Estimation

Max Welling

California Institute of Technology 136-93
Pasadena, CA 91125
welling@vision.caltech.edu

1 Preliminaries

Let $\mathbf{x}$ denote a random variable and $p(\mathbf{x})$ its probability density. $\mathbf{x}$ may be multidimensional and continuous or discrete. In the discrete case $p(\mathbf{x} = \mathbf{s}_i)$ is the probability that $\mathbf{x}$ assumes the value $\mathbf{s}_i$. Obviously, all probabilities have to sum to up to one,

$$\sum_{\mathbf{x}} p(\mathbf{x}) = 1. \tag{1}$$

In the continuous case, $p(\mathbf{x} = \mathbf{s})\Delta\mathbf{x}$ is the probability of finding the vector $\mathbf{x}$ in the range $[\mathbf{s}, \mathbf{s} + \Delta\mathbf{x}]$. We also have the constraint,

$$\int d\mathbf{x}\ p(\mathbf{x}) = 1 \tag{2}$$

Important characteristics of a probability density are its mean and its (co)variance, defined by,

$$\boldsymbol{\mu} = \mathbf{E}[\mathbf{x}] = \int d\mathbf{x}\ p(\mathbf{x})\ \mathbf{x} \tag{3}$$

$$\boldsymbol{\Sigma} = \mathbf{E}[(\mathbf{x} - \boldsymbol{\mu})(\mathbf{x} - \boldsymbol{\mu})^T] = \int d\mathbf{x}\ p(\mathbf{x})\ (\mathbf{x} - \boldsymbol{\mu})(\mathbf{x} - \boldsymbol{\mu})^T \tag{4}$$

The mean is the average value of the random variable, while the variance (diagonal terms) is a measure for the spread around the mean. The off-diagonal terms of the covariance matrix compute the (second order) dependencies between the different entries of $\mathbf{x}$. In general higher order moments are needed to characterize the probability density function (PDF). Indeed usually infinitely many are needed, just like a Taylor series approximates a function. For a normal distribution (Gaussian PDF), these two moments completely specify the PDF,

$$\mathcal{G}_{\mathbf{x}}[\boldsymbol{\mu}, \boldsymbol{\Sigma}] = \frac{1}{(2\pi)^{\frac{D}{2}}\sqrt{\det[\boldsymbol{\Sigma}]}} e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})} \tag{5}$$

If two random variables are independent, information about one does not reveal information about the other. Mathematically this is expressed as,

$$p(\mathbf{x}, \mathbf{y}) = p(\mathbf{x})\ p(\mathbf{y}) \tag{6}$$

and in particular,

$$\mathbf{E}[\mathbf{xy}] = \mathbf{E}[\mathbf{x}]\ \mathbf{E}[\mathbf{y}] \tag{7}$$

2 Supervised versus unsupervised Learning

The main topic of this class will be to train probabilistic models from observed data sequences. This can be divided into two categories; supervised and unsupervised learning. Unsupervised learning takes a set of input samples and fits a probabilistic model. Supervised learning uses input samples and target samples, provided by a "teacher", and learns a mapping from input to output, under a probabilistic model. Let's assume we observe N independent samples $\mathbf{x}^N = \{\mathbf{x}_1, ..., \mathbf{x}_N\}$, for unsupervised learning; and N independent input samples $\mathbf{x}^N$ and N independent target samples $\mathbf{t}^N = \{\mathbf{t}_1, ..., \mathbf{t}_N\}$ for supervised learning. We assume that we have a parametrized model $p(\mathbf{x}|\boldsymbol{\theta})$ for unsupervised learning and $p(\mathbf{t}|\mathbf{x}, \boldsymbol{\theta})$ for supervised learning. The main question we want to answer is, what is the optimal set of parameters $\boldsymbol{\theta}$ that can explain the data.

We would like to mention here that the most powerful models have *unobserved* random variables in their model. In that case, the probabilistic model can be written as follows,

$$p(\mathbf{x}|\boldsymbol{\theta}) = \int d\mathbf{y}\ p(\mathbf{x}, \mathbf{y}|\boldsymbol{\theta}) \tag{8}$$

$$p(\mathbf{t}|\mathbf{x}, \boldsymbol{\theta}) = \int d\mathbf{y}\ p(\mathbf{t}, \mathbf{y}|\mathbf{x}, \boldsymbol{\theta}), \tag{9}$$

where $\mathbf{y}$ is now the hidden (or latent or missing) variable. In this course all models will be of this sort!

3 Bayes Rule

The single most important equation that we will use is Bayes rule,

$$p(\mathbf{x},\mathbf{y}) = p(\mathbf{x}|\mathbf{y})\ p(\mathbf{y}) \tag{10}$$
$$p(\mathbf{x},\mathbf{y}) = p(\mathbf{y}|\mathbf{x})\ p(\mathbf{x}) \tag{11}$$

In words, the probability of the events $\mathbf{x}$ and $\mathbf{y}$ happening together, is equal to the probability of $\mathbf{x}$ happening, times the probability of $\mathbf{y}$ happening, given that $\mathbf{x}$ happened. Combining the above equations we find,

$$p(\mathbf{y}|\mathbf{x}) = \frac{p(\mathbf{x}|\mathbf{y})\ p(\mathbf{y})}{p(\mathbf{x})} \tag{12}$$

and the same where the roles of $\mathbf{x}$ and $\mathbf{y}$ are interchanged. $p(\mathbf{x})$ can be viewed as a normalization constant, i.e.

$$p(\mathbf{x}) = \int d\mathbf{y}\ p(\mathbf{x}|\mathbf{y})\ p(\mathbf{y}) \tag{13}$$

4 Bayesian estimation

A Bayesian would typically condider $\boldsymbol{\theta}$ as a random variable. He would probably have some prior knowledge about the distribution which he formalizes in a prior distribution $p(\boldsymbol{\theta})$. For instance, he could express his intuition that the distribution over $\boldsymbol{\theta}$ should be smooth, and that the values of $\boldsymbol{\theta}$ should not become unreasonably large. For instance, he could use a Gaussian prior, $p(\boldsymbol{\theta}) = \mathcal{G}_{\boldsymbol{\theta}}[0, \alpha^2\mathbf{I}]$, where the parameter α is called a hyperparameter, and could either be fixed or again be treated as a random variable (like $\boldsymbol{\theta}$). Next, the Bayesian would want to know how his prior knowledge of the random variable $\boldsymbol{\theta}$ changes in the light of the new observations $\mathbf{d}$. The data will consist of input samples $\mathbf{x}^N$ for unsupervised learning and pairs of input and target data $\{\mathbf{x}^N, \mathbf{t}^N\}$ for supervised learning. To inquire this, he calculates the posterior distribution,

$$p(\boldsymbol{\theta}|\mathbf{d}) = \frac{p(\mathbf{d}|\boldsymbol{\theta})\ p(\boldsymbol{\theta})}{p(\mathbf{d})} \tag{14}$$

The data will move the modes of the distributions to the most probable values of $\boldsymbol{\theta}$ and also determine a spread around those values, expressing how well the data actually determine $\boldsymbol{\theta}$. Notice, that we need to specify the *likelihood* $p(\mathbf{d}|\boldsymbol{\theta})$, while $p(\mathbf{d})$ acts as a normalization constant, which is hard to calculate because we need to integrate over all possible values of $\boldsymbol{\theta}$,

$$p(\mathbf{d}) = \int d\boldsymbol{\theta}\ p(\mathbf{d}|\boldsymbol{\theta})\ p(\boldsymbol{\theta}). \tag{15}$$

This normalization constant is sometimes called "evidence" or "marginal likelihood". For definiteness let's look at supervised learning and use the following likelihood, $p(\mathbf{t}|\mathbf{x},\boldsymbol{\theta}) = \mathcal{G}_{\mathbf{t}}[f(\mathbf{x},\boldsymbol{\theta}), \beta^2\mathbf{I}]$ where β represents the noise variance around the deterministic mapping $f(\mathbf{x},\boldsymbol{\theta})$. The supervised Bayesian would want to calculate the probability over target values, given the input datum and the previous data.

$$p(\mathbf{t}|\mathbf{x},\mathbf{d}) = \int d\boldsymbol{\theta}\ p(\mathbf{t}|\mathbf{x},\boldsymbol{\theta})\ p(\boldsymbol{\theta}|\mathbf{d}) \tag{16}$$

Similarly, an unsupervised Bayesian would want to be able to compute the probability of a new data point $\mathbf{x}$, given the data, i.e.

$$p(\mathbf{x}|\mathbf{d}) = \int d\boldsymbol{\theta}\ p(\mathbf{x}|\boldsymbol{\theta})\ p(\boldsymbol{\theta}|\mathbf{d}) \tag{17}$$

Notice that we never actually estimate or choose any value for $\boldsymbol{\theta}$. Instead, we determine the posterior density over all values for $\boldsymbol{\theta}$ and use it to integrate over all possible values of $\boldsymbol{\theta}$. The downside is that these integrals are usually very difficult, and only approximately computable using time consuming sampling techniques.

In our example, if we linearize the deterministic mapping $f(\mathbf{x}, \boldsymbol{\theta})$ around the 'maximum a posteriori' (MAP) value of $\boldsymbol{\theta}$ (the value which maximizes $p(\boldsymbol{\theta}|\mathbf{d})$), i.e.

$$f(\mathbf{x}, \boldsymbol{\theta}) = f_0(\mathbf{x}, \boldsymbol{\theta}_{MAP}) + \mathbf{f}_1(\mathbf{x}, \boldsymbol{\theta}_{MAP}) \times (\boldsymbol{\theta} - \boldsymbol{\theta}_{MAP}), \tag{18}$$

then we can actually perform the intergral in (16), resulting in a Gaussian $\mathcal{G}_{\mathbf{t}}[f_0(\mathbf{x}, \boldsymbol{\theta}_{MAP}), \sigma(\mathbf{x}, \boldsymbol{\theta}_{MAP})]$. The average value of this mapping (which is the most likely value for a Gaussian) is determined by taking the deterministic map using the parameters at their MAP-value. The variance of the Gaussian around its mean gives us an indication of our confidence in the estimate of the mean. In our example this can be worked out with the result,

$$\sigma^2(\mathbf{x}, \boldsymbol{\theta}) = \beta^2 + \mathbf{f}_1^T(\mathbf{x}, \boldsymbol{\theta}_{MAP}) \mathbf{A}^{-1} \mathbf{f}_1(\mathbf{x}, \boldsymbol{\theta}_{MAP}) \tag{19}$$

$$\mathbf{A} = \frac{1}{\alpha^2}\mathbf{I} + \frac{1}{\beta^2} \sum_{n=1}^{N} \mathbf{f}_1(\mathbf{x}_n, \boldsymbol{\theta}_{MAP}) \mathbf{f}_1(\mathbf{x}_n, \boldsymbol{\theta}_{MAP})^T + (f_0(\mathbf{x}_n, \boldsymbol{\theta}_{MAP}) - \mathbf{t}_n) \nabla \mathbf{f}_1^T(\mathbf{x}_n, \boldsymbol{\theta}_{MAP}) \tag{20}$$

where ∇ denote taking derivative with respect to $\boldsymbol{\theta}$. More important than this explicit form is that we notice that the mean and variance depend on the data through the value of $\boldsymbol{\theta}_{MAP}$, and the variance also through the matrix $\mathbf{A}$. Both are also functions of $\mathbf{x}$, the input value of the new datapoint. It is important to realize that the variance contains two contributions. One from the noise which is added to the deterministic mapping $f(\mathbf{x}, \boldsymbol{\theta})$. This terms is independent of the data. The other, data dependent term however, reflects our confidence in the value of $\boldsymbol{\theta}_{MAP}$. If the data cannot determine $\boldsymbol{\theta}_{MAP}$ accurately for certain values of $\mathbf{x}$ this term dominates the total variance. If there are enough data in a certain region of $\mathbf{x}$ space, than β^2 will give the main contribution (see demo-Bayes).

5 MAP Estimation

Because the integrals in the Bayesian approach are often intractible, we may use the following approximation for the unsupervised and supervised problems respectively,

$$p(\mathbf{x}|\mathbf{d}) \approx \int d\boldsymbol{\theta}\ p(\mathbf{x}|\boldsymbol{\theta}_{MAP})\ p(\boldsymbol{\theta}|\mathbf{d}) = p(\mathbf{x}|\boldsymbol{\theta}_{MAP}) \tag{21}$$

$$p(\mathbf{t}|\mathbf{x}, \mathbf{d}) \approx = \int d\boldsymbol{\theta}\ p(\mathbf{t}|\mathbf{x}, \boldsymbol{\theta}_{MAP})\ p(\boldsymbol{\theta}|\mathbf{d}) = p(\mathbf{t}|\mathbf{x}, \boldsymbol{\theta}_{MAP}) \tag{22}$$

It is important to realize that all information about our confidence in the value $\boldsymbol{\theta}_{MAP}$ is thrown away. I.e. in the example of the previous section we are left with a Gaussian with the same mean, but the variance now only contains the term β^2. So we are indeed trading off some information against computational efficiency.

In order to find $\boldsymbol{\theta}_{MAP}$ we need to solve,

$$\boldsymbol{\theta}_{MAP} = \mathbf{argmax}_{\boldsymbol{\theta}}\ p(\boldsymbol{\theta}|\mathbf{d}) \tag{23}$$

$$= \mathbf{argmax}_{\boldsymbol{\theta}}\ \frac{p(\mathbf{d}|\boldsymbol{\theta})\ p(\boldsymbol{\theta})}{p(\mathbf{d})} \tag{24}$$

$$= \mathbf{argmax}_{\boldsymbol{\theta}}\ p(\mathbf{d}|\boldsymbol{\theta})\ p(\boldsymbol{\theta}) \tag{25}$$

$$= \mathbf{argmax}_{\boldsymbol{\theta}}\ \log[p(\mathbf{d}|\boldsymbol{\theta})] + \log[p(\boldsymbol{\theta})] \tag{26}$$

since the log a monotonic function. Notice, that also in this calculation the integral term $p(\mathbf{d})$ is irrelevant. The prior $p(\boldsymbol{\theta})$ is still there and plays a crucial role in protecting against overfitting, an issue we will pick up later. It can be considered as a regularizer term if one expresses his belief in the prior that the function should be smooth. The term can be used to penalize too complex models. Also some complexity terms. like "minimum desrcition length" criterium (MDL, see later), Akaike's information criterium (AIC) and Baysian information criterium (BIC) can be understood as a prior on the complexity of the model.

To illustrate the effect of a prior we will solve the following toy problem (demo-MAP). Let's assume we have N datapoints which we think are generated by a one dimensional Gaussian, with unit variance and mean μ, $p(x|\mu) = \mathcal{G}_x[\mu, 1]$. Since we think that the mean should not be very big we use as a prior $p(\mu) = \mathcal{G}_\mu[0, \alpha^2]$, where α is a hyperparameter. The total objective function is thus,

$$\mathbf{E} \propto -\sum_{n=1}^{N} (x_n - \mu)^2 - \frac{\mu^2}{\alpha^2} \tag{27}$$

which is easily maximized to give,

$$\boldsymbol{\mu} = \frac{1}{N + \frac{1}{\alpha^2}} \sum_{n=1}^{N} x_n \tag{28}$$

The first thing to observe is that for very large $N >> \frac{1}{\alpha^2}$ the influence of the prior is neglicable, and the result is the Maximum Likelihood estimate for the mean of a Gaussian (see next section). On the other hand, for very strong belief in the prior, $\frac{1}{\alpha^2} >> N$, the estimate tends to zero. Thus, if few data are available, the prior will bias the estimate towards the prior expected value.

6 Maximum Likelihood Estimation

If we omit the prior, and thus maximize the likelihood alone, we are calculating the "maximum likelihood" (ML) estimate. Where MAP-estimation still relied on the interpretation of $\boldsymbol{\theta}$ as a random variable (since we had a prior over $\boldsymbol{\theta}$), in ML-estimation $\boldsymbol{\theta}$ is simply viewed as parameter. Frequentists think that it makes no sense to view $\boldsymbol{\theta}$ as a random variable, and ML-estimation is the right thing to do. The argument is that a Bayesian probablity is more like a belief that some event happens than the truely physical probability which must be measured by doing repeated experiments (the frequency of an event occurring is than the probability). We have already seen that when ample data are available, there is no significant difference between ML and MAP estimation (even Bayesian estimation would not add much). Only when the model complexity (number of parameters) becomes too large with respect to the number of data samples, will Bayesian analysis and MAP-estimation outperform ML-estimation. The ML estimate will overfit to the data, i.e. it will fit noise instead of underlying structure, and this will lead to bad generalization performance on a new (test) data set. We will come back to these issues lateron in this lecture, but for the next classes we will assume that there are enough data, and proceed with using the ML estimation procedure. To summarize we have,

$$\boldsymbol{\theta}_{ML} = \mathbf{argmax}_{\boldsymbol{\theta}} \quad \log[p(\mathbf{d}|\boldsymbol{\theta})] \tag{29}$$

Assuming independent samples this will give for unsupervised and supervised learning respectively,

$$\boldsymbol{\theta}_{ML} = \mathbf{argmax}_{\boldsymbol{\theta}} \sum_{n=1}^{N} \log[p(\mathbf{x}_n|\boldsymbol{\theta})] \tag{30}$$

$$\boldsymbol{\theta}_{ML} = \mathbf{argmax}_{\boldsymbol{\theta}} \sum_{n=1}^{N} \log[p(\mathbf{t}_n|\mathbf{x}_n, \boldsymbol{\theta})] \tag{31}$$

As an example we will work out the case of a Gaussian (demo-ML). The model is given by (5). The log-likelihood is given by,

$$L = \frac{1}{2} \sum_{n=1}^{N} \{\log(\det[\boldsymbol{\Sigma}^{-1}]) - (\mathbf{x}_n - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x}_n - \boldsymbol{\mu})\} \tag{32}$$

To maximize this we need to take derivatives with respect to $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}^{-1}$ and equate them to zero. The following identities are useful in doing so,

$$\mathbf{a}^T \mathbf{A} \mathbf{b} = \mathbf{tr}[\mathbf{A} \mathbf{b} \mathbf{a}^T] \tag{33}$$

$$\mathbf{tr}[\mathbf{AB}] = \mathbf{tr}[\mathbf{BA}] \tag{34}$$

$$\frac{\partial}{\partial \mathbf{A}} \mathbf{tr}[\mathbf{AB}] = \mathbf{B}^T \tag{35}$$

$$\frac{\partial}{\partial \mathbf{A}} \mathbf{tr}[\mathbf{A}^T \mathbf{B}] = \mathbf{B} \tag{36}$$

$$\log\det[\mathbf{A}] = -\log\det[\mathbf{A}^{-1}] \tag{37}$$

$$\frac{\partial}{\partial \mathbf{A}} \log\det[\mathbf{A}] = (\mathbf{A}^T)^{-1} \tag{38}$$

We find,

$$\frac{\partial L}{\partial \boldsymbol{\mu}} = \sum_{n=1}^{N} \boldsymbol{\Sigma}^{-1} (\mathbf{x}_n - \boldsymbol{\mu}) \Rightarrow$$

$$\boldsymbol{\mu} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n \tag{39}$$

For $\boldsymbol{\Sigma}$ we find,

$$\begin{aligned}\frac{\partial L}{\partial \boldsymbol{\Sigma}^{-1}} &= \frac{1}{2}\sum_{n=1}^{N}\boldsymbol{\Sigma} - (\mathbf{x}_n-\boldsymbol{\mu})(\mathbf{x}_n-\boldsymbol{\mu})^T \Rightarrow \\ \boldsymbol{\Sigma} &= \frac{1}{N}\sum_{n=1}^{N}(\mathbf{x}_n-\boldsymbol{\mu})(\mathbf{x}_n-\boldsymbol{\mu})^T \end{aligned}\tag{40}$$

These are called the sample mean (arithmetic mean) and sample covariance.

Before reviewing some good properties of ML-estimation let me list two bad ones. First of all there is the above mentioned problem of overfitting is the presence of too few data points. Another problem is that the method is not guaranteed robust. Robustness implies that the estimate of $\boldsymbol{\theta}$ is not too much influenced by deviations from the assumptions. For instance, we could use a slightly wrong model to describe the data, or there could be outliers present (not described by the model). The arithmetic mean and variance are examples of non robust estimators.

Now let me discuss the good properties. We will assume that we have an infinite amount of datasets, each consisting of N independent samples, $\mathbf{x}_i^N$, $i = 1....$ For practical purposes, we may assume a very large amount of such datasets. In the following we will denote by $\mathcal{E}$ the expectation taking over those datasets. We will also assume that the data are generated by a parametrized density $p(\mathbf{x}^N|\boldsymbol{\theta})$. For every dataset we estimate $\boldsymbol{\theta}$ using some kind of estimator $\hat{\boldsymbol{\theta}}(\mathbf{x}^N)$ and study the distribution of those estimates. In this stage this is not neccesarily the ML-estimate, but we will show that if we use the ML estimate, then some nice properties hold. We find for the expected squared error ($\boldsymbol{\theta}_*$ denotes the true value),

$$\begin{aligned}\mathcal{E}[(\hat{\boldsymbol{\theta}}-\boldsymbol{\theta}_*)^2] &= \mathcal{E}[\hat{\boldsymbol{\theta}}^2 - 2\hat{\boldsymbol{\theta}}\boldsymbol{\theta}_* + \boldsymbol{\theta}_*^2)] \\ &= \mathcal{E}[\hat{\boldsymbol{\theta}}^2] - 2\mathcal{E}[\hat{\boldsymbol{\theta}}]\boldsymbol{\theta}_* + \boldsymbol{\theta}_*^2 \\ &= \mathcal{E}[\hat{\boldsymbol{\theta}}^2] - 2\bar{\boldsymbol{\theta}}\boldsymbol{\theta}_* + \boldsymbol{\theta}_*^2 + 2\bar{\boldsymbol{\theta}}^2 - 2\bar{\boldsymbol{\theta}}^2 \\ &= \mathcal{E}[\hat{\boldsymbol{\theta}}^2] + (\boldsymbol{\theta}_*-\bar{\boldsymbol{\theta}})^2 + \bar{\boldsymbol{\theta}}^2 - 2\bar{\boldsymbol{\theta}}^2 \\ &= (\boldsymbol{\theta}_*-\bar{\boldsymbol{\theta}})^2 + \mathcal{E}[\hat{\boldsymbol{\theta}}^2] + \bar{\boldsymbol{\theta}}^2 - 2\mathcal{E}[\hat{\boldsymbol{\theta}}]\bar{\boldsymbol{\theta}} \\ &= (\boldsymbol{\theta}_*-\bar{\boldsymbol{\theta}})^2 + \mathcal{E}[(\bar{\boldsymbol{\theta}}-\hat{\boldsymbol{\theta}})^2] \end{aligned}\tag{41}$$

The first term is the squared bias of the estimate $\hat{\boldsymbol{\theta}}$, while the second term is the asymptotic variance. The bias term is independent of N, while the asymptotic variance scales like $\frac{1}{N}$ with the size of the dataset. This implies that for very large number of data, the bias dominates the error. Without proof we will now state an important theorem about the ML-estimator.

For large N, the distribution of the parameter estimates $\boldsymbol{\theta}_{ML}$ is Gaussian with mean $\boldsymbol{\theta}_*$, and variance $\frac{1}{N}\mathbf{J}^{-1}$. This implies that the estimator is unbiased, and maximally efficient (minimum variance). This follows because we can prove in general the Cramer-Rao bound $\boldsymbol{\Sigma} \geq \frac{1}{N}\mathbf{J}^{-1}$, where $\mathbf{J}$ is the Fisher information matrix.

Basically it says, that if the data are distributed according to $p(\mathbf{x}^N|\boldsymbol{\theta})$ for some $\boldsymbol{\theta}$, than for very large datasets, the mean of the ML estimates will give the correct parameter value (no bias) and more over, we could not do this more efficiently since the variance of the estimates is minimal.

For completeness let us proof the Cramer-Rao inequility. First let us define the score function as,

$$\mathbf{s}(\mathbf{x}^N,\boldsymbol{\theta}) = \frac{\partial}{\partial\boldsymbol{\theta}}\log p(\mathbf{x}^N,\boldsymbol{\theta}) \tag{42}$$

$$= \sum_{n=1}^{N}\mathbf{s}(\mathbf{x}_n,\boldsymbol{\theta}) \tag{43}$$

where $\mathbf{x}^N$ denotes N independent random variables (samples). It is easy to prove the following two properties

for $\mathbf{s}$,

$$\begin{aligned}\mathbf{E}[\mathbf{s}(\mathbf{x}^N,\boldsymbol{\theta})] &= \int d\mathbf{x}^N\ p(\mathbf{x}^N|\boldsymbol{\theta})\ \frac{\frac{\partial}{\partial\boldsymbol{\theta}}p(\mathbf{x}^N|\boldsymbol{\theta})}{p(\mathbf{x}^N|\boldsymbol{\theta})} \\ &= \frac{\partial}{\partial\boldsymbol{\theta}}\int d\mathbf{x}^N\ p(\mathbf{x}^N|\boldsymbol{\theta}) \\ &= \frac{\partial}{\partial\boldsymbol{\theta}}\ 1 = 0 \end{aligned} \tag{44}$$

and

$$\begin{aligned}\mathbf{E}[\boldsymbol{\theta}_{ML}(\mathbf{x}^N)\mathbf{s}(\mathbf{x}^N,\boldsymbol{\theta})] &= \int d\mathbf{x}^N\ p(\mathbf{x}^N|\boldsymbol{\theta})\ \boldsymbol{\theta}_{ML}(\mathbf{x}^N)\ \frac{\frac{\partial}{\partial\boldsymbol{\theta}}p(\mathbf{x}^N|\boldsymbol{\theta})}{p(\mathbf{x}^N|\boldsymbol{\theta})} \\ &= \frac{\partial}{\partial\boldsymbol{\theta}}\int d\mathbf{x}^N\ p(\mathbf{x}^N|\boldsymbol{\theta})\ \boldsymbol{\theta}_{ML}(\mathbf{x}^N) \\ &= \frac{\partial}{\partial\boldsymbol{\theta}}\boldsymbol{\theta} = \mathbf{I} \end{aligned} \tag{45}$$

Then we define the Fisher information as the variance of the score function, which is, since the mean is zero, equal to,

$$\begin{aligned}\mathbf{J}_N(\boldsymbol{\theta}) &= \mathcal{E}[\mathbf{s}(\mathbf{x}^N,\boldsymbol{\theta})\mathbf{s}(\mathbf{x}^N,\boldsymbol{\theta})^T] \\ &= \mathcal{E}[\sum_{n,m=1}^{N}\mathbf{s}(\mathbf{x}_n,\boldsymbol{\theta})\mathbf{s}(\mathbf{x}_m,\boldsymbol{\theta})^T] \\ &= \sum_{n=1}^{N}\mathbf{E}[\mathbf{s}(\mathbf{x}_n,\boldsymbol{\theta})\mathbf{s}(\mathbf{x}_n,\boldsymbol{\theta})^T] + \sum_{n\neq m}\mathbf{E}[\mathbf{s}(\mathbf{x}_n,\boldsymbol{\theta})]\mathbf{E}[\mathbf{s}(\mathbf{x}_m,\boldsymbol{\theta})]^T \\ &= \sum_{n=1}^{N}\mathbf{E}[\mathbf{s}(\mathbf{x}_n,\boldsymbol{\theta})\mathbf{s}(\mathbf{x}_n,\boldsymbol{\theta})^T] \\ &= N\mathbf{J}, \end{aligned} \tag{46}$$

where we used the independence of the samples and the fact that the score function has zero mean. Finally, we proof the inequality by defining,

$$\alpha = \mathbf{a}^T\boldsymbol{\theta}_{ML}(\mathbf{x}^N) \qquad \beta = \mathbf{b}^T\mathbf{s}(\mathbf{x}^N,\boldsymbol{\theta}) \tag{47}$$

From the Cauchy-Schwartz inequality we have,

$$\mathcal{E}[(\alpha-\mathcal{E}[\alpha])^2]\ \mathcal{E}[(\beta-\mathcal{E}[\beta])^2] \geq \mathcal{E}[(\alpha-\mathcal{E}[\alpha])(\beta-\mathcal{E}[\beta])]^2 \tag{48}$$

Using (44) and (45) we may write,

$$\begin{aligned}(\mathbf{a}^T\boldsymbol{\Sigma}\mathbf{a})(\mathbf{b}^T\mathbf{J}_N\mathbf{b}) &\geq (\mathbf{a}^T\mathbf{b})^2 \Rightarrow \\ \frac{(\mathbf{a}^T\mathbf{b})^2}{(\mathbf{a}^T\boldsymbol{\Sigma}\mathbf{a})(\mathbf{b}^T\mathbf{J}_N\mathbf{b})} &\leq 1 \end{aligned} \tag{49}$$

Now we will maximize the right hand site over $\mathbf{b}$ using the lemma

$$\mathbf{max_b}\frac{(\mathbf{a}^T\mathbf{b})^2}{\mathbf{b}^T\mathbf{J}_N\mathbf{b}} = \mathbf{a}^T\mathbf{J}_N^{-1}\mathbf{a} \tag{50}$$

giving

$$\mathbf{a}^T\boldsymbol{\Sigma}\mathbf{a} \geq \mathbf{a}^T\frac{\mathbf{J}^{-1}}{N}\mathbf{a} \quad \forall\mathbf{a} \tag{51}$$

which is the definition of $\boldsymbol{\Sigma} \geq \frac{1}{N}\mathbf{J}^{-1}$.

7 Generalization, Bias/Variance tradeoff, Minimum Description Length

We have already mentioned several times that ML-estimation does not solve the problem of overfitting. What we need is a criterium to choose from a set of different models the one which is expected to generalize best, i.e. the one which has not fitted the noise. This amounts to choosing the model with the right complexity. We may need to choose between different models with different number of parameters (model order selection) or equivalent models, but trained in a different way. In the latter case, the effective complexity can be different, e.g. smooth solutions will have a lower complexity than highly structured solutions.

To illustrate what is going on we will go back to the derivation of the squared error of an estimator. Now we will look at the integrated squared error between the true probability density and the estimated one and proof a similar bias-variance trade-off. Lets call $\hat{p}(\mathbf{x})$ the estimate of the pdf using the data $\mathbf{x}^N$. We could repeat the same analyis for $p(\mathbf{t}|\mathbf{x})$ for the supervised case. We are interested in the goodness of fit, i.e.,

$$L_2 = \int d\mathbf{x}\ \mathcal{E}[(p_*(\mathbf{x}) - \hat{p}(\mathbf{x}))^2] \tag{52}$$

where p_* denotes the true density. Following essentially the same procedure as for the bias/variance trade off for estimators, and using $\mathcal{E}[\hat{p}(\mathbf{x})] = \bar{p}(\mathbf{x})$ and $\mathcal{E}[p_*(\mathbf{x})] = p_*(\mathbf{x})$, we find,

$$L_2 = \int d\mathbf{x}\ \{(p_*(\mathbf{x}) - \bar{p}(\mathbf{x}))^2 + \mathcal{E}[(\hat{p}(\mathbf{x}) - \bar{p}(\mathbf{x}))^2]\} \tag{53}$$

The first term is again the bias and the second the variance around the mean estimate. The reason that this equation is important is that it illuminates a trade-off between the bias and variance. We may reason as follows. Let's assume we have a very simple model to describe the data. It is then very likely that there is a large bias in the estimation of $p(\mathbf{x})$, simply because our model is not flexible enough to capture all the structure. The variance however is expected to be small since the parameters are well determined by the data. On the other hand, imagine a very complex model. Due to the enormous flexibility we probably have a very small bias term, but since the data cannot determine all parameters, the different datasets drawn from the true PDF cause the parameters to acquire very different values every time we estimate them. Thus, the variance is expected to be large. What we want is a small expected error, and therefore we need to trade off the bias and the variance by choosing a model which has the right complexity.

A fully Bayesian approach can deal with these issues by simply integrating over all possible values of the parameters and/or summing over the different models. However, the choice of the priors for both parameter values and models is a difficult issue, and moreover the integrals involved are usually too complicated to do analytically or very time consuming to do approximately. As an alternative one could use MAP-estimation, with a log-prior which acts as a penalty term for too complex models. To choose between different models some penalty terms were suggested in the literature (AIC, BIC, MDL), one of which (MDL) we will explore a bit further below. These terms provide rough estimates for the correct number of parameters in the light of the available data. To avoid overfitting during training of a specific model, one may add regularizer terms which penalize too high derivatives for instance, i.e. they enforce smoothness. Adding noise to the data has a similar effect. Also weight decay in neural nets is an instance of a log-prior term which acts as a regularizer.

Yet another alternative is to split the dataset into a training set and a validation set. During training of the model, the training error is expected to decrease, but when the model starts overfitting its generalization performance will become worse, resulting in an increasing validation error. A simple heuristic is therefore to stop the training when the validation error increases (early stopping). The same procedure may be used to test different models with different numbers of degrees of freedom. The downside is of course that one cannot use all data for training. Cross-validation is designed to overcome this problem, by splitting the data in many small chunks, train on all but one chunk and test on the remaining chunck. Then cycle through the chunks and average the result. Of course, this procedure is computationally demanding.

Let me explain in a little bit more detail what the philosophy behind the "minimum description length" (MDL, mentioned above) principle is. The idea is that we image that we need to send the samples $\mathbf{x}^N$ over a communication channel to an imaginary receiver. If both sender and receiver agree beforehand on a probability distribution for $p(\mathbf{x}^N)$, then the minimum cost of sending these data is according to Shannon's theorem $-\log[p(\mathbf{x}^N)]$ bits, while we can approximate this bound arbitrarily closely by some coding scheme. Let's assume therefore that this is the actual cost of sending the data (for continuous data we need to quantize the space in bins of size $d\mathbf{x}^N$ and the coding cost is then multiplied by that binsize). There is however a smarter idea, which is to build a model $\mathcal{M}$, which captures the dependencies (structure) in the data, then send

- The specifics of the model

- The activities of the model when applied to the data
- The reconstruction errors which are the differences of the predicted values of the model and the true values.

Of course, for all continuous values we need to decide on the tolerances (precision) of the coding. With this information the receiver can losslessly recontruct the data, but we have send fewer bits! The principle behind MDL is the belief that the model for which the sum of those three costs is minimized also has the best generalization performance. To see how this can work we first image that we have very few datapoints. In this case it makes no sense to build a sophisticated model, since sending the specifics of the model require more bits than the data in the first place. Next, imagine we have enormous amounts of data at our disposal, than the specifics of the model are a fraction of the total cost and we may profit from building complicated models. Note also, that building accurate models reduces the error term, so the simplest, most accurate model that can describe the data will win. We see that there is a trade-off between the first cost and the last two costs, which will be minimized by a model of the "correct" complexity. As an example consider some data which almost on a straight line in two dimensions. We may send the values $\{\mathbf{x}_1, \mathbf{y}_1, ...\mathbf{x}_N, \mathbf{y}_N\}$ directly or we may recognize that it is cheaper to send the two parameters determining the straight line (cost 1), then project the datapoint onto the line and send distances along the line, for instance from the x-intersection (cost 2). Finally send the errors (cost 3). Notice that instead of sending two potentially large values for every datapoint, we now send one large value (distance along line) and a small one (error) per datapoint, which may be coded with fewer bits, plus two values for the line specifics which are independent of the number of datapoints.

Using this philosopy, Rissanan derived an error term which should be added to the ML criterium,

$$MDL = ML - \frac{1}{2}(\#\ \mathbf{parameters})\ \log(N) \tag{54}$$

which, when maximized, gives the model with the best generalization performance. It must be said however that for very small datasets (54) is not very accurate.

Recommended literature, (Bishop, 1995), (Ripley, 1996).

References

Bishop, C. (1995). *Neural networks for pattern recognition.* Clarendon Press.

Ripley, B. (1996). *Pattern recognition and neural networks.* Cambridge University Press.