Inverse Statistical Mechanics Notes

Jesse Young Lin

February 13, 2025

Contents

1 Bayesian statistics **1**
1.1 The question of Bayesian inference and maximum likelihood estimation 1
1.2 Maximum entropy inference 1

2 Gradient descent and machine learning **3**
2.1 Regression on moments is maximum likelihood estimation 3
2.2 Assignment 4

1 Bayesian statistics

1.1 The question of Bayesian inference and maximum likelihood estimation

The core question is, given high-dimensional data, i.e., $\{x_i\}$ with each $x_i \in \mathbb{R}^d$ and d very large, how does one fit a probability distribution $\rho(\{x_i\})$ over the data set which "respects" its statistics?

In general, this is the question answered by **Bayesian inference**. A common paradigm is **maximum likelihood estimation**. Here, one considers a probablity distribution ρ which is parameterized by variables $\{\theta_j\}$ with allowable values in the sets $\{\Theta_j\}$. Then, the **likelihood function** $\rho(\{x_i\} \mid \{\theta_j\})$ is defined over the data. Its interpretation is as a conditional probability: what is the probability of observing my dataset $\{x_i\}$ given specific values of the unknown fit parameters $\{\theta_j\}$? Maximum likelihood estimation is a way of estimating the $\{\theta_j\}$ by assuming that they take on the values which maximize the likelihood, i.e.,

$$\theta_j = \arg\max_{\theta_j \in \Theta_j} \rho(\{x_i\} \mid \{\theta_j\}).$$

1.2 Maximum entropy inference

We are using a specific technique for doing maximum likelihood estimatino called **maximum entropy inference**. Essentially, we posit a maximum entropy assumption when defining our model distribution ρ, which otherwise must be arbitrarily specified.

For concreteness, consider the Ising model with random variables S_i. If we denote a lattice configuration with size $N \times N$ as a vector[1]

$$\Lambda = (s_1, s_2, \ldots, s_{N\times N})$$

i.e., we specify the values of each of the random variables as s_i, then a dataset is a set of lattice configurations $\{\Lambda^j\}_{j=1}^M$. Therefore, we use subscript index to denote the lattice position and superscript index to denote which sample in the dataset we refer to.

Given such a dataset $\{\Lambda^j\}_{j=1}^M$, we want to fit a probability model ρ which reproduces the first and second moments of the dataset.[2] In other words, we want

$$\begin{aligned} \langle S_i \rangle_\rho &= \frac{1}{M}\sum_{k=1}^M s_i^k \\ \langle S_i S_j \rangle_\rho &= \frac{1}{M}\sum_{k=1}^M s_i^k s_j^k, \end{aligned} \tag{1}$$

where $\langle\cdot\rangle_\rho$ denotes the expectation value with respect to the distribution ρ.

It turns out that matching the moments via (1) does not uniquely specify the probability distribution, but if we also add the constraint that the entropy is maximized, the unique probability distribution that satisfies (1) with maximal entropy is

$$\rho(\Lambda) = Z^{-1} e^{-\sum_{ij} J_{ij} s_i s_j - \sum_k h_k s_k}. \tag{2}$$

Defining the **partition function** as[3]

$$Z = \sum_{\text{all possible } \Lambda} \rho(\Lambda),$$

we note

$$\begin{aligned} \frac{\mathrm{d}\log Z}{\mathrm{d}J_{ij}} &= \langle S_i S_j \rangle_\rho \\ \frac{\mathrm{d}\log Z}{\mathrm{d}h_k} &= \langle S_k \rangle_\rho . \end{aligned}$$

[1] Physically we imagine the $N \times N$ lattice like a matrix, but mathematically for convenience we denote it as a vector of length N^2. In computer science this is sometimes called "flattening" a matrix, and can be convenient when programming high-performance simulations because computer memory is linear and therefore matrices are all stored in flattened form. There are many ways to flatten a matrix, for example: for linear index k, doing division gives integers q, r such that

$$k = qN + r.$$

with $0 \le r < q$. Then the Cartesian index (i.e., x- and y-coordinates on the lattice) is (q, r).

[2] Note that due to the symmetries of the Ising model, namely its spatial homogeneity (no spatial position is special) and isotropy (no spatial direction is special), many of these moments are actually the same. For example, $\langle S_i \rangle$ is actually independent of the lattice position i: this is the physical statement that the net magnetization is uniform across the lattice.

[3] I prefer writing the summation as over all possible configurations of the lattice Λ. For the Ising model, which is a 2D lattice of binary spins, this is a summation over a Cartesian product $\Lambda \in \{\pm 1\}^{N^2}$.

Therefore the condition (1) is equivalent to

$$\frac{\mathrm{d}\log Z}{\mathrm{d}J_{ij}} = \frac{1}{M}\sum_{k=1}^{M} s_i^k$$
$$\frac{\mathrm{d}\log Z}{\mathrm{d}h_k} = \frac{1}{M}\sum_{k=1}^{M} s_i^k s_j^k.$$

2 Gradient descent and machine learning

Essentially all the success of machine learning is based on the fact that the gradient descent algorithm is very efficient to compute. Gradient descent amounts to defining a cost function $\mathcal{L}(\theta)$ over a parameter θ, and then noting that if

$$\theta \leftarrow \theta - \alpha\frac{\partial\mathcal{L}}{\partial\theta}. \tag{3}$$

where $\alpha > 0$ is the dimensionless **step size**, then θ will converge to a value where $\frac{\partial\mathcal{L}}{\partial\theta} = 0$. This corresponds to a local extremum, and the choice of minus sign in the algorithm (3) indicates that this should be a local minimum of $\mathcal{L}$. If $\mathcal{L}$ fulfills other conditions (for example, if it were a globally convex function of θ), then this will also be a global minimum, but in general it does not fulfill these conditions and we are satisfied if we can even find local minima.

2.1 Regression on moments is maximum likelihood estimation

Writing the distribution (2) as a likelihood function,

$$\rho(\{s_i\} \mid \{J_{ij}\},\{h_k\}) = Z^{-1}(\{J_{ij}\},\{h_k\})e^{-\sum_{ij} J_{ij}s_i s_j - \sum_k h_k s_k}$$

one can calculate

$$\frac{\partial\log\rho(\{s_i\} \mid \{J_{ij}\},\{h_k\})}{\partial J_{ij}} = \langle S_i S_j\rangle_\rho - s_i s_j$$
$$\frac{\partial\log\rho(\{s_i\} \mid \{J_{ij}\},\{h_k\})}{\partial h_k} = \langle S_k\rangle_\rho - s_k$$

which we saw on the board last time. Therefore, we can implement the following algorithm

$$\begin{aligned} J_{ij} &\leftarrow J_{ij} + \alpha(\langle S_i S_j\rangle_\rho - s_i s_j) \\ h_k &\leftarrow h_k + \alpha(\langle S_k\rangle_\rho - s_k) \end{aligned} \tag{4}$$

which is equivalent to gradient ascent on the function $\log\rho$, i.e., maximum likelihood estimation.

2.2 Assignment

You will create a dataset $\{\Lambda^k\}$ of binary images (it can even be a single image). Then, code an algorithm which implements (4) for the Ising model, starting from randomly initialized parameters $\{J_{ij}, h_k\}$.

Recall, for the Ising model that J_{ij} is zero unless sites i and j are adjacent on the lattice, and if so then J_{ij} equals a single constant $\frac{J}{N}$. The code we are using does not have the term that depends on h_k. You should add it into the code, with the same assumption that h_k is uniform across the lattice with $h_k = \frac{h}{N^2}$. (Note: if you work out the sums in (2) you will see where the factors of N come from. This is not that important because you are fitting the values of h and J anyway.)

Hints:

- The easiest dataset you could consider is just to pick a value of J and use the existing code to generate images for your Λ^k. Then when you do maximum likelihood estimation, you should converge on the original value of J you chose. This is bona fide "inverse statistical mechanics".

- Alternatively, you can find your own images from the internet. You may need an image processing library like `https://pypi.org/project/pillow/`. Then, one needs to crop and subsample your images down to the resolution $N \times N$ of the lattice, and also binarize them so they consist of pixels only of brightness value 0 or 1 (corresponding to the up and down spins of the Ising lattice).

- If you are using a dataset with more than a single image, it may be worth implementing the algorithm as follows:

$$J_{ij} \leftarrow J_{ij} + \alpha\left(\langle S_i S_j\rangle_\rho - \frac{1}{M}\sum_k s_i^k s_j^k\right)$$
$$h_k \leftarrow h_k + \alpha\left(\langle S_k\rangle_\rho - \frac{1}{M}\sum_\ell s_k^\ell\right).$$

  This produces statistically equivalent results as (4) assuming all your data are "independent". As we discussed, it is not always straightforward to determine when real-world data should be considered independent from each other. Note that playing around with the value of the step size α may be needed for best results.