RICE UNIVERSITY

Nonlinear Neural Codes

by

Qianli Yang

A THESIS SUBMITTED
IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE
Master

APPROVED, THESIS COMMITTEE:

Yanf Pitkow, Director
Professor of Electrical and Computer
Engineering and Computer Science

Behnam Aazhang, Chair
J.S. Abercrombie Professor of Electrical
and Computer Engineering

Don H. Johnson
J.S. Abercrombie Professor of Electrical
and Computer Engineering

Richard G. Baraniuk
Victor E. Cameron Professor of Electrical
and Computer Engineering

Andreas Tolias
Associate Professor of Neuroscience in
Baylor College of Medicine

Houston, Texas
December, 2015
ABSTRACT

Nonlinear Neural Codes

by

Qianli Yang

Most natural task-relevant variables are encoded in the early sensory cortex in a form that can only be decoded nonlinearly. Yet despite being a core function of the brain, nonlinear population codes are rarely studied and poorly understood. Interestingly, the most relevant existing quantitative model of nonlinear codes [1] is inconsistent with known architectural features of the brain. In particular, for large population sizes, such a code would contain more information than its sensory inputs, in violation of the data processing inequality. In this model, the noise correlation structures provide the population with an information content that scales with the size of the cortical population. This correlation structure could not arise in cortical populations that are much larger than their sensory input populations. Here we provide a better theory of nonlinear population codes that obeys the data processing inequality by generalizing recent work on information-limiting correlations [2] in linear population codes. Although these generalized, nonlinear information-limiting correlations bound the performance of any decoder, they also make decoding more robust to suboptimal computation, allowing many suboptimal decoders to achieve nearly the same efficiency as an optimal decoder. Although these correlations are
extremely difficult to measure directly, particularly for nonlinear codes, we provide a simple, practical test by which one can use choice-related activity in small populations of neurons to determine whether decoding is limited by correlated noise or by downstream suboptimality. Finally, we discuss simple sensory tasks likely to require approximately quadratic decoding, to which our theory applies.
# Contents

Abstract ii

1 Introduction 1
   1.1 Nonlinear neural codes 2
   1.2 Nonlinear choice correlation 4
   1.3 Information-limiting noise (‘bad noise’) 4

2 Materials and Methods 6
   2.1 Nonlinear neural codes 6
      2.1.1 Neural encoding 6
      2.1.2 Exponential family distribution and sufficient statistics 7
      2.1.3 Neural decoding 9
      2.1.4 Fisher information 10
   2.2 Information-limiting noise (‘bad noise’) 11
   2.3 Choice correlation 13

3 Results 15
   3.1 Nonlinear neural codes 15
   3.2 Information-limiting noise 16
   3.3 Nonlinear choice correlation 18

4 Discussion 21

5 Supplementary 23
Chapter 1

Introduction

Representations of stimulus in brain are tangled with each other. It’s a central task for the brain to untangle them and estimate the input stimulus accurately. Neural coding describes the relationship between stimulus and neural responses, which is the approach for us to examine how brain process this task. However, most task-relevant variables are encoded in the early sensory cortex in a form that can only be decoded nonlinearly. Thus, nonlinear neural coding is thus needed, which describes the information about stimulus is available in higher order statistics of neural responses.

To examine the existence of nonlinear neural coding in the brain, we introduce choice correlation [3] [4], which is the correlation between behavior choice and the neural responses. The choice correlation will tell us if the brain has made an optimal use of the information from neural responses to decide the behavior choices. In pure nonlinear neural coding, the linear choice correlation will be trivial. We will extend the idea of choice correlation in nonlinear framework and show that nonlinear choice correlation is the right variable worth consideration in experiments.

The current model for nonlinear neural coding [1] showed the necessity to capture the right form of nonlinearity in neural response space. However, the model violated the information inequality. The previous work in information-limiting correlation [2] can fix the the information inequality violation in linear neural coding. Here, we generalize information-limiting correlation in nonlinear neural coding and successfully realize the information threshold we can extract from a fixed input when the downstream
population keeps growing in size. Nonlinear information-limiting correlation can also make some suboptimal decoding schemes to achieve the information threshold. This allows the theoretical optimal nonlinear choice correlation to predict data in experiments successfully even if we don’t capture the exact right nonlinearity, thus making our nonlinear neural decoding to be more general and robust. [5]

1.1 Nonlinear neural codes

There are three reasons to introduce nonlinear neural codes. Firstly, task-relevant variables are nonlinear functions of sensory observations. Secondly, these task-relevant variables are entangled with task-irrelevant (nuisance) variables. Finally, perception requires nonlinear untangling to resolve the task.

Figure 1.1 : Tangly representations in sensory neurons require brain to do nonlinear computation.

Thus, we build an Encode-recode-decode model to describe information transferring process in brain. We assume information is represented and transmitted in neural populations. Encoding process is concerned with stimulus and upstream neural responses, which mostly come from sensory neurons. If the information about stimulus is encoded nonlinearly in sensory neurons, we have to do nonlinear computing, which
is done in recoding process. With all the true nonlinearity extracted in downstream neurons, we can linearly decode these downstream neurons to get the estimate of stimulus.

![Diagram](image1.png)

**Figure 1.2**: Encode, recode and decode scheme.

A simple example for nonlinear neural codes can be found in vision system. In the following example about checkershadow illusion, the task is to tell if square A and square B are the same image.

![Image](image2.png)

**Figure 1.3**: Checkershadow Illusion [6]. The information from the image is decoded nonlinearly in vision system with respect to reflectance and lighting. Image=Reflectance×Lighting.
If we decode the image linearly on lighting or reflectance we will come to the conclusion the squares A and B are the same shade of gray. However, we will have the illusion that the are different images because our brain has done nonlinear computation. This simple example shows the necessity to introduce nonlinear neural codes.

1.2 Nonlinear choice correlation

How can we test if brain do nonlinear neural codes and use the nonlinear information efficiently? We offer an innovative test named nonlinear choice correlation to answer this question.

Previous results show that the efficiency of a neural decoder can be related to the correlation between single neurons and behavioral decisions, a quantity known as choice correlation or choice probability [3] [4]. However, this is a test for linear neural codes. In pure nonlinear neural coding, linear choice correlation will be zero. This does not mean that the neuron has no effect on the behavioral choices. We will instead consider the effect of higher order statistics of neurons in the encoding process. Thus, the nonlinear choice correlation is necessary to be introduced to show how the information transmit from neural responses to behavioral choices.

1.3 Information-limiting noise (‘bad noise’)

Neuron population size in deeper layers is far greater than that in the sensory input. For instance, in the vision system, the population size of visual neuron in cortex is hundreds of times larger than that in retina. If readout information from the cortex kept growing with growing population size, this would exceed the information in the
input layer, which is not mathematically possible. This motivates us to understand
the noise structure that correctly limits the information in the output.

Information-limiting noise is noise that is indistinguishable from the signal, and
therefore cannot be averaged away. Thus it will bound the efficiency of decoding to
match the information in the input layer [2]. However, previous work in nonlinear
population codes [1] came to the conclusion that information will not saturate with
growing population size. In this paper, we will introduce information-limiting noise
into nonlinear population codes to make it more mathematically consistent.
Chapter 2

Materials and Methods

We will build models to simulate the encoding and decoding process when the information about stimulus is in higher order statistics of neural responses. The sufficient statistics of exponential family is used to analyze the choice of decoders. To measure the quality of decoding, we will introduce the Fisher information [7]. We will then extend the idea of choice correlation to verify the existence of nonlinear neural codes in brain. Lastly, nonlinear information-limiting noise will make sure it’s plausible to apply these data analysis methods when population size is large.

2.1 Nonlinear neural codes

2.1.1 Neural encoding

In a linear population code, all of the information about a stimulus is encoded in the mean neural responses. However, unlike a linear code, the information in nonlinear codes is not encoded in mean neural responses but instead by higher-order statistics. In what follows, we will quantify responses by a vector of spike counts $\mathbf{r}$ in an appropriate time window.

Consider a population of $N$ neurons that code for a scalar stimulus $s$, and denote by $r_i$ the response of $i$th neuron $(i = 1, 2, \ldots N)$. The response of neuron $i$ is given by

$$ r_i(s) = f_i + \eta_i $$  \hspace{1cm} (2.1)
where \( f_i \) is the tuning curve of neuron \( i \) and \( \eta_i \) is the trial-to-trial variability in the neural responses. The variability is assumed to follow a multivariate normal distribution with zero mean \( 0 = \langle \eta \rangle \) and covariance \( Q = \langle \eta \eta^\top \rangle \). When the tuning curve \( f(s) \) is function of stimulus, there is linear information encoded in the neuron responses. Whereas nonlinear information is encoded in responses when the trial-to-trial variability \( \eta(s) \) is a function of stimulus.

### 2.1.2 Exponential family distribution and sufficient statistics

We model the conditional probability distribution of neuronal responses given the stimulus following exponential family distributions. Many widely used distributions are in exponential family, such as Gaussian and Poisson distributions.

An exponential family distribution has the following form

\[
P(r|s) = \exp \left( \theta(s)^\top T(r) - \tau(s) + \phi(r) \right)
\]

where \( \theta(s) \) are the natural parameters, \( T(r) \) are the sufficient statistics, \( \tau(s) \) and \( \phi(r) \) are the log normalizer and underlying measure. The statistics \( T(r) \) are called sufficient because the likelihood for \( \theta(s) \) only depends on \( r \) through \( T(r) \).

When \( T(r) = r \), it’s the model for linear neural encoding. When \( T(r) = R(r) \), where \( R \) is nonlinear function of \( r \), we have a model for nonlinear neural encoding.

The simplest example would be quadratic neural encoding where \( R(r) = \text{vec}(rr^\top) \). Gaussian distribution with stimulus-dependent covariance would be a ideal setting for quadratic codes. Assume the distribution of neural responses follows

\[
P(r|s) = \frac{1}{\sqrt{(2\pi)^N \det(Q(s))}} \exp\left(-\frac{1}{2}(r - f)^\top Q(s)^{-1}(r - f)\right)
\]
where the natural parameters and corresponding sufficient statistics are

\[
\theta(s) = (\ldots (Q^{-1} f)_i, \ldots, (\frac{-1}{2} Q^{-1})_{ij}, \ldots)^\top
\]

\[
T(r) = (\ldots, r_i, \ldots, r_i r_j, \ldots)^\top. \tag{2.4}
\]

If the tuning curves \( f(s) \) are dependent on the stimulus but the covariance \( Q \) is not, then the sufficient statistics are linear, \( T(r) = (\ldots r_i\ldots) = r \). In this circumstance, linear population decoding is sufficient to readout the information about the stimulus. However, when the tuning curves \( f \) do not depend on the stimulus but the covariance \( Q(s) \) is dependent on the stimulus, then the sufficient statistics are quadratic, \( T(r) = (\ldots r_i r_j \ldots)^\top \). In this situation, a quadratic decoding scheme is necessary to extract the information about the stimulus.

Figure 2.1 : Linear and nonlinear neural encoding. Assuming it’s in two dimensional neural response space, left plot shows linear neural encoding where the tuning curve is dependent on stimulus. In contrast, right plot shows quadratic neural encoding where the covariance is stimulus-dependent.

To model the stimulus-dependent covariance, we can change the orientation or the scaling of covariance when we change stimulus. Thus we assume

\[
Q(s) = V \Lambda V^* \tag{2.5}
\]
Where $V$ is the rotation matrix, $V^*$ is the conjugate transpose of $V$ and $\Lambda$ is positive definite diagonal matrix [8]. If $V(s)$ is dependent on stimulus, the orientation of the covariance changes with stimulus. If $\Lambda(s)$ is stimulus-dependent, the scaling of the covariance changes with stimulus. We will discuss both settings in the following analysis when we study quadratic codes.

$$Q(s) = V\Lambda(s)V^*$$

$$Q(s) = V(s)\Lambda V^*(s)$$

Figure 2.2: Stimulus-dependent covariance. Left plot is stretching covariance and right plot is rotating covariance.

### 2.1.3 Neural decoding

In this quadratic neural codes, we need to collect all the product of lower stream neural responses in the recoding process. Thus we can get the downstream neural responses $R_k = r_ir_j$, $i, j = 1, 2, ..., N$, and $k = 1, 2, ..., N(N+1)/2$. Then in decoding process, this become a linear decoding for downstream neural responses.

The locally optimal weights for downstream neurons can be solved by minimizing of the mean square error $E = \langle (\hat{s} - s)^2 \rangle_{p(r|s)}$ under unbiased constraint $\partial_s \langle \hat{s} \rangle_{p(r|s)} = 0$ (Details in Supplementary),

$$w_{\text{opt}} = \frac{\Gamma^{-1}F'}{F'\Gamma^{-1}F'}$$

$$\Gamma = \text{Cov}(R), F = \langle R \rangle$$
where $F'$ is the derivative of $F$ with respect to stimulus.

An alternative way to get weights would be linear regression. In the readout layer, we have recoded all the nonlinear information using a quadratic nonlinearity, so the output can then be read out from the readout layer with a linear weighting. Thus we can using linear regression to calculate weights between the hidden and readout layer. The expression of linear regression weights is as follows

$$w_{\text{regression}} = \langle R^\top R \rangle^{-1} \langle R^\top s \rangle \quad (2.7)$$

In the supplementary material, it is shown that the decoding weights in two methods are equivalent except an unbiased constant.

### 2.1.4 Fisher information

How can we examine the efficiency of the decoder? How can we tell if we have extracted all accessible information from stimulus. Here we use the Fisher information, which measures the information locally around stimulus \[7\]. In general, the Fisher information is defined as

$$J(P(r|s)) = - \left\langle \frac{\partial^2}{\partial s^2} \log P(r|s) \right\rangle_{P(r|s)} = \left\langle \left( \frac{\partial}{\partial s} \log P(r|s) \right)^2 \right\rangle_{P(r|s)} \quad (2.8)$$

For the multivariate Gaussian distribution, the Fisher information can be divided into linear term and nonlinear term, $J = J_{\text{mean}} + J_{\text{cov}}$, where \[9\]

$$J_{\text{mean}}(s) = f'^\top(s)Q^{-1}f'(s)$$

$$J_{\text{cov}}(s) = F'(s)\Gamma^{-1}F'(s) \quad (2.9)$$
According to the Cramér – Rao Bound [10], the Fisher information bounds the variance of an unbiased estimator

\[
\frac{1}{\text{Var}(\hat{s})} \leq J
\]  
(2.10)

We can calculate the variance of estimate \( \hat{s} \) when we choose optimal weights in equation 2.6, we can get

\[
\hat{s} = \frac{1}{\mathcal{F}^T(s)\mathcal{F}(s)} = \frac{1}{J}. 
\]

This shows the optimal decoding gives the best decoding efficiency. For other suboptimal decoding, we can calculate the variance of estimate and compare it with the Fisher information. Thus we can know how much information we have got with these suboptimal decoding method.

### 2.2 Information-limiting noise (‘bad noise’)

Previous model in population codes [1] violates data processing inequality. Here we introduce information-limiting noise [2] to fix this flaw. Information-limiting noise is the noise that cannot be distinguished from signal. A possible source would be the input noise. To examine it in linear neural codes, we assume neural responses follows Gaussian distribution with mean determined by stimulus, \( p(r|s) \sim N(f(s), Q_0) \). The Fisher information will be \( J_0(s) = f^T(s)Q_0^{-1}f(s) \), which will increase linearly with growing population size.

Assume the bad noise follows zero mean Gaussian distribution, \( ds \sim N(0, \epsilon) \). Previous input stimulus is a specific number. However, with inherited bad noise, it will become a distribution. Although it will not affect the mean tuning, the covariance of \( r \) will be different. The distribution of responses given stimulus will become

\[
p(r|s) \sim N(f(s), Q_0 + \epsilon f^T f^T) 
\]  
(2.11)

The uncertainty of responses will include independent noise \( Q_0 \) and the uncertainty of input stimulus due to inherited bad noise.
To see the effect of the bad noise, we can calculate the Fisher information

\[ J = f'^T(s)Q^{-1}f'(s) = \frac{1}{\alpha + 1/J_0} \]  

(2.12)

where we applied Sherman-Morrison formula when we calculate the inverse of \( Q \).

From above expression, we can find that when \( \alpha \) is relatively large, \( J \approx \alpha \). This set a threshold to the output information, which matches the information limit from the input.

Figure 2.3 : Information-limiting noise will limit the Fisher information

We generalize the information-limiting noise in nonlinear neural codes. For instance, when we assume distribution of responses follows Gaussian distribution with stimulus-dependent covariance, \( \mathbf{r} \sim N(0, Q(s)) \). Because the bad noise cannot be distinguished from input, \( ds \sim N(0, \epsilon) \) will always go with the input stimulus \( s \). This bad noise will not change the covariance of responses in average, however, it will change the mean of fourth order statistics of responses,

\[ \Gamma = \text{Cov} (\mathbf{R}|s) = \Gamma_0 + \epsilon \mathbf{F}' \mathbf{F}'^T \]  

(2.13)

where \( \Gamma_0 \) and \( \mathbf{F} \) is defined in equation 2.6, and \( \mathbf{R}(\mathbf{r}) = \text{vec}(\mathbf{r}\mathbf{r}^T) \). Here vec mean to
flatten out a multidimensional array, making the variable to be a vector.

<table>
<thead>
<tr>
<th>Linear</th>
<th>Nonlinear</th>
</tr>
</thead>
<tbody>
<tr>
<td>decoder: $s = \mathbf{w} \mathbf{r}$</td>
<td>$s = \mathbf{W} \mathbf{R}(\mathbf{r})$</td>
</tr>
<tr>
<td>FI: $J_0 = \mathbf{f}' \mathbf{Q}_0^{-1} \mathbf{f}'$</td>
<td>$J_0 = \mathbf{F}' \mathbf{\Gamma}_0^{-1} \mathbf{F}'$</td>
</tr>
<tr>
<td>ILN: $\mathbf{Q} = \mathbf{Q}_0 + \epsilon \mathbf{f}' \mathbf{f}'^\top$</td>
<td>$\mathbf{\Gamma} = \mathbf{\Gamma}_0 + \epsilon \mathbf{F}' \mathbf{F}'^\top$</td>
</tr>
<tr>
<td>FI with ILN: $J = \frac{1}{J_0 + \epsilon}$</td>
<td></td>
</tr>
</tbody>
</table>

Figure 2.4: Comparison of information-limiting noise (ILN) in linear and nonlinear neural codes

In Figure 2.4, we compare the information-limiting noise in linear and nonlinear neural codes. For nonlinear neural codes, only nonlinear ILN can limit the information in the output, where linear ILN will fail.

### 2.3 Choice correlation

Choice correlation gives a solid test to see if brain uses information optimally. Here we generalize previous work in linear choice correlation [4] [3] to nonlinear neural codes. Linear choice correlation $C_{\text{linear}}^k$ is the Pearson correlation coefficient between response of one neuron and the behavioral choice $\hat{s} = \mathbf{w}^\top \mathbf{r}$

$$C_{\text{linear}}^k = \text{Corr}(\hat{s}, r_k) = \frac{\langle \hat{s} r_k \rangle - \langle \hat{s} \rangle \langle r_k \rangle}{\sqrt{\langle (\hat{s})^2 \rangle - \langle \hat{s} \rangle^2 \langle (r_k^2) - \langle r_k \rangle^2 \rangle}} = \frac{\langle \mathbf{Q} \mathbf{w} \rangle_k}{\sqrt{\mathbf{Q}_{kk} \mathbf{w}^\top \mathbf{Q} \mathbf{w}}} \quad (2.14)$$
For nonlinear choice correlation, we will consider the correlation between the nonlinear unit $R_k$ and the behavior choice $\hat{s} = w^\top R$

\[ C_{\text{nonlinear}}^k = \text{Corr}(\hat{s}, R_k) = \frac{\langle \hat{s}R_k \rangle - \langle \hat{s} \rangle \langle R_k \rangle}{\sqrt{\langle \hat{s}^2 \rangle - \langle \hat{s} \rangle^2} \sqrt{\langle R_k^2 \rangle - \langle R_k \rangle^2}} = \frac{(\Gamma w)_k}{\sqrt{\Gamma_{kk} w^\top \Gamma w}} \]  

(2.15)

Where $\Gamma_{kk}$ is the variance of $R_k$. 

Chapter 3

Results

We will study quadratic codes as an example. Through this example, we want to show how nonlinear neural code works with information-limiting noise. And we performed simulation to test if nonlinear choice correlation can give us hint about nonlinear coding.

3.1 Nonlinear neural codes

We assume neural encoding model described in Equation 2.3, where the stimulus-dependent covariance is Equation 2.5. In the downstream neurons, we have all the quadratic statistics of upstream neurons, $R(r) = \text{vec} (rr^\top)$. The decoding weights will be locally optimal weights as showed in Equation 2.6. To show the efficiency of decoding, we compare theoretical information bound described in Equation 2.9 and extracted information in the output $\frac{1}{\sigma_i^2}$. In the simulation, we change the number of upstream neurons $N_r$. Thus the number of downstream quadratic neurons is $N_R = N_r(N_r + 1)/2$. 
As shown in Figure 3.1, we can get all the information when we decode all quadratic units. The information grows linearly with growing number of decoded downstream neurons.

3.2 Information-limiting noise

With bad noise added, we hope to set a maximum for the information we can extract from a fixed input when the downstream population keeps growing in size. Thus, in above quadratic codes, we add Information-limiting noise as showed in Equation 2.13. We will also evaluate another suboptimal decoding scheme. In optimal decoding scheme, decoded downstream quadratic neuron number is $N_R = N_r(N_r + 1)/2$. In contrast, in suboptimal decoding scheme, we randomly choose $3N_r$ downstream neurons to decode. Thus we only have a fraction of sufficient statistics in downstream layer. When there has no information-limiting noise, the information ratio between
suboptimal decoding and optimal decoding will decrease with growing upstream neurons. The efficiency of this suboptimal decoding scheme will be very poor when we have large population size. However, when we have Information-limiting noise, the threshold will make suboptimal decoding to get some information even when the population size is large.

Figure 3.2: Information-limiting noise sets a information threshold and makes decoding to be more robust. Left plot shows how the information grows when we add bad noise. Right plot shows the information ratio between suboptimal decoding and optimal decoding. With bad noise, The ratio will saturate with growing population size.

Figure 3.2 shows that the suboptimal decoding can still extract almost half of the total information if we have ILN. Thus we can claim that information-limiting noise can help decoding scheme to be more robust. Because the information threshold set by ILN, the decoding scheme doesn’t need to be perfectly right. This makes the neural decoding to be more biological plausible because the plasticity of brain is enormous and mistakes are inevitable when information transferring between different layers. ILN can assure that at the decision layer, we still have considerable information even brain made some mistakes.
3.3 Nonlinear choice correlation

The linear choice correlation is defined in Equation 2.14. When applying the weights for locally optimal linear decoding $w_{opt} \propto Q^{-1} f'$, it has been showed the linear choice correlation can be expressed as follows [3] [4],

$$
C_{\text{linear}}^k = \text{Corr}(\hat{s}, r_k) = \frac{(QQ^{-1}f')_k}{\sqrt{Q_{kk}f'Q^{-1}f'}} = \frac{f'_k}{\sigma_k} \sigma_{\hat{s}} = \sqrt{\frac{J_k}{J}}
$$

(3.1)

where $\sigma_k$ is the variance of $r_k$, $\sigma_{\hat{s}}$ is the variance of the estimate for stimulus and $f'$ is the derivative of the tuning curve over stimulus. $J_k$ is the Fisher information contains in one neuron and $J$ is the Fisher information for the neuron population. From the perspective of information transmission, choice correlation is related to fraction of the information contained in one neuron to the information contained in the whole population.

In pure quadratic codes, we have $f'_k = 0$. Thus the linear choice correlation is zero. According to this result, if we arrive at the conclusion that the behavior choice has no connection to the single neuron, this will be an enormous error. The linear choice correlation in our model is zero because we don’t have information about stimulus in the mean of the response. However, what really affects the behavior choice in our model is the second order statistics of neuron responses. We need to use the nonlinear choice correlation to describe decoding process in this model.

The definition for nonlinear choice correlation is in Equation 2.15. Here we also apply optimal decoding weights, $w_{opt} \propto \Gamma^{-1} F'$. Nonlinear choice correlation can be expressed as follows

$$
C_{\text{nonlinear}}^k = \text{Corr}(\hat{s}, R_k) = \frac{(\Gamma\Gamma^{-1}F')_k}{\sqrt{\Gamma_{kk}F'\Gamma^{-1}F'}} = \frac{F'_k}{\sigma_k} = \sqrt{\frac{J_k}{J}}
$$

(3.2)

With respect to information transmission, nonlinear and linear choice correlation is consistent to each other. For pure quadratic codes, $R_k = r_i r_j$ is the product of two
upstream neural responses. The nonlinear choice correlation for $R_k$ would be the information ratio in one pair of neurons respect to the information in the population. In the following simulation, we will calculate choice correlation in quadratic codes. To calculate quadratic choice correlation as showed in Equation 3.2, we need to first decode every pair of neurons and use the inverse of the estimates’ variance to be information in pairs. Dividing that over population information, we can get theoretical choice correlation for quadratic codes. We can compare the results with simulated choice correlation, which is the correlation between population estimate and the correlation between neuron pairs. In other word, it’s showed in the correlation(significant relationship) between correlations(with choice) of correlations(quadratic code).

We also evaluate another suboptimal decoding methods. The suboptimal weights are $w_{\text{sub}} = \text{sgn}(F')$. This decoding method only consider the sign of $F'$ where as optimal decoding weights will divide $F'$ with covariance of $R$.

Moreover, we will introduce information-limiting noise to see if it will help theoretical choice correlation to be more interpretable for suboptimal decoding.
From Figure 3.3, we can find that blue dots in both plots are in the diagonal line, showing theoretical choice correlation can predict simulated choice correlation when decoding weights are optimal. This is because the calculation of theoretical choice correlation is based on optimal decoding assumption. Red dots are choice correlations when decoding is suboptimal. They are on the diagonal line even with information-limiting noise. Thus we can conclude that information-limiting noise can enhance the interpretability of theoretical choice correlation.
Chapter 4

Discussion

Previous works in neural codes pay most attention on linear codes. However, if we want to figure out the principle of brain, we have to study nonlinear codes. Because if natural tasks could be solved with linear computation, then we wouldn’t need a brain. We could just wire our sensors to our muscles and accomplish the same goal, because multiple iterations of linear processing is just another single linear processing step. Real problems require the brain to expand its representation nonlinearly, so that simple operations to solve them. Complex tasks require a lot of untangling to isolate the task-relevant properties.

This paper build a nonlinear population coding scheme and offer a effective test, nonlinear choice correlation. Most of the analysis in this paper is focused in quadratic codes, which is the simplest nonlinear codes. However, we can extend the analysis and results in complex nonlinearity. We can change the expression of downstream neurons $R$. However, another question is needed to be answered: Which nonlinearities to consider?

One sensible strategy is an expansion in powers of upstream neuron responses $r$. We may consider powers of neural responses up to the number of neurons we record, such that the highest order interaction would be $\prod_i r_i$. This approach is akin to using a Taylor series expansion of the neural nonlinearities, $\{r_i, r_ir_j, r_ir_jr_k, \ldots\}$, to approximate the sufficient statistics that the brain assumes in decoding the stimulus. We will also consider other sets of nonlinearities, such as radial basis functions, random
nonlinearities [11] [12] [13], or nonlinear features learned in deep discriminative networks [14] [15]. These expansions need not exactly match those used by the brain. As long as they can approximate the real nonlinearities then we can still find the effective nonlinearity (Figure 4.1). This insensitivity to fine details of neural nonlinearities, and sensitivity to the computational form of the nonlinearities, is a benefit of modeling computation in the informational space, rather than trying to model individual neural nonlinearities.

![Image of Figure 4.1](image.jpg)

**Figure 4.1**: Model for how an effective nonlinearity can replicate the true nonlinearity. The true nonlinearity is one function, $T(r)$, and it is decoded according to some weights $W$. We infer an approximation to the function $W \cdot T(r)$ by a linear combination $W_p \cdot R_p(r)$ of other nonlinearities. We can compute the nonlinear choice correlations for these latter nonlinearities $R_p$ using weights inferring method in [3].
Chapter 5

Supplementary

5.1 Fisher information for exponential family distribution

The log probability of an exponential family distribution is

\[ L(s, r) = \log P(r | s) \]

\[ = \theta(s)^\top T(r) - \tau(s) + \phi(r) \tag{5.1} \]

Let’s denote the mean of the sufficient statistics as \( F = \langle T \rangle \).

The Fisher information is defined as

\[ J(P(r | s)) = - \left\langle \frac{\partial^2}{\partial s^2} \log P(r | s) \right\rangle_{P(r | s)} \]

\[ = \left\langle \left( \frac{\partial}{\partial s} \log P(r | s) \right)^2 \right\rangle_{P(r | s)} \tag{5.2} \]

Another condition is

\[ \frac{\partial \langle L \rangle}{\partial \theta_i} = F_i - \frac{\partial \tau}{\partial \theta_i} = 0 \tag{5.3} \]

Using (4), we can get \( \tau' \) and \( \tau'' \)

\[ \tau' = \sum_i \frac{\partial \tau}{\partial \theta_i} \frac{\partial \theta_i}{\partial s} = \theta^\top F \tag{5.4} \]

\[ \tau'' = \theta''^\top F + \theta'^\top F' \tag{5.5} \]
So we can get Fisher information’s two expressions.

\[
J(P(r|s)) = -\left(\frac{\partial^2}{\partial s^2} \log P(r|s)\right)_{P(r|s)} \\
= \tau'' - \theta''^T F \\
= \theta'^T F'
\]  

\[
J(P(r|s)) = \left(\frac{\partial}{\partial s} \log P(r|s)\right)^2_{P(r|s)} \\
= \theta'^T (TT^T - FF^T) \theta' \\
= \theta'^T \Gamma_{TT} \theta'
\]

Combining equation (28) and (29), we can get

\[
\theta' = \Gamma_{TT}^{-1} F'
\]  

So the Fisher Information for exponential family will be

\[
J(P(r|s)) = F'^T \Gamma_{TT}^{-1} F'
\]

This result will be useful for future studies where the relevant nonlinear code is neither linear nor quadratic. In Multivariate Gaussian distribution, we assume the tuning curve is zero and the covariance has linear dependence on stimulus in diagonal elements and constants in off-diagonal elements:

\[
f = 0 \\
Q_{ij}(s) = \delta_{ij} s + (1 - \delta_{ij}) c
\]

where \(\delta_{ij}\) is a Kronecker delta, \(c\) is constant. Thus all the information is in quadratic sufficient statistics.

For \(n\) neurons, we vectorize quadratic sufficient statistics to be

\[
T(r) = (r_i r_i, ..., r_i r_j (i \neq j), ...) 
\]
Apply 4th-order central moment of multivariate Gaussian distribution

\[ E[r_ir_jr_kr_n] = Q_{ij}Q_{kn} + Q_{ik}Q_{jn} + Q_{in}Q_{jk} \]  

(5.12)

Considering the mean of response is zero, we can express \( \Gamma_{FF} \) with 7 different elements.

\[
\begin{align*}
\mu_6 &= \langle r_i^2r_jr_k \rangle - \langle r_ir_j \rangle \langle r_ir_k \rangle \\
\mu_5 &= \langle r_i^2r_j^2 \rangle - \langle r_ir_j \rangle \langle r_j^2 \rangle \\
\mu_4 &= \langle r_i^4 \rangle - \langle r_i^2 \rangle \langle r_i^2 \rangle \\
\mu_3 &= \langle r_i^3r_j \rangle - \langle r_i^2 \rangle \langle r_ir_j \rangle \\
\mu_2 &= \langle r_i^2r_j^2 \rangle - \langle r_j^2 \rangle \langle r_j^2 \rangle \\
\mu_1 &= \langle r_i^2r_jr_k \rangle - \langle r_j^2 \rangle \langle r_jr_k \rangle \\
\mu_0 &= \langle r_ir_jr_kr_n \rangle - \langle r_i^2 \rangle \langle r_jr_n \rangle
\end{align*}
\]  

(5.13)

Considering the symmetry of covariance matrix of sufficient statistics, the inverse matrix of \( P = \Gamma_{TT}^{-1} \) can also be expressed with 7 different elements, correspondingly we name them \( \nu_6 \) to \( \nu_0 \). Thus we can get the inverse matrix with equation

\[
\sum_{kl} P_{ijkl} \Gamma_{kln} = I_{im}I_{jn} \]  

(5.14)

Where \( I \) is identity matrix. The derivative over stimulus of the mean of the sufficient statistics is \( F' = (...1..., ...0...) \). Applying Equation (31), we can get the Fisher information for our model.

\[
J(P|r|s) = \nu_2 (n^2 - n) + \nu_4n = \frac{n[c^2(n^2 - 3n + 3) + 2c(n - 2)s + s^2]}{2(c - s)^2[c(n - 1) + s]^2}
\]  

(5.15)

5.2 Optimal weights for linear codes

How can we get the optimal weights to estimate a specific stimulus \( s_0 \)? We need to introduce a distribution of \( s \), otherwise we have no knowledge about signal. The
covariance around $s_0$ can only give us noise. This is why we want to introduce prior distribution $p(s)$. So the following analysis has nothing to do with bad noise. It’s just the need from the regression’s assumption.

The relationship between regression and bad noise is both of them add a distribution for input stimulus, regression knows what is stimulus and bad noise doesn’t. Thus bad noise sets a threshold for accessible information.

### 5.2.1 Assumption

Assumption is

$$P(s) \sim N(s_0, \epsilon)$$

$$P(r|s) \sim N(f(s), Q)$$

(5.16)

Here $Q$ is the covariance matrix doesn’t depend on $s$, which should be

$$Q = \langle (r - f(s))(r - f(s))^\top \rangle_{p(r|s)}$$

(5.17)

Note it should be $f(s)$ instead of $f_0 = f(s_0)$. The estimate to $s$ should be

$$\hat{s} = w^\top (r - f_0) + s_0$$

(5.18)

When we do taylor expansion for $f$, $f = f_0 + f'_0(s - s_0)$, we will have the following results

$$\hat{s} - s = w^\top (r - f) + w^\top (f - f_0) - (s - s_0)$$

$$= w^\top (r - f) + w^\top f'_0(s - s_0) - (s - s_0)$$

(5.19)
\[(\hat{s} - s)^2 = w^\top (r - f)(r - f)^\top w + (s - s_0)^2 w^\top f_0 f_0^\top w + (s - s_0)^2 + 2(s - s_0)w^\top (r - f) f_0^\top w - 2(s - s_0)^2 w^\top f_0^t - 2(s - s_0)w^\top (r - f)\]

\[(5.20)\]

Then we take expectation over \(p(r|s)\)

\[\langle (\hat{s} - s)^2 \rangle_{p(r|s)} = w^\top f_0^t (s - s_0) - (s - s_0)\]

\[(5.21)\]

\[\langle (\hat{s} - s)^2 \rangle_{p(r,s)} = w^\top (Q + (s - s_0)^2 f_0^t f_0^\top f_0^\top) w - 2(s - s_0)^2 w^\top f_0^t + (s - s_0)^2\]

\[(5.22)\]

### 5.2.2 Biased optimal weights

To get our optimal weights, we can minimize the mean square error. The question is how to define mean square error. Let’s denote \(\langle (\hat{s} - s)^2 \rangle_{p(r|s)} = E_1\), \(\langle (\hat{s} - s)^2 \rangle_{p(r,s)} = E_2\).

We will discuss it separately.

Setting \(\partial_w E_1 = 0\), we can get

\[\arg\min_w E_1 = (s - s_0)^2 (Q + (s - s_0)^2 f_0^t f_0^\top)^{-1} f_0^t\]

\[(5.23)\]

If we make \(s = s_0\), then \(w = 0\). This tells us nothing because if you want to do get the estimate about the weights, you need to take an average around \(s_0\). Thus the right definition for mean square error should be \(E_2\). Interestingly the relationship between \(E_1\) and \(E_2\) is \(E_2 = \langle E_1 \rangle_{p(s)}\).

\[E_2 = \langle (\hat{s} - s)^2 \rangle_{p(r,s)} = w^\top (Q + \epsilon f_0^t f_0^\top) w - 2\epsilon w^\top f_0^t + \epsilon\]

\[(5.24)\]
argmin \ E_2 = \epsilon (Q + \epsilon f_0 f_0^\top)^{-1} f_0' \\
= \frac{\epsilon}{1 + \epsilon J_0} Q^{-1} f_0' \tag{5.25}

where \( J_0 = f_0^\top Q^{-1} f_0' \). This is different from Xaq’s note because we have another term which resembles bad noise. But this term doesn’t depend on stimulus. This is the regression weights. But it’s biased.

### 5.2.3 Unbiased optimal weights

If it’s unbiased weights, we need to include a constraint

\[
\begin{aligned}
\text{argmin}_{w} \ E_1 & \quad \text{s. t.} \quad \partial_s \langle \hat{s} \rangle_{p(r|s)} = 1 \\
\tag{5.26}
\end{aligned}
\]

The constraint can be simplified to \( w^\top f_0' = 1 \).

Using Lagrange Multiplier, we can get

\[
\begin{aligned}
\partial_w \left( E_1 - \lambda (\partial_s \langle \hat{s} \rangle_{p(r|s)} - 1) \right) &= 0 \\
(Q + (s - s_0)^2 f_0' f_0'^\top) w &= (s - s_0)^2 + \lambda) f_0' \\
(5.27)
\end{aligned}
\]

\[
w = B Q^{-1} f_0' 
\]

where \( B = \frac{(s-s_0)^2+\lambda}{1+(s-s_0)^2 J_0} \).

Substitute the solution to the constraint, we can get \( B = 1/J_0 \) Thus the unbiased weight is

\[
w = \frac{Q^{-1} f_0'}{J_0} \tag{5.28}
\]

Note here we solve the problem with two steps. The first is to take expectation over \( p(r|s) \), the second step is to take expectation over \( p(s) \).

This is unbiased weights, if we want to get unbiased weights from regression weights, we need to include a scale.

\[
w_{\text{unbiased}} = (1 + \frac{1}{\epsilon J_0}) w_{\text{biased}} \tag{5.29}
\]
5.2.4 Linear regression weights

What is the typical process for linear regression process?

Firstly, we have the distribution of inputs \( s \), size should be \( m \times 1 \), where the sample number is \( m \). Then we produce responses according to the probability \( p(r|s) \), size of \( r \) is \( n \times m \), \( n \) is the neuron number. The linear regression weights are produced by

\[
\mathbf{w}_{\text{regression}} = \left( (\mathbf{r} - \langle \mathbf{r} \rangle_{p(s)}) \cdot (\mathbf{r} - \langle \mathbf{r} \rangle_{p(s)})^\top \right)^{-1} \cdot (\mathbf{r} - \langle \mathbf{r} \rangle_{p(s)}) \cdot (s - s_0)
\]

\[
= \epsilon (Q + \epsilon f_0^T f_0)^{-1} f_0'
\]

\[
= w_{\text{biased}}
\]

Notice that when we do matrix multiplication, it’s equivalent to do average over the distribution of \( p(s) \). Why don’t we have the denominator \( m \) in every term? Because they cancel each other! Thus linear regression weight is in the exact same form as biased optimal weights.

5.2.5 Linear regression gives wrong answer?

Textbooks tell us linear regression gives us unbiased estimate for weights. How can that be in our example here?

This is because we keep digging at \( p(r|s) \), however, the right decoding process should look at posterior \( p(s|r) \). Following we will start from the posterior, we will just
focused at terms related to $s$. $r$ terms will only gives us normalized terms.

\[
p(s|r) \propto p(r|s)p(s)
\]

\[
\propto \exp\left(-\frac{1}{2}(r - f)^\top Q^{-1}(r - f) - \frac{(s - s_0)^2}{2\varepsilon}\right)
\]

\[
\propto \exp\left[-\frac{1}{2}(ds^2(J_0 + 1/\varepsilon) - 2dsf'_0Q^{-1}r)\right]
\]

\[
\propto \exp\left[-\frac{J_0 + 1/\varepsilon}{2}(ds - \frac{\varepsilon}{1+\varepsilon J_0}f'_0Q^{-1}r)^2\right]
\]

\[
= \exp\left[-\frac{1}{2\sigma^2_{s|r}}(ds - \text{unbiased } r)^2\right]
\]

where $ds = s - s_0$, $\sigma^2_{s|r} = \frac{\varepsilon}{1+\varepsilon J_0}$.

This tells us linear regression gives unbiased estimate for the parameter in $p(s|r)$. However, the meaning of this unbiasedness is not the same as our definition that $\hat{s} = s$.

### 5.2.6 MAP or MLE

From equation 16, we can find that MAP gives us biased weights. Likelihood function $p(r|s)$ can be expressed as

\[
p(r|s) \propto \exp\left(-\frac{1}{2}(r - f)^\top Q^{-1}(r - f)\right)
\]

\[
\propto \exp\left[-\frac{J_0}{2}(ds - \frac{f'_0Q^{-1}r}{J_0})^2\right]
\]

\[
= \exp\left[-\frac{1}{2\sigma^2_{r|s}}(ds - \text{unbiased } r)^2\right]
\]

Thus MLE gives us unbiased weights. This is because Fisher information is applied with likelihood instead of posterior.
5.2.7 Conclusion

When we don’t know the accurate expression for Fisher information or we just want to decode subset of neurons, linear regression is a powerful method. But it can only give us biased weights because it evaluates with MAP. However, we notice that the only difference between biased and unbiased weights is only the scale. To solve that, we can apply the constraint condition $\mathbf{w}^\top \mathbf{f}_0' = 1$, where we only need to get numerical expression for $\mathbf{f}_0'$. 
Bibliography


