeneous density also converges to the weighted decoder. Furthermore, we have shown how one can invert tuning curve transform operators for type-I (theta model) and type-II firing rates. It turns out that the inversion is non-unique,

which results in an infinite-dimensional family of scale-invariant decoder surfaces and thus synaptic weights.

The second contribution of this paper is a method of finding the scale-invariant decoders analytically from the decoded function and the probability density of the neuronal parameters. Additionally, we have demonstrated that as the weighted decoder is non-unique for any particular dynamics, the NEF solution to the weights with scale-invariant decoders also becomes non-unique.

Finally, we have demonstrated how the weights that couple neurons together can actually be thought of as surfaces defined entirely by the source(s) of heterogeneity in the network. In particular, a weight ω_{ij} is defined as the inner product of a scale-invariant decoder $\gamma(\boldsymbol{e}_j, a_j)/N$ of the post-synaptic neuron j with the encoding vector $\boldsymbol{e}_i$ of the presynaptic neuron i. Thus, one can consider this inner product as a function over a higher dimensional space and the weights are merely sample points on this hypersurface. Furthermore, the weights generated via this approach have the advantage that the coupling between the jth presynaptic neuron and the ith postsynaptic neuron depends only upon the local properties of neuron i and neuron j for any specified dynamics. This is different from Hebbian plasticity, in that the coupling does not depend on presynaptic and postsynaptic activity, but rather on an intrinsic property of the presynaptic neuron times an intrinsic property of the postsynaptic neuron. While this perspective has been used before in the literature, for example to model hypercolumns in the primary visual cortex in (Shriki et al., 2006), here we show how it can be extended to arbitrary (smooth) dynamics.

4.1. Relationships between the Analytical and Optimal Decoders

The optimal decoders appear to have an asymptotic weighted decoder, the product of the density function multiplied by the scaled optimal decoder, $N\Phi_i$, when one computes this quantity numerically after large-matrix inversion (see **Figure 2**). For the optimal decoders, the weighted decoder has high frequency oscillations that are related to the idiosyncracies of the particular sample of random neurons generated. These seem to attenuate with increasing network size, and regularization parameter λ. These high frequency oscillations are for example eliminated when the neurons are drawn from a grid. Indeed, when the neurons are drawn from a grid, the optimal decoders for a much larger network can be approximated by simply interpolating between the decoder values for the smaller grid network, and rescaling the interpolated decoders in accordance with the network size.

Thus, one may ask is the weighted decoder generated by the optimal decoders (1) convergent as $N \to \infty$, and (2) does it converge to any particular weighted decoder in the set defined by the requirement that $\hat{g}(x) = M(P^+, P^-) = L^+(P^+) + L^-(P^-) = g(x)$? It seems that numerically the quantity $N\rho_a(a_i)\Phi_i$ does converge to some surface $P^{\pm}$ that varies depending on the identity of neuron i as an ON/OFF neuron. The likely candidate for the specific $\hat{P}^{\pm}$ in the set defined by $M(P^+, P^-)$ is the surface that minimizes Equation (42). However, the relationship between the optimal decoders, and any particular scale-invariant decoder as $N \to \infty$ is outside of the scope of this paper and is best left for future work. Relatedly, there seem to be some systematic differences between scale-invariant and optimal decoders due to finite neurons. For example, a scale-invariant decoder surface can have discontinuities, while corresponding regularized optimal decoder surfaces do not, even with very large numbers of neurons

4.2. Relationship to Other Approaches

We have demonstrated that if one defines a scale-invariant decoder that is a function of the source(s) of heterogeneity of the neurons, one can obtain arbitrary macroscopic dynamics with the NEF weight solution, which is the dot product of the decoder for the presynaptic neuron, and the preferred orientation vector or encoding vector for the post-synaptic neuron. Thus, both the weights and the decoders for the neurons can be thought of as surfaces as they are merely functions of the sources of heterogeneity of the network. While this perspective is novel in the sense that we have shown one can obtain arbitrary dynamics with this approach, similar decoders and weights have been suggested in previous work. In particular, the scale-invariant decoders here can be thought of as an extension of the neuronal population vectors of Georgopolous (Georgopoulos et al., 1986, 1994). It has been previously noted (for example in Eliasmith and Anderson, 2004) that the population vector from (Georgopoulos et al., 1986, 1994) is similar to the case where one uses the encoding vectors e_i as the decoding vectors ϕ_i. Here, we have derived a more general approach where ϕ_i is a function of $\boldsymbol{e}_i$ and the other sources of heterogeneity in the tuning curves. Additionally, weights that are a function of the presynaptic and post-synaptic preferred orientations have been also used in the literature (Ben-Yishai et al., 1995; Shriki et al., 2006).

There are also methods in the literature that construct networks of neurons with prescribed dynamics. For example, spiking networks of leaky integrate-and-fire models have been constructed that can display arbitrary linear dynamics with weights that are seemingly unrelated to the NEF solution for the synaptic weights (Boerlin et al., 2013). The solution obtained in (Boerlin et al., 2013) is derived through minimizing the time integral of the L2 norm of the decoded estimate of the dynamics, $\hat{x}(t)$ and the intended dynamics in addition to other terms intended to minimize the spiking, and distribute the spikes equitably across the network. The end result is a network of leaky integrate and fire neurons and weights that when coupled together display the desired dynamics. While the solution is very elegant, it is unknown if it can be extended to non-linear dynamics.

4.3. Numerical Applications

We considered whether this work has practical applications for neural simulations. As an example, we considered whether (having found the weighted decoders for a given network) it would be useful to adjust ρ in order to make the weights uniform, thus avoiding large fluctuations associated with the spiking of heavily weighted neurons. This sometimes led to modest reductions in spike noise in simulations.

As discussed above, scale-invariant decoders can serve as a starting point for iterative optimization, quickly leading to highly optimized weights for large networks. However, it should be pointed out that since the optimization problem is convex, any starting point can be used, and in our experience scale-invariant decoders are only moderately better than other reasonable choices.

It is possible that our approach could be applied to groups of neurons that are so large as to be impractical even for efficient iterative methods. The cortex consists of a fairly continuous sheet of neurons with few distinct boundaries, suggesting that it may be somehow useful to simultaneously consider the activity of billions of neurons. On the other hand, if the goal is simply to optimize synaptic weights, the degree of convergence onto single neurons ($< 200,000$) is within a practical range for iterative methods. Furthermore, our solution for multivariable functions requires encoder distributions that are separable in hyper-spherical coordinates, which may be a limitation for modeling extensive sheets of neurons with overlapping tuning.

In light of these experiences, we consider this work to be valuable mainly as a source of new insights into network function and dynamics, rather than as a basis for new numerical tools.

4.4. Analyzing Measured Synaptic Weights

Using experimental methods, it is possible to measure both the synaptic weights (defined as the peak post-synaptic current) in addition to fitting integrate-and-fire type neurons with heterogeneity using the dynamic current-voltage curve approach (Badel et al., 2008; Harrison, 2014; Harrison et al., 2015). While there are typically many parameters for more complicated integrate-and-fire models (such as the AdEx), one can always reduce the number of heterogeneous parameters to a much smaller set governing the properties of the tuning curves. We refer to these generically as $\boldsymbol{b}_i$, a vector with the parameter values for the ith neuron. Given these assumptions, and the work done in this paper, one kind of analysis that can be conducted without much effort is the construction of a non-linear regression model of the weights:

$$\omega_{ij} = F(\boldsymbol{b}_i, \boldsymbol{b}_j) + \epsilon_{ij} \tag{92}$$

where $F(\boldsymbol{b}_i, \boldsymbol{b}_j)$ is either a non-linear or linear model with a number of free coefficients and ω_{ij} are the experimentally determined synaptic weights of a recurrently coupled neural network. The coefficients can be estimated using optimization techniques to minimize ϵ_{ij}. For this weight analysis to be valid, we require the following

$$|\epsilon_{ij}| \ll |F(\boldsymbol{b}_i, \boldsymbol{b}_j)|, \forall i, j$$

where $F(\boldsymbol{b}_i, \boldsymbol{b}_j) = O(N^{-1})$. The core result of this paper is that any dynamics are possible with synaptic weights of the form (Equation 92), and their is no unique weight matrix that confers specfic dynamics. Thus, a regression analysis of the synaptic weights (Equation 92) is a reasonable analysis to conduct if one knows the sources of heterogeneity for the neurons in the network and provided that the residuals are sufficiently small.

4.5. Future Work

While the networks constructed here display the desired macroscopic dynamics, this is not always the case. In particular, if the time constants are too small, then the collective macroscopic state can destabilize. For the weight solution we have determined to be valid, one needs to prove that the macroscopic dynamics form a stable attractor in the large network limit. Unfortunately, resolution of this problem is well outside the scope of this paper. While a great deal of work has been done in determing the stability of asynchronous states in large network limits (for example in Abbott and van Vreeswijk, 1993; van Vreeswijk, 1996), to our knowledge no work has been when the weights have structure present here. The majority of work done on large network stability analysis is devoted to weights that are constant throughout the network, $\omega_{ij} = \omega$ however there is some work on non-constant, randomly distributed weights (Hermann and Touboul, 2012). The authors of (Hermann and Touboul, 2012) note that oscillations arise when considering randomly distributed weights. Here, we demonstrate that with a little bit more structure to the weights/network (the weights are functions of the properties of the $f(I)$ curves of the neurons), arbitrary prescribed macroscopic dynamics can be generated by the network.

Networks of heterogeneous theta oscillators have been extensively analyzed in (Barreto et al., 2008; Luke et al., 2013; So et al., 2014) by using the Ott-Antonsen Anzats initially applied to networks of Kuramoto Oscillators (Martens et al., 2009; Ott and Antonsen, 2009). Additionally, one of the weight solutions for a network with one-dimensional dynamics that arises from the scale-invariant decoders sets all the weights to $\pm\omega$ by setting the density to $\rho \propto |\hat{P}(a)|$ where the constant of proportionality normalizes $|\hat{P}(a)|$. Given that, it may be possible to apply some of the existing literature on the stability analysis of networks of heterogeneous theta neurons to this network.

In addition to stability analysis of the large network, the weight solutions were derived here under a pair of simplifying assumptions. In particular the two strongest assumptions were that the FI curves were constant in time, and that the neurons were coupled using current-based synapses instead of conductance based synapses. We intend to extend the approach we have taken here with scale-invariant decoders to neurons with conductance based synapses, and $f(I)$ curves that vary due to forces like spike frequency adaptation (Ermentrout, 2006). Fortunately, some of the initial work on generating macroscopic rate-equations (a necessary initial step) for conductance based neurons has been done in (Ermentrout, 1994; Shriki et al., 2006), in addition to work on rate equations for adapting neurons (Nicola and Campbell, 2013b)

AUTHOR CONTRIBUTIONS

WN and MS performed much of the analysis. The numerical experiments were performed by WN with assistance from BT with regards to the Neural Engineering Framework. WN, BT, and MS contributed to the writing of the manuscript.

FUNDING

This research was funded by an NSERC Postgraduate Scholarships-Doctoral Program (WN) and NSERC discovery grants (BT, MS).

SUPPLEMENTARY MATERIAL

The Supplementary Material for this article can be found online at: http://journal.frontiersin.org/article/10.3389/fncom.2016.00015

REFERENCES

Abbott, L. F., and van Vreeswijk, C. (1993). Asynchronous states in networks of pulse-coupled oscillators. *Learn. Mem.* 48, 1483–1490.

Abbott, L. F. (1999). Lapicques introduction of the integrate-and-fire model neuron (1907). *Brain Res. Bull.* 50, 303–304.

Ahmed, B., Anderson, J. C., Douglas, R. J., Martin, K. A., and Whitteridge, D. (1998). Estimates of the net excitatory currents evoked by visual stimulation of identified neurons in cat visual cortex. *Cereb. Cortex* 8, 462–476.

Azouz, R., Gray, C. M., Nowak, L. G., and McCormick, D. A. (1997). Physiological properties of inhibitory interneurons in cat striate cortex. *Cereb. Cortex* 7, 534–545.

Badel, L., Lefort, S., Brette, R., Petersen, C. C. H., Gerstner, W., and Richardson, M. J. E. (2008). Dynamic iv curves are reliable predictors of naturalistic pyramidal-neuron voltage traces. *J. Neurophysiol.* 99, 656–666. doi: 10.1152/jn.01107.2007

Barreto, E., Hunt, B., Ott, E., and So, P. (2008). Synchronization in networks of networks: the onset of coherent collective behavior in systems of interacting populations of heterogeneous oscillators. *Phys. Rev. E* 77:036107. doi: 10.1103/PhysRevE.77.036107

Bekolay, T., Laubach, M., and Eliasmith, C. (2014). A spiking neural integrator model of the adaptive control of action by the medial prefrontal cortex. *J. Neurosci.* 34, 1892–1902. doi: 10.1523/JNEUROSCI.2421-13.2014

Ben-Yishai, R., Bar-Or, R. L., and Sompolinsky, H. (1995). Theory of orientation tuning in visual cortex. *Proc. Natl. Acad. Sci. U.S.A.* 92, 3844–3848.

Bishop, C. M. (1995). *Neural Networks for Pattern Recognition*. Oxford, UK: Oxford University Press.

Bobier, B., Stewart, T. C., and Eliasmith, C. (2014). A unifying mechanistic model of selective attention in spiking neurons. *PLoS Comput. Biol.* 10:e1003577. doi: 10.1371/journal.pcbi.1003577

Boerlin, M., Machens, C. K., and Denève, S. (2013). Predictive coding of dynamical variables in balanced spiking networks. *PLoS Comput. Biol.* 9:e1003258. doi: 10.1371/journal.pcbi.1003258

Brette, R., and Gerstner, W. (2005). Adaptive exponential integrate-and-fire model as an effective description of neuronal activity. *J. Neurophysiol.* 94, 3637–3642. doi: 10.1152/jn.00686.2005

Brunel, N., and Van Rossum, M. C. W. (2007). Lapicques 1907 paper: from frogs to integrate-and-fire. *Biol. Cybern.* 97, 337–339. doi: 10.1007/s00422-007-0190-0

Conklin, J., and Eliasmith, C. (2005). A controlled attractor network model of path integration in the rat. *J. Comput. Neurosci.* 18, 183–203. doi: 10.1007/s10827-005-6558-z

Dayan, P., and Abbott, L. F. (2001). *Theoretical Neuroscience, Vol. 806*. Cambridge, MA: MIT Press.

DeWolf, T., and Eliasmith, C. (2011). The neural optimal control hierarchy for motor control. *J. Neural Eng.* 8:065009. doi: 10.1088/1741-2560/8/6/065009

Eliasmith, C., and Anderson, C. H. (2004). *Neural Engineering: Computation, Representation, and Dynamics in Neurobiological Systems*. MIT press.

Eliasmith, C., Stewart, T. C., Choo, X., Bekolay, T., DeWolf, T., Tang, C., and Rasmussen, D. (2012). A large-scale model of the functioning brain. *Science* 338, 1202–1205. doi: 10.1126/science.1225266

Eliasmith, C. (2005). A unified approach to building and controlling spiking attractor networks. *Neural Comput.* 17, 1276–1314. doi: 10.1162/0899766053630332

Ermentrout, G. B., and Kopell, N. (1986). Parabolic bursting in an excitable system coupled with a slow oscillation. *SIAM J. Appl. Math.* 46, 233–253.

Ermentrout, G. B. (1994). Reduction of conductance-based models with slow synapses to neural nets. *Neural Comput.* 6, 679–695.

Ermentrout, B. (2006). Linearization of f-i curves by adaptation. *Neural Comput.* 10, 1721–1729. doi: 10.1162/089976698300017106

Georgopoulos, A. P., Schwartz, A. B., and Kettner, R. E. (1986). Neuronal population coding of movement direction. *Science* 233, 1416–1419.

Georgopoulos, A. P., Lurito, J. T., Petrides, M., Schwartz, A. B., and Massey, J. T. (1994). Mental rotation of the neuronal population vector. *Biol. Comput. Physicist. Choice* 3, 183.

Harrison, P. M., Badel, L., Wall, M. J., and Richardson, M. J. E. (2015). Experimentally verified parameter sets for modelling heterogeneous neocortical pyramidal-cell populations. *PLoS Comput. Biol.* 11:e1004165. doi: 10.1371/journal.pcbi.1004165

Harrison, P. M. (2014). *Experimentally Verified Reduced Models of Neocortical Pyramidal Cells*. Ph.D. thesis, University of Warwick.

Hermann, G., and Touboul, J. (2012). Heterogeneous connections induce oscillations in large-scale networks. *Phys. Rev. Lett.* 109:018702. doi: 10.1103/PhysRevLett.109.018702

Izhikevich, E. M. (2003). Simple model of spiking neurons. *Neural Netw. IEEE Trans.* 14, 1569–1572. doi: 10.1109/TNN.2003.820440

Izhikevich, E. M. (2007). *Dynamical Systems in Neuroscience*. Cambridge, MA: MIT Press.

Lapicque, L. (1907). Recherches quantitatives sur lexcitation électrique des nerfs traitée comme une polarisation. *J. Physiol. Pathol. Gen.* 9, 620–635.

Lorenz, E. N. (1963). Deterministic nonperiodic flow. *J. Atmos. Sci.* 20, 130–141.

Luke, T. B., Barreto, E., and So, P. (2013). Complete classification of the macroscopic behavior of a heterogeneous network of theta neurons. *Neural Comput.* 25, 3207–3234. doi: 10.1162/NECO_a_00525

Martens, E. A., Barreto, E., Strogatz, S. H., Ott, E., So, P., and Antonsen, T. M. (2009). Exact results for the kuramoto model with a bimodal frequency distribution. *Phys. Rev. E* 79:026204. doi: 10.1103/PhysRevE.79.026204

MATLAB (2014). *version 7.10.0 (R2010a)*. Natick, MA: The MathWorks Inc.

Naud, R., Marcille, N., Clopath, C., and Gerstner, W. (2008). Firing patterns in the adaptive exponential integrate-and-fire model. *Biol. Cybern.* 99, 335–347. doi: 10.1007/s00422-008-0264-7

Nesse, W., Borisyuk, A., and Bressloff, P. (2008). Fluctuation-driven rhythmogenesis in an excitatory neuronal network with slow adaptation *J. Comput. Neurosci.* 25, 317–333. doi: 10.1126/science.1225266

Nicola, W., and Campbell, S. A. (2013a). Bifurcations of large networks of two-dimensional integrate and fire neurons. *J. Comput. Neurosci.* 35, 87–108. doi: 10.1007/s10827-013-0442-z

Nicola, W., and Campbell, S. A. (2013b). Mean-field models for heterogeneous networks of two-dimensional integrate and fire neurons. *Front. Comput. Neurosci.* 7:184. doi: 10.3389/fncom.2013.00184

Ott, E., and Antonsen, T. M. (2009). Long time evolution of phase oscillator systems. *Chaos* 19:023117. doi: 10.1063/1.3136851

Parisien, C., Anderson, C. H., and Eliasmith, C. (2008). Solving the problem of negative synaptic weights in cortical models. *Neural Comput.* 20, 1473–1494. doi: 10.1162/neco.2008.07-06-295

Rasmussen, D., and Eliasmith, C. (2014). A spiking neural model applied to the study of human performance and cognitive decline on raven's advanced progressive matrices. *Intelligence* 42, 53–82. doi: 10.1016/j.intell.2013.10.003

Salinas, E., and Abbott, L. F. (1995). Vector reconstruction from firing rates. *J. Comput. Neurosci.* 1, 89–107.

Shriki, O., Hansel, D., and Sompolinsky, H. (2006). Rate models for conductance-based cortical neuronal networks. *Neural Comput.* 15, 1809–1841. doi: 10.1162/08997660360675053

Singh, R., and Eliasmith, C. (2006). Higher-dimensional neurons explain the tuning and dynamics of working memory cells. *J. Neurosci.* 26, 3667–3678. doi: 10.1523/JNEUROSCI.4864-05.2006

So, P., Luke, T. B., and Barreto, E. (2014). Networks of theta neurons with time-varying excitability: Macroscopic chaos, multistability, and final-state uncertainty. *Phys. D Nonlinear Phenomena* 267, 16–26. doi: 10.1016/j.physd.2013.04.009

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

Touboul, J. (2008). Bifurcation analysis of a general class of nonlinear integrate-and-fire neurons. *SIAM J. Appl. Math.* 68, 1045–1079. doi: 10.1137/070687268

Van der Pol, B. (1926). Lxxxviii. on relaxation-oscillations. *Lond. Edinburgh Dublin Philos. Mag. J. Sci.* 2, 978–992.

van Vreeswijk, C. V. (1996). Partial synchronization in populations of pulse-coupled oscillators. *Phys. Rev. E* 54, 5522.

Werfel, J., Xie, X., and Seung, H. S. (2005). Learning curves for stochastic gradient descent in linear feedforward networks. *Neural Comput.* 17, 2699–2718. doi: 10.1162/089976605774320539

Conflict of Interest Statement: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

APPENDIX

Determining the Decoder Surface

Here, we will make use of the Laplace transform to analytically determine the functions one can represent and numerically invert for the decoder surfaces in the NEF approach. In particular, consider the operator $L^+(\hat{P})$ which is given by

$$\hat{g}^+(x) = L^+(\hat{P}) = \int_{-1}^{x} \hat{P}(a)\sqrt{x-a}\,da \tag{A1}$$

Using a series of substitutions we can write the following:

$$\begin{aligned}\hat{g}^+(x) &= \int_0^{x+1} \hat{P}(b-1)\sqrt{x+1-b}\,db &\text{(A2)}\\ &= \int_0^{y} \hat{P}(b-1)\sqrt{y-b}\,db \\ &= \int_0^{y} Q(b)R(y-b)\,db = m(y) &\text{(A3)}\end{aligned}$$

where $Q(b) = \hat{P}(b-1) = \hat{P}(a)$ and $R(y-b) = \sqrt{y-b} = \sqrt{x+1-b} = \sqrt{x-a}$, and $m(y) = \hat{g}(y-1) = \hat{g}(x)$. In this case, the expectation is a convolution when written in the appropriate variables:

$$m(z) = (Q \star R)(z) \tag{A4}$$

where $\star$ denotes the convolution operator. Taking a Laplace transform yields:

$$\begin{aligned}\mathcal{L}(m(z)) &= \mathcal{L}(Q(z))\mathcal{L}(R(z)) = \mathcal{L}(Q(z))\frac{\sqrt{\pi}}{2s^{3/2}} &\text{(A5)}\\ \Rightarrow \mathcal{L}(Q(z)) &= \frac{2}{\sqrt{\pi}}\mathcal{L}(m(z))s^{3/2} &\text{(A6)}\\ &= \mathcal{L}(m(z))\mathcal{L}(C(z)) &\text{(A7)}\end{aligned}$$

for some function $C(z)$. Unfortunately $\mathcal{L}(C(z)) = \frac{2}{\sqrt{\pi}}s^{3/2}$ is not invertible as we have:

$$\mathcal{L}(\sqrt{t}) = \frac{1}{2}\frac{\sqrt{\pi}}{s^{3/2}} \quad \mathcal{L}(f''(t)) = s^2\mathcal{L}(f(t)) + sf(0) + f'(0) \tag{A8}$$

from the general properties of the Laplace transform. This implies that if $f(t) = -4\sqrt{t}$, $f''(t) = t^{-3/2}$ and the Laplace transform is undefined as $f'(0) \to \infty$. Suppose instead that we assume that $m(z)$ is twice differentiable and that $m(0) = 0$ and $m'(0) = 0$. Then we can write the following:

$$\begin{aligned}\mathcal{L}(Q(z)) &= \frac{2}{\sqrt{\pi}}s^{3/2}\mathcal{L}(m(z)) &\text{(A9)}\\ &= \frac{2}{\sqrt{\pi}}s^{-1/2}\mathcal{L}(m''(z)) \\ &= \mathcal{L}\left(\frac{2}{\pi\sqrt{z}}\right)\mathcal{L}(m''(z)) \\ \Rightarrow Q(z) &= \int_0^z \frac{2}{\pi\sqrt{t}}m''(z-t)\,dt &\text{(A10)}\end{aligned}$$

This is how to find $Q(z)$ as a convolution with $m''(z)$, the second derivative of the function you want to approximate and $1/\sqrt{t}$. Based on these assumptions, we require that $m(z) = bz^2 + O(z^3)$. Writing the convolution in terms of the original variables, we have the following:

$$\hat{P}(a) = \int_0^{a+1} \frac{2}{\pi\sqrt{t}}\hat{g}''(a-t)\,dt \tag{A11}$$

This implies that we can approximate any function that vanishes to two orders at $x = -1$ using the following decoder for the ON neurons:

$$\phi_i^+(a_i) = \frac{2}{N}\frac{1}{\rho_a(a_i)}\int_0^{a_i+1} \frac{2}{\pi\sqrt{t}}\hat{g}''(a_i-t)\,dt \tag{A12}$$

Additionally, if one repeats this process for a population of OFF neurons, then we can immediately write down the decoder for a function g that vanishes to two orders at $x = 1$:

$$\phi_i^-(a_i) = \frac{2}{N}\frac{1}{\rho_a(a_i)}\int_0^{a_i+1} \frac{2}{\pi\sqrt{t}}\hat{g}''(-a_i-t)\,dt \tag{A13}$$

Given the constraints on the derivatives of $\hat{g}$, it is clear that using a population of ON and OFF neurons, the resulting approximant has the form:

$$\begin{aligned}\hat{g}(x) &= \hat{g}^+(x) + \hat{g}^-(x) \\ &= \frac{1}{2}\left(g(x) - g(-1) - (x+1)g'(-1)\right) \\ &\quad + \frac{1}{2}\left(g(x) - g(1) - (x-1)g'(1)\right) \\ &= g(x) - A - Bx &\text{(A14)}\end{aligned}$$

by using the ON and OFF populations, where $g(x)$ is the function we want to approximate. The remainder is a linear term, $A + Bx$. Thus, if we can approximate an arbitrary linear function with a population of type-1 neurons with heterogeneity, then we can accomodate the remainder term. Note the following:

$$\begin{aligned}\frac{2}{\pi}\int_{-1}^{\pm x}\frac{\sqrt{\pm x - a}}{\sqrt{a+1}}\,da &= \frac{4}{\pi}\int_0^{\sqrt{x+1}}\sqrt{x+1-u^2}\,du \\ &= 1 \pm x &\text{(A15)}\end{aligned}$$

Thus, with the functions $\hat{P}^+ = \frac{2C}{\pi\sqrt{1+a}}$ and $\hat{P}^- = \frac{2D}{\pi\sqrt{1-a}}$, for the ON and OFF neurons respectively, we can approximate

$$C(x+1) + D(1-x) = (C-D)x + (C+D) = Bx + A$$

with $C = (A+B)/2$ and $D = (A-B)/2$. Thus, to approximate the function $g(x)$, we can use the following $\hat{P}$:

$$\begin{aligned}\hat{P}^+(a) &= \frac{2}{\pi}\left(\frac{g(-1)+g(1)}{2} + g'(-1)\right)\frac{1}{\sqrt{1+a}} \\ &\quad + \int_0^{a+1} g''(a-t)\frac{2}{\pi\sqrt{t}}\,dt &\text{(A16)}\end{aligned}$$

$$\hat{P}^{-}(a) = \frac{2}{\pi}\left(\frac{g(-1)+g(1)}{2} - g'(1)\right)\frac{1}{\sqrt{1+a}} + \int_0^{a+1} g''(-t-a)\frac{2}{\pi\sqrt{t}}\,dt \tag{A17}$$

To remove the linear error, we used a linear combination of $1+x$ and $1-x$. This choice is not unique. For example, we could have used

$$g^{\pm}(x) = \frac{(1\pm x)^2}{1+x^2} \tag{A18}$$

to eliminate constant term and a separate linear combination of

$$g^{\pm}(x) = 1 \pm x^2 \tag{A19}$$

to eliminate the x term. The reason Equations (A18, A19) are not used for type-I neurons is due to complexity in the associated weighted decoders. However, these forms are simpler for Type-II firing rates and hence they are used there.

The same process can be carried out with the type-II firing rate form, which yields the operators:

$$L^{\pm}(\hat{P}^{\pm}) = \int_{-1}^{\pm x} \hat{P}^{\pm}(a)(\pm x - a + c)\,da \tag{A20}$$

with the resulting values for $\hat{P}^{\pm}$ being

$$\hat{P}^{\pm}(a) = 2A + 2Ba\frac{(a^2-3)}{(a^2+1)^3} + g''(\pm a) \tag{A21}$$

for $c = 0$. For $c > 0$, the inversion for functions that vanish to second order at $x = \pm 1$ yields

$$\hat{P}^{\pm}(a) = \frac{1}{c}\int_0^{a+1} \exp\left(-\frac{t}{c}\right) g''(\pm(a-t))\,dt \tag{A22}$$

One can use the convolution (A22) to compute the weighted decoders for

$$\hat{g}^{\pm}(x) = \frac{(1\pm x)^2}{1+x^2}$$

the solution is lengthy and thus we do not include it here.