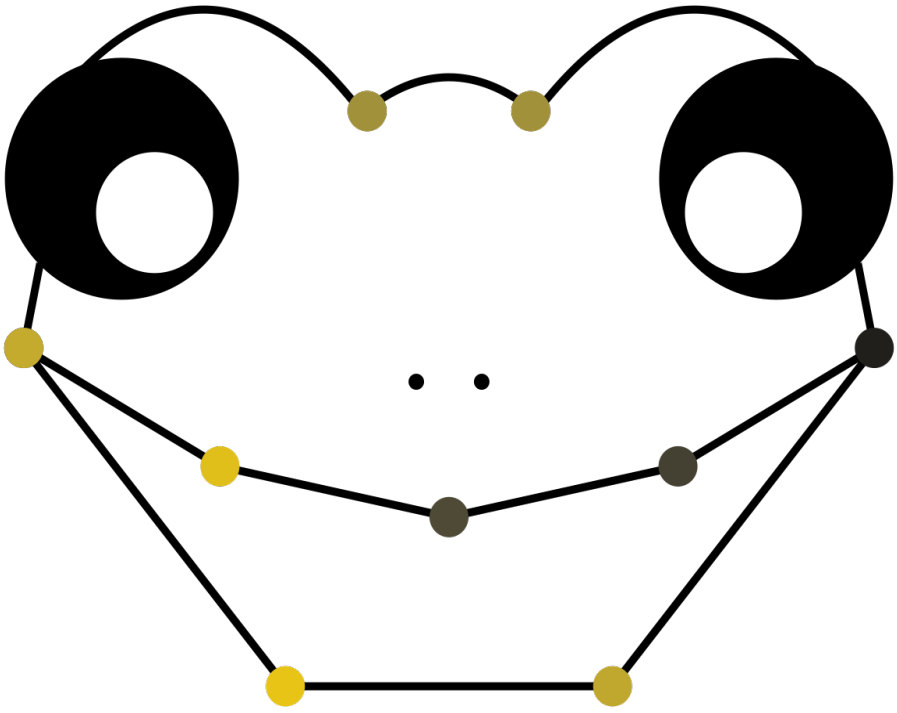
M-LAB

computational physics

Statistical Inference

Inverse Ising Problem: The Salamander's Brain



Johannes Berg
Markus Schmitt

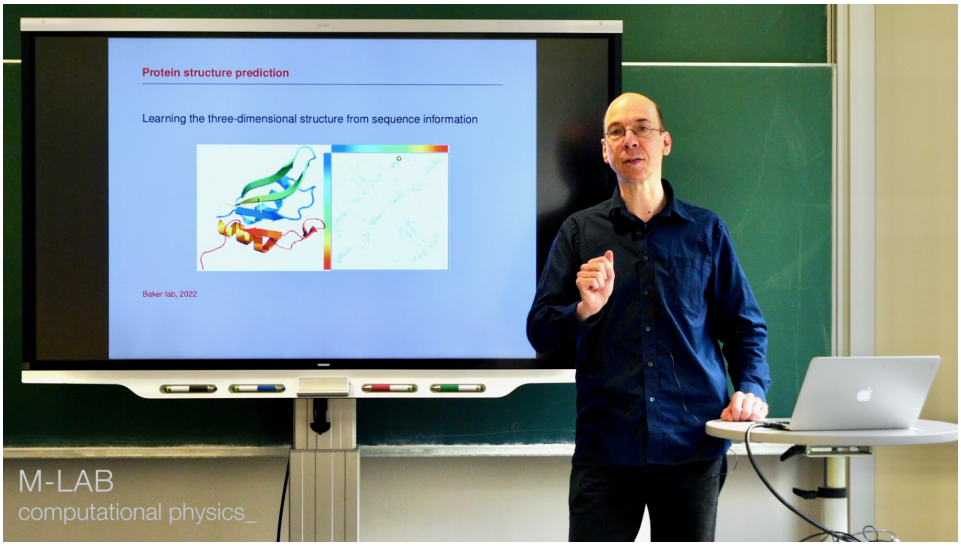
Institute for Theoretical Physics
Cologne

v1.0

Contents

1 Theory

1.1 Teaser video



1.2 Introduction

Statistical physics aims to derive observable quantities from a microscopic system. For instance, the Ising model is defined by interactions between elementary magnets (spins), and the goal is to derive, among others, the magnetisations and correlations of the spins from the parameters of the model (interactions and external fields).

In an inverse statistical problem, this direction is reversed: given observed data, like spin magnetisations and correlations, we aim to infer the underlying model parameters.

In the last two decades a number of applications of the inverse Ising problem have arisen in different fields, ranging from the inference of neural and gene regulatory networks, to the inference of protein structures and the structure of complex quantum phase transitions. The inverse Ising problem has also turned out to be an algorithmically challenging problem (we will see why) and a wide range of different approaches has been developed. See Ref. [1] for a review article.

1.3 Boltzmann machine learning

In the problem that we are going to consider in the following we are given observed data in the form of vectors of binary variables, $\mathbf{s} = (s_1, \ldots, s_N)$ with $s_i = \pm 1$. These can, for example, correspond to Ising spin configurations or represent the brain activity of a salamander. The question we want to answer is what is the probability distribution that best describes the observed data. For

this purpose, we choose a *model*, i.e., a specific parametrized function $P_\theta(\mathbf{s})$ with parameters θ. The objective is then to find the optimal parameters θ^* in the sense that $P_{\theta^*}(\mathbf{s})$ is the best choice to explain the given data.

This problem of statistically inferring a good model for the given observations is today often framed as a machine learning problem. Therefore, we will in the following talk about "learning" the model or a "learning algorithm". In this section, we will discuss the three central components of this approach, namely the choice of a model, the formulation of a learning objective, and a possible learning algorithm.

1.3.1 Ising model

We consider a system of N binary variables (Ising spins) $\{s_i\}, i = 1, \ldots, N$ with $s_i = \pm 1$. These spins are coupled by pairwise interactions and are subject to external magnetic fields.

$$P_{\{h_i, J_{ij}\}}(\mathbf{s}) = \frac{1}{Z} \exp\left[\sum_i h_i s_i + \sum_{i<j} J_{ij} s_i s_j\right] \tag{1}$$

denotes the corresponding Boltzmann equilibrium distribution $P(\{s_i\}) = e^{-H(\{s_i\})}/Z$, where we have set the (inverse) temperature to 1 by subsuming it into the interactions J_{ij} and fields h_i: The interaction J_{ij} hence denotes $\beta = 1/(k_B T)$ times the energy difference associated with parallel and antiparallel spin configurations. Interactions and fields are unknown, and many of them may be zero.

The Hamiltonian

$$H(\{s_i\}) = -\sum_i h_i s_i - \sum_{i<j} J_{ij} s_i s_j \tag{2}$$

specifies the energy of the spin system as a function of the spin variables and the parameters of the Ising model: the local fields and pairwise interactions. The inverse Ising problem is the determination of the couplings J_{ij} and local fields h_i given a set of pairwise correlations between spins and magnetisations.

The factor Z in Eq. (1) denotes the partition sum, i.e., the normalization that guarantees that all probabilities sum up to one,

$$Z = \sum_{\mathbf{s}} \exp\left[\sum_i h_i s_i + \sum_{i<j} J_{ij} s_i s_j\right] . \tag{3}$$

1.3.2 Learning objective

Now that we chose a model for the probability distribution, we have to formalize what we mean by "inferring the optimal parameters". This means, in particular, that we need a quantitative approach to evaluate how "good" a chosen set of parameters θ is.

For this purpose, we assume that there exists a true underlying probability distribution of the data $P_0(\mathbf{s})$ – the ground truth. Our goal is then that $P_{\theta^*}(\mathbf{s})$ is as close as possible to the ground truth. The similarity of probability distributions can be quantified by the Kullback-Leibler divergence, which is for two distributions $P(x)$ and $Q(x)$ given as

$$D_{KL}(P||Q) = \sum_x P(x) \log \frac{P(x)}{Q(x)} \,. \tag{4}$$

This quantity informs us about the similarity of both distributions, because $D_{KL}(P||Q) \geq 0$ and it vanishes if and only if the two distributions are equal. Notice, however, that it is not a distance, because the function is not symmetric under exchange of P and Q.

Based on the Kullback-Leibler divergence, inferring the optimal parameters means to minimize $D_{KL}(P_0||P_\theta)$. Clearly, this objective needs further thought, because we do not know the ground truth $P_0(\mathbf{s})$, meaning that it is not immediately possible for us to evaluate Eq. (4). All that we know is the training data, which we view as a finite sample from the distribution, $\mathbf{s}^{(k)} \sim P_0$, $k = 1, \ldots, M$. Using this finite sample, we can approximate the Kullback-Leibler divergence as

$$\begin{aligned} D_{KL}(P_0||P_\theta) &= \sum_{\mathbf{s}} P_0(\mathbf{s}) \log \frac{P_0(\mathbf{s})}{P_\theta(\mathbf{s})} \\ &\approx \frac{1}{M} \sum_k \log \frac{P_0(\mathbf{s}^{(k)})}{P_\theta(\mathbf{s}^{(k)})} \\ &= -\frac{1}{M} \sum_k \log P_\theta(\mathbf{s}^{(k)}) + \frac{1}{M} \sum_k \log P_0(\mathbf{s}^{(k)}) \,. \end{aligned} \tag{5}$$

In the last expression we deliberately split the logarithm of the ratio into the difference of the logarithms. Thereby, we separate the remaining occurrence of the ground truth $P_0(\mathbf{s})$ to underline that this is a constant contribution to the expression. This additive term is not affected by changes of the parameters θ, and, therefore, the minimization of $D_{KL}(P_0||P_\theta)$ is reduced to the minimization of

$$\mathcal{L}(\theta) = -\frac{1}{M} \sum_k \log P_\theta(\mathbf{s}^{(k)}) \,. \tag{6}$$

This quantity is called the (empirical) negative log-likelihood.

Notice that by discarding the constant term in the last line of Eq. (5) we lost the property of the Kullback-Leibler divergence, that the absolute minimum is zero. This means that the absolute minimum of the negative log-likelihood (6) is unknown and it's bare value is not informative about the distance between model and ground truth. It is nonetheless useful to monitor the negative log-likelihood during the process of the training algorithm discussed in the next section in order to determine whether learning continues and when convergence is achieved.

1.3.3 Learning algorithm

As a last ingredient, we need an algorithm that allows us to minimize the negative log-likelihood. Since $\mathcal{L}(\theta)$ is a differentiable function of the continuous parameters θ, we can use a gradient descent approach to find the optimum. This means that after choosing an initial guess of the parameters $\theta^{(0)}$ we follow an iterative procedure with the update step

$$\theta^{(n+1)} = \theta^{(n)} - \eta \nabla \mathcal{L}(P_\theta) \, . \tag{7}$$

Here, η is the learning rate.

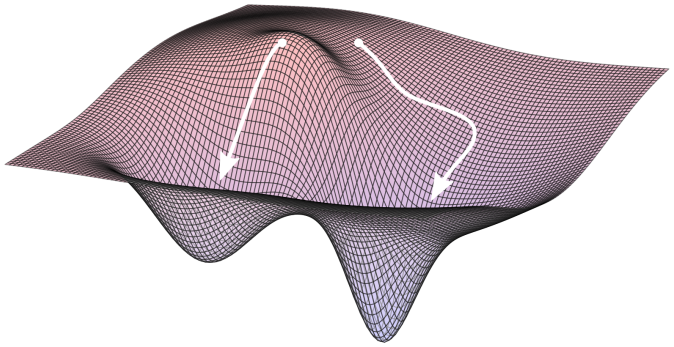


Figure 1 – Schematic of gradient descent optimization in an exemplary cost landscape. The white arrows indicate paths that a gradient descent algorithm would follow during optimization. Notice that in the presence of multiple extrema convergence to the global minimum is not guaranteed and the outcome can depend on the initial condition as indicated by the two exemplary paths.

Plugging the Ising model (1) into the log-likelihood (6), we obtain the prescription

$$\begin{aligned} h_i^{(n+1)} &= h_i^{(n)} + \eta \left(\langle s_i \rangle^D - \langle s_i \rangle_\theta \right) \\ J_{ij}^{(n+1)} &= J_{ij}^{(n)} + \eta \left(\langle s_i s_j \rangle^{\mathrm{D}} - \langle s_i s_j \rangle_\theta \right) \, . \end{aligned} \tag{8}$$

for the parameters of the Ising Hamiltonian. In these expressions, $\langle \cdot \rangle^{\mathrm{D}}$ denotes the mean obtained from the training data D and

$$\langle \cdot \rangle_\theta = \sum_{\mathbf{s}} P_\theta(\mathbf{s}) \big(\, \cdot \, \big) \tag{9}$$

denotes the expectation value with respect to P_θ. This particular way to find the optimal parameters θ model parameters is called Boltzmann machine learning, because it relies on the fact that the model function P_θ takes the form of a Boltzmann factor as in Eq. (1).

In order to calculate the expectation values $\langle s_i \rangle$ and $\langle s_i s_j \rangle$ on the right-hand side of these equations, one needs to perform thermal averages over all 2^N configurations, which is generally infeasible for all but the smallest system sizes.

To proceed, we will set up a Monte-Carlo Markov chain (MCMC, described in the following section) to estimate the expectation values $\langle s_i \rangle$ and $\langle s_i s_j \rangle$ under the Boltzmann equilibrium distribution (1) for a given set of model parameters. These then enter the Boltzmann machine learning algorithm (8).

1.4 Non-equilibrium reconstruction

The Boltzmann equilibrium distribution in Eq. (1) relies on the assumption that the spin degrees of freedom of the model are coupled symmetrically, i.e., $J_{ij} = J_{ji}$. This assumption is, however, not necessarily a good match for the given data. In this project you will analyze the brain activity of a Salamander, where the the spin variables correspond to the activity of individual neurons. The connections between these neurons are directed and therefore their mutual influence is not necessarily identical.

The stochastic dynamics of systems with asymmetric couplings breaks *detailed balance*. Therefore, the steady states reached at late times do not necessarily correspond to the Boltzmann distribution. One way to incorporate the breaking of detailed balance in the inverse Ising problem is to consider a stochastic *Glauber dynamics*. For this purpose, we assume discrete time steps and a probabilistic update rule for the new spin configuration conditioned on the current configuration,

$$P\big(\mathbf{s}(t+1)|\mathbf{s}(t)\big) = \frac{\exp\left[\sum_i s_i(t+1)\Theta_i(t)\right]}{\prod_i 2\cosh\big(\Theta_i(t)\big)} \,. \tag{10}$$

Here, we introduced the effective local field

$$\Theta_i(t) = \sum_j J_{ij} s_j(t) + h_i \,. \tag{11}$$

Notice that the joint probability (10) factorizes and one can write

$$P\big(\mathbf{s}(t+1)|\mathbf{s}(t)\big) = \prod_i \frac{\exp\left[s_i(t+1)\Theta_i(t)\right]}{2\cosh\big(\Theta_i(t)\big)} \equiv \prod_i p_i\big(s_i(t+1)|\mathbf{s}(t)\big) \tag{12}$$

with normalized marginal distributions for the individual spins, $p_i\big(s_i(t+1)|\mathbf{s}(t)\big)$. This means that in a simulation the update step can be implemented by individually drawing the new configuration for each spin from the respective Bernoulli distribution.

1.4.1 Learning from time series data

The objective is now to infer couplings J_{ij} and local fields h_i such that the resulting Glauber dynamics matches given time series data $D = \{\mathbf{s}(t=1), \ldots, \mathbf{s}(t=$

$M)\}$. For this purpose we consider a modified negative log-likelihood

$$\begin{aligned}\mathcal{L}(\theta) &= \frac{1}{M}\sum_{t=1}^{M-1} \ln P\big(\mathbf{s}(t+1)|\mathbf{s}(t)\big) \\ &= \frac{1}{M}\sum_{t=1}^{M-1}\sum_{i}\left[s_i(t+1)\Theta_i(t) - \ln 2\cosh\left(\Theta_i(t)\right)\right] . \end{aligned} \tag{13}$$

On this basis, the optimal parameters can again be obtained using the gradient descent approach. Notice that the non-equilibrium negative log-likelihood (13) is *tractable*, because it can be evaluated in MN^2 computational steps.

1.5 Markov chain Monte Carlo

For the Boltzmann machine learning algorithm we need to be able to evaluate expectation values of the form

$$\langle O\rangle_\theta = \sum_{\mathbf{s}} P_\theta(\mathbf{s})O(\mathbf{s}) , \tag{14}$$

where $O(\mathbf{s})$ is some function of the microscopic configuration $\mathbf{s}$. Assume that we have a means to obtain a finite sample $\mathcal{S}_M = \{\mathbf{s}_1, \ldots, \mathbf{s}_M\}$ such that in the limit of large N the relative frequency of a configuration $\mathbf{s}$ in $\mathcal{S}_M$ is proportional to its probability $p_\beta(\mathbf{s})$. Then, according to the law of large numbers, the sample mean converges to the expectation value with increasing sample size,

$$\langle\!\langle O\rangle\!\rangle_N \equiv \frac{1}{N}\sum_{\mathbf{s}\in\mathcal{S}_M} O(\mathbf{s}) \overset{M\to\infty}{\longrightarrow} \langle O\rangle_\theta . \tag{15}$$

1.5.1 Metropolis-Hastings algorithm

The idea of the Metropolis-Hastings algorithm is to realize a Markov process whose stationary distribution is the probability distribution $\pi(\mathbf{s})$ that we are interested in. Once the process reached the stationary state, the states $\mathbf{s}_i$ visited can be used as a sample $\mathcal{S}_M$ and therefore to estimate expectation values.

In a Markov process the system is described by a state $\mathbf{s}$ and in each time step it transitions into a new state $\mathbf{s}'$ according to a transition probability $p_{MC}(\mathbf{s}\to\mathbf{s}') \equiv p_{MC}(\mathbf{s}'|\mathbf{s})$ that only depends on the current state of the system. In the Metropolis-Hastings algorithm the Markov transition probabilities are constructed in two steps, given the current state $\mathbf{s}$. First, a new configuration $\mathbf{s}'$ is proposed following a proposal probability $p_P(\mathbf{s}'|\mathbf{s})$ that can be conditioned on the current state. Given the proposed configuration $\mathbf{s}'$ the update is accepted with a probability

$$p_A(\mathbf{s},\mathbf{s}') = \min\left(1, \frac{\pi(\mathbf{s}')}{\pi(\mathbf{s})}\frac{p_P(\mathbf{s}|\mathbf{s}')}{p_P(\mathbf{s}'|\mathbf{s})}\right) \tag{16}$$

Hence, the probability to move from state s to state s′ in the resulting Markov process is

$$p_{MC}(\mathbf{s} \to \mathbf{s}') = p_P(\mathbf{s}'|\mathbf{s}) p_A(\mathbf{s}, \mathbf{s}') \ . \tag{17}$$

This form of the transition probability is chosen to satisfy the *detailed balance* condition

$$\pi(\mathbf{s}) p_{MC}(\mathbf{s} \to \mathbf{s}') = \pi(\mathbf{s}') p_{MC}(\mathbf{s}' \to \mathbf{s}) \ , \tag{18}$$

which is a sufficient condition for the Markov process to have $\pi(\mathbf{s})$ as stationary distribution. Uniqueness of the stationary distribution is guaranteed if the process is *ergodic*, i.e., if it is possible for the process to go from any state s to any other state s′ in a finite number of steps. Clearly, the latter condition has to be kept in mind when designing the proposal probability $p_P(\mathbf{s}'|\mathbf{s})$.

For Ising spin systems single spin flip updates are often sufficient. This means that from the given configuration s one flips the sign of a single degree of freedom, $s_i' = -s_i$, and the proposal probability $p_P(\mathbf{s}'|\mathbf{s})$ is a uniform distribution over all configurations s′ that differ from s by a single spin flip. To reduce the cost of estimating expectation values and to mitigate the effect of autocorrelation discussed in the following section, one typically performs *sweeps* of local updates between subsequent configurations that are added to the sample $\mathcal{S}_M^{\mathrm{MC}}$. One sweep consists of one Markov update step per degree of freedom, which means that the configuration obtained after the update sweep can globally differ from the initial configuration.

1.6 The Salamander's brain activity

We now turn to an empirical dataset, a neural recording taken from the retina of a salamander. The retina is an extension of the brain. It consists of neural tissue, which has been isolated and placed on a 2D square array of electrodes at a distance of about $60\mu m$. In this case, there are 160 electrodes connected to the neurons (there will be some crosstalk between electrodes).

The measurements of neural signals are used to specify the state of different neurons (active or inactive, $s_i(t) = 1$ or -1) at different times while a movie is projected on the retina ($19s$ of movie are repeated 297 times). Specifically, neural spikes are binned in intervals of $20ms$, and a neural spike during an interval leads to the neuron being labelled 'active' in that interval.

If two cells tend to fire simultaneously, they will frequently exhibit 1 in the same bin, corresponding to pairwise spin correlations.

The data is available at `https://research-explorer.app.ist.ac.at/download/5562/5623/IST-2017-61-v1%2B2_bint_fishmovie32_100.zip` in the form of a matrix whose rows correspond to different electrodes and columns describe the different time bins.

2 Preparation

2.1 Theory

While you familiarize yourself with the theoretical background, find answers to the following questions:

Q1 How is the Boltzmann machine learning algorithm (8) obtained from the objective to maximize the log-likelihood (6)?

Q2 The model (1) involves the partition sum Z with an exponential number of terms. Why does this exponential sum not affect the efficiency of the learning algorithm?

Q3 Derive the derivatives of the non-equilibrium likelihood (13), which you need to implement the corresponding gradient descent optimization.

3 Tasks

3.1 Warm-up: learning thermal distributions

Clearly, there are several problems nested within each other, and we proceed step-by-step. To begin with, pick a small system size, say $N = 5$.

1. Generate fields and interactions that you will later infer ("target parameters"). Generally, they can be anything, but a vector of $(0, 1)$-Gaussian distributed numbers for fields and a symmetric matrix with zero diagonal of $(0, 1/N)$-Gaussian distributed numbers is a good starting point.

2. Set up a MCMC algorithm based on flipping individual spins. Make sure you compute the change in the energy (2) efficiently, because this will be what the algorithm will spend most of its time doing[1].

3. After running the chain for a sufficient number of steps to reach equilibrium, record the average values of spins s_i and spin pairs $s_i s_j$ to compute magnetisations and correlations[2].

4. Now suppose we do not know what the couplings and fields are, but need to infer them from the magnetisations and correlations you just computed. Set the initial couplings J_{ij}^0 and fields h_i^0 to some (randomly chosen, see above) values, and change them according to (8) until convergence. Monitor the negative log-likelihood (6) during the training procedure. Is the

[1] Hint: it should take only $O(N)$ steps.

[2] To figure out what sufficient means here, try halving that number of steps. If that changes the outcome significantly the number of steps was too small. If ever you are unsure how to pick some quantity in your work or your life, remember this principle. You are welcome.

value being reduced systematically? How does this depend on the learning rate?

5. Check how well the inference works by plotting the inferred model parameters against the target parameters. How does the quality of the inference depend on the number of training data?

3.2 The brain of the salamander

3.2.1 Inferring an equilibrium model

Use your experience with Boltzmann machines to explore this data

1. Calculate the "magnetisations"' and "two-spin correlations" of the data
2. Infer couplings and fields which describe this data
3. To probe how well your model describes the data, pretend for a moment you are missing some individual entries of the data. How well does the distribution of that spin *conditioned on the others* describe the missing data?
4. To further test the model, calculate the correlations between triplets of spins both in the data and from your model and plot them against each other.
5. Devise other tests and try them out!

3.2.2 Inferring a non-equilibrium model

Now, we turn to non-equilibrium reconstruction as discussed in Section 1.4. We again start with the reconstruction of a known model and subsequently turn to the Salamander's brain activity.

1. Generate a set of couplings and fields that you will later infer. Implement the Glauber dynamics and generate a time series as training data.
2. Infer the couplings and fields used to generate the training data. How accurately are the parameters recovered?
3. Infer symmetric couplings $J_{ij} = J_{ji}$ and fields that best describe the Salamander's brain activity. How well does the inferred model describe temporal correlations of the data?
4. Infer couplings J_{ij} that are not necessarily symmetric and fields that best describe the Salamander's brain activity. By comparing the resulting likelihoods, does the model with asymmetric couplings lead to a better description of the data?

References

[1] H. Nguyen, R. Zecchina, and J. Berg, Advances in Physics **66**, 197 (2017).