# Inverse Ising Algorithms for Biological Data

Hanlin Tang
*Graduate Program in Biophysics, Harvard University*
(Dated: May 16, 2011)

The Ising Model provides a powerful description of many biological phenomena with correlated activity between multiple components, from neuronal spiking to correlated mutations in proteins. A central problem to utilizing this model is the 'Inverse Ising' challenge – to derive coupling constants from observable data. Fortunately, many algorithms for this problem have been developed to find couplings for neural spiking data, that can then be applied to other biological problems. We review several these approaches, and then experimentally test two such approximation algorithms on actual neuronal firing data. We find that the simplest approximation works remarkably well for network sizes of $N < 40$.

## I. CORRELATIONS

Many biological phenomena, from neuronal firing in the retina [1] to mutations along a protein [2], involve correlations among multiple components. This provides a convenient analogy to the Ising Model, where the framework of statistical mechanics can be applied to describe such complex behavior.

Consider a system of $N$ spins denoted as $\vec{\sigma} = \{\sigma_1, \sigma_2, \cdots, \sigma_N\}$, where each spin can take values of $\sigma_i = 0$ or 1. If we work in units where $k_B T = 1$, the probability of observing a particular configurations of spins is then given by the usual boltzmann factor

$$P(\vec{\sigma}) = \frac{1}{\mathcal{Z}} \exp\left[\sum_i h_i \sigma_i + \sum_{i \neq j} J_{ij} \sigma_i \sigma_j\right] \tag{1}$$

where $h_i$ is a local weight and $J_{ij}$ is a matrix of interaction terms of that capture the correlations between different spins. This general model has wide applicability to many biological systems. For example, each spin $\sigma_i$ can represent a single neuron, with $\sigma_i = 1$ denoting spiking, and $\sigma_i = 0$ denoting no activity. Then, the Ising Model can describe correlations between neuronal spiking at a population level.

The Ising Model is also particularly useful in describing the pattern of correlated mutations in a give protein, which may provide clues as to the 3-D structure. The main practical challenge in all these applications is the estimation of the Ising Model parameters from the observed data. Fortunately, many of these 'Inverse Ising' algorithms have been developed to analyze neuronal firing data that can be readily applied to other biological phenomenon.

We first formulate the basics of the 'Inverse Ising' problem, then review several approaches applied in computation neuroscience. We conclude with a computational experiment that tests two of these approaches against actual neuronal firing data. In particular, the two tested approaches represent the simplest and most efficient approximation, and the more accurate, but more expensive algorithmic approach.

## II. MAXIMUM ENTROPY MODEL

The application of the Ising Model is not just a matter of convenience; the model is the maximum entropy model that includes pairwise correlations [3][4]. To demonstrate this, we define a set of features that describe a particular observation $\vec{\sigma}$.

$$\vec{f}(\vec{\sigma}) = (\sigma_1, \cdots, \sigma_N, r_1 r_2, r_1 r_3, \cdots, r_{N-1} r_N) \tag{2}$$

We seek a distribution $p(\vec{\sigma})$ that maximizes the entropy subject to the constraint that it much describe the expectations of $\vec{f}$ in the data. Formally, we wish solve

$$\max_{\lambda, p(\vec{\sigma})} \mathcal{L}_\lambda[p(\vec{\sigma})] \tag{3}$$

where we construct the objective function as the entropy with Lagrangian multiplers $\lambda_j$ for each feature $f_j$ constraint, as well as $\lambda_0$ for the normalization requirement of the probability distribution:

$$\mathcal{L}_\lambda[p(\vec{\sigma})] = -\sum_{\vec{\sigma}} p(\vec{\sigma}) \log(p(\vec{\sigma})) + \lambda_0 \left[\sum_{\vec{\sigma}} p(\vec{\sigma}) - 1\right] \tag{4}$$

$$+ \sum_j \lambda_j \left[\sum_{\vec{\sigma}} p(\vec{\sigma}) f_j(\vec{\sigma}) - \langle f_j \rangle_{\text{data}}\right] \tag{5}$$

We solve by extremizing with respect to $p(\vec{\sigma})$,

$$\frac{\partial \mathcal{L}}{\partial p(\vec{\sigma})} = \log[p(\vec{\sigma})] + 1 + \lambda_0 + \sum_j \lambda_j f_j(\vec{\sigma}) = 0 \tag{6}$$

yielding the distribution

$$p(\vec{\sigma}) = \frac{\exp(-\sum_j \lambda_j f_j(\vec{\sigma}))}{\exp(1 + \lambda_0)} = \frac{1}{\mathcal{Z}} \exp\left[-\sum_j \lambda_j f_j(\vec{\sigma})\right] \tag{7}$$

where $\mathcal{Z}$ is the partition function and normalization condition

$$\mathcal{Z} = \sum_{\vec{\sigma}} \exp\left[-\sum_j \lambda_j f_j(\vec{\sigma})\right] \tag{8}$$

Recall that the features $\vec{f}$ include individual activity $\sigma_i$ and every pairwise combination $\sigma_i\sigma_j$. Substituting that set of features in, the Lagrangian multipliers $\lambda_j$ are simply the parameters $h_i$ and $J_{ij}$, and $p(\vec{\sigma})$ is simply the Ising Model. Thus, the Ising Model is not a random physical model we apply to these biological problems, but the maximum entropy model for describing pairwise interactions.

## III. INVERSE ISING

Similar to many modeling problems in biology, the challenge is to, given a set of observable data, estimate the optimal parameters $\{h_i, J_{ij}\}$ for the Ising Model that most closely match the feature expectations [5]. Formally, we seek a set of parameters $\vec{\lambda}$ that minimizes the set of loss functions:

$$L(\lambda_j) = \langle f_j\rangle_{\text{data}} - \langle f_j\rangle_{\text{model}} \tag{9}$$

Note that for simplicity of notation, we will use $\lambda$ and $\vec{f}$ through the rest of this section to denote the parameters and variables in the Ising model. To be clear, the above equation is equivalent to writing

$$\begin{aligned}&\langle\sigma_i\rangle_{\text{data}} - \langle\sigma_i\rangle_{\text{model}} \qquad \forall\, i\\ &\langle\sigma_i\sigma_j\rangle_{\text{data}} - \langle\sigma_i\sigma_j\rangle_{\text{model}} \;\forall\, i \neq j\end{aligned}$$

Suppose the data is comprised of $K$ observed activity patterns, denoted by $\vec{\sigma}_k$ Then, the observed features are simply the average across the observed data

$$\langle f_j\rangle = \frac{1}{K}\sum_{k=1}^{K} f_j(\vec{\sigma}_k) \tag{10}$$

As expected in statistical mechanics, the model feature expectations can be computed from derivatives of the logarithm of the partition function:

$$\langle\sigma_i\rangle = -\frac{\partial \ln \mathcal{Z}}{\partial h_i} \tag{11}$$

$$\langle\sigma_i\sigma_j\rangle = -\frac{\partial \ln \mathcal{Z}}{\partial J_{ij}} \tag{12}$$

Then, we can write the loss function for $j$th feature as

$$\begin{aligned}L(\lambda_j) &= \frac{1}{K}\sum_{k=1}^{K} f_j(\vec{\sigma}_k) + \frac{\partial \ln \mathcal{Z}}{\partial \lambda_j}\\ &= \frac{\partial}{\partial\lambda_j}\frac{1}{K}\sum_{k=1}^{K}\left[\sum_j \lambda_j f_j(\vec{\sigma}_k) + \ln\mathcal{Z}\right]\\ &= \frac{\partial}{\partial\lambda_j}\frac{1}{K}\sum_{k=1}^{K} -\ln\left[\frac{1}{\mathcal{Z}}\exp\left(-\sum_j \lambda_j f_j(\vec{\sigma}_k)\right)\right]\\ &= \frac{\partial}{\partial\lambda_j}\frac{1}{K}\sum_{k=1}^{K} -\ln[p(\vec{\sigma}_k|\lambda)]\\ &= \frac{\partial}{\partial\lambda_j}\Big\langle -\ln[p(\vec{\sigma}|\lambda)]\Big\rangle_{\text{data}}\end{aligned}$$

In other words, minimizing the loss function $L(\lambda_j) \to 0$ is equivalent to minimizing the average log probability of observing the data along the individual parameter $\lambda_j$ [5]. This provides a plausible algorithm for computing the parameters – compute the gradient of the expected log likelihood to alter the parameters and iteratively step towards the minimum. The major challenge here is the computational tractability of accurately sampling the partition function $\mathcal{Z}$ in order to compute the expected values. There are $2^N$ states in $\mathcal{Z}$, and for large $N$, such calculations become infeasible.

## IV. APPROXIMATIONS

Previous implementations of the naive approach suggested in the previous section have proven computationally intractable for $N > 30$ [6], and several approximations and modifications must be made to solve this improve the scaling of this Inverse Ising problem. A wide variety of approximations have been developed by handle neuronal spiking data, which may be applicable to other biological topics such as correlated mtuations. Here we review several of these developments.

### A. Histogram Monte Carlo Sampling

Instead of generating new Monte Carlo samples for each iteration, the algorithm uses the well-known method of Histogram Monte Carlo [7], where previously generated samples for parameters $\lambda$ are used to compute samples with some new parameters $\lambda'$ via

$$p(\vec{\sigma}|\lambda') = \frac{p(\vec{\sigma}|\lambda)\exp\left[(\lambda'-\lambda)\cdot\vec{f}(\vec{\sigma})\right]}{\left\langle\exp\left[(\lambda'-\lambda)\cdot\vec{f}(\vec{\sigma})\right]\right\rangle_\lambda} \tag{13}$$

This more efficient method eliminates the repetitive resampling of the naive approach. In [5], the authors combine this sampling method with a coordinate descent

method, where individual parameter dimensions $\lambda_j$ are explored in sequence instead of calculating the gradient in parameter space. The resulting algorithm is relatively efficient and accurate for network sizes of $N < 200$.

### B. Independent Pair

The independent pair approximation treats every pair of interactions as independent of the rest of the system[8]. Then, we can write the probability distribution of this two-spin system as

$$p(\sigma_i, \sigma_j) = \frac{1}{\mathcal{Z}_{ij}} \exp(h_i\sigma_i + h_j\sigma_j + J_{ij}\sigma_i\sigma_j) \tag{14}$$

To isolate $J_{ij}$, we take advantage of the property of exponentials to write

$$\log\left(\frac{p(+1,+1)}{p(+1,0)p(0,+1)}\right) = \log\left(\frac{e^{h_j+h_j+J_{ij}}}{e^{h_i}e^{h_j}}\right) = J_{ij} \tag{15}$$

Thus, we can extract the coupling constants from the empirical estimations of $p(+1,+1), p(+1,0)$, and $p(0,+1)$.

### C. Discretized Gaussian

A Discretized Gaussian (DG) model forces spins to interact through the covariance matrix of a multi-dimensial gaussian function [9][3]. This assumption makes learning the parameters very efficient, and can easily model systems where the spin can take $> 2$ states, but at the sacrifice of some accuracy. Suppose we allow the spin to take three states: 0, 1, and 2. The DG model assumes that a spin configuration $\vec{\sigma}$ is generated from a hidden $N$-dimensional Gaussian variable $\mathbf{z} \sim \mathcal{N}(0, \Lambda)$ with thresholds $\gamma_i = \{\gamma_{i,1}, \gamma_{i,2}\}$ such that

$$\sigma_i = \begin{cases} 0 & \text{if } z_i < \gamma_{i,1} \\ 1 & \text{if } \gamma_{i,1} < z_i < \gamma_{i,2} \\ 2 & \text{if } z_i > \gamma_{i,2} \end{cases}. \tag{16}$$

Basically the threshold parameters $\gamma_i$ discretizes the domain into multiple states. The covariance matrix $\Lambda ij$, is easily written as

$$\Lambda_{ij} = \mathbb{E}\left[\sigma_i\sigma_j\right] = \sum_{\sigma_i,\sigma_j} \sigma_i\sigma_j p(\sigma_i, \sigma_j), \tag{17}$$

Learning the parameters is far more efficient than the Ising Model, and requires no Monte Carlo sampling. Given the empirical estimate for $p(\sigma_i)$, the thresholds $\gamma_i$ can be obtained by using the definition of the gaussian distribution:

$$\gamma_{i,1} = \Phi^{-1}[p(\sigma_i = 0)] \qquad \gamma_{i,2} = -\Phi^{-1}[p(\sigma_i = 2)] \tag{18}$$

where $\Phi^{-1}(x)$ is the inverse error function.

The matrix $\Lambda$ can be computed by solving Eq. 17 using the empirical estimate for the joint distribution $p(\sigma_i, \sigma_j)$. Note that, since the interaction term $\Lambda_{ij}$ for a single spin pair can be determined independent of the other spins, learning the parameters is very efficient. Thus, the DG model can be viewed as a further approximation of the independent pair model that allows for multiple states, and easier evaluation. Probabilities with this model are computed by numerical integration over a $N$-dimensional multivariate Gaussian:

$$P_{DG}(\vec{\sigma}) = \frac{1}{(2\pi)^{N/2}|\Lambda|^{1/2}} \int_{a_1}^{b_1} \dots \int_{a_N}^{b_N} \exp(-\vec{\sigma}^T\Lambda^{-1}\vec{\sigma}), \tag{19}$$

where $a_i = \gamma_{i,r_i}$ and $b_i = \gamma_{i,r_i+1}$ are the limits of integration. Fast numerical integration of this high-dimensional Gaussian can be accomplished with algorithms from [10][11]. Since the normalization requirement is computed analytically, there is no required summation over a large dimensional space such as in the Ising Model. Algorithms implementing the DG model can learn parameters exactly up to network sizes of $N \approx 1000$.

## V. EXPERIMENT

We test specifically here the relevance of the Independent Pair approximation (Algorithm B) in describing a set of neuronal data, as benchmarked against the Histogram Monte Carlo algorithm (Algorithm A). Previous work has established Algorithm A as an accurate, but expensive, estimator of $J_{ij}$ [5], so if the Independent Pair approximation (a much faster algorithm) can produce similar results, then larger network size analysis becomes computationally tractable. This is particularly important when moving beyond existing neural network recording capabilities, where $N < 200$, to other biological topics such as correlated mutations, where the number of amino acids in a protein can be in the $N \sim 10^3$ range.

The neuronal data is obtained from multi-electrode recordings of the salamander retina [12][13], and consists of 5,700 trials where stimulus (geometric shapes) were shown to the retina. Each neuron response $i$ to the stimulus was denoted with $\sigma_i = 1$ if there were any action potentials in the first 100ms post-stimulus, and $\sigma_i = 0$ otherwise.

For different network sizes, parameters were estimated from this dataset using Algorithm A, henceforth known as the Ising algorithm [14], and Algorithm B, the independent pair model. As a measure of performance,the correlation coefficient between these two sets was computed.

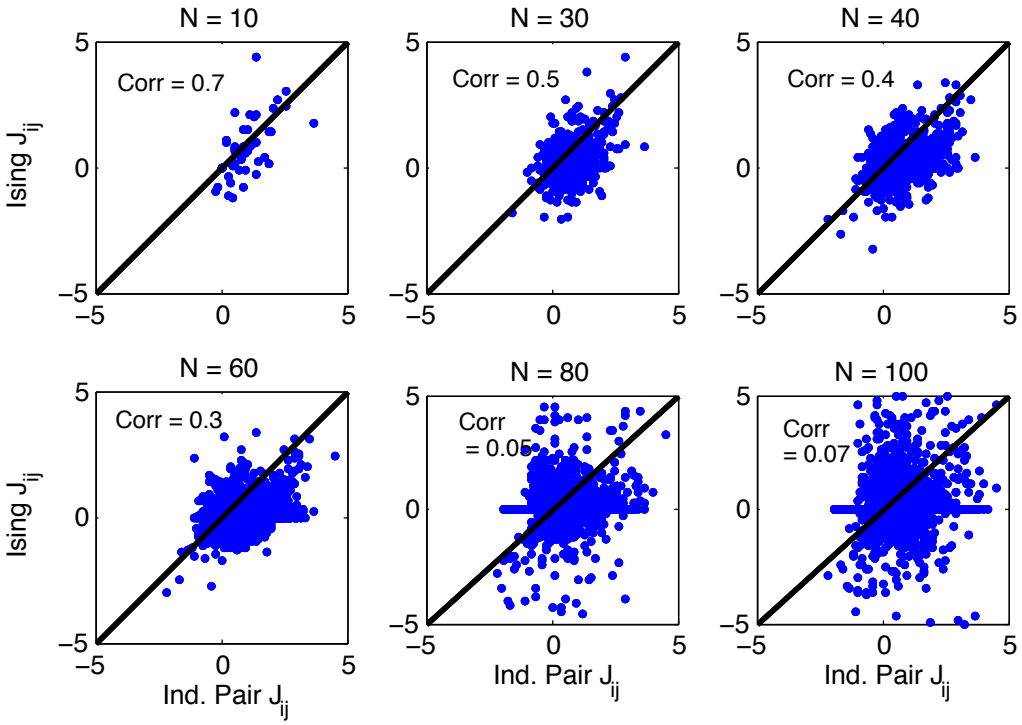


FIG. 1. Comparison between coupling parameters $J_{ij}$ from the Ind. Pair algorithm (x-axis) to those from the Histogram Monte Carlo algorithm (y-axis, labeled as 'Ising'). Black line denotes unity, and the correlation coefficient is shown

As shown in Figure 1, where the coupling constants for the two models are compared, the constants are similar until $N > 40$. For large $N$, the Ising algorithm returns a relatively sparse coupling matrix, while the independent pair model does not. The independent pair assumption breaks down for such large $N$ because the effect of small correlations from the larger system on a particular two-spin pair system begins to accumulate.

The effect of scaling on the accuracy of independent pair approximation is shown in Figure 2, which plots the correlation coefficient between the two. Evidently, the correlation coefficient scales linearly with the network size for this particular dataset.

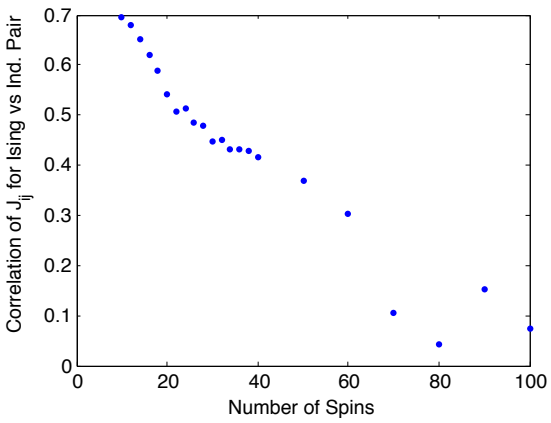


FIG. 2. Correlation coefficient between $J_{ij}$ estimates as a function of network size.

## VI. DISCUSSION

We have reviewed the literature for 'Inverse Ising' algorithms, and used experimental data to implement and test the simplest approximation – the Independent Pair algorithm – against one of the most robust, but expensive, learning algorithms. For this particular dataset, the independent pair approximation captures the coupling constants reasonably well for $N < 40$, suggesting that such techniques can be used for those network sizes.

However, there are several caveats to this analysis. The accuracy of the independent pair algorithm depends on how independent any two pairs are from the rest of the system. In this particular dataset, where the neuronal spiking is tied to a stimulus, and thus more strongly correlated, this approximation is not very accurate for large network sizes. For other biological datasets, however, with even more sparse correlations, this approximation can be valid for large network sizes.

In addition, while our measure here is the correlation coefficient between the coupling constants, a far more accurate measure would be the actual loss function $L_\lambda$ – especially for large network sizes, where the loss function landscape has many local minima, different combinations

of the parameters may produce similar performance. Unfortunately, calculating such a measure was beyond the scope of this project.

Because of these caveats, the experimental portion of this analysis is not a rigorous and generalizable finding. Nevertheless, that the far simpler approximation can reproduce $J_{ij}$ estimates remarkably similar to that of the Histogram Monte Carlo algorithm suggests that the sparsity of correlations in neuronal firing data can be an advantage. In addition, other biological phenomena with sparse correlations (such as correlated mutations) can also be amenable to this approach.

Future work should test these algorithms on other biological datasets, where the underlying characteristics may be very different. Hopefully, other fields can benefit from the advances in the 'Inverse Ising' problem developed for neuroscience, as detailed and experimentally tested here.


[1] E. Schneidman, M. J. Berry, R. Segev, and W. Bialek, Nature **440**, 1007 (2006).
[2] F. Pazos, M. Helmer-Citterich, G. Ausiello, and A. Valencia, J. Mol. Biol. **271**, 511 (1997).
[3] H. Tang, Princeton University Senior Thesis , 1 (2008).
[4] Mathemtical derivations here drawn from [3][5].
[5] T. Broderick, M. Dudik, G. Tkacik, R. E. Schapire, and W. Bialek, arXiv **q-bio.QM** (2007), 0712.2437v2.
[6] E. Schneidman, S. Still, M. J. n. Berry, and W. Bialek, Phys Rev Lett **91**, 238701 (2003).
[7] A. Ferrenberg and R. Swendsen, Phys Rev Lett **63**, 1658 (1989).
[8] Y. Roudi, J. Tyrcha, and J. Hertz, Phys. Rev. E **79**, 051915 (2009).
[9] M. Bethge and P. Berens, NIPS Conference (2007).
[10] D. Cox and N. Wermuth, Biometrika **81**, 403 (1994).
[11] A. Genz, Journal of Computational and Graphical Statistics **1**, 141 (1992).
[12] G. Schwartz, Ph.D. thesis.
[13] R. Segev, J. Goodhouse, J. Puchalla, and M. Berry, Nat. Neurosci. **7**, 1154 (2004).
[14] Code for this learning algorithm was written in [3].