Bayesian Modelling in Machine Learning: A Tutorial Review

Matthias Seeger
Probabilistic Machine Learning and Medical Image Processing
Saarland University
Room 116, Campus E1.4, 66123 Saarbruecken
mseeger@mmci.uni-saarland.de

March 21, 2009

Abstract

Many facets of Bayesian Modelling are firmly established in Machine Learning and give rise to state-of-the-art solutions to application problems. The sheer number of techniques, ideas and models which have been proposed, and the terminology, can be bewildering. With this tutorial review, we aim to give a wide high-level overview over this important field, concentrating on central ideas and methods, and on their interconnections. The reader will gain a basic understanding of the topics and their relationships, armed with which she can branch to details of her interest using the references to more specialized textbooks and reviews we provide here.

1 Introduction

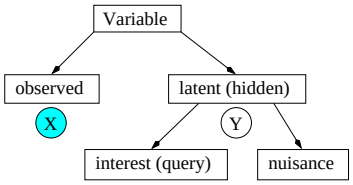
Machine Learning is a hybrid of Statistics and algorithmic Computer Science. In this review, we will mostly be concerned with the statistical side. Statistics is about *managing and quantifying* uncertainty. Uncertainty may arise due to many different reasons, for example:

- *Measurement noise*: Measurements of physical processes are always subject to inaccuracies. Sometimes, low quality data may be obtained more economically. Data items may be missing
- *Model uncertainty*: Models are almost never exact, we abstract away complexity in order to allow for predictions to be feasible
- *Parameter uncertainty*: Variables in a model can never be identified exactly and without doubt from finite data

The calculus of uncertainty is *probability theory*, [22] gives a good introduction. Some phenomenon of interest is mapped to a *model*, being a set of *random variables* and probabilistic relationships between them. Variables are *observed* or *latent (unobserved)*. To give an example, consider the *linear model*

$$y = \boldsymbol{w}^T \boldsymbol{\phi}(x) + \varepsilon, \tag{1}$$

where ε is independent noise. This model describes a functional relationship $x \to y \in \mathbb{R}$. It is a cornerstone of Statistics, and we will see much of it in the following. Suppose we can measure (x, y) repeatedly and independently. Here, x and y are observed, $\boldsymbol{w}$ and ε are latent. Latent variables are *query* or *nuisance*, we want to know only about the former. For example, $\boldsymbol{w}$ may be query (is there a linear trend in the data? are some features more relevant than others?), ε is nuisance. $\boldsymbol{w}$ is also called *parameter* or *weights*. It is important to note that the classification of model variables into observed, query, or nuisance depends on the task which is addressed by the model.



Some statistical tasks for this model: What is the "best" value for $\boldsymbol{w}$ representing our data? We are looking for an *estimate* $\hat{\boldsymbol{w}}$ for $\boldsymbol{w}$ computed from the data. But maybe we are uncertain about $\boldsymbol{w}$ (we should always be!). How does the data change our uncertainty from *before* to *after* having seen data? For example, can we specify a region that $\boldsymbol{w}$ lies in with "high confidence"? Bayesian Statistics goes beyond estimators in order to address the second problem.

Maybe we merely want to *predict* y for x not seen in our measurements. Now, y is latent query, while $\boldsymbol{w}$ becomes nuisance. This ambivalence in the role of variables is resolved in the Bayesian paradigm by *treating all variables as random variables*. We note that the failure to acknowledge uncertainty can lead to unexpected problems. A good example is the phenomenon of *overdispersion*. When fitting basic models to data with maximum likelihood (see Section 2.1), it is often noted that the variance of responses predicted from the fitted model is significantly smaller than the variance apparent in the observed data. This "phenomenon" indicates that uncertainties have been ignored in the process of fitting. "Best" parameter values have been plugged in rather than admitting to uncertainties in this choice. If these sources of uncertainty are properly accounted for, the excess in variance typically all but disappears.

Given several models, how to *compare* them? Which one explains the data "best"? For model comparison, all latent variables are nuisance, and we need a *score* to rank the models, given the data. Model comparison is related to *hierarchical models* in Bayesian Statistics, where *hyperparameters* index models or variants of a model (an example is given in Section 2.2), and we need to estimate a "good" value. Finally, *objective model validation* measures the behaviour of the fitted model on *heldout test data*.

The *overfitting problem* is central in Statistics. At some point, we may have to estimate a single value $\hat{\boldsymbol{w}}$ for $\boldsymbol{w}$, or to prefer a single model over others. However, for finite data we can obtain better and better fits with more and more "complicated" models, say in the linear model above with larger and larger weights $\boldsymbol{w}$. Such "overfits" eventually reproduce the observed data exactly, but perform poorly in test data validation. This problem arises only because we do make a fixed choice and ignore some uncertainty. It can be controlled however by defining appropriate notions of *complexity* and implementing a reasonable trade-off between *good fit* of observed data and *low complexity*. This trade-off is inherent in Bayesian procedures, as we will see. The overfitting problem is illustrated in Figure 1.

The Bayesian approach to Statistics is sometimes depicted as more of a "religion" than a pragmatic problem-oriented approach. We think this misses the point, and we will not take part in such discussions. In a nutshell, Bayesianism is about the consequent usage

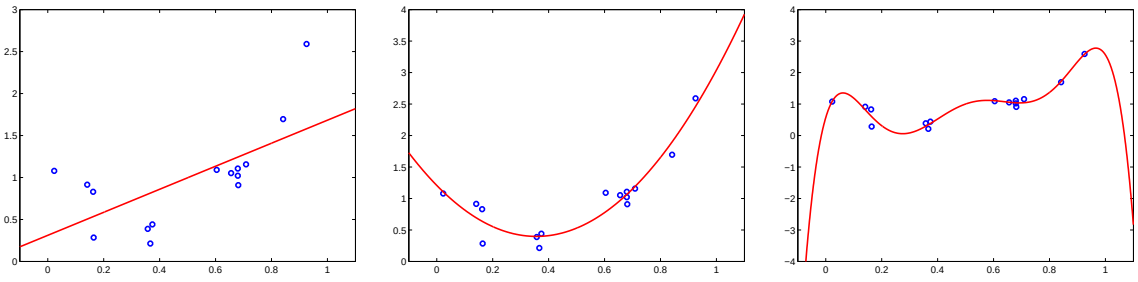


Figure 1: Illustration of overfitting problem. Noisy data from a quadratic is fitted by a line (left), a quadratic (middle), and a sixth-order polynomial (right). Fitting is done by least squares, which is the maximum likelihood estimate for Gaussian noise. The sixth-order polynomial fit attains smaller residual error on the data, but at the expense of an over-complicated solution whose specifics are determined by noise (and therefore arbitrary) away from the data. Generalization requires some form of complexity control.

of two elementary probability operations: *conditioning* and *marginalization*. Conditioning embodies the "what if E were known" notion via

$$P(A \cap E) = P(A|E)P(E)$$

for events A, E, and the corresponding extension to random variables via the sigma algebras generated by them.[1] Marginalization is the operation for abstracting away facts, "not wanting to know about":

$$P(Y) = \int P(X, Y)\, dX$$

Both operations are integral parts of probability, and all Statisticians make use of them. In general, Bayesian Statisticians agree that the calculus of probability may have a wider applicability than abstracting from limits of count ratios in repeatable experiments, in that it has the potential of formalizing *subjective beliefs*, connecting them with real observations, and making valid inferences by updating the beliefs using conditioning and marginalization. Second, Bayesians insist on the complete specification of a model as joint distribution over all relevant variables.[2] This means that even variables about which no definite prior knowledge whatsoever is available, have to be endowed with prior distributions.[3] Given a complete model, the Bayesian program for a given task simply involves identifying the roles of all variables, conditioning on the observed ones, and marginalizing over the latent nuisance ones, in order to make an inference about the latent query variables.

The structure of this review is as follows. In Section 2, concepts of Bayesian statistics are introduced using the concrete example of the linear model (Eq. 1). It is shown that complexity control is directly embodied in Bayesian analysis via Occam's razor. In Section 3, we introduce graphical models as convenient framework for model building and computing inferences, stressing the importance of latent variables in order to construct realistic models. Several different examples of graphical models are discussed. The classification problem is

[1] The conditional expectation $\mathrm{E}[Y|X]$ is defined as best predictor of Y (in least squares) given X (a function of X).

[2] Covariates, *i.e.* variables which are always observed, may be an exception, as we will see.

[3] Much work has been devoted to formalize such appropriate uninformative priors or reference priors [4]. Vague but informative priors can be created using hierarchical models.

introduced together with the logistic regression model in Section 4, where we also introduce the notion of nonparametric models and regularized estimation. Finally, Gaussian process models are discussed in Section 5 as example of nonparametric Bayesian models.

Sections marked with an asterisk "$*$" contain additional advanced material which can be skipped at first reading without disrupting the main flow of the text.

2 Bayesian Statistics for Machine Learning

In this Section we define and motivate basic terms of probability and Bayesian statistics relevant for Machine Learning. The linear model is introduced, the notion of complexity control via Occam's razor is motivated.

2.1 Concepts of Bayesian Statistics

In this Section we introduce basic concepts of Bayesian Statistics, using the example of the linear model (Eq. 1). A good general textbook for Bayesian analysis is [3], while [4] focus on theory. The Bayesian approach to Machine Learning has been promoted by a series of papers of [40] and by [47]. [7] provides an introductory textbook with emphasis on neural networks, [41] has a wider scope and provides links with coding and information theory.

The linear model is of elementary importance in Statistics, being the essential building block in many more elaborate models. We present some simple motivating examples in Section 2.1.1, and the linear model will lead like a red thread though the remainder of this text. To complete the model specification, assume that the noise is Gaussian: $\varepsilon \sim N(0, \sigma^2)$, where σ^2 is the noise variance (note that $y \in \mathbb{R}$). Here, $N(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ denotes the Gaussian distribution over $\boldsymbol{x}$ with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$, and we write $N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ if $\boldsymbol{x}$ is clear from the context. The *probability density function* (pdf) of the Gaussian is given by

$$N(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = |2\pi\boldsymbol{\Sigma}|^{-1/2} \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right).$$

The Gaussian is another cornerstone of Statistics, it is maybe the most important distribution family. There are several reasons for this. First, when comparing all distributions with a given mean and covariance matrix, the Gaussian has the least structure (or additional information) among them.[4] All cumulants higher than second order vanish. Losely speaking, a Gaussian embodies mean and covariance matrix, but nothing else. Second, it is very easy to work with Gaussians. If $\boldsymbol{x}$ is Gaussian, conditioning on or marginalizing over some components leaves you with another Gaussian, as does any affine mapping. No other non-trivial family of continuous distributions has such wide-ranging closure properties. Manipulating Gaussians requires some linear algebra which can seem bewildering, but becomes manageable by concepts such as Schur complements. [61], Appendix A, provides details.

σ^2 is an example for a *hyperparameter*. The distinction between (primary) parameters (here: $\boldsymbol{w}$) and hyperparameters is ultimately artificial, but often useful in structuring a problem into a *hierarchy*. Think about *generating the data*: first, sample hyperparameters, then

[4] Absence of structure or information content can be measured by the differential entropy $\mathrm{H}[P] = \mathrm{E}_P[-\log P(\boldsymbol{x})]$.

parameters given hyperparameters, then data given all parameters. Such *hierarchical models* are a Bayesian answer to the apparent dilemma of having to specify models and priors which are vague (in order not to exclude important aspects of the true phenomenon) using hard model assumptions and concise distribution families with clear semantics (such as Gaussian, linear, *etc*). We come back to this aspect in Section 3. The data is $D = \{(x_i, y_i) \,|\, i = 1, \ldots, n\}$. We assume that the (x_i, y_i) are *independent and identically distributed* (i.i.d.), given all parameters. The (complete, or joint) *likelihood* is the probability of the data given all parameters:

$$P(D|\boldsymbol{w}, \sigma^2) = \prod_{i=1}^{n} N(y_i|\boldsymbol{w}^T \boldsymbol{\phi}(x_i), \sigma^2).$$

$P(D|\boldsymbol{w}, \sigma^2)$ is a product of independent terms, due to the i.i.d. assumption on D. Note that the likelihood is really only a density over the targets $\{y_i\}$, this is because our model is *conditional*: we do not care about modelling the input point x, because it will always be available at prediction (see end of Section 2.1.1).

An estimator for $\boldsymbol{w}$ is given by

$$\hat{\boldsymbol{w}}_{ML} = \operatorname*{argmax}_{\boldsymbol{w}} P(D|\boldsymbol{w}, \sigma^2),$$

the *maximum likelihood* (ML) estimator. Maximum likelihood is a very powerful estimation principle, and is at the heart of very many Machine Learning algorithms. Any estimator of $\boldsymbol{w}$ implies a *plug-in* prediction rule as $\hat{y}_{ML}(x) = \hat{\boldsymbol{w}}_{ML}^T \boldsymbol{\phi}(x)$. ML estimators are traditionally analyzed in a *frequentist* framework, where the assumption is that there is a true underlying parameter $\boldsymbol{w}_*$ for which repeated samples D are drawn for the purpose of analyzing properties of the estimator. Such analysis terms are generally of the form $\mathrm{E}_{D|\boldsymbol{w}_*}[f(\hat{\boldsymbol{w}}_{ML}(D), \boldsymbol{w}_*)]$. Examples include the bias $\mathrm{E}_D[\hat{\boldsymbol{w}}_{ML}(D)] - \boldsymbol{w}_*$, the covariance $\mathrm{Var}_D[\hat{\boldsymbol{w}}_{ML}(D)]$, or the expected squared error $\mathrm{E}_D[\|\hat{\boldsymbol{w}}_{ML}(D) - \boldsymbol{w}_*\|^2]$. An estimator is called *unbiased* if its bias is zero for any $\boldsymbol{w}_*$. Many proven frequentist properties are *asymptotic*, in the sense that they hold true as the sample size $n \to \infty$. Many theoretical results about ML can be found in [11].

However, with ML estimation problems of overfitting arise if $\boldsymbol{w}$ is high-dimensional as compared to the number of datapoints n. We also note that the model so far is not completely specified. Namely, what are the distributions of the parameters? The Bayesian paradigm which allows us to move beyond parameter estimation, requires that *prior distributions* are specified for all variables. In our example, we might assume the prior $P(\boldsymbol{w}) = N(\boldsymbol{0}, \boldsymbol{I})$, favouring no direction in particular, but coding a bias for smaller weights $\boldsymbol{w}$ rather than larger ones.[5] Having done so, we can obtain the *posterior distribution* of $\boldsymbol{w}$ *after* having seen the data. This is a simple exercise in probability, recalling that *knowing* a variable means *conditioning* on it. First, we have $P(D, \boldsymbol{w}|\sigma^2) = P(D|\boldsymbol{w}, \sigma^2)P(\boldsymbol{w})$. Next,

$$P(\boldsymbol{w}|D, \sigma^2) = \frac{P(D, \boldsymbol{w}|\sigma^2)}{P(D|\sigma^2)}, \quad P(D|\sigma^2) = \int P(D, \boldsymbol{w}|\sigma^2)\, d\boldsymbol{w}. \tag{2}$$

This is sometimes called *Bayes formula*, but is really a basic fact of probability. Bayesians use this formula in order to implement coherent and consistent *inference* (going from prior to posterior).

[5] We consider σ^2 fixed and given in the moment. We can always do such a *conditional* Bayesian analysis first, then extend it later.

The posterior is the Bayesian solution to the "confidence region" task mentioned above. It exactly quantifies our remaining uncertainty in $\boldsymbol{w}$ given the data. We can obtain a "Bayesian estimate" of $\boldsymbol{w}$ by extracting mean, mode, or median, but a more useful report includes size and shape of the region of high posterior mass. Of course, the posterior depends on the prior, thus our report will be *subjective*, but it is important to note that *any* report from finite data will *always* be subjective: why this model (and no other)? Why only these variables (and no more)? Why Gaussian noise? While it is sometimes possible to at least motivate certain individual choices in a more or less objective manner, this is hardly ever possible for the *combination* of many choices which lead to a model specification.

The prediction task mentioned above is solved by *marginalizing* over (*i.e. integrating out*) the parameters, in order to obtain the *predictive distribution*:

$$P(y|x, D, \sigma^2) = \int P(y|x, \boldsymbol{w}, \sigma^2) P(\boldsymbol{w}|D, \sigma^2)\, d\boldsymbol{w}.$$

Marginalization is the correct way of dealing with nuisance variables (recall that for the prediction task, $\boldsymbol{w}$ *is* nuisance).[6] Again, we can estimate our "best" prediction of y as mean or mode of $P(y|x, D, \sigma^2)$, but reporting more features of the predictive distribution is more versatile. For example, the variance of $P(y|x, D, \sigma^2)$ can be used to obtain "Bayesian confidence intervals".

We should really marginalize over σ^2 as well, but we run into the most common problem with Bayesian inference in practice: the required integral is not analytically tractable. We can either use an approximation, or we can settle for an estimate at this point. The Bayesian community splits at this point: some "fundamentalists" reject estimates in general, while other *empirical Bayesians* advocate careful estimation procedures at least for hyperparameters, keeping in mind that overfitting might arise. The normalization quantity $P(D|\sigma^2)$ in Eq. 2 is called *marginal likelihood* (or *evidence*, or *partition function*). A general empirical Bayes estimate of σ^2 is obtained as a maximum of $P(D|\sigma^2)$, or of the hyperposterior $P(\sigma^2|D) \propto P(D|\sigma^2)P(\sigma^2)$. We will motivate this estimator in Section 2.3, and note here merely that it is an instance of the *maximum a posteriori* (MAP) approximation: instead of averaging $P(y|x, D, \sigma^2)$ over the posterior $P(\sigma^2|D)$, we plug in a maximum point $\hat{\sigma}^2$ of $P(\sigma^2|D)$, which can be seen as "averaging" over the point distribution $\delta_{\hat{\sigma}^2}$.

More general, marginal likelihoods can be used for model comparison, for example the *Bayes factor* $P(D|M_1)/P(D|M_2)$ can be used to compare models M_1 and M_2 (with all parameters marginalized over) [37].

2.1.1 Examples of the Linear Model [*]

In this Section, we motivate the linear model of Eq. 1 using some elementary examples. A classic text on basic pattern recognition techniques and their analysis is [16], which contains details about the examples here and many others. Suppose that data $\boldsymbol{x}_i \in \mathbb{R}^d$ can come from two different populations $c_i \in \{1, 2\}$. We can model this situation by assuming that $\boldsymbol{x}_i \sim N(\boldsymbol{\mu}_1, \boldsymbol{\Sigma})$ if $c_i = 1$, and $\boldsymbol{x}_i \sim N(\boldsymbol{\mu}_2, \boldsymbol{\Sigma})$ if $c_i = 2$. Having specified $P(\boldsymbol{x}_i|c_i)$ this way,

[6] It is sometimes advocated to maximize over latent nuisance variables, but in general this leads to unwanted bias ([72], Sect. 4.3.4, describes the problem and gives references), or to overfitting.

and assuming that $P(c_i = 1) = p,\ P(c_i = 0) = 1 - p$, it is an easy exercise to the reader in probability to compute the conditional distribution

$$P(c_i = 1|\boldsymbol{x}_i) = \lambda\left(\boldsymbol{a}^T\boldsymbol{x}_i + b\right), \quad \boldsymbol{a} = \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2),\ b = \log\frac{p}{1-p} + \frac{1}{2}\boldsymbol{\mu}_2^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_2 - \frac{1}{2}\boldsymbol{\mu}_1^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_1,$$

where $\lambda(u) = (1 + e^{-u})^{-1}$ is the logistic function, strictly increasing from 0 to 1. Thus, our simple class-conditional Gaussian setup implies a linear model for the conditional distribution $P(c_i|\boldsymbol{x}_i)$ which can be used to predict the class c_i for unseen points $\boldsymbol{x}_i$. The setup also renders particular semantics to the parameters $(\boldsymbol{a}, b)$, which hold if the class-conditional model is reasonable for the task at hand. In this case, the parameters may be estimated by ML (see Section 2.1) which amounts to plugging in sample averages for $\boldsymbol{\mu}_1$, $\boldsymbol{\mu}_2$, and $\boldsymbol{\Sigma}$. This example is interesting also because the linear model is *embedded* (into the logistic function) rather than directly applied to real-valued targets as in Eq. 1. We discuss such embedded (or generalized) linear models in Section 4.1.

Another example is given by *analysis of variance* (ANOVA) models of the additive type. The general idea is that a measurement y may exhibit variance from different sources, and an ANOVA model allows us to unravel these factors. ANOVA models are hierarchical, associating the entity for each observation to groups at different levels. In a clinical study, $y_{i,j,k}$ may be associated with gender k, age group j, and index i within gender-age group. We could then model $y_{i,j,k} = w + w_k + w_{j,k} + \varepsilon_{i,j,k}$, w being the over-all mean, w_k the gender-specific contribution, $w_{j,k}$ the gender-age-specific contribution, and $\varepsilon_{i,j,k}$ accounting for variability within gender-age group (j, k). If we collect all parameters into $\boldsymbol{w}$, this becomes a linear model, where the $x_{i,j,k}$ contain the group attributes, and $\phi(x_{i,j,k})$ is a vector of zeros and ones coding these attributes. $\boldsymbol{w}$ is given a zero-mean Gaussian prior whose covariance matrix may encode assumptions about the levels of the hierarchy. For example, w may be thought to have higher variance than $w_{j,k}$. Treating this ANOVA model in a Bayesian way would amount to determine posterior covariances for the $\boldsymbol{w}$ components which gives an answer to the distribution of variance problem. Details about this view on ANOVA can be found in [8].

Finally, the *Naive Bayes* (or Idiot's Bayes) classifier [42] is popular in Machine Learning. Suppose, $\boldsymbol{x}_i \in \{0, 1\}^d$ has to be classified according to $c_i \in \{1, \ldots, C\}$. If d is large, the class-conditional distributions $P(\boldsymbol{x}_i|c_i)$ cannot be fitted reliably from moderate amounts of data. The Naive Bayes classifier is obtained by making the model assumption that $P(\boldsymbol{x}_i|c_i) = \prod_{j=1}^d P_j(x_{i,j}|c_i)$, namely that the components of $\boldsymbol{x}_i$ are independent if the class c_i is known. This assumption leads to a dramatic reduction in the number of parameters to be estimated from data. Namely, let $w_{j,c} = \log[P_j(x_j = 1|c)/(1 - P_j(x_j = 1|c))]$. We have that

$$P_j(x_j|c) = P_j(x_j = 1|c)^{x_j}(1 - P_j(x_j = 1|c))^{1-x_j} = \exp\left(x_j w_{j,c} - \log(1 + e^{w_{j,c}})\right).$$

Introduce $w_{0,c}$ and let $\boldsymbol{w}_c = (w_{0,c}, \ldots, w_{d,c})^T$ and $\tilde{\boldsymbol{x}} = (1, \boldsymbol{x}^T)^T$. Then,

$$P(\boldsymbol{x}, c) = P(c)\prod_{j=1}^d P_j(x_j|c) = \exp(\boldsymbol{w}_c^T\tilde{\boldsymbol{x}}), \quad w_{0,c} = \log P(c) - \sum_j \log(1 + e^{w_{j,c}}).$$

Again, $(P(c|\boldsymbol{x}))_c$ is given as a increasing mapping of the linear functions $\boldsymbol{w}_c^T\tilde{\boldsymbol{x}}$. The mapping is a multivariate generalization of the logistic function and is known as softmax mapping (see Section 4.1). The likelihood for some data $\{(\boldsymbol{x}_i, c_i)\}$ is given by $\exp(\sum_i \boldsymbol{w}_{c_i}^T\tilde{\boldsymbol{x}}_i)$, which

can be maximized under a constraint on the $w_{0,c}$ (since $\sum_c P(c) = 1$). The ML estimate can be computed analytically and amounts to plug in empirical sample counts for the corresponding $P_j(x_j|c)$ and $P(c)$.

Finally we note an important point. Both in the class-conditional Gaussian and in the Naive Bayes model, the linear model can be identified as underlying building block. However, by the class-conditional property of these models (meaning that $P(\boldsymbol{x}|c)$ are specified explicitly), these models also come with fixed parameter semantics. In other words, if we talk about parameter estimation in these models, we mean estimation of the class-conditional densities (for example, of the class means and covariance matrix in the Gaussian model). The role of the linear model in these cases is very different from the role it will play in much of the remainder of this text, where $\boldsymbol{w}$ is typically unconstrained, and estimation of $\boldsymbol{w}$ is based directly on the likelihood of the data $y_i|\boldsymbol{x}_i$ given $\boldsymbol{w}$. The reader will appreciate this difference by contrasting the treatment in Section 2.1 from the class-conditional examples here, where $\boldsymbol{w}$ is in fact a mapping from the parameters of the class-conditional densities. Members of the latter class (like Naive Bayes) are called *generative*, *joint*, or *sampling* methods, while methods like the inference of Section 2.1 are called *diagnostic*, *predictive*, *conditional*, or sometimes *discriminative*. An important property of generative methods is that an estimate of the marginal $P(\boldsymbol{x})$ is always obtained as side product, while this is not the case for predictive techniques. On the other hand, predictive methods often come with many less parameters to be fitted from data. Both paradigms come with their strong and weak points, and it is not possible to favour one uniformly over the other. [49] give some comparative arguments. Hybrids between joint and conditional methods are used to address the problem of *semi-supervised learning* ([60] provides a review) where an additional pool of unlabeled data is available to aid predictions $x \to y$ from labeled data D.

2.2 The Linear Model: A Bayesian View

In this Section, we give a concrete example for Bayesian analysis for the linear model (Eq. 1).

Recall our Bayesian setup of the linear model from Section 2.1. Consider σ^2 fixed for the moment. Let $\boldsymbol{y} = (y_i)_i,\ \boldsymbol{\varepsilon} = (\varepsilon_i)_i,\ i = 1, \ldots, n$. Also, write $\boldsymbol{X} = (\boldsymbol{\phi}(x_1) \ldots \boldsymbol{\phi}(x_n))^T \in \mathbb{R}^{n,p}$, where $\boldsymbol{w} \in \mathbb{R}^p$. The joint distribution $P(\boldsymbol{w}, \boldsymbol{y}|\sigma^2)$ is Gaussian, therefore the posterior $P(\boldsymbol{w}|\boldsymbol{y}, \sigma^2)$ is Gaussian as well and can be obtained by collecting the terms depending on $\boldsymbol{w}$ only (exercise to the reader!):

$$P(\boldsymbol{w}|\boldsymbol{y}, \sigma^2) = N\left(\boldsymbol{M}^{-1}\boldsymbol{X}^T\boldsymbol{y}, \sigma^2\boldsymbol{M}^{-1}\right), \quad \boldsymbol{M} = \boldsymbol{X}^T\boldsymbol{X} + \sigma^2\boldsymbol{I}. \tag{3}$$

Since the posterior is Gaussian, mean and mode are the same and give rise to the Bayesian estimate $\mathrm{E}[\boldsymbol{w}] = \boldsymbol{M}^{-1}\boldsymbol{X}^T\boldsymbol{y}$. Furthermore, regions of large posterior mass are ellipsoids around that point. The shape of these reveals how well certain linear combinations of feature components of $\boldsymbol{\phi}(\cdot)$ can be estimated from the data. For example, if $\boldsymbol{v}$ is an unit eigenvector with small eigenvalue λ of the posterior covariance $\sigma^2\boldsymbol{M}^{-1}$ (*i.e.* with large eigenvalue of $\boldsymbol{M}$), then the contribution of $\boldsymbol{v}^T\boldsymbol{\phi}(x)$ to y is well determined by the data, because the posterior variance $\mathrm{Var}[\boldsymbol{v}^T\boldsymbol{w}] = \sigma^2\boldsymbol{v}^T\boldsymbol{M}^{-1}\boldsymbol{v} = \lambda$ is small. The predictive distribution is obtained by marginalizing over $\boldsymbol{w}$ in $y = \boldsymbol{w}^T\boldsymbol{\phi}(x) + \varepsilon$:

$$P(y|x, D, \sigma^2) = N\left(\boldsymbol{y}^T\boldsymbol{X}\boldsymbol{M}^{-1}\boldsymbol{\phi}(x), \sigma^2(1 + \boldsymbol{\phi}(x)^T\boldsymbol{M}^{-1}\boldsymbol{\phi}(x))\right). \tag{4}$$

If we ignore the noise in the predictive distribution (why should our prediction be corrupted by noise?), the "1+" is dropped. We can now compute a best estimate of y under any loss function (by minimizing the expected loss, where we use predictive expectation) and an uncertainty estimate from the predictive mean and variance.

What to do with the noise variance σ^2? We can compute the posterior of σ^2 for a reasonable prior, but marginalizing over σ^2 is not tractable. The marginal likelihood $P(\boldsymbol{y}|\sigma^2)$ is Gaussian. We have that $\boldsymbol{y} = \boldsymbol{X}\boldsymbol{w} + \boldsymbol{\varepsilon}$. Since $\boldsymbol{w} \sim N(\boldsymbol{0}, \boldsymbol{I})$ and $\boldsymbol{\varepsilon} \sim N(\boldsymbol{0}, \sigma^2\boldsymbol{I})$ are independent, we have that

$$P(\boldsymbol{y}|\sigma^2) = N\left(\boldsymbol{0}, \boldsymbol{X}\boldsymbol{X}^T + \sigma^2\boldsymbol{I}\right), \tag{5}$$

because $\mathrm{E}[\boldsymbol{y}\boldsymbol{y}^T] = \boldsymbol{X}\,\mathrm{E}[\boldsymbol{w}\boldsymbol{w}^T]\boldsymbol{X}^T + \mathrm{E}[\boldsymbol{\varepsilon}\boldsymbol{\varepsilon}^T] = \boldsymbol{X}\boldsymbol{X}^T + \sigma^2\boldsymbol{I}$. We can estimate σ^2 by empirical Bayesian maximization of $\log P(\boldsymbol{y}|\sigma^2)$, which can be done using a gradient-based nonlinear optimizer. We then plug in this estimate $\hat{\sigma}^2$ into the predictive distribution. As mentioned in Section 2.1, this is an example for a MAP approximation.

If the linear model is treated in a Bayesian way, many parallels to the classical treatment open up. [8] make this point in all details, here we just note some obvious ones. The maximum likelihood estimator for $\boldsymbol{w}$ is known to be obtained as a solution to Gauss' *normal equations*:

$$\hat{\boldsymbol{w}}_{ML} = \left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}\boldsymbol{X}^T\boldsymbol{y}, \tag{6}$$

if $\boldsymbol{X}^T\boldsymbol{X}$ has full rank. We note that if $\sigma^2 \to 0$, then the posterior mean $\mathrm{E}[\boldsymbol{w}] \to \hat{\boldsymbol{w}}_{ML}$, and the posterior covariance converges to $\boldsymbol{0}$ as $\sigma^2(\boldsymbol{X}^T\boldsymbol{X})^{-1}$. Therefore, as the noise variance goes to 0, we retrieve the maximum likelihood estimate together with the ML "asymptotic variance". It is known that noisy, high-dimensional data leads to stability problems with the ML estimator, through the presence of small eigenvalues in $\boldsymbol{X}^T\boldsymbol{X}$. The Bayesian method suggests that this may be due to non-accountance for the noise. In fact, the "trick" of replacing $\boldsymbol{X}^T\boldsymbol{X}$ by $\boldsymbol{X}^T\boldsymbol{X} + \sigma^2\boldsymbol{I}$ with a small $\sigma^2 > 0$ has been given a justification as *shrinkage estimator* [31] seemingly independent of the Bayesian view, although the underlying idea is the common one of having to control complexity in the presence of noise.

2.2.1 Data and Feature Dimensionality [*]

Recall that the linear model has feature (or weight space) dimensionality p and is fitted using n datapoints. Call n the data dimensionality. Some terms important for Bayesian analysis have been shown to be analytically tractable in Section 2.2, but for practical applications it is important to understand how these computations scale, and whether there are alternative forms which are cheaper and numerically more stable to compute. For the linear model, all relevant terms (posterior mean, predictive distribution, and marginal likelihood) can be computed in $O(\min\{p, n\}p\,n)$. We need the elementary Sherman-Morrison-Woodbury formula, [52], Sect. 2.7:

$$\left(\boldsymbol{A} + \boldsymbol{B}\boldsymbol{C}^{-1}\boldsymbol{D}\right)^{-1} = \boldsymbol{A}^{-1} - \boldsymbol{A}^{-1}\boldsymbol{B}\left(\boldsymbol{C} + \boldsymbol{D}\boldsymbol{A}^{-1}\boldsymbol{B}\right)^{-1}\boldsymbol{D}\boldsymbol{A}^{-1}, \tag{7}$$

which holds whenever all inverses exist. We also have that

$$|\boldsymbol{I} + \boldsymbol{U}\boldsymbol{V}| = |\boldsymbol{I} + \boldsymbol{V}\boldsymbol{U}|\,. \tag{8}$$

Recall that $\boldsymbol{M} = \boldsymbol{X}^T\boldsymbol{X} + \sigma^2\boldsymbol{I} \in \mathbb{R}^{p,p}$. The posterior of Eq. 3 and the predictive distribution of Eq. 4 are computed using the *Cholesky factorization*, [28], Sect. 7.2: $\boldsymbol{M} = \boldsymbol{L}\boldsymbol{L}^T$ where $\boldsymbol{L}$

is lower triangular. Computing the factorization costs $O(p^3)$ and is much more numerically stable than inverting $\boldsymbol{M}$. We have that $\boldsymbol{M}^{-1}\boldsymbol{x} = \boldsymbol{L}^{-T}\boldsymbol{L}^{-1}\boldsymbol{x}$, which can be computed in $O(p^2)$ using back-substitution. But what if $p > n$? Let $\boldsymbol{A} = \boldsymbol{X}\boldsymbol{X}^T + \sigma^2\boldsymbol{I} \in \mathbb{R}^{n,n}$. We can employ Eq. 7 to rewrite both expressions Eq. 3, Eq. 4, arriving at (exercise to the reader!)

$$P(\boldsymbol{w}|\boldsymbol{y}, \sigma^2) = N\left(\boldsymbol{X}^T\boldsymbol{A}^{-1}\boldsymbol{y}, \boldsymbol{I} - \boldsymbol{X}^T\boldsymbol{A}^{-1}\boldsymbol{X}\right)$$

and

$$P(y|x, D, \sigma^2) = N\left(\boldsymbol{y}^T\boldsymbol{A}^{-1}\boldsymbol{X}\boldsymbol{\phi}(x), \sigma^2 + \|\boldsymbol{\phi}(x)\|^2 - (\boldsymbol{X}\boldsymbol{\phi}(x))^T\boldsymbol{A}^{-1}\boldsymbol{X}\boldsymbol{\phi}(x)\right). \quad (9)$$

For later use, we note that Eq. 9 is written entirely in terms of inner products $\boldsymbol{\phi}(x)^T\boldsymbol{\phi}(x')$ for x, x', we never need to represent $\mathbb{R}^p$ vectors explicitly. Apart from that, the Cholesky factorization of $\boldsymbol{A}$ (but not of $\boldsymbol{M}$) is required.

The marginal likelihood is $P(\boldsymbol{y}|\sigma^2) = N(\boldsymbol{0}, \boldsymbol{A})$ from Eq. 5 and can be computed using the Cholesky factorization of $\boldsymbol{A}$. If $n > p$, we can use Eq. 7 once more to obtain $\boldsymbol{A}^{-1} = \sigma^{-2}(\boldsymbol{I} - \boldsymbol{X}\boldsymbol{M}^{-1}\boldsymbol{X}^T)$, furthermore Eq. 8 results in $|\boldsymbol{A}| = (\sigma^2)^{n-p}|\boldsymbol{M}|$. Therefore, the marginal likelihood can be computed in $O(np^2)$, requiring the Cholesky decomposition of $\boldsymbol{M}$.

To conclude, if data dimensionality n and feature dimensionality p are very different, the Bayesian computations can be done in time complexity $O(\min\{p, n\}^2 \max\{p, n\})$, scaling linearly in the larger one. The space complexity is $O(\min\{p, n\}^2)$, and the computations which scale linear in $\max\{p, n\}$ are in fact inner products of columns or rows of $\boldsymbol{X}$. Working in the smaller of the two dimensionalities is important not only for time and memory reasons, but also to ensure better numerical stability.

2.3 Occam's Razor in Bayesian Statistics

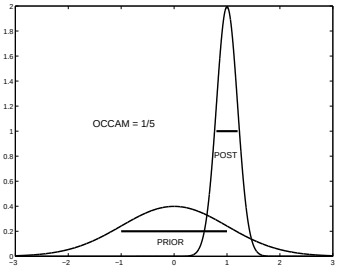
We already noted in Section 1 that a central trade-off to be faced when approaching statistical problems in a feasible way, is one between data fit and model complexity. Recall that in principle, Bayesian marginalization allows to circumvent the problem entirely: the predictive distribution makes use of *all* variations of the model, or even of a number of models, with the correct weighting being supplied by the posterior.

In practice, Bayesian marginalization is not feasible for all but very simple models. For example, in the linear model (Eq. 1) we cannot marginalize over σ^2 analytically. As noted in Section 2.1, a general idea is to replace the marginalization of a hyperparameter α by plugging in a maximum point of the *marginal likelihood* $P(D|\alpha)$. As with any estimation process, overfitting may arise as a problem if we do not somehow penalize "complicated" solutions (which always achieve a better fit than "simple" ones). However, we present some arguments why this particular empirical Bayes procedure in fact has a strong complexity control built in. The principle of *Occam's razor* in Science makes us prefer a simpler theory over a more complicated one (*i.e.* one that needs more basic axioms) if both explain observations equally well. To us, Occam's razor is no more than an empirical observation about how the world works, albeit an extremely powerful one. We follow [40] in motivating how Occam's razor is embodied in empirical Bayes. Suppose we have a model with parameters $\boldsymbol{w}$ and hyperparameter α. If the posterior $P(\boldsymbol{w}|D, \alpha)$ is sharply peaked around $\boldsymbol{w}_{MAP}$ with

covariance matrix $\mathbf{\Sigma}_{post}$, and the prior $P(\boldsymbol{w}|\alpha)$ spreads widely with some covariance matrix $\mathbf{\Sigma}_{prior}$, such that $\boldsymbol{w}_{MAP}$ does not lie in its extreme tails, we can write

$$P(D|\alpha) = \int P(D|\boldsymbol{w},\alpha)P(\boldsymbol{w}|\alpha)\,d\boldsymbol{w} \approx P(D|\boldsymbol{w}_{MAP},\alpha)\left(\frac{|\mathbf{\Sigma}_{post}|^{1/2}}{|\mathbf{\Sigma}_{prior}|^{1/2}}\right). \tag{10}$$

The proof is given at the end of this Section. The right hand side is the product of the likelihood for $\boldsymbol{w}_{MAP}$, measuring the goodness of fit of our guess $\boldsymbol{w}_{MAP}$ given α, and of the so called *Occam factor*. The latter measures the complexity of the model for fixed α. If the model is complicated, the Occam factor is small, because the prior mass must be spread over many values of $\boldsymbol{w}$, and the posterior is sharply peaked due to the specificity of the model.



On the other hand, for a simple model the Occam factor is large, because the prior needs to cover less options, and also the posterior has a larger spread, because more variants of the simple model are close to the best fit (which may not be very good). Therefore, maximizing $P(D|\alpha)$ implements a trade-off between good fit (large $P(D|\boldsymbol{w}_{MAP},\alpha)$) and low complexity (large Occam factor). Maximizing a marginal likelihood is often much more robust against overfitting than maximizing a joint one. The Bayesian paradigm states that latent variables should be marginalized rather than maximized over, and usually it is observed to be better in practice to do part of the job than not doing it at all. As a rule of thumb, one should preferentially integrate out variables which are directly related to the data, in the sense that their posterior is most strongly dependent on the data. In terms of a hierarchical model, variables at lower levels should preferentially be marginalized over.[7] In the example of the linear model, it is more sensible to integrate out $\boldsymbol{w}$ and maximize for σ^2, than vice versa (which is indeed possible as well, see Section 2.4).

We conclude that empirical Bayesian estimation embodies a strong notion of complexity control by maximizing the *marginal likelihood*, in contrast to classical maximum (joint) likelihood estimation where complex models are always preferred (there is no Occam factor). Techniques such as regularization or penalized ML (see Section 4.2) are used to remedy the classical techniques. However, other than Bayesian inference and marginalization, empirical Bayes is *not* a generally coherent procedure. It can fail badly, and one should always try to find independent justifications in a special case.

We finally provide the details for Eq. 10. Dropping the conditioning on α, we have that $P(D) = P(D|\boldsymbol{w})P(\boldsymbol{w})/P(\boldsymbol{w}|D)$ for any $\boldsymbol{w}$. We plug in $\hat{\boldsymbol{w}}_{MAP}$ and approximate the posterior by a Gaussian with the same mean and covariance matrix, which is justified if the posterior is sharply peaked. Thus, $P(D) \approx P(D|\hat{\boldsymbol{w}}_{MAP})P(\hat{\boldsymbol{w}}_{MAP})|2\pi\mathbf{\Sigma}_{post}|^{1/2}$. For a fairly flat prior, $|\mathbf{\Sigma}_{prior}|$ is large, and $P(\hat{\boldsymbol{w}}_{MAP}) \approx |2\pi\mathbf{\Sigma}_{prior}|^{-1/2}$ can be justified. Plugging this in, we obtain Eq. 10.

2.4 Sparse Bayesian Learning. Automatic Relevance Determination [*]

Automatic relevance determination (ARD) [47] is a powerful application of empirical Bayesian estimation relevant for Machine Learning. When applied to feature selection in

[7] This is merely a rule of thumb, and examples violating it can be constructed.

the linear model (Eq. 1), one obtains the framework of *sparse Bayesian learning* (SBL) [70]. Suppose that the p features contain many candidates, but we expect that only a small number of them are required to explain our data. In other words, we expect a good weight vector $\boldsymbol{w}$ to be *sparse*, in that most components $w_j = 0$. In SBL, this is addressed by placing a prior $P(\boldsymbol{w})$ on $\boldsymbol{w}$ which favours such sparse vectors. The prior $P(\boldsymbol{w}) = N(\boldsymbol{0}, \boldsymbol{I})$ of Section 2.1 does not have this property.[8] Let us introduce a hyperparameter λ_j for every component w_j of $\boldsymbol{w}$, and let $P(\boldsymbol{w}|\boldsymbol{\lambda}) = N(\boldsymbol{w}|\boldsymbol{0}, \operatorname{diag}\boldsymbol{\lambda})$. λ_j is the prior variance of w_j. Furthermore, the λ_j^{-1} are given independent identical Gamma hyperpriors: $P(\lambda_j^{-1}) \propto \lambda_j^{1-a} \exp(-b/\lambda_j)$. The important point to understand is that a, b are chosen s.t. most of the hyperprior mass is at small values of λ_j: effectively λ_j is pushed to attain very small values. Since λ_j is the (prior) variance of w_j around 0, this also forces w_j *a priori* to near-zero values. On the other hand, significant hyperprior mass is available for large λ_j as well, allowing some of the w_j to become large. This effect can be studied directly by observing that the marginal prior $P(\boldsymbol{w})$ (obtained by integrating out $\boldsymbol{\lambda}$) is a product of Student t distributions, one for each w_i. The reader is encouraged to consult the plots given by [70], who compares $P(\boldsymbol{w})$ with the conventional $N(\boldsymbol{0}, \boldsymbol{I})$ and motivates the sparsity-favouring characteristics of the former. Much of the mass lies in spines along the coordinate axes (for $p = 2$), *i.e.* in regions where most of the w_j are almost zero. This is not the case for $N(\boldsymbol{0}, \boldsymbol{I})$, where high mass regions are balls or shells around $\boldsymbol{0}$. The setup is an example for ARD where sparsity-favouring priors are placed on weight vectors, which lets inference make use of *relevant* components only (in light of the data), forcing irrelevant ones close to zero. ARD is quite different in spirit to other "combinatorial" feature selection techniques, such as greedy forward selection. In order to find a relevant subset, several iterations of inference with the "full" model (all features are relevant) have to be done within a smooth non-convex optimization for the relevance weights (the λ_j in our example).

A strict Bayesian treatment of this model would not even amount to the selection of features, since $\boldsymbol{w}$ and $\boldsymbol{\lambda}$ were integrated out for predictions. Note that selection can be seen as a form of estimation. However, while it is possible to marginalize over either of them in isolation, the marginalization over both is intractable. [70] suggests to integrate out $\boldsymbol{w}$, then to maximize the remaining marginal likelihood $P(\boldsymbol{y}|\boldsymbol{\lambda})$ w.r.t. $\boldsymbol{\lambda}$. On most datasets, this results in many of the λ_j becoming very small, whereby they can be eliminated from the model. SBL can be contrasted with other commonly used sparsity-favouring linear model techniques by considering the alternative option of integrating out $\boldsymbol{\lambda}$ and maximizing $\log P(\boldsymbol{y}, \boldsymbol{w})$ w.r.t. $\boldsymbol{w}$, equivalent to maximizing the sum of the log likelihood and a penalty $\log P(\boldsymbol{w})$ which is a sum of Student t distributions. Related techniques are obtained by replacing this penalty by $\|\boldsymbol{w}\|_1 = \sum_j |w_j|$ or by $\|\boldsymbol{w}\|_0 = \sum_j \mathrm{I}_{\{w_j \neq 0\}}$. Theoretical arguments suggest that $\|\cdot\|_0$ penalization leads to sparsest solutions, then Student t, then $\|\cdot\|_1$. On the other hand, for $\|\cdot\|_1$ penalization the search for $\boldsymbol{w}$ is a simple convex problem (with a unique solution and no local minima), while for $\|\cdot\|_0$ penalization the problem is NP hard in general. [77] show that at least for the noiseless case ($\sigma^2 \to 0$ in the linear model), a corresponding limit of the SBL technique has far fewer local minima one can get trapped in than direct local $\|\cdot\|_0$ minimization methods, while the methods have the same global minimum. This strongly suggests that the maximization of $P(\boldsymbol{y}|\boldsymbol{\lambda})$, while being a hard non-convex problem, is less prone to getting trapped in a shallow local minimum than direct local "hill-climbing" methods which focus on maintaining a small $\|\boldsymbol{w}\|_0$.

[8] In high dimensions, most of the mass of $N(\boldsymbol{0}, \boldsymbol{I})$ lies in a thin shell at radius $\approx \sqrt{p}$.

2.5 Markov Chain Monte Carlo [*]

In Section 2.1, we have already encountered a case where high-minded Bayesian inference got stuck with an intractability (of marginalization over σ^2). In fact, the basic linear model is very special in that some of the basic inference tasks are tractable, but what if there are uncertainties in the features, or in hyperparameters of the prior distributions? It is fair to say that the value of the Bayesian framework to practitioners would be very low indeed, were it not for a very general and powerful computational technique with which the required marginalizations can be approximated: *Markov chain Monte Carlo* (MCMC). Originated in Physics, this idea is used in many fields beyond Statistics. We cannot provide more than very basic ideas here. There is a bewildering array of more or less specialized techniques, and certainly none of them can be recommended uniformly, even for small classes of models. MCMC in Statistics is far from a black box method and often requires considerable expert knowledge to even assess basic validity of results. On the other hand, MCMC is converging to the correct answer in the limit of large running time, which is not the case for most other approximate inference techniques. A very good and detailed overview of MCMC is given by [46], see also [19]. BUGS [66] is a general system which allows the graphical specification of a model, for which inference is approximated using Gibbs sampling.

Marginalization boils down to computing integrals $I = \int f(\boldsymbol{x})p(\boldsymbol{x})\,d\boldsymbol{x}$, where $p(\boldsymbol{x})$ is a pdf. For discrete $\boldsymbol{x}$, replace the integral by a sum. If we had an i.i.d. sample $\{\boldsymbol{x}^{(i)}\}$ from p, and f is well-behaved, then $n^{-1}\sum_i f(\boldsymbol{x}^{(i)})$ converges to I in probability with a rate of about $O(n^{-1/2})$. This is called a Monte Carlo estimate, and MCMC is a technique to obtain such a sample $\{\boldsymbol{x}^{(i)}\}$. To this end, we start from an arbitrary value $\boldsymbol{x}_1$ and apply randomized transitions $\boldsymbol{x}_t \to \boldsymbol{x}_{t+1}$ until after some T steps we can be sure that the distribution of $\boldsymbol{x}_T$ is close to p. Repeating this process for independent starting values, we obtain the desired sample. For MCMC, $(\boldsymbol{x}_t)$ is a Markov chain with homogeneous transition kernel $T(\boldsymbol{x}_*|\boldsymbol{x})$, denoting the conditional pdf of $\boldsymbol{x}_{t+1}$ given $\boldsymbol{x}_t$, for any t. The kernel has to fulfil the somewhat opposite properties of ergodicity and nonperiodicity (ensuring that all transitions between any two states do have positive probability) and of leaving the target density $p(\boldsymbol{x})$ invariant, meaning that if $\boldsymbol{x}_t \sim p$, then $\boldsymbol{x}_{t+k} \sim p$ for all $k \geq 0$. Since any kernel is a contraction (by definition of being a pdf), the MC will converge against its unique limit (or stationary) distribution p, meaning that for large T, the distribution of $\boldsymbol{x}_T$ is ever closer to p. *Detailed balance* is a simple criterion implying invariance for p: we require that $T(\boldsymbol{x}_*|\boldsymbol{x})p(\boldsymbol{x}) = T(\boldsymbol{x}|\boldsymbol{x}_*)p(\boldsymbol{x}_*)$ for all $\boldsymbol{x}, \boldsymbol{x}_*$.

Maybe the most general way of constructing such a kernel $T(\boldsymbol{x}_*|\boldsymbol{x})$ for given $p(\boldsymbol{x})$ is the *Metropolis-Hastings* (MH) technique [44, 23]. We need some conditional proposal distribution $q(\boldsymbol{x}_*|\boldsymbol{x})$. For the step $\boldsymbol{x}_t \to \boldsymbol{x}_{t+1}$, sample $\boldsymbol{x}_* \sim q(\cdot|\boldsymbol{x}_t)$, compute

$$\alpha = \min\left\{1, \frac{q(\boldsymbol{x}_t|\boldsymbol{x}_*)p(\boldsymbol{x}_*)}{q(\boldsymbol{x}_*|\boldsymbol{x}_t)p(\boldsymbol{x}_t)}\right\}.$$

Now, with probability α accept $\boldsymbol{x}_{t+1} = \boldsymbol{x}_*$, otherwise reject $\boldsymbol{x}_*$ and let $\boldsymbol{x}_{t+1} = \boldsymbol{x}_t$. This update satisfies detailed balance for p. Note that if $q(\boldsymbol{x}_*|\boldsymbol{x}) = q(\boldsymbol{x}|\boldsymbol{x}_*)$ (symmetry), then $\alpha = \min\{1, p(\boldsymbol{x}_*)/p(\boldsymbol{x}_t)\}$, so states of higher density than the current state are always accepted, while states of lower density are rejected with probability $1 - p(\boldsymbol{x}_*)/p(\boldsymbol{x}_t)$. There is a tendency to seek regions of higher p volume (or lower p energy). It is important to note that in order to use the MH rule, $p(\boldsymbol{x})$ has to be known only up to a multiplicative

constant. For example, in Bayesian marginalization p is the posterior proportional to the joint distribution of data and parameters, and the latter can often be computed easily, while the normalization is intractable.

A special case of the MH rule is very popular in Machine Learning under the name of *Gibbs sampling* [17]. Suppose that $\boldsymbol{x} = (x_j)_j$, and denote $\boldsymbol{x}_{-i} = (x_j)_{j \neq i}$. Assume that the full conditional densities $p(x_i|\boldsymbol{x}_{-i})$ can all be sampled from feasibly. A single Gibbs sweep starts from $\boldsymbol{x}_* = \boldsymbol{x}_t$, runs over the components in some order, replacing $x_{*,i}$ by a sample from $p(\cdot|\boldsymbol{x}_{*,-i})$, finally setting $\boldsymbol{x}_{t+1} = \boldsymbol{x}_*$. Note that there is no rejectance here, and that the full conditionals serve as natural proposal distributions.

The main problem with MCMC in practice is to assess when convergence of the chain to p or a distribution very close by has occurred. For simple, yet nontrivial methods, the idea of exact sampling [53] provides an exact test, but in general for a complicated model one can never be sure. A key idea for many convergence diagnostics is the notion that the chain has converged once all information about the starting state $\boldsymbol{x}_1$ is lost. A good sampler takes into account properties of the model and the data. For some applications, it makes sense to analytically marginalize over parts of $\boldsymbol{x}$ if that is possible (*Rao-Blackwellization*; the variance in the sample is reduced that way). For others, it pays to *extend* the state space by auxiliary variables $\boldsymbol{a}$, and jointly sample $(\boldsymbol{x}, \boldsymbol{a})$, which can help faster exploration by allowing for bolder transitions (w.r.t. $\boldsymbol{x}$) to be accepted more frequently. The auxiliary variables can also be used to allow the resampling of $\boldsymbol{x}$ given $\boldsymbol{a}$ to somehow adapt to local properties of p (which is important to prevent inefficient random walk exploration) while maintaining detailed balance for the joint $(\boldsymbol{x}, \boldsymbol{a})$ chain.

3 Latent Variables. Graphical Models

In this Section, we will describe some general techniques of how to build more complicated and useful models from simple ingredients. We will also see that Bayesian computations such as inference and learning can be treated in a powerful common framework.

3.1 Graphical Models

The area of graphical models is of central importance today in Statistics, Machine Learning, Statistical Physics, and Information Theory, and whole conferences (such as UAI) are devoted to this field. We cannot provide more than a simple flavour, but refer the reader to [34, 32, 39, 51, 33]. We will be concerned mainly with directed graphical models (Bayesian networks), alluding to undirected ones (Markov random fields) only briefly in Section 3.5, and skipping other variants such as chain graphs.

If you scribble a model on a piece of paper, chances are you will draw something like a Bayesian network. You will probably draw a graph with variables as nodes, and then represent direct causal relationships by arrows (see Figure 2, left). If a node (variable) V has parents (nodes with outgoing edges pointing to V) $W_1, \ldots, W_k$, you probably intend to say that one knows how to sample a value for V, given that values for $W_1, \ldots, W_k$ are known (and values of other nodes are not required). In probability speech:

$$P(V|\text{all others}) = P(V|W_1, \ldots, W_k)$$

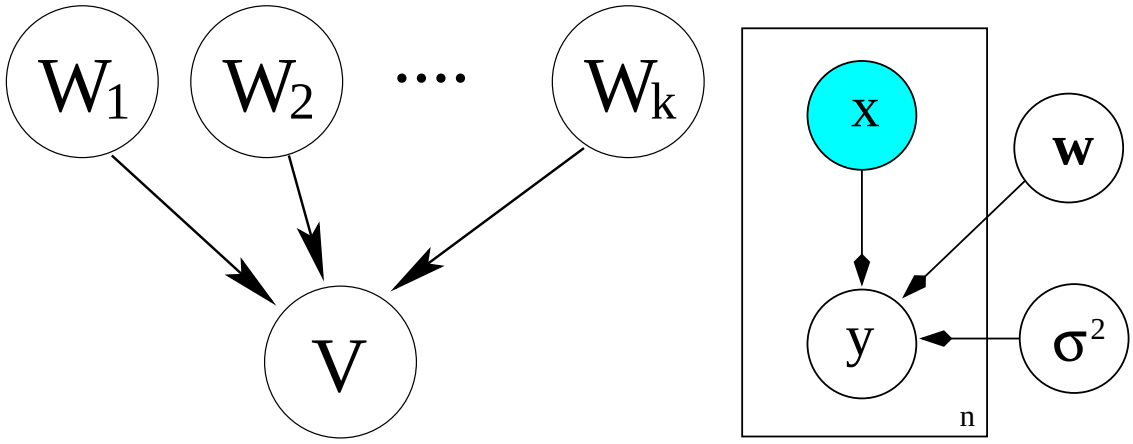


Figure 2: Left: Node V and its parents W_j in a Bayesian network. Right: The linear model as Bayesian network. The CPT $P(y|x, \boldsymbol{w}, \sigma^2)$ is $N(\boldsymbol{w}^T \boldsymbol{\phi}(x), \sigma^2 \boldsymbol{I})$, and $P(\boldsymbol{w}) = N(\boldsymbol{0}, \boldsymbol{I})$. ε is a pure nuisance variable and has been pruned from the model.

You will just have re-invented graphical models then! The Bayesian network graph for the linear model is given in Figure 2, right. Given a graph, a model is completely specified by fixing a *conditional distribution* $P(V|W_1, \ldots, W_k)$ for every node V with parents $W_1, \ldots, W_k$ (historically called *conditional probability table*, CPT). An exception are *covariates* (variables which are always known), whose CPTs do not have to be specified. x is a covariate in the linear model. The box in Figure 2, right, is a *plate*: the variables inside a plate are sampled i.i.d., given the variables projecting into it. Observed variables are typically represented by shaded nodes.

Graphical models are immensely useful for designing and communicating models. Surprisingly, they also allow for a unified treatment of common statistical tasks such as Bayesian inference (computing posteriors for latent variables) or reading off implied conditional independencies. Inference can be done by clustering nodes in a certain way, then passing "messages" (conditional distributions) between the node clusters. The complexity of inference can be read off this cluster graph. For certain classes of models, all nodes are discrete and all CPTs are really given as tables. For others, such as our linear one (or multi-layer perceptrons), the CPTs are represented by (complex) functional relationships, and the graphical model view is maybe less useful. However, it is of immense use to understand all models in a common framework. We will give an example for this advantage through unification in the Sections to follow.

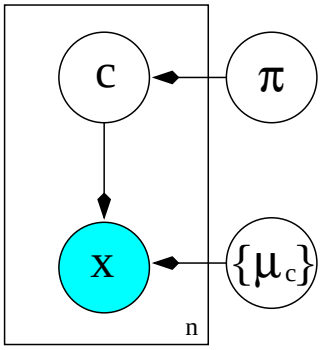
3.2 Mixture Models. EM Algorithm

Statisticians have developed and studied a wide range of elementary distribution families, such as the Gaussian, Gamma, multinomial, Poisson, Student t,*etc*, whose characteristics are very well known. But how to model a variable x which shows non-elementary characteristics? For example, what if we know that the density of x has more than one mode? We can build complicated distributions from simple ones by introducing new latent variables. In almost all cases, such latent variables are nuisance: we "invented" them to make the model more expressive, so there is no gain in obtaining explicit information about them.

For example, we could introduce a discrete $c \in \{1, \ldots, C\}$ and assume that $P(x|c) = N(\mu_c, \sigma^2)$, $P(c) = \pi_c$. Marginally we have

$$P(x) = \sum_{c=1}^{C} P(x|c)P(c) = \sum_{c=1}^{C} \pi_c N(x|\mu_c, \sigma^2), \qquad (11)$$

a *mixture distribution*. Note that $P(x)$ can be much more complicated than a single Gaussian. It can have multiple modes, and if C is large enough, a very large class of distributions can be well approximated by a Gaussian mixture.



In fact, it is safe to say that almost all useful model building can be reduced to combining elements from a rather small "statistical toolbox" by way of introducing convenient latent variables, and using the "language" of graphical models. The step from an elementary family (say, Gaussians) to a mixture of these is a very useful template for model building and has been used frequently in Machine Learning, even in order to extend the linear model [35]. A standard textbook on mixture distributions is given by [72].

Latent variables introduced for the purpose of a better model are nuisance, and the correct way of dealing with them is to marginalize over them. The *expectation maximization* (EM) algorithm [15] can be used to maximize the marginal likelihood for a mixture model. It is a special case of a *variational technique* (see Section 3.2.1), and can be motivated as follows. We have i.i.d. pairs (x_i, c_i), where c_i is latent. If c_i were known, we could write $q^*_{i,c} = \mathrm{I}_{\{c_i=c\}}$ (here, $\mathrm{I}_A = 1$ if A is true, $\mathrm{I}_A = 0$ otherwise), and the ML estimate would simply be

$$\mu_c = (\sum_i q^*_{i,c})^{-1} \sum_i q^*_{i,c} x_i, \quad \pi_c = n^{-1} \sum_i q^*_{i,c}. \qquad (12)$$

Note that $\boldsymbol{q}^*_i = (q^*_{i,c})_c$ are deterministic distributions. If the c_i are completely unknown, we can adopt a "Münchhausen strategy" of bootstraping. We first use the current setting $\{\mu_c, \boldsymbol{\pi}\}$ in order to estimate the $\boldsymbol{q}^*_i$ by the posterior marginals:

$$q_{i,c} \leftarrow P(c_i = c|x_i, \boldsymbol{\mu}, \boldsymbol{\pi}) = \frac{N(x_i|\mu_c, \sigma^2)\pi_c}{\sum_{c'} N(x_i|\mu_{c'}, \sigma^2)\pi_{c'}}.$$

The computation of $\boldsymbol{q}_i$ is known as *E step*. Next, we plug in these estimates for the true $\boldsymbol{q}^*_i$ into Eq. 12, obtaining new parameters. This is the *M step*. EM involves iterating E and M steps until convergence. We show in Section 3.2.1 that EM converges monotonically to a local maximum of the marginal likelihood. Note that we can also maximize the marginal likelihood by gradient ascent, the gradient computation involves marginal inference like in E steps. The advantage of EM in practice lies in the M steps being very simple to execute. For example, if the corresponding ML estimator is analytically tractable for all c_i being known, the M steps come in exactly the same form after some reweighting of the data.

We close this Section by a very important remark which transcendents the area of graphical models, and for which EM is but a simple example. *Learning* parameters by maximum likelihood or related Bayesian approximations can in general be reduced to *marginal inference* for the latent nuisance variables. Here, "marginal" means over *cliques* (maximal fully connected components) of the model. This fact remains true for undirected graphical models (see Section 3.5), but we need marginal inference over observed response variables as well. In general, the key to successful parameter learning is a good approximation of marginal inference.

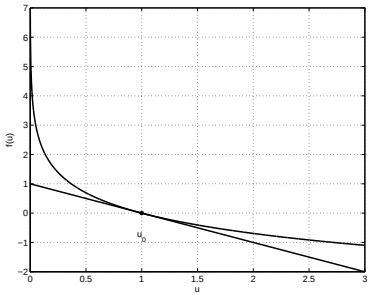
3.2.1 Variational Mean Field [*]

In this Section, we show why the EM technique of Section 3.2 works, and how the idea can be generalized in order to obtain variational mean field approximations to learning in the presence of latent variables. [74] gives a wide and deep overview of variational approximations, see also [33]. In a nutshell, a variational technique solves a problem of inference or marginalization (approximately) by phrasing it as an equivalent optimization problem (a relaxed version of) which is solved. Many variational techniques are based on *convexity properties*. A good elementary book on convexity and convex optimization is given by [9], while [56] contains a wealth of material.

Recall that the EM technique is used to minimize the negative log likelihood of data $\{x_1, \ldots, x_n\}$ for the mixture model of Eq. 11. The criterion is a sum of terms $-\log \sum_c \pi_c P(x_i|c)$. If we could pull $\sum_c$ outside, this would be a standard ML estimation problem which we know how to solve. We use a property of $-\log u$ called *convexity*.

$f(u)$ is convex if at any u_0 one can find a *linear lower bound* on f which touches at u_0:

$$f(u) \geq g(u_0)(u - u_0) + f(u_0). \tag{13}$$



Clearly, $-\log u$ is convex on its domain $u > 0$. If f is convex, we have *Jensen's inequality*:

$$\mathrm{E}[f(u)] \geq f(\mathrm{E}[u])$$

This holds for any distribution of u with support contained in the domain of f. It is easy to see from Eq. 13 by setting $u_0 = \mathrm{E}[u]$: $\mathrm{E}[f(u)] \geq \mathrm{E}[g(u_0)(u - u_0) + f(u_0)] = f(u_0)$. As for our marginalization problem, we have

$$\log \sum_c \pi_c P(x_i|c) = \log \sum_c q_{i,c} \frac{\pi_c}{q_{i,c}} P(x_i|c) = \log \mathrm{E}_{\boldsymbol{q}_i} \left[\frac{\pi_c}{q_{i,c}} P(x_i|c) \right] \geq \sum_c q_{i,c} \log \frac{\pi_c P(x_i|c)}{q_{i,c}}$$

for any distribution $\boldsymbol{q}_i$ over $\{1, \ldots, C\}$, using Jensen's inequality with $f = -\log$ and $\mathrm{E}_{\boldsymbol{q}_i}[\cdot]$. We can maximize the lower bound (thus tighten the inequality) by choosing $q_{i,c} \propto \pi_c P(x_i|c)$, thus $q_{i,c} = P(c_i = c|x_i)$, the marginal posterior. Since the bound touches the true log marginal likelihood at these $\boldsymbol{q}_i$, it is clear that the latter is never decreased in M steps. In theory, convergence of EM may be hampered by the marginal likelihood being unbounded above, but in practice this poses no problems. Since the bound and the log marginal likelihood have the same gradient at the current parameters, convergence of EM implies that a local maximum of the marginal likelihood is attained.

More generally, we have latent variables $\boldsymbol{h}$ and observed variables $\boldsymbol{o}$, and Jensen's inequality gives

$$\log P(\boldsymbol{o}) \geq \mathrm{E}_Q[\log P(\boldsymbol{o}, \boldsymbol{h})] + \mathrm{E}_Q[-\log Q(\boldsymbol{h})]. \tag{14}$$

This bound is maximized by choosing $Q(\boldsymbol{h}) = P(\boldsymbol{h}|\boldsymbol{o})$, the posterior, but it holds for *any* Q. In a generalization of EM, we maximize the bound w.r.t. Q in the E step, and then maximize the lower bound w.r.t. model parameters in the M step. We see from Eq. 14 that ML for known $\boldsymbol{h} = \boldsymbol{h}^*$ and the M step are closely related: the former uses $Q = \delta_{\boldsymbol{h}^*}$.

The E step in EM for mixture models is simple, because the posterior over the c_i factorizes and can be computed tractably. In general, this is not the case, and we may have to settle for an approximate bound maximizer Q. The *variational mean field*[9] approximation works by allowing for factorized variational distributions Q only (completely in all components, or in blocks of components). We can maximize the bound for any single of the Q factors, keeping all others fixed, and the E step involves iterating this until convergence.[10] *Structured variational* approximations follow a different line, in that they constrain Q to lie in a simpler distribution family, for example with a graphical structure which allows for tractable marginal inference (for example: trees, see Section 3.3). Note that the property of EM to attain a local maximum of the true marginal likelihood is lost in general if we use posterior approximations Q only. The link between EM and variational approximations has been introduced to Machine Learning by [26]. EM is an instance of the class of alternating minimization procedures which has been studied in Information Theory [14].

3.3 Hidden Markov Models. Belief Propagation

How do we model data with a dependency structure? Suppose we observe a sequence $(x_1, \ldots, x_T)$, where it does not make sense to assume the single x_t are independent given some parameters. For example, the sequence could be the acoustic recording of a spoken sentence, a nucleotide sequence of a chromosome, or a time series of weather-related measurements. We might observe a number of independent sequences of potentially different length, but within a sequence we would like to model some temporal or spatial coherence. A powerful and frequently used model in Machine Learning is the *Hidden Markov model* (HMM) [54]. The HMM is maybe the simplest model for which the full power of the Bayesian network view can be demonstrated, and in fact methods such as inference by message passing were developed for HMMs long before the general view was available. Its simplicity implies that one can fit HMM parameters over very large datasets efficiently, which has lead to enormously successful applications. State-of-the-art systems for large vocabulary continuous speech recognition use HMMs as backbone. The HMM is the standard model used to compare and analyze nucleotide sequences from genomes.

We introduce a corresponding sequence of latent variables $(c_1, \ldots, c_T)$, $c_t \in \{1, \ldots, C\}$ is the latent state for observation x_t. We assume some emission probabilities $P(x_t|c_t)$ which do not depend on t but only on the value of c_t, for example $P(x_t|c_t) = N(x_t|\mu_{c_t}, \sigma^2)$. If all (x_t, c_t) were assumed independent, this would just be the mixture model introduced above. The HMM is obtained by inserting edges $c_{t-1} \to c_t$ as well, together with some transition probabilities $P(c_t|c_{t-1})$ again assumed to be independent of t. In other words, the c_t sequence is modelled by a (first order stationary) *Markov chain*. The transition from mixture model to HMM is depicted in Figure 3. Note that the Markov assumption applies to the latent state sequence, while the observed sequence can have a much more complicated behaviour for large state spaces.

[9] The term "mean field" comes from Physics and alludes to the fact that by making a factorization assumption, we ignore dependencies between the latent components, replacing the posterior by its marginal field.

[10] A subtle issue is that the E step is not a convex optimization problem, so that the iterative block updates can get trapped in a local maximum which is not a global maximum of the bound w.r.t. Q, subject to the factorization constraints. See [74].

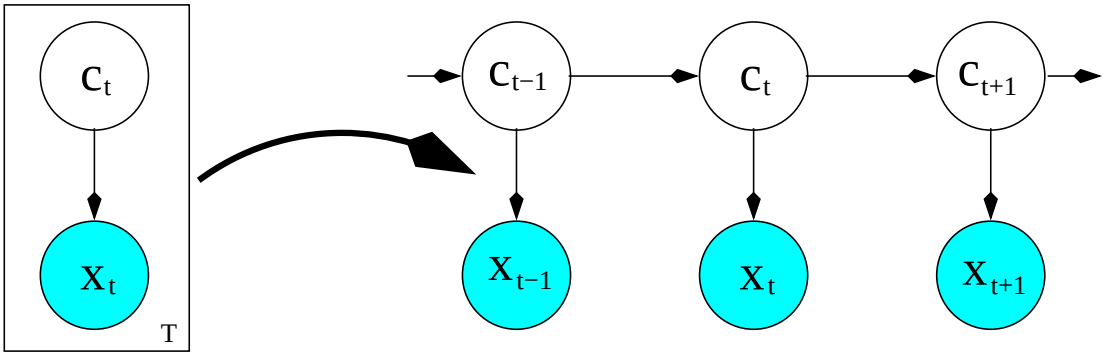


Figure 3: Illustrates dynamic Markovian extension of the mixture model to the Hidden Markov model.

How do we learn the parameters of an HMM, *i.e.* the μ_c and transition probabilities in our example? Again, we would like to maximize the marginal likelihood of the observed sequence $(x_1, \ldots, x_T)$ only, marginalizing over the c_t. We can use the EM algorithm in the same way as above, however what we need to compute in the E step are the posterior marginals[11] $P(c_t, c_{t+1}|x_1, \ldots, x_T)$. In the independent mixture model, these reduce to $P(c_t|x_t)$ factors due to the independence. It turns out that all these marginals $P(c_t, c_{t+1}|x_1, \ldots, x_T)$ can be computed in time $O(T\,C^2)$ using the *belief propagation* (BP) algorithm. We will motivate this powerful method in the sequel, but have to refer to the graphical models literature cited above for any details.

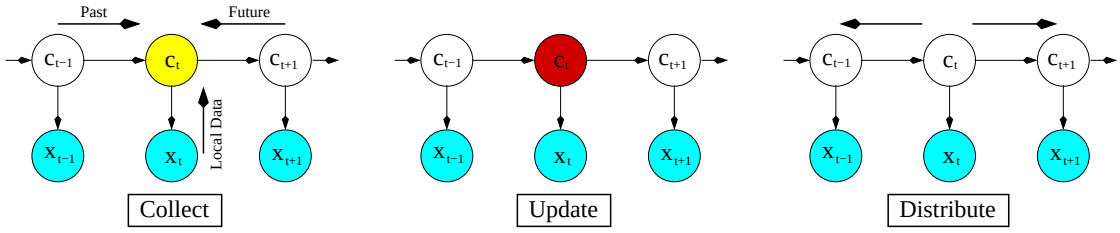


Figure 4: Illustration of a local message passing update at node c_t in belief propagation. Messages are collected from all neighbors. The local belief is updated. New messages are distributed towards all neighbors. The message that c_t sends to c_{t+1} depends on the messages c_t received from all neighbors *except* c_{t+1}.

The graph of the latent variables of an HMM as Bayesian network is a chain, which is a special case of a *tree*, being a graph without cycles.[12] The tree is the basic information structure which enables dynamic programming, which is the general principle underlying BP. Pick any node c_t in a tree. The subgraphs starting from the different neighbours of c_t are trees again, and they are disjoint. Therefore, it seems possible that many functions global to the whole tree, but centered on c_t, can be computed in a "divide-and-conquer" fashion: compute them for the neighbours of c_t, then combine the results. This idea is implemented by passing *messages* (local information representations) along the structure of the graph, a

[11] Recall from the end of Section 3.2 that learning needs inference for the latent variable *cliques*, which are pairs c_t, c_{t+1} in the HMM graphical model.

[12] We can ignore the observed variables in this argument, because each x_t is connected to a different c_t.

single local message update is depicted in Figure 4. The semantics of a message $c_t \to c_{t+1}$ is the belief of c_t of what c_{t+1} should be, this belief is based on messages to c_t of all neighbors *except* c_{t+1}.

For a tree-structured graphical model, BP converges in a single outward and inward sweep starting from an arbitrary root and results in all clique marginals. The application to HMMs is known as *Baum-Welch* algorithm and was in fact the first instance of BP to be proposed. Apart from marginal inference needed for learning, one is also interested in the most likely state sequence for an observed sequence, this is an MAP problem. Interestingly, we can use a simple variant of BP, the *max-product algorithm*, in order to solve this problem with the same complexity (for HMMs, this is known as *Viterbi decoder*). All we need to do is to replace product (sum) in BP by maximum (product), together with some bookkeeping. BP and max-product are instances of a more general distributive law [1]. If the graphical model is not a tree, BP may not converge, and if it does, the marginals are only approximations to the true posterior. However, this particular approximation, known as *loopy belief propagation*, can be very accurate in challenging cases, and has received widespread applications in Machine Learning and coding theory, also due to its simplicity of implementation. Meanwhile, loopy BP has been characterized as a variational approximation, and higher order generalizations are available, we refer to [74] for details.

3.4 Continuous Latent Variables

Another important class of graphical models make use of the idea of a latent low-dimensional state. Suppose we want to model the distribution of $\boldsymbol{x} \in \mathbb{R}^d$, where d is fairly large. A model such as $\boldsymbol{x} \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ makes sense only if we have enough independent data in order to be able to estimate $\boldsymbol{\Sigma}$ reliably. A linear latent state model assumes that $\boldsymbol{x} = \boldsymbol{\mu} + \boldsymbol{A}\boldsymbol{u} + \boldsymbol{\varepsilon}$, where $\boldsymbol{u} \in \mathbb{R}^l$ is the latent state, $l < d$, $\boldsymbol{A} \in \mathbb{R}^{d,l}$ and $\boldsymbol{\mu} \in \mathbb{R}^d$ are parameters, and $\boldsymbol{\varepsilon}$ is independent noise. Different models arise with the specifications of $P(\boldsymbol{u})$ and $P(\boldsymbol{\varepsilon})$.

If $P(\boldsymbol{u}) = N(\boldsymbol{0}, \boldsymbol{I})$, we have a *linear Gaussian latent state model* (LGLSM). Examples include *probabilistic PCA* for $P(\boldsymbol{\varepsilon}) = N(\boldsymbol{0}, \sigma^2\boldsymbol{I})$ and *factor analysis* (FA) for $P(\boldsymbol{\varepsilon}) = N(\boldsymbol{0}, \boldsymbol{\Psi})$, where $\boldsymbol{\Psi}$ is diagonal. Note that such linear Gaussian models do specify a marginal Gaussian distribution $P(\boldsymbol{x})$, however the covariance matrix is of the restricted form $\boldsymbol{A}\boldsymbol{A}^T + \boldsymbol{\Psi}$, where $\boldsymbol{\Psi}$ is diagonal. By controlling l, we can estimate or infer these parameters even for small datasets. If we fit $\boldsymbol{A}$ by maximum likelihood or estimate it in a Bayesian way, the main directions of variation in the data will be extracted. In fact, [71] showed that the ML estimate for a probabilistic PCA model is equivalent to the result of the *principal components analysis* (PCA) technique, a method very widely used for dimensionality reduction, visualization, and preprocessing for data analysis.[13] By viewing PCA as a graphical model, a host of interesting and useful extensions could be explored, and a better understanding of the traditional PCA method was gained. Furthermore, the graphical models view allows for generalizations to be conceived easily. For example, by introducing *both* a discrete latent variable c and a continuous one $\boldsymbol{u}$, we obtain mixture of factor analyzers models [18] which are among the most powerful density estimation techniques used in Machine Learning. Our presentation here is based partly on [57].

[13]The equivalence holds for $\sigma \to 0$, while the ML solution for finite σ^2 is given in terms of a *damped* eigendecomposition of the data covariance matrix.

If we assume instead that $P(\boldsymbol{u})$ factorizes, *i.e.* $P(\boldsymbol{u}) = \prod_j P(u_j)$, we obtain models used for *independent components analysis* (ICA) [2]. Such models are of use to "unmix" linear combinations of sources whose statistics are not Gaussian. There are a host of parametric ICA methods which basically perform ML estimation of $\boldsymbol{A}$ under certain distributional assumptions on the marginals $P(u_j)$. Ideally, one would like to use "model free" nonparametric approaches (see Section 4.2) in order to be able to estimate $\boldsymbol{A}$ for a wide range of source distributions. On the other hand, characteristics such as temporal or spatial structure in the sources may be present, and modelling such prior knowledge typically leads to better unmixing algorithms. We note that if $\boldsymbol{u}$ lies in a finite set, *i.e.* the support of $P(\boldsymbol{u})$ is finite, the Gaussian mixture model of Section 3.2 arises as a special case of a LGLSM. However, there are technical and didactical reasons to clearly distinguish between discrete and continuous variables in graphical models.[14]

Recall that in Section 3.3, we obtained HMMs as dynamic generalization of mixture models via a simple extension of the model graph. Given that LGLSMs and mixture models are closely related, it is natural to ask whether we can generalize the LGLSM accordingly. Indeed, this is possible along exactly the same lines. Namely, assume that successive latent states $\boldsymbol{u}_t$ form a Markov chain with linear Gaussian transition dynamics: $P(\boldsymbol{u}_t|\boldsymbol{u}_{t-1}) = N(\boldsymbol{u}_t|\boldsymbol{B}\boldsymbol{u}_{t-1}, \boldsymbol{\Lambda})$, where $\boldsymbol{B}$ is invertible. The parameters can now be learned by EM, where the E step requires inference for the $P(\boldsymbol{u}_t, \boldsymbol{u}_{t+1}|\text{all data})$. These marginals are Gaussian and can be computed using exactly the same message passing scheme as for the HMM, again an instance of BP. In fact, once more the whole setup was known long before in the control literature as *Kalman filter* or *Kalman smoother* [36]. It is very instructive to compare the Baum-Welch algorithm for HMMs with Kalman smoothing and to recognize that they are simply two variants of the same underlying BP scheme, the reader is encouraged to do so and to fill in the details.

3.5 Undirected Graphical Models [*]

Another important class of graphical models are *undirected* ones, also known as *Markov random fields* (MRF). The underlying graphs have undirected edges, and nonnegative potential functions are placed on the cliques of the graph, they do not have to be normalized conditional distributions. The distribution represented by an MRF is obtained by multiplying all potentials and dividing through a normalization constant Z (also called partition function). Conditional independence can be read off a MRF easily: variable groups A and B are independent given C, if every path from A to B goes through C. The Hammersley-Clifford theorem shows that for each undirected graph, the distributions with these conditional independence constraints are exactly the ones obtained by placing potentials on the cliques [6]. MRFs are of central importance in low-level vision, where the task is to associate a label to each pixel or local patch in a bitmap. MRFs for this task are distributions over the label field, the observed pixels are covariates. Note that the Gibbs sampling scheme (see Section 2.5) was introduced for MRFs by [17]. MRFs originate from Statistical Physics [29], where they are used to study spin systems such as magnets and spin glasses. In Machine Learning, MRF variants are known as *Boltzmann machines* [27], *energy-based models*, or *products of experts* [25].

[14] For example, the statement (made in Section 3.2) that inference is efficient in a tree-structured graphical model is true in general only if the model does not contain a mix of discrete and continuous latent variables.

The reader may wonder why two different notions of graphical models are required. Are directed graphical models not powerful enough to encompass everything we need? The answer is that there are practically important situations which can naturally be written as an MRF, while formalizing them as causal directed Bayesian network is awkward and may require more constraints than necessary,[15] and vice versa. MRFs allow to softly enforce local constraints by choosing the clique potentials appropriately, *without* having to specify a causal generative process for the variables. In situations such as low-level vision, specifying a causal model for the generation of a natural scene is extremely hard, while important characteristics such as spatial coherence of texture and colour or edge features can be modelled easily using potentials. In this respect, MRFs are more closely related to random field models (see Section 5).

Recall from Section 3.2 that for Bayesian networks, we can typically reduce parameter learning to marginal inference for the latent variables. If all variables in a directed model are observed, ML estimation is straightforward. However, learning of potential parameters in a MRF requires us to perform marginal inference over the response variables of the model (the labels in the low-level vision example), even though they are given for the training cases. This is due to the presence of the partition function, whose computation is typically intractable. Belief propagation (see Section 3.3) works for MRFs as well,[16] and loopy BP and its higher order extensions are standard methods for MRFs nowadays. Alternatively, the idea of *brief Gibbs sampling* [25] for learning is successful and very fast in practice, although its properties are maybe less well understood. Very powerful MCMC methods for MRF inference have been imported to Machine Learning from the Statistical Physics community [68]. Alongside marginal inference, the MAP problem (finding the most likely reponses for given covariates) is hard for general MRFs. This energy minimization problem has been studied extensively in Machine Vision, and very efficient relaxations[17] of the problem are available which make use of graph cuts [10]. When comparing MAP and conditional inference, the former can often be computed or approximated more efficiently. *Large margin learning* is a way of learning potential parameters using MAP instead of conditional inference in the inner loop [69]. While MRFs are typically applied to graphs such as grids, which feature hugely many cycles, they have also been applied to tree graphs where conditional inference (and therefore learning) is tractable and efficient (see Section 3.3). This variant is known as *conditional random field* (CRF) [38]. CRFs (conditional, undirected) can be compared directly to the joint, directed HMM. Both have a chain structure, whereby conditional inference is efficient, but they have significantly different characteristics (global versus local normalization, see [38]). CRFs are instances of *log-linear models*, in the sense that their log potentials are linear functions of the parameters. ML (or MAP) estimation for a log-linear MRF is a convex problem with a unique solution which can be found efficiently, given that marginal inference is tractable (see Section 4.1 for the role of convexity). These properties may explain the large success of CRFs in fields such as text and language modelling.

[15] We alluded to the fact that in general, the complexity of inference in a graphical models grows rapidly with the number of constraints (or edges).

[16] In fact, directed Bayesian networks are usually converted to undirected ones before BP is run on them.

[17] Hard optimization problems can often be *relaxed* by dropping or losening problematic constraints until the remaining relaxed problem is easy to solve (for example, because it is convex). Ideally, some worst case guarantees of how much is lost by the relaxation are sought.

4 Classification, Regularization, and Kernels

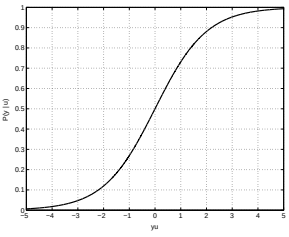
In this Section, we show how to extend the linear model for classification tasks. We also introduce the idea of regularization and nonparametric models, and show how penalized likelihood kernel methods arise as limit of the linear model.

4.1 Logistic Regression

The linear model of Eq. 1 is useful for *regression estimation* tasks, but for other problems such as classification, ranking, rate estimation, survival analysis, it needs to be modified. The idea is again to introduce a latent variable, namely let $u = \boldsymbol{w}^T \boldsymbol{\phi}(x) \in \mathbb{R}$. The linear model is obtained by assuming that $P(y|u) = N(y|u, \sigma^2)$, which is often appropriate if $y \in \mathbb{R}$. In binary classification, we have $y \in \{-1, +1\}$. We will retain the latent setup of the linear model, but choose a different distribution $P(y|u)$ supported on $\{-1, +1\}$, arriving at a *generalized linear model* (GLIM) [43]. The framework of GLIM is once more a unification of a number of common concepts valid for all members. Tasks such as maximum likelihood estimation have generic solution methods which can trivially be adapted to special GLIMs. We are free to choose $P(y|u)$ from any *exponential family*, namely families of the form $P(y|u) = \exp(u\psi(y) + g(u) + f(x))$. Examples of exponential families are Gaussian, Bernoulli, binomial, Poisson, Gamma, Beta.

For binary classification, the Bernoulli distribution is a useful likelihood:

$$P(y|u) = \lambda(y(u + b)), \quad \lambda(s) = (1 + e^{-s})^{-1},$$



where λ is called the *logistic function*, and b is an intercept hyperparameter. The GLIM with a Bernoulli likelihood is known as *logistic regression model*.

ML estimation for logistic regression can be dealt with in the framework of GLIM, giving rise to a method called *iteratively reweighted least squares* (IRLS), which is in fact the Newton-Raphson algorithm applied to the ML problem. In each iteration, a reweighted version of the normal equations of Eq. 6 have to be solved:

$$\boldsymbol{w}' = \left(\boldsymbol{X}^T \boldsymbol{D} \boldsymbol{X}\right)^{-1} \boldsymbol{X}^T \boldsymbol{r},$$

where $\boldsymbol{D}$ is diagonal, and both $\boldsymbol{D}$, $\boldsymbol{r}$ depend on the data $\boldsymbol{y}$ and the current $\boldsymbol{w}$ in general.

Importantly, for each GLIM there is a unique ML point $\boldsymbol{w}_{ML}$, and IRLS finds this point in very few iterations. Note that for the Gaussian likelihood, we have $\boldsymbol{D} = \sigma^{-2}\boldsymbol{I}$ and $\boldsymbol{r} = \sigma^{-2}\boldsymbol{y}$, so that IRLS in this case converges in a single step as expected. Therefore, in terms of tractability, ML estimation for a general GLIM resides somewhat in between a direct analytical solution as for the linear model or ML estimation of a single Gaussian, and first-order convergence to some local maximum as for the mixture model or HMMs. This is because fitting GLIM parameters is a *convex problem* (see Section 3.2.1). By construction, the negative log density of an exponential family is convex in u (see remark at the end of this Section), and since concatenations of linear and convex functions are convex again, it is easy to see that the negative log likelihood of a GLIM is convex in $\boldsymbol{w}$. A convex function cannot have more than a single mode, and this mode can be found in typically very few steps

using the well known Newton-Raphson algorithm. In fact, convexity is such an important concept in optimization theory that it is nowadays equated with "simple to solve globally" (a good textbook on convex optimization is [9]).

What about $C > 2$ classes? It is straightforward to generalize logistic regression (and GLIMs in general) to this case, by allowing for C latent linear functions. We have $y \in \{1, \ldots, C\}$, and the likelihood is $P(y|u_1, \ldots, u_c) \propto \exp(u_y)$, the *softmax* mapping. For simplicity, we restrict ourselves to the binary setup in the rest of this note.

While ML estimation is about as tractable for a GLIM than for the linear model, an exact Bayesian analysis of the logistic regression model is not analytically tractable, and a simple analytic form of the posterior $P(\boldsymbol{w}|D)$ is not known. Many approximations have been suggested. If $P(\boldsymbol{w})$ is Gaussian, it is sensible to approximate $P(\boldsymbol{w}|D)$ by a Gaussian as well (from the unimodality of the likelihood, we know that $P(\boldsymbol{w}|D)$ is unimodal as well). Variational approximations (see Section 3.2.1) obtain bounds on the marginal likelihood $P(D)$ and posterior approximations by essentially bounding the log likelihood terms or using Eq. 14. *Expectation propagation* (EP) [45] is an iterative approximation technique recently proposed in Machine Learning, which gives very promising results. If n (number datapoints) is much larger than p (size of $\boldsymbol{w}$), the simpler *maximum a posteriori* (MAP) approximation (also *Laplace* approximation) can be used. We first find the mode of the posterior $\boldsymbol{w}_{MAP}$ by minimizing $g(\boldsymbol{w}) = -\log P(D|\boldsymbol{w}) - \log P(\boldsymbol{w})$, using an obvious variant of IRLS. We then expand $g(\boldsymbol{w})$ to second order around $\boldsymbol{w}_{MAP}$ in order to obtain the Gaussian approximation $N(\boldsymbol{w}_{MAP}, \boldsymbol{H}^{-1})$ of $P(\boldsymbol{w}|D)$, where $\boldsymbol{H} = \nabla\nabla g(\boldsymbol{w}_{MAP})$. Both EP and Laplace lead to approximations of the marginal likelihood $P(D)$ as well, so that hyperparameters may be adjusted by empirical Bayes, as discussed in Section 2.1. We note that the difference between ML and MAP is the presence of the $\log P(\boldsymbol{w})$ term, which in the case of a Gaussian prior *penalizes* large $\boldsymbol{w}$. This *regularization* of the ML estimator is discussed in more detail in the next Section.

We close by noting that a central feature of GLIMs is that the likelihood $P(y|u)$ is a *log-concave* function of u, in that $-\log P(y|u)$ is convex. This implies convexity of ML estimation, and also makes it much easier to approximate inference in GLIMs by variational techniques or even by MCMC. As such, one could generalize GLIMs further by allowing for log-concave likelihoods which are not exponential families. An example is the *probit* binary classification likelihood $P(y|u) = \int_{-\infty}^{y(u+b)} N(s|0,1)\, ds$, where the sigmoidal function is the c.d.f. of a Gaussian. The concept of log-concavity is reviewed in [9], Sect. 3.5.

4.2 Regularization and Nonparametric Models

We have seen that a sensible estimation method from noisy data has to incorporate some sort of complexity control, if the number of parameters is not much smaller than the number of datapoints. Complexity can be restricted *a priori* by choosing a simple model. For example, in the linear model we can choose a small number p of features. However, in most cases we do not know a small number of features from the start which lead to a useful model. Another idea is to use a large model, but to *penalize complexity* within the model family itself. In the linear model, even if the number of features is large, very complicated functions $u(x)$ can only arise for large $\boldsymbol{w}$. We can *regularize* ML estimation by adding a *complexity penalty* (or regularizer) to the negative log likelihood to be minimized. For example, a regularizer $\alpha^{-1}\|\boldsymbol{w}\|^2$ penalizes large weights. In contrast to the classical ML estimate, the penalized one

satisfies a trade-off between data fit and complexity, and overfitting problems are typically alleviated.

Recalling the MAP (Laplace) approximation to Bayesian inference from the previous Section, we note that there is a simple connection to regularization. Namely, for any prior $P(\boldsymbol{w})$, the term $-\log P(\boldsymbol{w})$ can be used as a penalty. The squared penalty $\alpha^{-1}\|\boldsymbol{w}\|^2$ corresponds to a Gaussian prior $P(\boldsymbol{w}) = N(\boldsymbol{0}, \alpha\boldsymbol{I})$. This does not mean that the Bayesian framework is equivalent to the idea of regularization. Recall from Section 2.3 that marginalization should *always* be preferred over regularized optimization, if it is feasible. And even in intractable situations such as logistic regression, better approximations to Bayesian marginalization than MAP are often available (see Section 4.1).

Suppose now that we want to perform binary classification with the linear logistic regression model discussed in Section 4.1, but in fact we do not really want to spend the effort of carefully constructing or selecting features for $\boldsymbol{\phi}(\cdot)$. Rather, we would like to impose some properties on the latent function $u(\cdot)$ directly, for example it should vary smoothly. Given that regularization is in place, there is really no reason anymore to restrict the number of features at all, we could even allow for infinitely many, as long as we make sure that most of them can make no more than a infinitesimal contribution to the latent function. Following these ideas will provide us with a bridge between GLIMs and a class of models which have very different properties: *nonparametric models*. For example, a nonparametric model gives rise to the well known *nearest neighbour rule*. In order to classify a pattern x based on data $\{(x_i, y_i)\}$, find k such that $|x - x_k| \leq |x - x_i|$ for all i, then output y_k. This is very different from logistic regression (which is a *parametric model*). There are no apparent parameters to be estimated or to be inferred, and the complete dataset has to be used for each prediction. Nonparametric regression techniques try to estimate or infer the latent function $u(\cdot)$ directly from the data. For example, the Parzen windows method represents u as weighted sums of kernels placed on each point x_i.

However, in general the term "nonparametric" is frequently misunderstood to mean that the model has no parameters at all. It rarely makes sense to conceive of a statistical method without any free parameters to be adjusted from observed data. Learning about unknown parameters from data is what much of Statistics is about. Even a simple histogram method comes with a bin width and a smoothing parameter which have to be adjusted. The correct distinction between parametric and nonparametric methods is as follows. Suppose a model $P(x)$ for x is fitted to data D, in order to predict x (if there are covariates, we ignore them in the moment). A method gives rise to a predictor $P(x|D)$, which could be a predictive distribution or simply a point estimate. A method is *parametric* if there exists a representation $\boldsymbol{r}(D) \in \mathbb{R}^q$ for some finite q independent of D and its size: $P(x|D) = \tilde{P}(x|\boldsymbol{r}(D))$ for all x, D. Otherwise, the method is *nonparametric*. $\boldsymbol{r}(D)$ is also called *sufficient statistic*. For example, the ML plug-in rule for the model $P(x|\mu) = N(x|\mu, 1)$ is parametric, because the ML estimate $\hat{\mu} = n^{-1}\sum_i x_i \in \mathbb{R}$ is a sufficient statistic of size 1. The ML plug-in rule for the linear model has sufficient statistic $\hat{\boldsymbol{w}}_{ML} \in \mathbb{R}^p$, and Bayesian prediction for the linear model has a sufficient statistic of size $O(p^2)$. In contrast, for many nonparametric methods, we cannot find any sufficient statistics significantly smaller than D itself, and D in a more or less uncompressed form has to be available for each prediction.

Parametric and nonparametric models have very different characteristics. A parametric model is fundamentally limited by the fixed form of the distribution it represents. A linear model can represent linear functions only, a single Gaussian will represent multimodal

data poorly. These limitations do not vanish with growing dataset size, as opposed to the situation for many nonparametric models which can be shown to be universally consistent, *i.e.* they can represent any data source given enough data. On the other hand, if the data-generating process or important aspects of it are well understood, a parametric model can deliver a much better representation from finite data than a nonparametric model whose prior assumptions are typically much weaker. A major advantage of parametric models is scalability for large datasets. ML estimation or inference approximations typically scale linearly with the dataset size n, and both prediction time and learned representation size are independent of n. The reader may object that in practice, the representation size of parametric models such as mixtures or multi-layer perceptrons are chosen depending on n. Indeed, some nonparametric models are close to parametric ones, in that they extend them by making the size choice automatically from the data. An example are Dirichlet process[18] mixture models [48] which are effectively used to infer a mixture model with a number of components appropriate for the given data, although from another point of view they are universal density estimators.

It is important to note that while regularization or Bayesian complexity control can be beneficial with parametric models, it becomes indispensable with nonparametric ones. It is not possible to estimate the optimal $u(\cdot)$ from finite data without any restriction of how u may change with x. For many nonparametric models, the concrete regularizer or the Bayesian model are opaque or not even explicitly known, but we will see in the sequel that a proper limit of the linear model gives rise to regularized and to Bayesian nonparametric models.

4.3 Penalized Likelihood Kernel Methods

In Section 4.2, we noted that with proper regularization in place, we could take the number of features in the linear model of Eq. 1 to infinity. This astonishing idea leads to *kernel methods*.

Recall from Section 2.2.1 that if $p > n$, it makes sense to write the Bayesian predictive distribution in the form of Eq. 9. Nothing in these formulae really scales with p, except for inner product terms $\boldsymbol{\phi}(x)^T\boldsymbol{\phi}(x')$. If we define a *kernel* $K(x,x') = \boldsymbol{\phi}(x)^T\boldsymbol{\phi}(x')$, we have that $\boldsymbol{X}\boldsymbol{X}^T = \boldsymbol{K} = (K(x_i,x_j))_{i,j}$ and $\boldsymbol{X}\boldsymbol{\phi}(x) = (K(x_i,x))_i$, and any variable scaling as p has disappeared. This means that within the linear model, it is actually sufficient to know the "correlations" between any two feature vectors $\boldsymbol{\phi}(x)$, $\boldsymbol{\phi}(x')$ in the form of $K(x,x')$, we do not need to specify the $\boldsymbol{\phi}(x)$ themselves. Merely the kernel has to be specified. Now, a kernel does not constitute a prior distribution, and in any case: a prior distribution over what? The correct link between the Bayesian linear model and kernels will be drawn in Section 5.1, where we will also gain more understanding about the role of the kernel.

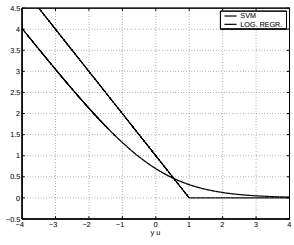
Suppose now we use the MAP approximation for the linear model with Gaussian prior $P(\boldsymbol{w}) = N(\boldsymbol{0}, \alpha\boldsymbol{I})$, and we want to allow for $p \to \infty$, specifying the kernel $K(x,x')$ only. Since $K(x,x')$ is an inner product, it must be *positive definite*:

$$\sum_{i,j} a_i a_j K(x_i,x_j) \geq 0 \tag{15}$$

[18] DP mixture models are another important example of nonparametric Bayesian models which have been employed successfully in Machine Learning. They are useful for density estimation and clustering. Their introduction requires some advanced MCMC arguments, which are not in the scope of this review.

for all x_i, $a_i \in \mathbb{R}$. We also have that each positive definite form $K(x, x')$ is an inner product, although we may not be able to compute $\boldsymbol{\phi}(\cdot)$. The latent function is $u(x) = \boldsymbol{w}^T \boldsymbol{\phi}(x)$, and the likelihood is a function of the $u(x_i)$ only. One can show that the correct "limit" of the regularizer $\alpha^{-1}\|\boldsymbol{w}\|^2$ is in fact $\alpha^{-1}\|u(\cdot)\|_K^2$, the squared norm in a *reproducing kernel Hilbert space* (RKHS; a function space depending on K) of $u(\cdot)$. *Penalized likelihood kernel methods* can therefore be seen as minimizing the sum of a negative log likelihood and the penalty $\alpha^{-1}\|u(\cdot)\|_K^2$ *directly for the function* $u(\cdot)$. Here, the regularization term penalizes complicated behaviour of $u(\cdot)$, the specifics depend on K. We will obtain a clearer picture of the role of K in the next Section.[19] Kernel and RKHS inner product are linked by the reproducing property: $(K(\cdot, x), K(\cdot, y))_K = K(x, y)$. Since the likelihood depends on the $u(x_i)$ only, one can show that the minimizer must have the form $u(x) = \sum_i \alpha_i K(x_i, x)$, a *kernel expansion*, and by the reproducing property we have that $\|u(\cdot)\|_K^2 = \boldsymbol{\alpha}^T \boldsymbol{K} \boldsymbol{\alpha}$ in this case. Let us recapitulate the steps in this chain. (1) The limit $p \to \infty$ of the MAP linear model results in the $\|u(\cdot)\|_K^2$ penalty. (2) The form of the likelihood implies[20] that any minimizer of the penalized criterion is a kernel expansion. (3) $\|u(\cdot)\|_K^2$ for a kernel expansion can be computed tractably by the reproducing property. Our problem is now reduced to finding $\boldsymbol{\alpha} \in \mathbb{R}^n$, which in the penalized GLIM case can be done by a variant of IRLS (again, there is a unique minimum point $\hat{\boldsymbol{\alpha}}$). This argument is the "kernel trick", which allows to reduce the optimization of a penalized likelihood functional over a function space to the estimation of n coefficients. Note that the resulting method is clearly nonparametric, since all of the x_i are required for each prediction. Kernelized GLIMs are discussed in detail by [21]. They are equivalent to what is called "exponential family kernel methods" in Machine Learning.

The *support vector machine* (SVM) is among the most popular penalized likelihood kernel methods in Machine Learning. We cannot go into details, but refer to [58] who also give details on RKHS. In the SVM, the negative log likelihood is replaced by a particular empirical risk term of the form $\sum_i [1 - y_i u(x_i)]_+$, where $[z]_+ = z \mathrm{I}_{\{z>0\}}$. Just as in logistic regression, a pattern (x_i, y_i) is misclassified iff $y_i u(x_i) < 0$.



In SVM, a clear separation with "margin" is enforced by penalizing $y_i u(x_i) < 1$ already. On the other hand, if $y_i u(x_i) \geq 1$, the pattern does not induce any loss, a fact which can be shown to lead to *sparse* solutions $\hat{\boldsymbol{\alpha}}$ (many $\hat{\alpha}_i = 0$) on simpler classification problems. $\hat{\boldsymbol{\alpha}}$ can be obtained as the solution of a *quadratic program*, and much work has been concentrated in obtaining specialized algorithms and implementing efficient software.[21] Many theoretical aspects of SVM are well understood now, and due to their attractive computational properties, they have had a host of successful Machine Learning applications ([58] give many references).

Note that the SVM problem is *not* an MAP approximation to inference for any probabilistic model [59], because the hinge loss does not correspond to a negative log likelihood. Indeed, [30] emphasize that the SVM and other large margin techniques (see Section 3.5) are in-

[19] Within much of the non-probabilistic kernel methods field in Machine Learning, the kernel K is unfortunately regarded as something of a "black box", and the precise regularization characteristics of different kernels are typically ignored.

[20] The likelihood for a density estimation model does not have this property, it depends on global nonlinear functionals of $u(\cdot)$. It is an open problem to do tractable Bayesian density estimation with a GP model.

[21] The web page *www.kernel-machines.org* has links to papers and software for SVM and related methods.

stances of *maximum entropy methods* (ME), an inference paradigm quite different from the Bayesian one. Rather than combining observed data with prior knowledge by using a probabilistic likelihood function which is multiplied with the prior to obtain the posterior through renormalization, the maximum entropy approach is to find a predictive distribution (which is the analogue to the posterior) closest to a vague prior (which often has maximal entropy, *i.e.* uncertainty), *subject to* constraints given by the observations. Typically, the constraints are linear in the predictive distribution, and frequently, they are "softened" by allowing for slack variables. For a large margin method, an observation introduces one or more constraints forcing this observation to be classified correctly by some margin, *i.e.* to outvote all possible competitors distinctively. ME and Bayesianism come with respective strengths and weaknesses. In simple cases, ME methods result in straightforward convex problems with linear constraints which can be solved exactly and efficiently. The deep algorithmic knowledge surrounding structured linear programs can be tapped [69]. The presence of hard constraints means that final solutions often exhibit *sparsity*, in the sense that many representation variables are zero. On the other hand, it is not clear how to deal with latent variables and hierarchical models correctly and efficiently in ME. More importantly, the predictive distribution computed by ME does not serve as a useful uncertainty estimate, in a way the Bayesian posterior does. Among all solutions fulfilling the data constraints, ME always picks the most uncertain one, rather than ranking the solutions w.r.t. a concept such as likelihood.

We finally note that nonparametric methods are sometimes branded as "infinite models". While this can be a useful way of thinking about these models, as demonstrated here, the view has its pitfalls. What is really meant is that the statistics we extract from the data to fit our model simply cannot be limited in size *before* we have seen D. The advantage of nonparametric models is not that they are "infinite" for finite data, but rather that their representation size *grows* with D in an automatic fashion.

5 Gaussian Processes for Machine Learning

In this Section we introduce probabilistic models based on *Gaussian processes* (GP). This will turn out to be the correct nonparametric generalization of the linear model (and GLIMs) to infinitely many features, so Bayesian analysis becomes possible. We will also get a clear intuition about the role of the kernel w.r.t. regularization in these methods. We can only give some intuition here, for details and many more facettes the reader may look at [62, 55]. A wealth of material is collected at the web page `http://www.gaussianprocess.org/`.

5.1 The Infinite Limit of the Linear Model

The Bayesian treatment of the linear model has been given in Section 2.2. We have already motivated the step to penalized likelihood kernel methods in Section 4.2, by letting the number of features p grow to infinity, while assuming that the kernel $K(x, x') = \boldsymbol{\phi}(x)^T \boldsymbol{\phi}(x')$ can be computed without having to access the feature map directly.

We noted that predictive distributions and marginal likelihood for the linear model are finite expressions even for $p = \infty$, if all feature inner products are replaced by kernels. However, in order to arrive at a sound probabilistic model, we have to clarify the relevant

latent variables and their prior distributions exactly. Recall that $u(x) = \boldsymbol{w}^T \boldsymbol{\phi}(x)$ with $\boldsymbol{w} \sim N(\boldsymbol{0}, \boldsymbol{I})$ *a priori* for any finite p. We are looking for a distribution over $u(\cdot)$, also called a *process*. Note that $\mathrm{E}[u(x)] = 0$ and $\mathrm{E}[u(x)u(x')] = \boldsymbol{\phi}(x)^T \boldsymbol{\phi}(x') = K(x, x')$. Indeed, for any finite $x_1, \ldots, x_m$, we have that $\boldsymbol{u} = (u(x_i))_i \in \mathbb{R}^m$ is jointly Gaussian with zero mean and covariance $\boldsymbol{K} = (K(x_i, x_j))_{i,j}$. All these properties do not depend on $\boldsymbol{\phi}(\cdot)$ or on p, and in fact they characterize[22] a valid process, called *Gaussian process* (GP). We merely require that $K(x, x')$ is positive definite (Eq. 15), which is clearly necessary, since $0 \leq \mathrm{E}[(\sum_i a_i u(x_i))^2] = \sum_{i,j} a_i a_j K(x_i, x_j)$. We can close the link to the infinite linear model and in fact construct a GP $u(\cdot)$ explicitly by invoking the Mercer eigenexpansion of the kernel and the Karhunen-Loeve representation $u(x) = \sum_{i \geq 1} w_i \phi_i(x)$ with infinitely many independent $w_i \sim N(0, 1)$, [62] gives the details.

While an explicit construction of a GP $u(\cdot)$ with covariance function $K(x, x')$ is necessary in order to arrive at a well defined nonparametric model, for practical purposes the working definition of a GP via its finite-dimensional marginals is all that is needed in order to perform inference. For example, the marginal likelihood of Eq. 5 can be computed using $\boldsymbol{u} = (u(x_i))_i \sim N(\boldsymbol{0}, \boldsymbol{K})$ only, with $\boldsymbol{u} \in \mathbb{R}^n$, noting that $\boldsymbol{X}\boldsymbol{X}^T = \boldsymbol{K}$. For the predictive distribution of Eq. 9, we need one more variable $u(x)$, so $n + 1$ in total. The limit of the linear regression model becomes the standard *GP regression model* with Gaussian noise, and Bayesian inference is done in the same way as described in Section 2.2.1. Rather than having to deal with infinite feature vectors, we have to be able to do numerical linear algebra with large dense matrices, expressed in terms of the *kernel matrix* $\boldsymbol{K} \in \mathbb{R}^{n,n}$. Bayesian GP models as developed here have been suggested by [50]. The topic has been introduced to Machine Learning by [47] who observed that under certain assumptions, the limit of an infinitely large Bayesian multi-layer perceptron gives rise to a GP model. Early work in Machine Learning has been done by [75, 76], a more complete list of references can be found in [62, 55]. It is clear that the link between linear and GP models extends to GLIMs, although just as in the linear case, Bayesian analysis becomes intractable in general. We mentioned some approximation techniques in Section 4.1, they can be applied to GP GLIMs just as well. The general idea is to approximate the intractable posterior process by a GP itself, whose mean function constitutes our best prediction of $u(\cdot)$.

It is instructive to compare the predictive distributions from a linear model with features $\boldsymbol{\phi}(x) = (1, x)^T$ and from a GP regression model. In Figure 5, we depict these by drawing samples from the posteriors. For the linear model, these sample paths must be lines, while for the GP model they are smooth functions of appropriate complexity. The reader should note the significant differences in the predictive uncertainties. For the linear model, these local uncertainties are mediated entirely by the global uncertainty in the line itself. This is correct, given that the model cannot even hypothesize any nonlinear contributions. If most of the data is represented well by a narrow range of lines, the predictive uncertainty at *all* locations will be small. For example, the point with x closest to 0 lies far from the predicted line, yet the uncertainty there is very small. The local predictive uncertainties behave much more reasonably for the GP model: they grow where data is sparse.

GP models can be contrasted with Gaussian random fields [13]. The latter is an undirected graphical model (see Section 3.5) on a grid. Since a GRF has continuous variables and Gaussian potentials, conditional inference is tractable and is made efficient (linear time

[22] The GP is not uniquely characterized by these properties, but differences between GP distributions with the same mean and covariance function are technical and not important in our context here.

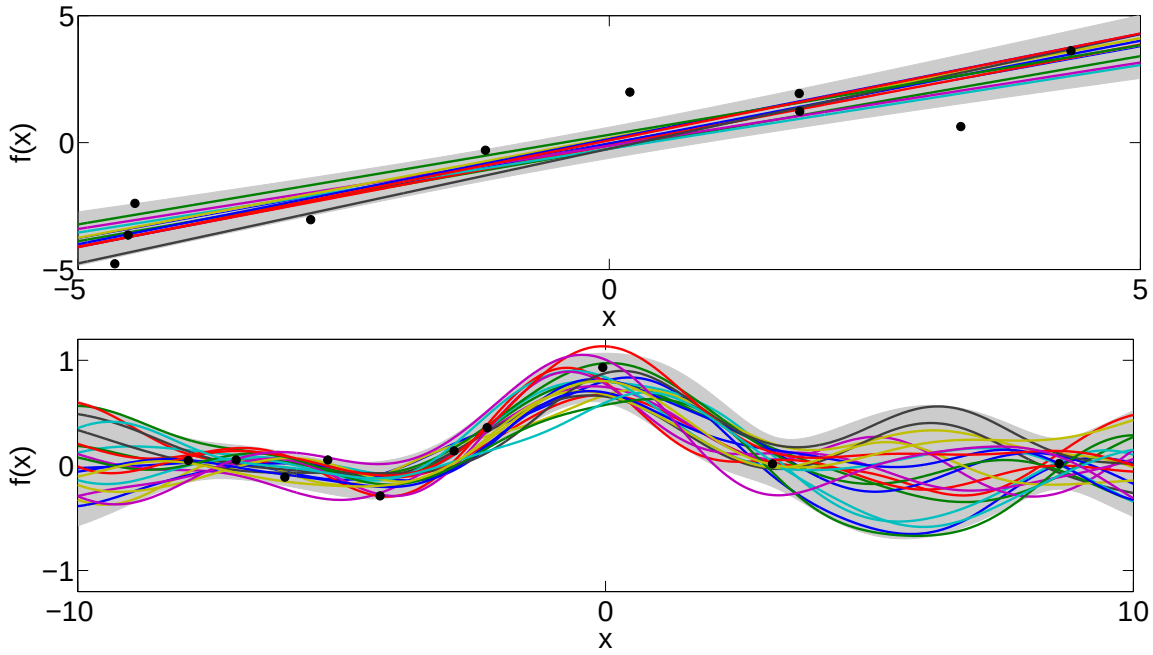


Figure 5: Comparison of predictive distributions from linear model (above) and GP regression model (below). Pictures courtesy of Malte Kuss.

in the number of nodes) by the fact that the inverse covariance matrix is sparse [73]. A GRF with stationary potentials does not correspond to a GP model,[23] because the joint distribution is not consistent under marginalization of some of the variables. GRFs have been used extensively in Computer Vision and Geostatistics, and they have recently been applied to semi-supervised learning [78].

We finally note a remaining subtle issue. In Section 4.3, we obtain penalized likelihood kernel methods as limit of an MAP approximation to the finite linear model. In this Section, we showed that the limit of the finite Bayesian model is well defined and corresponds to a GP regression model. But can a kernel method be obtained as MAP approximation to a GP regression model? The difficulty here is that it is not immediately clear how to define a density for a GP, for example what is an appropriate dominating measure? We can obtain a density for $u(\cdot)$ using its Karhunen-Loeve representation, and one can show that $\log P(u(\cdot)) \doteq -(1/2)\|u(\cdot)\|_K^2$, up to a term independent of u, which however diverges[24] as $i \to \infty$ in the series representation for u. If we drop this normalization term, we see that MAP for a GP GLIM is indeed equivalent to penalized likelihood. Details for this argument are given by [62].

[23] Although some Machine Learning researchers confused the two concepts.

[24] This is in line with the somewhat surprising fact that for the RKHS $\mathcal{H}_K = \{u \,|\, \|u\|_K < \infty\}$, we have that $\Pr\{u \in \mathcal{H}_K\} = 0$ under the GP distribution. This contradicts the (mistaken!) claim that a GP is a distribution over its RKHS.

5.2 The Kernel as Covariance Function

In a model based on a GP $u(\cdot)$, the kernel K has a clear meaning as (prior) *covariance function*:

$$K(x, x') = \mathrm{E}[u(x)u(x')].$$

Note that this assumes that the GP is zero mean: $\mathrm{E}[u(x)] = 0$. It is now possible to clearly understand *how* functions are regularized through the GP prior, and what role the kernel K plays. For example, if $\boldsymbol{x} \in \mathbb{R}^d$ and $K(\boldsymbol{x}, \boldsymbol{x}') = K(d),\ d = \|\boldsymbol{x} - \boldsymbol{x}'\|$ (such covariance functions are called *isotropic*), the correlation between the values $u(\boldsymbol{x})$, $u(\boldsymbol{x}')$ decay with increasing distance d. For very close $\boldsymbol{x}$, $\boldsymbol{x}'$, the correlation (therefore the dependence: we talk about Gaussians!) will be ≈ 1, while points far apart are almost uncorrelated (negative correlations are possible under some kernels as well). Although all $u(\boldsymbol{x})$ are random, variables for nearby $\boldsymbol{x}$ are strongly correlated, implying *smoothness* of the underlying random process. There is a rich theory of how characteristics of K imply properties of the process $u(\cdot)$ and its sample paths [67]. If we restrict kernels to be *stationary*, meaning that $K(\boldsymbol{x}, \boldsymbol{x}') = K(\boldsymbol{x} - \boldsymbol{x}')$, then Bochner's theorem provides a characterization. $K(\boldsymbol{d})$ is a covariance function with $K(\boldsymbol{0}) = 1$ iff it is the characteristic function of some distribution: $K(\boldsymbol{d}) = \mathrm{E}[\exp(i\boldsymbol{d}^T\boldsymbol{\omega})]$ for some random $\boldsymbol{\omega} \in \mathbb{R}^d$. This connection provide us with a rich set of (stationary) covariance functions which can be taylored to prior knowledge about our task. For example, the *Gaussian covariance function*

$$K(\boldsymbol{x}, \boldsymbol{x}') = v \exp\left(-\frac{b}{2}\|\boldsymbol{x} - \boldsymbol{x}'\|^2\right), \quad v, b > 0, \tag{16}$$

is frequently used in Machine Learning, it is the characteristic function of $N(\boldsymbol{\omega}|\boldsymbol{0}, b\boldsymbol{I})$. This kernel comes with a variance parameter v and an inverse squared length-scale parameter b. We have $\mathrm{Var}[u(\boldsymbol{x})] = v$ for each $\boldsymbol{x}$, while $b^{-1/2}$ is the average distance along which $u(\cdot)$ varies significantly.

We can generalize the Gaussian covariance function of Eq. 16 by replacing the radial squared distance $b\|\boldsymbol{x} - \boldsymbol{x}'\|^2$ by the weighted distance $\sum_j d_j(x_j - x'_j)^2,\ d_j > 0$. Here, we allow for a different length-scale along each dimension. Bayesian estimation of d_j via marginal likelihood maximization implements a form of ARD (see Section 2.4), in that if a component j of $\boldsymbol{x}$ is not relevant for predicting y, the corresponding d_j is driven towards zero. This is a powerful feature (or attribute) selection technique.

Probably the best way of choosing a kernel for a particular application is to consider theoretical arguments such as given by [67], and ideally vote for a class which comes with free hyperparameters determining the implied smoothness, variance, and length-scales for $u(\cdot)$. These parameters may be fitted from data using the empirical Bayesian method, for example. Stein recommends the *Matérn class* whose members arise as characteristic functions of t distributions. This class comes with a hyperparameter regulating the degree of smoothness of the sample functions $u(\cdot)$, and it contains the Gaussian kernel in the limit of very high smoothness. If $\boldsymbol{x}$ is of low dimensionality (for example in spatial problems), using a Matérn kernel with appropriate degree of smoothness can result in much better predictions than using the Gaussian kernel. We can also get some feeling for a covariance function by simply plotting samples of the corresponding $u(\cdot)$, examples can be found in [62, 55].

The use of finite-dimensional covariance functions is widespread in Machine Learning. By this we mean kernels which do have a feature map $\boldsymbol{\phi}(\cdot)$ of finite dimensionality. An example

is the polynomial kernel $K(\boldsymbol{x}, \boldsymbol{x}') = (v + \boldsymbol{x}^T \boldsymbol{x}')^q,\ v \geq 0, q \in \mathbb{N}$, whose feature map contains all monomials in $\boldsymbol{x}$ of total degree $\leq q$. In fact, following our definition above, methods based on such kernels are parametric, since they can always be written as finite-dimensional linear models. Kernel methods have also been applied to settings where input points x are structured objects, such as variable-length sequences of symbols or graphs. Kernels for such objects can be constructed using the principles described by [24]. Applications are in Bioinformatics, language modelling, and in Computer Vision [20]. For polynomial and string kernels, the feature space dimensionality is vastly larger than the input dimensionality, so the "kernel trick" is still important in those situations.

5.3 Practical Inference with Gaussian Process Models

In this Section, we provide details for the GP regression model. We then discuss a principal drawback of inference in GP models, namely the heavy scaling with the number of datapoints, and suggest approximations which overcome this problem.

The GP regression model is the limit of the linear one, so we only need to rewrite the equations of Section 2.2. Let $\boldsymbol{u} = (u(x_i))_i \in \mathbb{R}^n$ and $\boldsymbol{K} = (K(x_i, x_j))_{i,j}$. We have that $P(\boldsymbol{u}) = N(\boldsymbol{0}, \boldsymbol{K})$. From Eq. 9 we see that we need to solve linear systems with $\boldsymbol{A} = \boldsymbol{K} + \sigma^2 \boldsymbol{I}$. Since $\boldsymbol{A}$ is symmetric positive definite, the numerical method of choice for doing this is to compute the Cholesky decomposition (see Section 2.2.1):

$$\boldsymbol{A} = \boldsymbol{K} + \sigma^2 \boldsymbol{I} = \boldsymbol{L}\boldsymbol{L}^T,$$

where $\boldsymbol{L}$ is lower triangular with positive elements on the diagonal. We can now write the predictive distribution as

$$P(y|x, D, \sigma^2) = N\left(\boldsymbol{b}^T \boldsymbol{v}, \sigma^2 + K(x, x) - \|\boldsymbol{v}\|^2\right), \quad \boldsymbol{v} = \boldsymbol{L}^{-1}(K(x_i, x))_i., \ \boldsymbol{b} = \boldsymbol{L}^{-1}\boldsymbol{y}.$$

Here, $\boldsymbol{L}^{-1}\boldsymbol{a}$ is computed by a *backsubstitution* which can be done in $O(n^2)$ due to the lower triangular structure of $\boldsymbol{L}$. If only the predictive mean is required, we can also precompute $\boldsymbol{p} = \boldsymbol{L}^{-T}\boldsymbol{b}$ (another backsubstitution), whence the mean is $\boldsymbol{p}^T(K(x_i, x))_i$, which costs $O(n)$ only.

The empirical Bayesian estimation of σ^2 and parameters of the covariance function K, for example of v, b in Eq. 16, uses the criterion

$$-\log P(\boldsymbol{y}|\sigma^2) = -\log N(\boldsymbol{y}|\boldsymbol{0}, \boldsymbol{A}),$$

namely the negative log marginal likelihood. This criterion is continuously differentiable in the hyperparameters, but it is usually not convex and can have local minima. It can be optimized using a gradient-based scheme (the gradient can be computed in $O(n^3)$) such as conjugate gradients or Quasi-Newton.

The GP regression model is a very powerful technique to address the curve smoothing task, especially if the covariance function is chosen carefully and free parameters are chosen using empirical Bayes (or are even marginalized over using MCMC, see Section 2.5). However, it cannot be used for many Machine Learning applications due to its scaling behaviour. The running time requirements are $O(n^3)$, and $n \times n$ matrices have to be stored, either is prohibitive for large datasets. Do we have to drop the model and settle for simpler linear

ones, or multi-layer perceptrons? The answer is no, if we are prepared to make further approximations. Much recent work has concentrated on finding *sparse approximations* to inference in GP models. We can only provide basic intuitions here. A detailed discussion of aspects of sparse approximations can be found in [61, 63], also containing many relevant references which we omit here. Recall that we have $P(\boldsymbol{u}) = N(\boldsymbol{0}, \boldsymbol{K})$ (prior) and $P(\boldsymbol{y}|\boldsymbol{u}) = N(\boldsymbol{y}|\boldsymbol{u}, \sigma^2\boldsymbol{I})$ (likelihood) in the regression model, and note that the likelihood is a function of $\boldsymbol{u}$. Let $I \subset \{1, \ldots, n\}$ be a subset of size $d < n$, called *active set*, and let $\boldsymbol{u}_I = (u_i)_{i \in I}$ be the corresponding subvector of latent variables. Suppose that we replace the likelihood by a positive function $f(\boldsymbol{u}_I)$ of $\boldsymbol{u}_I$ only, so that the posterior $P(\boldsymbol{u}|D)$ is approximated by $Q(\boldsymbol{u}) \propto P(\boldsymbol{u}) f(\boldsymbol{u}_I)$. Under this replacement, the running time scaling for GP inference reduces to $O(n\, d^2)$, an improvement by a factor of $(d/n)^2$.

Given this observation, we need to address two points: how to choose $f(\boldsymbol{u}_I)$ for fixed I, and how to choose I? For the first task, some principled methods have been suggested. First, $f(\boldsymbol{u}_I)$ should be chosen s.t. $Q(\boldsymbol{u})$ is a good approximation to the true posterior $P(\boldsymbol{u}|D)$. For example, we can choose $f(\boldsymbol{u}_I)$ in order to minimize the relative entropy $\mathrm{D}[Q(\boldsymbol{u}) \,\|\, P(\boldsymbol{u}|D)]$.[25] Another related likelihood approximation is suggested by [64]. An even faster approximation (by a constant factor) is obtained by simply choosing $f(\boldsymbol{u}_I) = P(\boldsymbol{y}_I|\boldsymbol{u}_I)$, effectively throwing away the observations y_i, $i \notin I$ [63]. The second and harder problem is how to choose[26] I. This is a combinatorial problem in general, but one we *have* to address in time no more than $O(n\, d^2)$. At the expense of $O(n\, d)$ memory, this can be done using a greedy forward selection strategy. The idea is to keep the marginal distributions $Q(u_i)$ for all components of $\boldsymbol{u}$ up-to-date at any time, and to score candidates for inclusion into i by comparing the marginal with the true target. Care has to be taken to choose a good representation of $Q(\boldsymbol{u})$ which can be updated in a numerically stable manner, and which allows for efficient updating of the marginals. Sparse approximations can be generalized to GP GLIMs with non-Gaussian likelihood, by using the expectation propagation scheme mentioned in Section 4.1. We can even compute a sparse approximation to the marginal likelihood and optimize it in order to select hyperparameters by Bayesian estimation [61, 63].

6 Discussion

With this tutorial review, we aim to give a wide high-level overview over concepts of probabilistic modelling and Bayesian Statistics relevant for Machine Learning. Our focus is on stressing key concepts and cornerstones, on pointing out where they loom in the background of more complicated techniques, on showing connections between seemingly unrelated concepts, and in general on preparing the interested reader for further detailed studies along the references we provide here.

It has not been our goal to treat any of the topics here in great or even proper details, but in all cases we know of comprehensive reviews or textbooks, we have included their references. Apart from this guideline, our choice of references is subjective and pragmatic rather than complete in any sense. Given the limited size of this review versus the enormous scope of a

[25] In a sense, the best $f(\boldsymbol{u}_I)$ would be the one minimizing $\mathrm{D}[P(\boldsymbol{u}|D) \,\|\, Q(\boldsymbol{u})]$, but it *cannot* be determined efficiently (see [61], Lemma 4.1).

[26] It has been noted that $\boldsymbol{u}_I$ do not have to be a part of $\boldsymbol{u}$, they could be situated at any d points x_j. Under this more general view, the problem of selecting I as a subset of $\{1, \ldots, n\}$ is replaced by optimizing for the d active points x_j.

heterogeneous field such as probabilistic Machine Learning, we also had to be selective in terms of topics, skipping central fields such as multi-layer perceptrons and other "neural" architectures (see [7] for a textbook), reenforcement learning (see [5] for a textbook), density estimation and clustering, or information and coding theory (see [12, 41] for textbooks). Neither did we discuss any of the many application areas close to Machine Learning in much detail, although we hope that this review will be useful to practitioners of these areas as well. We can only hope to have met a central goal of ours, namely to convey some of the excitements about the dynamic and diverse field of probabilistic Machine Learning.

References

[1] S. Aji and R. McEliece. The generalized distributive law. *IEEE Transactions on Information Theory*, 46(2):325–343, 2000.

[2] T. Bell and T. Sejnowski. An information maximisation approach to blind separation and blind deconvolution. *Neural Computation*, 7(6):1129–1159, 1995.

[3] J. O. Berger. *Statistical Decision Theory and Bayesian Analysis*. Springer, 2nd edition, 1985.

[4] José M. Bernardo and Adrian F. M. Smith. *Bayesian Theory*. John Wiley & Sons, 1st edition, 1994.

[5] D. Bertsekas and J. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, 1st edition, 1996.

[6] J. Besag. Spatial interaction and the statistical analysis of lattice asystems. *Journal of Roy. Stat. Soc. B*, 36(2):192–236, 1974.

[7] C. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1st edition, 1995.

[8] G. Box and G. Tiao. *Bayesian Inference in Statistical Analysis*. John Wiley & Sons, 1992.

[9] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2002.

[10] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1222–1239, 2001.

[11] G. Casella and E. Lehmann. *Theory of Point Estimation*. Texts in Statistics. Springer, 1st edition, 1998.

[12] Thomas Cover and Joy Thomas. *Elements of Information Theory*. John Wiley & Sons, 1st edition, 1991.

[13] N. Cressie. *Statistics for Spatial Data*. John Wiley & Sons, 2nd edition, 1993.

[14] I. Csiszár and G. Tusnády. Information geometry and alternating minimization procedures. In et. al. E. F. Dedewicz, editor, *Statistics and Decisions*, pages 205–237. Oldenburg Verlag, Munich, 1984.

[15] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of Roy. Stat. Soc. B*, 39:1–38, 1977.

[16] R. Duda, P. Hart, and D. Stork. *Pattern Classification*. John Wiley & Sons, 2nd edition, 2000.

[17] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721–741, 1984.

[18] Zoubin Ghahramani and Matthew J. Beal. Variational inference for Bayesian mixtures of factor analysers. In Solla et al. [65], pages 449–455.

[19] W. Gilks, S. Richardson, and D. Spiegelhalter, editors. *Markov Chain Monte Carlo in Practice*. Chapman & Hall, 1st edition, 1996.

[20] K. Grauman and T. Darrell. The pyramid match kernel: Discriminative classification with sets of image features. In *Proc. IEEE Int. Conf. Computer Vision, Beijing*, 2005.

[21] P.J. Green and B. Silverman. *Nonparametric Regression and Generalized Linear Models*. Monographs on Statistics and Probability. Chapman & Hall, 1994.

[22] G. Grimmett and D. Stirzaker. *Probability and Random Processes*. Oxford University Press, 3rd edition, 2001.

[23] W. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57:97–109, 1970.

[24] David Haussler. Convolution kernels on discrete structures. Technical Report UCSC-CRL-99-10, University of California, Santa Cruz, July 1999. See `http://www.cse.ucsc.edu/~haussler/pubs.html`.

[25] G. E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14:1771–1800, 2002.

[26] G. E. Hinton and R. M. Neal. A new view on the EM algorithm that justifies incremental and other variants. In Jordan [34].

[27] G. E. Hinton and T. J. Sejnowski. Learning and relearning in boltzmann machines. In D. Rumelhart and J. McClelland, editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Volume 1: Foundations*, pages 282–317. 1986.

[28] R. Horn and C. Johnson. *Matrix Analysis*. Cambridge University Press, 1st edition, 1985.

[29] E. Ising. Beitrag zur theorie des ferromagnetismus. *Zeitschrift für Physik*, 31:253–258, 1925.

[30] Tommi Jaakkola, Marina Meila, and Tony Jebara. Maximum entropy discrimination. In Solla et al. [65], pages 470–476.

[31] W. James and C. Stein. Estimation with quadratic loss. In J. Neyman, editor, *Proceedings of 4th Berkeley Symp. Math. Stat. Prob.*, volume 1, pages 361–379, 1961.

[32] Finn V. Jensen. *An Introduction to Bayesian Networks*. UCL Press, 1st edition, 1996.

[33] M. Jordan, Z. Ghahramani, T. Jaakkola, and L. Saul. An introduction to variational methods in graphical models. In Jordan [34].

[34] M. I. Jordan, editor. *Learning in Graphical Models*. Kluwer, 1997.

[35] M. I. Jordan and R. A. Jacobs. Hierarchical mixtures of experts and the EM algorithm. *Neural Computation*, 6:181–214, 1994.

[36] R. E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME*, 82(Series D):35–45, 1960.

[37] R. Kass and A. Raftery. Bayes factors. *Journal of the American Statistical Association*, 90:773–795, 1995.

[38] J. Lafferty, F. Pereira, and A. McCallum. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In C. E. Brodley and A. P. Danyluk, editors, *International Conference on Machine Learning 18*. Morgan Kaufmann, 2001.

[39] S. Lauritzen. *Graphical Models*. Oxford Statistical Sciences. Clarendon Press, 1996.

[40] D. MacKay. Bayesian interpolation. *Neural Computation*, 4(3):415–447, 1992.

[41] D. MacKay. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2003.

[42] M.E. Maron. Automatic indexing: An experimental inquiry. *Journal of the ACM*, 8:404–417, 1961.

[43] P. McCullach and J.A. Nelder. *Generalized Linear Models*. Number 37 in Monographs on Statistics and Applied Probability. Chapman & Hall, 1st edition, 1983.

[44] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller. Equations of state calculations by fast computing machines. *Journal of Chemical Physics*, 21:1087–1092, 1953.

[45] T. Minka. Expectation propagation for approximate Bayesian inference. In J. Breese and D. Koller, editors, *Uncertainty in Artificial Intelligence 17*. Morgan Kaufmann, 2001.

[46] R. M. Neal. Probabilistic inference using Markov chain Monte Carlo methods. Technical Report CRG-TR-93-1, University of Toronto, 1993. See `www.cs.toronto.edu/~radford`.

[47] R. M. Neal. *Bayesian Learning for Neural Networks*. Number 118 in Lecture Notes in Statistics. Springer, 1996.

[48] Radford M. Neal. Markov chain sampling methods for Dirichlet process mixture models. Technical Report 9815, Department of Statistics, University of Toronto, Semptember 1998.

[49] A. Ng and M. Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive Bayes. In T. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*. MIT Press, 2002.

[50] A. O'Hagan. Curve fitting and optimal design. *J. Roy. Stat. Soc. B*, 40(1):1–42, 1978.

[51] J. Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann, 1990.

[52] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C*. Cambridge University Press, 2nd edition, 1992.

[53] J. Propp and D. Wilson. Exact sampling with coupled Markov chains and applications to statistical mechanics. *Random Structures and Algorithms*, 9(1):223–252, 1996.

[54] L. R. Rabiner and B. H. Juang. An introduction to hidden Markov models. *IEEE ASSP Magazine*, pages 4–15, January 1986.

[55] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[56] R. Rockafellar. *Convex Analysis*. Princeton University Press, 1970.

[57] S. Roweis and Z. Ghahramani. A unifying review of linear Gaussian models. *Neural Computation*, 11(2), 1999.

[58] B. Schölkopf and A. Smola. *Learning with Kernels*. MIT Press, 1st edition, 2002.

[59] M. Seeger. Bayesian model selection for support vector machines, Gaussian processes and other kernel classifiers. In Solla et al. [65], pages 603–609.

[60] M. Seeger. Learning with labeled and unlabeled data. Technical report, Institute for ANC, University of Edinburgh, UK, 2000. See `www.kyb.tuebingen.mpg.de/bs/people/seeger`.

[61] M. Seeger. *Bayesian Gaussian Process Models: PAC-Bayesian Generalisation Error Bounds and Sparse Approximations*. PhD thesis, University of Edinburgh, July 2003. See `www.kyb.tuebingen.mpg.de/bs/people/seeger`.

[62] M. Seeger. Gaussian processes for machine learning. *International Journal of Neural Systems*, 14(2):69–106, 2004.

[63] M. Seeger, N. Lawrence, and R. Herbrich. Efficient nonparametric Bayesian modelling with sparse Gaussian process approximations. Technical report, MPI Biological Cybernetics, Tübingen, 2006. See `www.kyb.tuebingen.mpg.de/bs/people/seeger`.

[64] E. Snelson and Z. Ghahramani. Sparse Gaussian processes using pseudo-inputs. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems 18*. MIT Press, 2006.

[65] S. Solla, T. Leen, and K.-R. Müller, editors. *Advances in Neural Information Processing Systems 12*. MIT Press, 2000.

[66] D. Spiegelhalter, A. Thomas, N. Best, and W. Gilks. BUGS: Bayesian inference using Gibbs sampling. Technical report, MRC Biostatistics Unit, Cambridge University, 1995.

[67] M. Stein. *Interpolation of Spatial Data: Some Theory for Kriging*. Springer, 1999.

[68] R. Swendsen and J.-S. Wang. Nonuniversal critical dynamics in Monte Carlo simulations. *Physical Review Letters*, 58(2):86–88, 1987.

[69] B. Taskar, C. Guestrin, and D. Koller. Max-margin Markov networks. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16*. MIT Press, 2004. To appear.

[70] M. Tipping. Sparse Bayesian learning and the relevance vector machine. *Journal of Machine Learning Research*, 1:211–244, 2001.

[71] M. Tipping and C. Bishop. Mixtures of probabilistic principal component analyzers. *Neural Computation*, 11(2):443–482, 1998.

[72] D. Titterington, A. Smith, and U. Makov. *Statistical Analysis of Finite Mixture Distributions*. Wiley Series in Probability and Mathematical Statistics. Wiley, 1st edition, 1985.

[73] M. Wainwright, E. Sudderth, and A. Willsky. Tree-based modeling and estimation of Gaussian processes on graphs with cycles. In T. Leen, T. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 661–667. MIT Press, 2001.

[74] M. J. Wainwright and M. I. Jordan. Graphical models, exponential families, and variational inference. Technical Report 649, UC Berkeley, Dept. of Statistics, 2003.

[75] C. Williams and C. Rasmussen. Gaussian processes for regression. In D. Touretzky, M. Mozer, and M. Hasselmo, editors, *Advances in Neural Information Processing Systems 8*. MIT Press, 1996.

[76] C. K. I. Williams and D. Barber. Bayesian classification with Gaussian processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(12):1342–1351, 1998.

[77] D. Wipf and B. Rao. L0-norm minimization for basis selection. In L. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*. MIT Press, 2005.

[78] X. Zhu, Z. Ghahramani, and J. Lafferty. Semi-supervised learning using Gaussian fields and harmonic functions. In T. Fawcett and N. Mishra, editors, *International Conference on Machine Learning 20*. Morgan Kaufmann, 2003.