The Bayesian Approach to Linear Regression

Vidhi Ramesh Lalchand

Department of Physics

Supervisors:
Dr. Anita Faul
Dr. Christopher Lester

Christ's College
The Alan Turing Institute

September 29, 2017

The report seeks to serve as a stand-alone introduction on the subject of Bayesian Inference in the context of Regression. The first half of the report acts as a primer to the latter half which talks about a specific probabilistic interpretation of the linear regression problem, the approach is called *Sparse Bayesian Learning* (SBL). The reason for this is that SBL interweaves a host of concepts which deserve explanations of their own.This not only seems to be critical for understanding but also clarifies how the probabilistic solution actually differs from other estimates.Towards the end this report, sub-areas in the field which offer opportunities for research and exploration are discussed without attempting to offer any solutions at this time. The report preserves a generic style allowing it to be used as groundwork chapters towards a larger thesis on the subject.

Contents

List of Figures

Part I

Chapter 1 of this report is a brief introduction to some rudimentary concepts in machine learning. The intention is to concentrate on ideas which provide intuition for the Bayesian approach by contrasting it with the frequentist approach. The chapter also introduces probability densities which are the bedrock of Bayesian learning. As a supplement to 1, Appendix A is a tutorial on the Gaussian density. It provides a suitably detailed discussion around its nature and properties. Chapter 2 narrows its focus on Bayesian inference and Chapter 3 pursues a recap of the linear regression problem and highlights the different methods of estimation.

Part II of the report moves on to Sparse Bayesian Learning and offers a compendium of derivations for the inference procedure in the context of regression. Sparse Bayesian learning (SBL) is a type of Bayesian inference which leads to *sparse* solutions. In a regression system this means that the parameter of interest, the coefficients have very few non-zero values. Sparsity is a desirable property as it is an efficient way to control the complexity of a model and prevent over-fitting. Further, making predictions with sparse models is very computationally efficient. Traditionally, a regression system with input/output pairs $\{\mathbf{x}_i, y_i\}_{i=1}^N$ where $\mathbf{x}_i \in R^d$ assumes a fixed design matrix $\Phi = [\phi_1(\mathbf{x}), \phi_2(\mathbf{x}), \ldots, \phi_m(\mathbf{x})]$ with potentially non-linear basis functions $\phi_i(\mathbf{x})$. The choice of candidate basis functions is generally fashioned from the estimated relationship between the dependent and independent variables. If one wants to model the dependent variable y as an nth degree polynomial in $\mathbf{x}$ then a design matrix with polynomial functions of different orders is chosen. One of the research goals of this Ph.D is to take a closer look at the task of selection of basis functions, as the performance of a regression system is tightly linked to the choice of basis functions. Can the selection of these functions be automated? Can it be a part of the learning/training step? Further, what approaches are there to automatically select good parameters for the basis functions.

Chapter 6 is a short introduction to kernels with a focus on the nature of the Gaussian kernel. Chapter 7 of this report consists of regression experiments using the SBL framework on artificial data, this is to demonstrate SBL at a conceptual level. We conduct experiments with different choices of basis functions on a one-dimensional input domain (to aid visualization) and output. We narrow our focus on Gaussian kernels and show the impact of altering the internal parameters of the kernel on the regression outcome. Chapter 8 outlines the research tasks and goals ahead.

A large chunk of standard results surrounding probability densities have been derived and included in Appendix B. In Appendix C there is a parallel version of Chapter 4 which presents the derivation of the Sparse Bayesian learning marginal likelihood and computes first derivatives with respect to the hyper-parameters and noise variance.

Chapter 1

Preliminary Concepts

In machine learning parlance there are two main classes of learning - *supervised* and *unsupervised*, the former uses labelled data[1] and learning occurs by finding a generalized mapping between the input and output, also known as *targets*. This mapping can then be used to map new inputs to their output. Unsupervised learning, as the name suggests deals with unlabelled data and the goal is to discover hidden or latent structure. Cluster analysis, is an example of unsupervised learning where the task is to identify clusters from unlabelled data. Clusters are a group of objects which are similar in some sense to each other than to others which are outside the cluster. Unsupervised learning comprises a host of other methodologies the most popular of which are anomaly/outlier detection and dimensionality reduction.

Supervised learning refers to the task of inferring from labelled data of the form $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$ a mathematical function $h : D \rightarrow Y$ where $D = \{\mathbf{x}\}_{i=1}^{N}$ is the input space, $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in Y$ (y_i's are typically scalar) is the output space. This function h can then be used to predict labels on unseen data. When the labels y_i are categorical the task is *classification*, alternatively, in the case of real-valued labels $y_i \in \mathbb{R}$ the task is that of *regression*. $\mathbf{x}_i$ refers to the d dimensional feature vector of the i-th data point. Hence, each data point is characterized by several features.

A supervised learning algorithm needs to have the ability to generalize to unseen data. In this sense, the function needs to capture the relationship between the input and output well enough to 'learn' something from the data but not overfit the observed data. Section 3 and 4 discuss this aspect of supervised learning.

Most of the supervised learning algorithms predict point estimates (regression) and categories (classification) without providing an uncertainty window around the prediction or an associated probability value that captures the confidence in the prediction. Support vector machines, neural networks, decision trees in their traditional usage provide point estimates. A common way to extract uncertainty estimates when using classification

[1] Data which consists of input/output mappings, the outputs are typically scalars and the inputs are n-dimensional vectors

algorithms that do not provide them directly is through bootstrapping (sampling with replacement) and repeating the training and prediction step. The idea is that a perturbation of the training data will yield a modified model and predictions. Some other techniques for coercing probability estimates is to apply a logistic transform to the output of an algorithm to give estimates between 0 and 1 which can then be interpreted as a probability, this technique is called Platt scaling [P+99]. Such calibration techniques offer a way to quantify uncertainty around predictions but for uncertainty estimates around the parameters of a trained model one needs an approach that is probabilistic by construction.

Bayesian learning uses rules from probability theory to provide a complete framework for probabilistic learning from data. The first key element is that it treats parameters of a model as random variables rather than fixed unknowns. The 'learning step' entails inferring a probability distribution over the parameters which in turn lead to probability distributions for the predictions.

To make this clearer, we will contrast the Bayesian and non-Bayesian approach to modelling with the aid of some notation.

Given some input/output data $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$, we are interested in a mathematical function $h : D \to Y$ where D denotes the input. As a first step we make an educated guess about the parametric form of the function h reducing the modelling task to finding good values of the parameters of the chosen model.

Without loss of generality lets denote this parametric form $h(D, w)$ where w are parameters that feature in the function and are the main target of the learning exercise.

Until now there is no distinction between the Bayesian and the non-Bayesian world.

The distinction arises in how w is learnt in the two settings. In the non-Bayesian (NB) setting, w is learnt by a process of tuning, which entails changing values of w to optimise a certain quality metric. In regression, an example quality metric could be the bias of a model, in a classification context this could be the accuracy rate.

In the non-Bayesian setting, we take an inherently different approach. The problem of finding the relationship between the inputs and outputs is encoded as a conditional probability.

$$p(Y|D) = h(D, w) \tag{1.1}$$

and most importantly, w is treated as a random vector with a probability distribution which is inferred from the Bayes' rule. How this inference takes place is the subject of chapter 2.

Crucially, Bayesian learning provides a representation of uncertainty in the model through the use of probability distributions. The result of Bayesian learning algorithms is a

posterior[2] probability distribution of the model parameters given the data $p(\mathbf{w}|D)$ rather than a point estimate. This gives information about the uncertainty bounds around the estimated parameters $\mathbf{w}$. In order to obtain the posterior distribution over the parameters one has to specify a 'prior' distribution $p(\mathbf{w})$ encoding our beliefs about the parameter values. While this may seem controversial at first, the ability to specify a prior summarizing ones knowledge about the parameter $\mathbf{w}$ might in some practical situations be an advantageous feature of Bayesian learning.

.

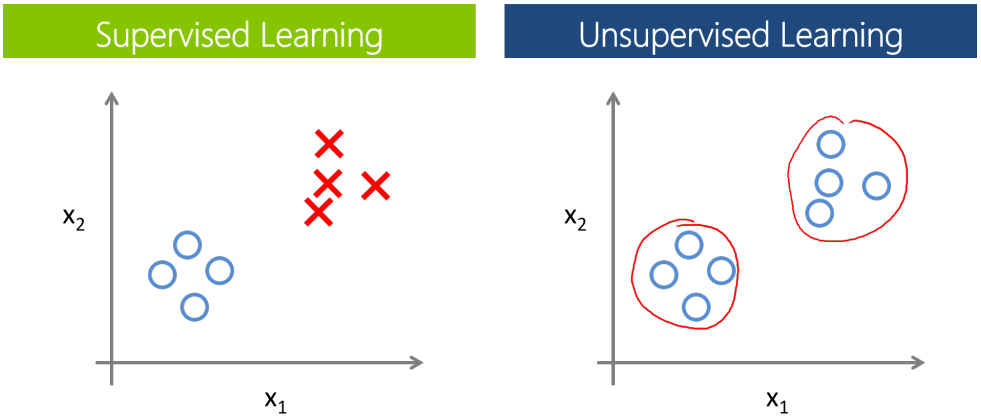


Figure 1.1: This illustration depicts the nature of the input data in supervised and unsupervised learning. In supervised learning, the input data are labelled and the task of the learning algorithm is to learn the mapping between the inputs and labels so as to be able to correctly label unseen inputs. In unsupervised learning, the task of the learning algorithm is to identify latent patterns or structure in the data.

1 Frequentism vs. Bayesianism

Herein we will use the symbol $p()$ to denote a probability density and $P()$ to denote a probability (of an event).

A fair introduction to Bayesian inference cannot be complete without contrasting it with inference in frequentism. Frequentists define probabilities in terms of frequencies or repeated events [Van14]. It is in fact calculated as the limiting case of repeated events. For instance, if we flip a coin a large number of times, the proportion of those tosses that lead to heads is the unconditional probability of a head. In the limit of a large number of repeated measurements, the probability of an event is computed as the frequency of any given value over the total number of measurements. The equation,

$$p = \lim_{n\to\infty} \frac{k}{n} \tag{1.2}$$

[2]The probability distribution after the data/observations have been taken into account.

where k is the number of successes and n is the number of trials computes the probability of success. In strict frequentist statistics there is no way to assign a probability distribution over a fixed measured value. Parameters are treated as fixed quantities.

A 95% confidence interval around a parameter value θ in the frequentist view indicates that if the experiment is repeated statistically *large* [3] number of times, in 95% of the cases the computed confidence interval will contain the true value of θ. In other words, the end points of the confidence interval are interpreted as random variables rather than the parameter. We can only talk about the probability that the interval includes the true parameter, rather than the probability that the parameter is inside the interval.

In Bayesian statistics parameters are described as random variables described with an associated probability distribution. Bayesians use probabilities to express degrees of certainty about events and the construction of this probability is tied to a prior. A prior refers to past knowledge about a certain event. It offers a way to codify knowledge about an event into a probability.

The Bayes' rule is the central equation in Bayesian inference and it provides the machinery to update probabilities in as more data/evidence becomes available.

The qauntity of interest is the probability distribution of the parameter (say, k) given the data D, $p(k|D)$. A standard application of Bayes' theorem enables one to compute this through,

$$\underbrace{p(k|D)}_{posterior} = \frac{\overbrace{p(D|k)}^{likelihood}\,\overbrace{p(k)}^{prior}}{p(D)} \tag{1.3}$$

Further, in Bayesian inference one talks about credible regions which have the more natural interpretation that given a 95% credible region we assign a 95% probability that the parameter is inside that region. These are simply regions in the parameter space over which the posterior distribution is integrated to have a pre-specified probability.

2 The Bayes' Rule and Bayesian Inference

In probability theory, Bayes' rule states for events A and B:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{1.4}$$

[3] How large is large enough depends on the effect we are trying to measure, if we are trying to prove that a coin is fair and toss a coin around 10 times we might not get a heads to tails ratio of 1:1 however if we conduct 1000 tosses we might find that the ratio is very close to 1:1.

where $P(B) \neq 0$, and $P(A)$ and $P(B)$ are the unconditional probabilities and $P(A|B)$ and $P(B|A)$ are the conditional probabilities. In probability theory, a conditional probability denotes the probability of an event of interest A given that another event B has occurred. It is denoted as $P(A|B)$, they play an important role in probability theory as if events A and B are connected, knowledge about one of them occurring can affect the probability of the other event occurring. If the events A and B are independent and do not impact each other then the conditional probability is the same as the unconditional probability .i.e. $P(A|B) = P(A)$. The power of Bayes' rule is that in many situations where we want to compute $P(A|B)$ it turns out that it is difficult to do so directly, yet we might have direct information about $P(B|A)$. Bayes' rule enables us to compute $P(A|B)$ in terms of $P(B|A)$.

Bayes rule finds its main application in Bayesian inference. Advantageously, Bayes' rule (explained in terms of events) can be generalized to probability densities. In Bayesian inference we have some data, X and a model defined by some parameters w. The aim is to find the distribution over the parameters w after taking into account the data X. The different building blocks in Bayesian inference are,

1. A prior distribution of the parameters $p(w)$.
2. A likelihood term $p(X|w)$ viewed as a function of w.
3. A posterior probability $p(w|X)$.

Bayesian inference relies on the computation of a posterior probability of model parameters $p(w|X)$, given some data/evidence X. A prior $p(w)$ is pre-specified based on best available past knowledge. Further, $p(X|w)$ is the likelihood which captures the probability of the evidence given a model choice. $p(X)$ is the unconditional probability of observing the data/evidence X. $p(X)$ is sometimes also referred to as the marginal likelihood as it is the probability of the data X where the dependence on the parameter w has been marginalised (integrated) out. It is probability of X independent of all parameter values and hence can be computed as,

$$p(X) = \int p(X|w)p(w)dw. \tag{1.5}$$

$$p(w|X) = \frac{p(X|w)p(w)}{p(X)} \tag{1.6}$$

This equation is the cornerstone for Bayesian inference.

The shorthand version (which ignores the denominator as the different values of the parameter w do not depend on the probability distribution of the data $p(X)$.)

$$Posterior \propto Likelihood \times Prior \tag{1.7}$$

where $\propto$ denotes proportionality.

2.1 An example to contrast the two approaches

It might be worthwhile to present an example where a question of inference is answerable in both frequentist and Bayesian approaches.

Consider the classical coin flip example, where we are trying to establish if a coin is fair or not based on the results of 10 tosses. Let k be the probability of heads ($1-k$ is the probability of tails). We flip the coin 10 times and it comes up with 7 heads, the frequentist approach would deem the best estimator of k to be equal to the ratio $7/10 = 0.7$. This is also, the maximum likelihood estimate.

Under the Bayesian approach we are interested in the posterior probability density of k rather than a single estimate. In order to construct this we apply the Bayes' rule.

$$p(k|h=7,t=3) = \frac{p(h=7,t=3|k)p(k)}{p(h=7,t=3)} \tag{1.8}$$

The value $p(h=7,t=3|k)$ represents the probability of obtaining 7 heads in 10 tosses of a coin where the probability of obtaining heads is k, this follows a binomial distribution,

$$p(h=7,t=3|k) = \binom{10}{7} k^7(1-k)^3 = \frac{10! \times k^7(1-k)^3}{7! \times 3!} \tag{1.9}$$

Further, the unconditional probability of $p(h=7,t=3)$ can be estimated by summing over all possibilities of the value k. We know that k is a value between 0 and 1 hence, the term is the integral $\int_0^1 \binom{10}{7} p^7(1-p)^3 dp$.

$p(k)$ denotes the prior which is chosen to be uniform in the interval [0,1], hence $p(k) = 1$

Substituting this information in equation 1.8. We get,

$$p(k|h=7,t=3) = \frac{k^7(1-k)^3}{\int_0^1 p^7(1-p)^3 dp} \tag{1.10}$$

Notice that the binomial coefficients in the numerator and denominator get cancelled.

Also, the denominator is a beta function, $B(8,4) = \dfrac{\Gamma(8)\Gamma(4)}{\Gamma(12)} = \dfrac{7! \times 3!}{11!}$. This simplifies equation 1.10 to,

$$p(k|h=7,t=3) = \frac{11!}{7! \times 3!} k^7(1-k)^3 = 1320 \times k^7(1-k)^3 \tag{1.11}$$

Expressing this posterior for different values of k.

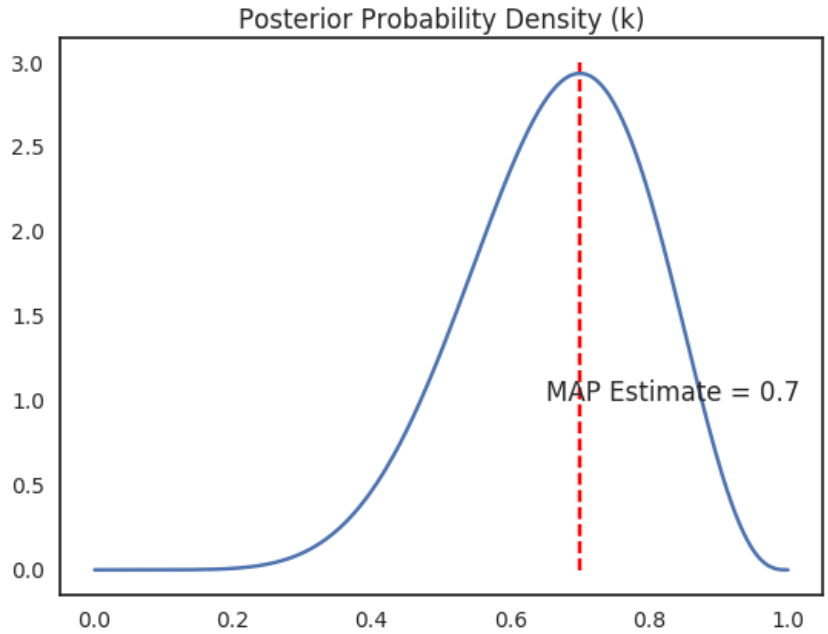


Figure 1.2: Posterior probability distribution of k with MAP Estimate

Note that, the posterior is maximised at the point $k = 0.7$, this is the maximum a posteriori estimate (MAP)[4]. In this case it is the same as the frequentist solution due to the fact that a uniform prior was used. Maximum likelihood estimates (MLE) and MAP estimates coincide in the case of uniform priors.

The probability for an unbiased coin (which in this example is the probability of the coin coming down heads 50% of the times) can then be inferred from this posterior distribution as the integral of the curve between (say, for practical purposes) 0.45 and 0.55.

$$P(0.45 < k < 0.55) = \int_{0.45}^{0.55} p(s|h = 7, t = 3)ds \tag{1.12}$$

$$= 1320 \int_{0.45}^{0.55} s^7(1-s)^3 ds \tag{1.13}$$

$$\approx 13\% \tag{1.14}$$

Further, the expected value of k under this posterior is,

$$E(k) = \int_0^1 kp(k|h = 7, t = 3)dk = \frac{2}{3} \tag{1.15}$$

[4]The mode of the posterior probability distribution.

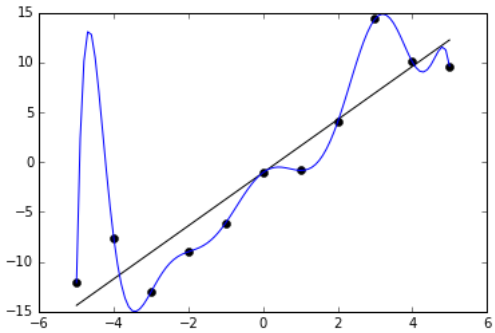


Figure 1.3: This figure illustrates a 15 degree polynomial and a linear line fitted to a group of data points. It is clear that while the curve fits the data exactly, it is a poor model of the data. The linear model is likely to fair better on new data.

If one were to use the expected value as the Bayesian estimate then we deduce that the Bayesian estimate for k (0.66) is more skewed to the fair coin hypothesis than the frequentist estimate (0.7). This is because in computing the Bayesian estimate we integrate (average or marginalise out) over all possibilities of k.

3 Occam's Razor

In model selection, it is not just enough to select the model which 'fits' the data the best. In curve fitting, fitting a polynomial with several terms will fit the training data better than a quadratic curve or a linear line however, choosing the former over the latter exposes one to the risk of over fitting and poor generalization on new data.

Occam's razor (also translated as the *law of parsimony*) refers to the principle of selecting the model with the least assumptions, in essence the simpler model to explain the data. It is a heuristic technique and meant to offer some guidance in the scientific practice for model selection. The idea is that simpler models usually generalize better to unseen data and avoid over-fitting. The Bayesian justification for Occam's Razor is that simpler models are just more probable than over-complex ones. In this sense, Bayesian inference embodies Occam's razor by attributing higher posterior probabilities to simpler models [Mac92b].

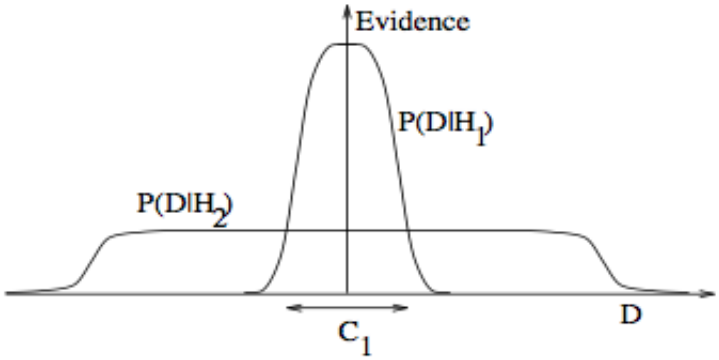


Figure 1.4: Bayesian learning and Occams Razor

Figure 1.4 a schematic first presented in [Mac92b] and then used by several authors in discussions around Bayesian model comparison illustrates how Bayesian inference embodies Occam's razor. The horizontal axis is the abstract space of all possible datasets which have been ordered in such that more 'simple' data sets are concentrated in the center of the axis.

The quantity of interest in Bayesian analysis the posterior probability distribution of the model given the data $P(H_i|D)$. This is inferred by applying the Bayes' rule:

$$P(H_i|D) = \frac{P(D|H_i)P(H_i)}{P(D)} \tag{1.16}$$

The term $P(D|H_i)$ on the right hand-side depicts how much the model H_i predicted the data that occurred, this is also called *model evidence*. Assuming that the subjective term $P(H_i)$ is flat such that $P(H_i|D)$ is directly proportional to the evidence term. The key idea is that the evidence $P(D|H_i)$ becomes the basis for comparing between models. A simple model H_1 is able to predict only a small range of the datasets as opposed to model H_2 which is a more complex model with a large number of free parameters. However, H_2 is not able to predict the datasets in region C_1 as strongly as H_1. If the dataset falls in region C_1 the less powerful model H_1 would be the more probable and preferred one [Mac92a]. In the event that unequal or biased priors come into play, just looking at the evidence will not provide the complete picture for model selection.

Further, datasets suffer from statistical noise which are minor unexplained variations in the data. More complex models are at risk of modelling the noise rather than capturing the patterns in real data.

4 Bias-Variance Tradeoff

A related concept in machine learning parlance is the bias-variance trade-off refers to the problem of balancing model accuracy and model generalization ability, it is usually

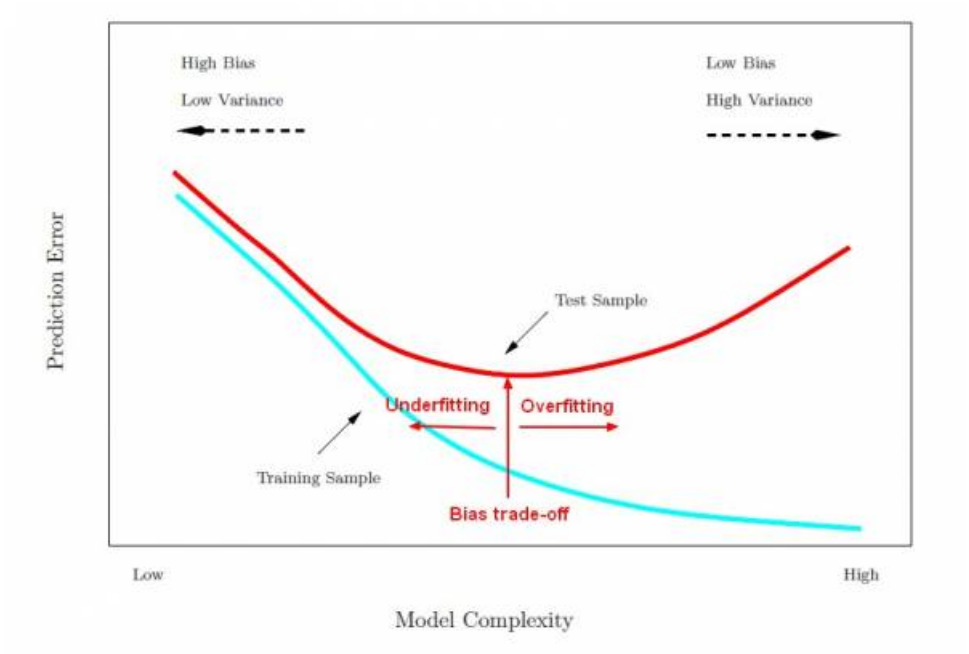


Figure 1.5: This figure illustrates the principle of model selection within the context of the bias-variance tradeoff. The red curve depicts generalization error, the blue curve depicts training error. The figure shows that it is possible to achieve arbitrarily accurate models in fitting to a training dataset, such models usually define complex rules to capture the hidden relationships in training data. However, the effectiveness of such models on unseen data is poor as they are too over-fitted to the training set. There is an inflection point where the error on test data starts to degrade (increase). Good models position themselves at the point when this starts to happen.

impossible to improve both simultaneously. A highly accurate model that captures the regularities of the training data well is a low-bias model. A low-bias model comes at the cost of high model complexity which can lead to the problem of over-fitting. An over-fitted model is sensitive to small fluctuations in the training data set. Even a slight perturbation of the training data leads to a substantially different model structure. Hence, low bias usually occurs with high variance. The high degree polynomial in fig. 1.3 is an example of a low-bias high-variance model. Any new data points between -5 and -4 would fit poorly due to the drastic over swings of high order polynomials.

On the flip side, a simple model which is more deterministic in its output is a high-bias low-variance model. It is too simple to capture the hidden relationships between the training data and output leading to high-bias (low accuracy) and under-fitting but has low variance. An under fitted model is stable to small perturbations in the training data. Hence, high bias usually occurs with low variance.

The error or misclassification rate on the training data is usually used to measure bias of a model. The variance of the model is calculated as the variance of the predictions of the

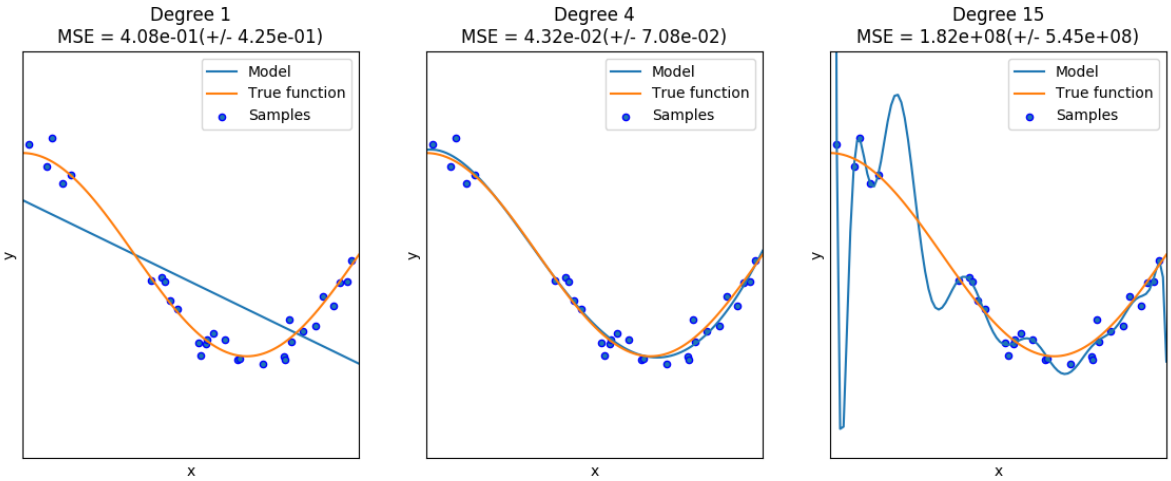


Figure 1.6: Overfitting and Underfitting in a regression context. The mean squared error is calculated on the validation set.

model on test data. There is a negative relationship between the bias and variance of a model, this makes it difficult to moderate both measures simultaneously. Increasing the accuracy of a model (lowering bias) is usually accompanied by increasing variance. This is because increasing model accuracy is done by fitting the model more to the training data[5], this increases the complexity of a model, hence increasing the variance of new predictions. Lowering the accuracy of a model is usually done by taking away some free parameters, this makes the model fit less accurately to the training data, a simple model will exhibit lower variance in new predictions. Hence, increasing the bias is accompanied by reducing the variance.

Robust models lie roughly in the middle of the bias-variance spectrum with an acceptable amount of bias and variance. The goal of parameter optimization is usually to find this point on the bias-variance trade-off curve.

4.1 Inference, Learning and Prediction

The use of these words under alternate phrasings has prompted this note to clarify the difference between these words commonly encountered in the machine learning lexicon.

Prediction can be thought of as an aim of learning (training a model). When the trained system is put to work using what it has learned, for instance to recognize images, cells, particles or suggest the direction the stock price is likely to move - it performs a prediction. What precedes prediction is *learning* which is specific to the model being used. For instance, in a linear regression model,

[5]For instance, this can be done by increasing the number of parameters in the model.

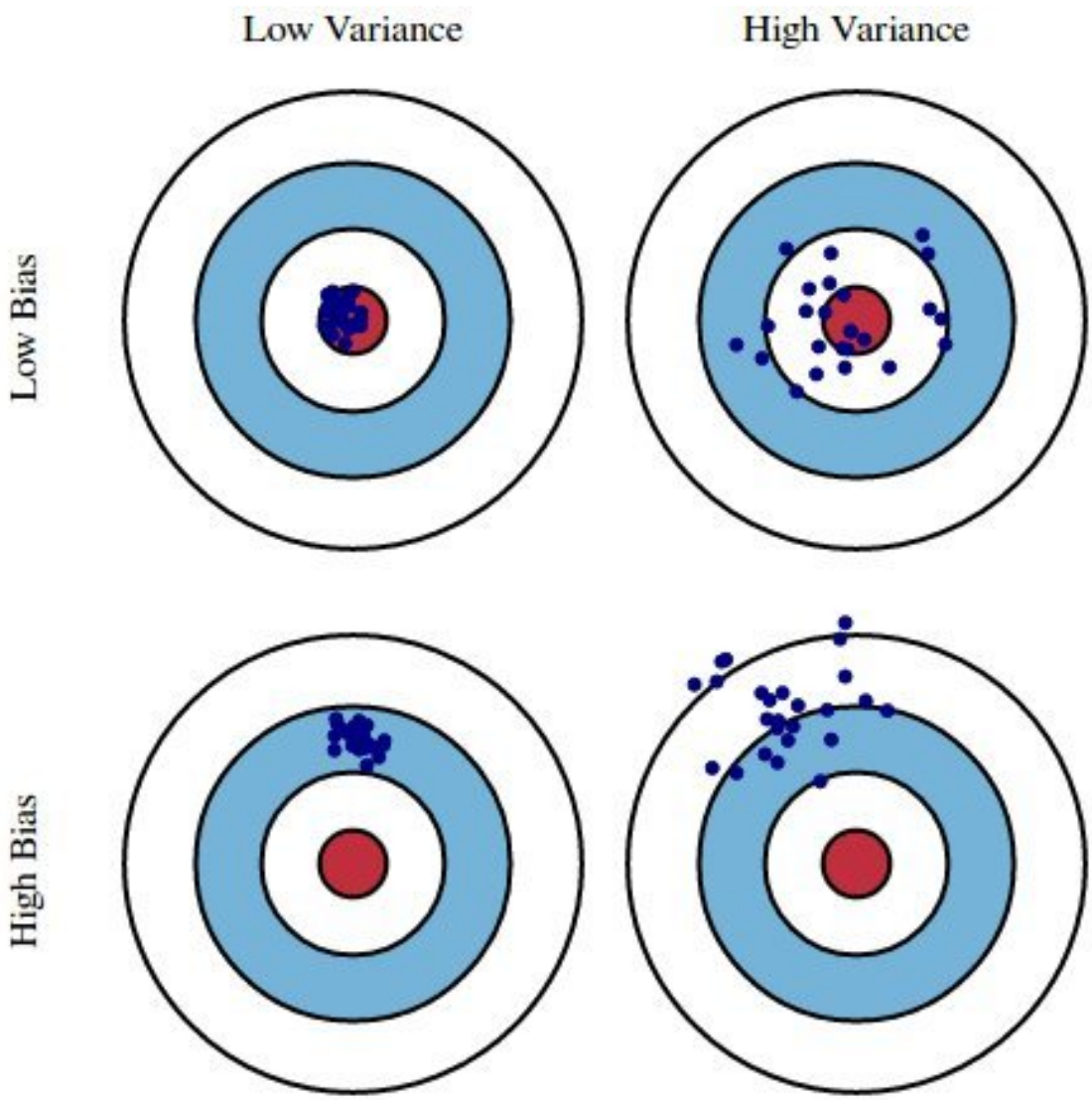


Figure 1.7: This figure illustrates the different combinations of bias and variance. The red center represents the true value target that the predictions need to fit to. The concentric circles represent different levels of diffusion. Low bias-low variance models present a theoretical standard - this is where models want to be. High-bias high variance models represent the attributes of an ill-defined model. Most models fall into the other two categories of being either low-bias high-variance or high-bias low-variance.

$$y_i = \beta_0 + \beta_1 x_i \tag{1.17}$$

the quantities of interest β_0 and β_1 need to be learnt to be able to compute targets for new data. Hence, parameter estimation can be thought of as learning.

Inference is something that usually follows the learning step. In a neural network for instance, the weights are 'learnt' by careful fine-tuning of the model for generalisation. Once we have the trained neural network, we can infer the values of the hidden layer neurons for a specific data point. When we pass a new data point through the trained neural network and get an answer out, we are essentially performing a prediction.

In the Bayesian world, learning is thought of as performing an inference. The term probabilistic inference typically means calculating posterior distribution given the data.

Caveat: It is important to note that the usage of this terminology as described here is by no means universal. Bishop's book uses 'inference' and 'decision' to mean 'learning' and 'prediction'. There are some sources that point out the difference in usage is tied to the field of the modeller. What is an 'inference' to a statistician could be 'learning' to a computer scientist.

We use the term 'Bayesian learning' to mean the general application of Bayesian methods of inference. The specific task of deriving a posterior probability distribution for the model unknowns is referred to as Bayesian inference.

5 Probability Densities

In probability theory, a random variable (rv) X has an associated probability distribution which summarizes the probabilities for different outcomes of the random variable (in a sample space S). If the random variable is discrete, for instance if X is the rv denoting the outcome of 2 coin tosses, X can take values, [0,1,2] with probabilities [0.25,0.5,0.25] which sum upto 1. This is an example of a discrete probability distribution. The function which assigns the probabilities to outcomes is called the *probability mass function* (pmf). Bernoulli, Poisson and Binomial are all examples of discrete probability distributions where the associated rv takes on discrete values.

$$\sum_{x \in S} p(X = x) = 1 \tag{1.18}$$

In fig. 1.8 the bars represent the discrete values X can take, and the height of the bars represent the probability of X taking on that value. To calculate the probability of X taking more than one value, the respective probabilities are just summed across the bars.

When a random variable X takes values in a continuous range, then it is a continuous

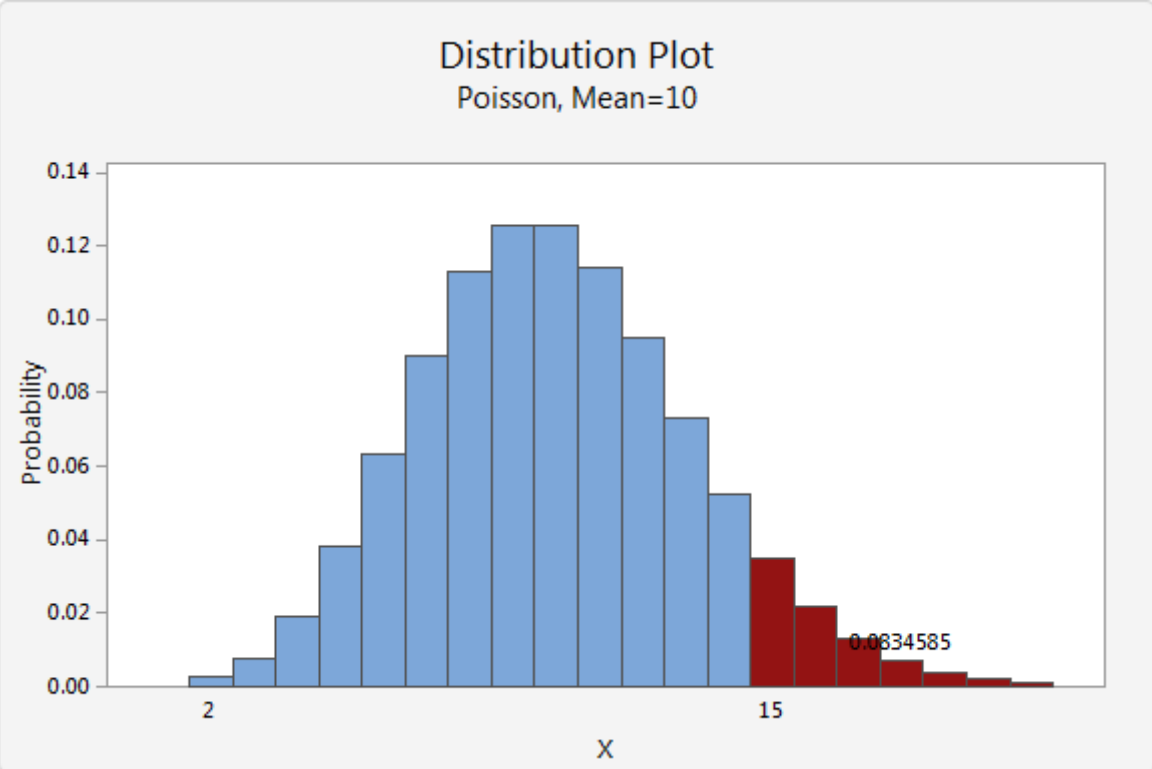


Figure 1.8: Discrete probability distribution function

random variable with an associated probability density function. The probability density function does not give probabilities directly, it gives a density and must be integrated over an interval to give a probability.

The fig. 1.9 gives the example of a continuous probability density function. The density shown here is the normal density[6] where the rv X takes values in the range [140, 210]. The probability of X taking a value between 160 and 170 denoted as $Pr(160 < X < 170)$ is given by the integral,

$$\int_{160}^{170} p(x)dx = P(160 < X < 170) \tag{1.19}$$

Probability in the continuous case is given by area under the curve of the probability density function in the interval of interest. The continuous analog of the probability mass function is the probability density hence the total area under it must sum to 1,

$$\int_{-\infty}^{\infty} p(x)dx = 1 \tag{1.20}$$

[6]Also called the Gaussian density which is discussed in detail in the Appendix A.

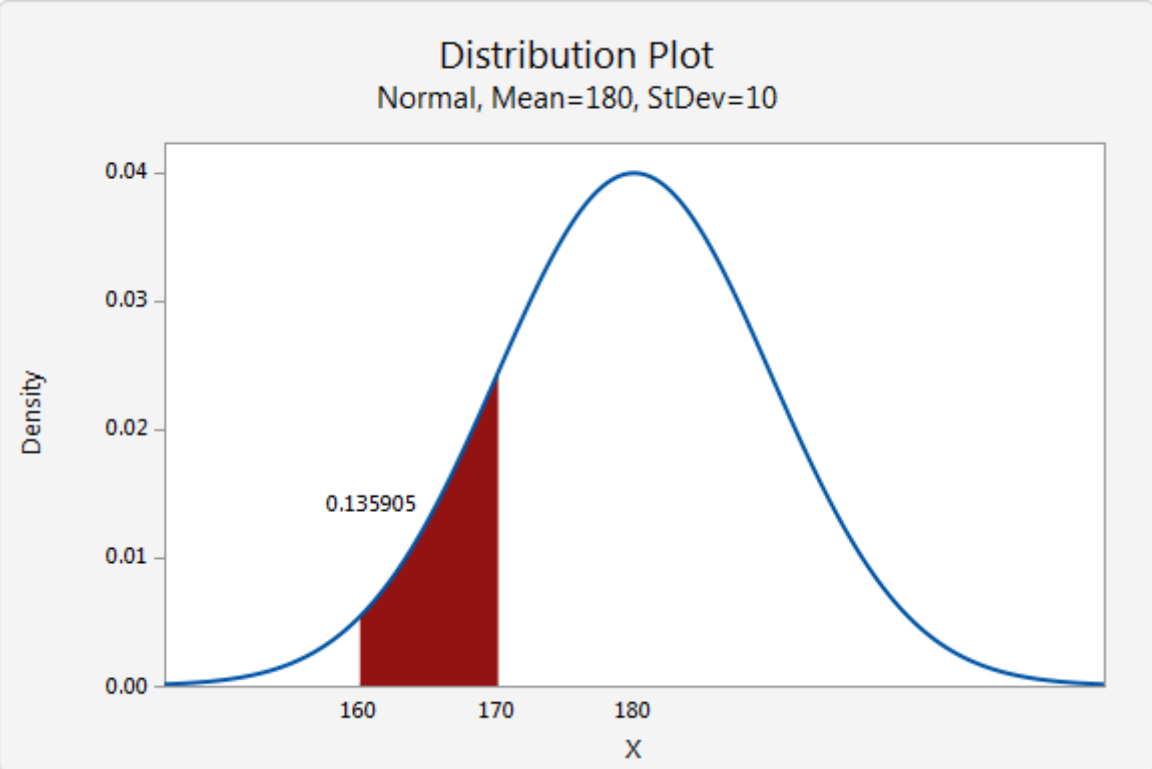


Figure 1.9: Continuous Probability density function

The probability of a continuous variable X taking on a specific value x is 0 as the integral with coinciding upper and lower limits is 0. Probability density functions are the bare bones foundation on which all of Bayesian learning relies.

5.1 Joint, Marginal and Conditional densities

This section delves into some fundamental extensions to probability density functions which are encountered repeatedly in Bayesian learning. It is important to re-iterate that in the discrete case, there is no distinction between a probability and the result of a probability mass function, they are essentially the same. In the continuous case, the result of a probability density function is a not a probability but a density, which when integrated over a desired range gives a probability. This motivates the idea of using integrals to compute probabilities in the continuous case.

For a set of continuous random variables $X_1, \ldots, X_n$ the joint probability density function denotes the probability that each of the n random variables take values in a range or domain.

$$P(X_1, \ldots, X_n \in S) = \int_S p(x_1, x_2, \ldots, x_n) dx_1, dx_2, \ldots, dx_n. \tag{1.21}$$

If the rvs are mutually independent then their joint density is just a product of their individual densities.

$$p(X_1, \ldots, X_n) = p(X_1), \ldots, p(X_n) \tag{1.22}$$

If a collection of random variables $X_1, \ldots, X_n$ form a random vector and have a joint density, the density of a single component $p(X_i)$ is the marginal probability density. The joint density describes the multivariate density (of the random vector) whereas the marginal density describes the density of just one component of this random vector. The two are closely related as it is possible to derive the marginal density from the joint density by integrating over all the other components, this is called marginalizing.

$$p(X_i) = \int_{-\infty}^{\infty}, \ldots, \int_{-\infty}^{\infty} p(x_1, \ldots, x_n) dx_1 \ldots dx_{i-1}, dx_{i+1} \ldots dx_n \tag{1.23}$$

The integral is replaced by a sum, if the variable is discrete.

Fig.1.10 depicts the joint density of two Gaussian random variables.

The conditional probability links the joint probability and the marginal probability via the product rule. The concept of a conditional probability density is best understood when considering two continuous variables X and Y. The conditional probability density of Y given the value x of X is,

$$p(Y|X = x) = \frac{p(X, Y)}{p(X)} \tag{1.24}$$

where $p(X, Y)$ is the joint density and $p(X)$ is the marginal density.

Now, consider the conditional density of X given some value y of Y,

$$p(X|Y = y) = \frac{p(X, Y)}{p(Y)} \tag{1.25}$$

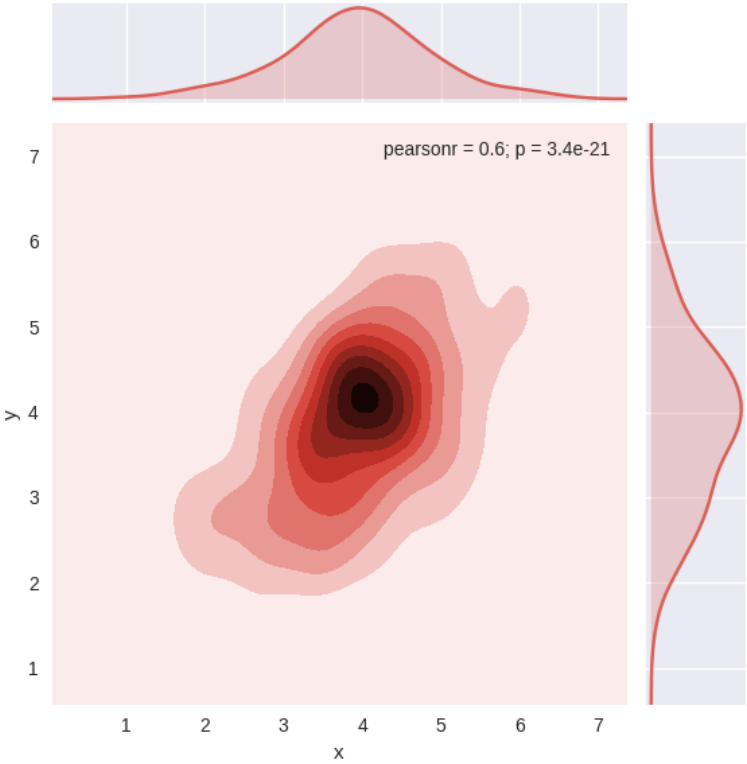
Amalgamating the two together gives,

$$p(Y|X = x) = \frac{p(X|Y = y)p(Y)}{p(X)} \tag{1.26}$$

This is the Bayes' rule which was introduced in section 2.

Further,

$$p(X, Y) = p(X|Y)p(Y) = p(Y|X)p(X) \tag{1.27}$$



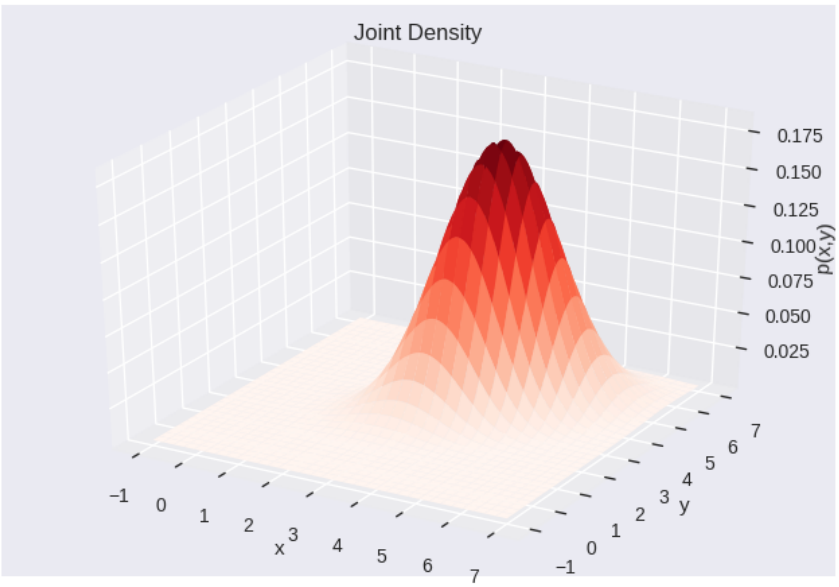


Figure 1.10: The mean vector is the [4, 4] and the covariance matrix is $\begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix}$

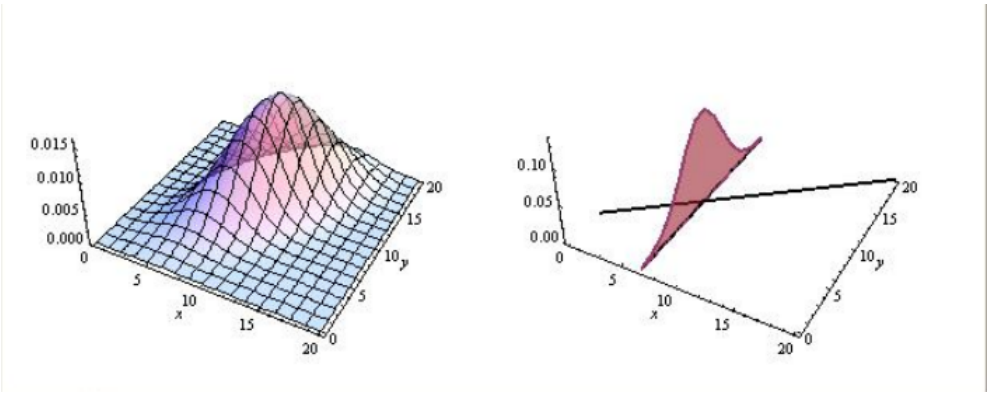


Figure 1.11: The graph visualises the conditional density $p(y|x = 7)$, for this imagine the line $x = 7$ on the xy plane and then imagine a plane containing that line perpendicular to the xy plane cutting through the bivariate bump. The intersection of the bivariate bump and the plane, (normalized to give unit area necessary for a pdf) gives the conditional density [Bou].

(we just write $p(X|Y)$ instead of $p(X|Y = y)$ as the conditioning variable is unambiguous)

Eq.1.27 is the product rule of probability.

$$p(Y) = \int p(x, y)dx \tag{1.28}$$

Eq. 1.28 is the sum rule of probability.

If X and Y are independent then the conditional density $p(X|Y = y)$ is just the marginal density $p(X)$, this is also evident from the definition.

Chapter 2

Bayesian Learning: A technical summary

Bayesian learning is the term given to methods of inference which involve computing a posterior probability distribution using two antecedents,

1. A prior probability distribution

2. A likelihood function

Given some data D and a chosen model M fully characterized by its parameters θ, the prior probability distribution $p(\theta)$ encodes our knowledge or prior belief about this model without taking the data D into account. After having observed the data D, we can compute the likelihood of D given the choice of the model parameters, $p(D|\theta)$. The fundamental objects of interest are the model parameter unknowns θ which are represented by the posterior probability distribution $p(\theta|D)$. Multiplying the prior with the likelihood and renormalizing the result (to obtain a probability distribution) gives the posterior probability distribution. The reason we get the posterior probability distribution out is precisely due to the Bayes' rule.

$$p(\theta|D) = \frac{p(D|\theta)p(\theta)}{p(D)} \tag{2.1}$$

Posterior distribution $\propto$ Likelihood function $\times$ Prior distribution.

In order to compute the posterior probability distribution, the Bayes' rule is applied. In essence, Bayesian learning offers a probability distribution over all possible parameter values of θ informed by the data.

Taking a look at each term in this expression:

1. $\mathbf{p}(\theta|\mathbf{D})$ is the posterior probability distribution.

2. $\mathbf{p}(\theta)$ is the prior which encodes what we know about the model a priori. It is important to note that this is the distribution of parameters before any data is observed.

3. $\mathbf{p}(\mathbf{D}|\theta)$ is the distribution of the observed data conditioned on its parameters. This is also called, the likelihood.

4. $\mathbf{p}(\mathbf{D})$ which is simply the normalization constant.

A critical point about Bayesian inference is that it provides a practical way of combining new evidence with prior beliefs through the application of the Bayes' rule. The posterior distribution from a previous study can often serve as a prior distribution for subsequent studies. This gives Bayesian inference an iterative flavour as the posterior probability can be recursively updated.

Formally, if $\mathbf{X}$ represents the data and θ represents the vector of parameters whose probability distribution we are interested in, and α is some vector of hyperparameters [1] .i.e. $\theta \sim p(\theta|\alpha)$ then, the posterior distribution is given by:

$$p(\theta|\mathbf{X}, \alpha) = \frac{p(\mathbf{X}|\theta)p(\theta|\alpha)}{p(\mathbf{X}|\alpha)} \tag{2.2}$$

The primary difficulty in Bayesian inference arises in computing the integral in the denominator,

$$p(\mathbf{X}|\alpha) = \int p(\mathbf{X}|\theta)p(\theta|\alpha)d\theta \tag{2.3}$$

as sometimes it could entail summing over too large a space of parameters. This computational intractability is sometimes addressed using approximate Bayesian methods like MCMC (Markov chain Monte Carlo).

In a Bayesian setting the posterior predictive distribution is obtained by marginalizing the posterior distribution of θ over all possible values. If x' is a new data point, instead of a fixed point estimate as a prediction, the distribution over all possible values of θ is returned.

$$p(x'|\mathbf{X}, \alpha) = \int p(x'|\theta)p(\theta|\mathbf{X}, \alpha)d\theta \tag{2.4}$$

This is in contrast to non-Bayesian methods of inference which involve coming up with a point estimate of the parameter. Determining the uncertainty around the parameter

[1] A hyperparameter is a parameter of a prior distribution. For example, if one is using a Gaussian distribution to model the distribution of the parameter θ, then, the scale (σ) of the Gaussian distribution is an example of a hyperparamter. They are distinguished from standard model parameters as they are not directly optimized in model training.

value can only be done by mapping the point estimates to a probability score via post-processing [P+99].

The MAP (maximum a posteriori) estimate is a pointwise estimate with a Bayesian flavor. Technically, it is the mode of the posterior probability distribution. We saw in fig. 1.2 an example of a MAP estimate.

$$\theta_{MAP} = \underset{\theta}{\operatorname{argmax}} \quad p(\theta|\mathbf{X}, \alpha) \tag{2.5}$$

Note that the denominator of the posterior distribution has no dependence on θ as it is marginalized out (eq. 2.3). Further, if the prior is uniform (A uniform prior refers to a probability distribution that does not favour any particular parameter value over the other, as such it is treated as having no impact on the posterior) then the MAP estimate coincides with the maximum likelihood estimate.

MAP estimates can be computed either in closed form (in the case of conjugate priors) or numerically via a monte carlo method.

Conjugate priors refer to choices of prior which give a posterior probability distribution belonging to the same family as the prior. As such a conjugate prior ensures that we can derive a closed form expression for the posterior (thus precluding the need for monte carlo like methods). A Gaussian prior is a conjugate prior.

Chapter 3

Taxonomy of Regression Systems

In this short chapter we shift our focus to regression in general, we take a look at the different taxonomies used to classify regression systems. This sets the tone for Part II of the report where we look at linear regression but under a Bayesian lense.

Based on the number of dependent and independent variables:

1. **Simple** Regression pertains to one dependent variable y and one independent variable x: $y = f(x) = w_1 x + w_0 + \epsilon$.

2. **Multiple** Regression pertains to one dependent variable y and multiple independent variables: $y = f(x_1, \ldots, x_n)$.

3. **Multivariate** Regression pertains to multiple dependent variables and multiple independent variables: $y_1, \ldots, y_m = f(x_1, \ldots, x_n)$.

Based on the parameters or coefficients:

1. **Linear** Regression assumes that the relationship between the dependent variable y and the vector of regressors $\mathbf{x} = (x_1, \ldots, x_p)$ is linear .i.e. the coefficient vector $\mathbf{w} = \{w_i\}_{i=1}^{p}$ appears linearly in the model, $y = w_0 + w_1 x_1 + \ldots + w_p x_p + \epsilon$.

2. **Non-linear** Regression expresses the relationship between y and $\mathbf{x}$ using a non-linear combination of model parameters $\mathbf{w} = (w_0, w_1)$. An example model of this type of regression would be, $y = w_0 e^{w_1 x}$

Based on the functional form of the model:

1. **Parametric** Regression refers to the case where the shape/form of the function linking the dependent and independent variable is known or specified in advance, only the coefficients $\mathbf{w}$ are the object of estimation.

2. **Non-parametric** Regression refers to the case where the form of the function f is not predetermined but can be adjusted to capture unexpected features in the

data.

Based on estimation procedures (by no means a comprehensive list but describe the two fundamental approaches):

1. **Least squares** estimates analytically minimize the sum of squared residuals giving a simple closed form expression for the estimated value of the coefficients $\mathbf{w}$.

2. **Bayesian** approach treats the regression coefficients $\mathbf{w}$ as random variables with a specified prior and in addition produces not only a single point estimate for the best value but an entire posterior distribution. Thus, describing uncertainty around the predictions.

Based on the **basis** of regression:

When the dependent variable y is modelled as a linear combination of a list of pre-specified functions $\phi_1(\mathbf{x}), \phi_2(\mathbf{x}), \ldots, \phi_m(\mathbf{x})$ it is said to be regression by linear combination of basis functions. Performing the regression still entails estimating the coefficients $\mathbf{w} = \{w_i\}_{i=1}^{M}$.

$$y = \sum_{i=1}^{m} w_i \phi_i(\mathbf{x}) \tag{3.1}$$

1. **Linear Basis** refers to the simplest case where $\phi_i(\mathbf{x}) = \mathbf{x}$.

2. **Polynomial Basis** refers to the case where $\phi_i(\mathbf{x}) = \mathbf{x}^i$ for $i = 1, 2, \ldots p$ are polynomial functions.

3. **Gaussian (RBF) Basis** refers to case where $\phi_i(\mathbf{x}) = \exp\{-||\mathbf{x} - \mathbf{x}_i||^2/2\sigma^2\}$ where σ is a pre-set variance parameter and the $\mathbf{x}_i's$ are typically the training data points[1]

Note that all of the above basis types are special cases of multiple linear regression as there is linear dependence on the object of estimation, the coefficients $\mathbf{w}$.

Further there are other distinctions, when the weights are modelled as an output of a kernel function, it is called *kernel* regression [Nad64]. Regression systems are also distinguished on the basis of their regularization terms, for instance, in LASSO (Least Absolute Shrinkage and Selection Operator) the penalty term is an L_1 norm of the coefficient vector $\mathbf{w}$, while Ridge regression uses a form of L_2 regularization.

[1]We delve deeper into regression using Gaussian basis functions in later sections.

1 Review of Linear Regression

In this section we provide an overview of the most common approaches to the linear regression problem and highlight the similarities and principal differences in the methods. We assume the standard multiple regression set-up with input/output pairs with N data points and p regressors.

Conside a generative function $f(\mathbf{x})$ that yields noisy measurments (with additive noise) y_i,

$$y_i = f(\mathbf{x}_i) + \epsilon_i \tag{3.2}$$

with ϵ_i typically iid with $Var(\epsilon_i) = \sigma^2$.

Given, $\mathcal{D} = \{\mathbf{x}_i, y_i\}_{i=1}^N, \mathbf{x}_i \in \mathbb{R}^p, y_i \in \mathbb{R}$ we want to estimate the weights $\mathbf{w} = (w_1, \ldots, w_p)^T$ given the parametric form:

$$y_i = w_1\mathbf{x}_{i1} + \ldots + w_p\mathbf{x}_{ip} + \hat{\epsilon}_i \tag{3.3}$$

The N equations corresponding to each target y_i can be stacked vertically to give the model,

$$\mathbf{y} = \mathbf{X}\mathbf{w} + \boldsymbol{\epsilon} \tag{3.4}$$

where $\mathbf{X}$ is a $n \times p$ matrix of regressors ($\mathbf{X}_i^T$ denotes the i-th data point) and $\mathbf{w}$ is the column vector of weights.

Note, the term 'linear' in general linear models refers to the linearity of the weights $\mathbf{w}$, models where the regressors appear non-linearly are still general linear models. Non-linear models refer to models which are non-linear in the weights, for example, the model $y = w_1 e^{w_2 x}$.

1.1 Least Squares and Marginal Likelihood

In least squares the usual estimation goal is to minimize the sums of squared residuals with respect to the weights $\mathbf{w}$ given by,

$$\sum_{i=1}^N \hat{\epsilon}_i^2 = \sum_{i=1}^N (y_i - \mathbf{X}_i^T\mathbf{w})^2 \tag{3.5}$$

The $\hat{\mathbf{w}}$ that minimizes this is called the least squares estimate. It is a relatively straight forward exercise to derive $\hat{\mathbf{w}}$ which is given in closed form,

$$\hat{\mathbf{w}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} \tag{3.6}$$

The term $(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T$ is called the Moore-Penrose pseudoinverse matrix of $\mathbf{X}$. Least squares estimates can be computed only if there is no perfect multicollinearity between the explanatory variables, as that would cause $(\mathbf{X}^T\mathbf{X})$ to have less than full column rank.

Using the residuals, $\hat{\boldsymbol{\epsilon}} = \mathbf{y} - \mathbf{X}\hat{\mathbf{w}}$ we can get an unbiased estimate for the value of σ^2 as,

$$s^2 = \frac{\hat{\boldsymbol{\epsilon}}^T\hat{\boldsymbol{\epsilon}}}{n-p} \tag{3.7}$$

where $n-p$ is the degrees of freedom to adjust for the estimation of the p dimensional parameter $\mathbf{w}$. s is also called the standard error of regression.

Among the cardinal assumptions of least squares are:

1. The linearity of the model
2. No linear dependence $\Rightarrow$ regressor matrix $\mathbf{X}$ should have full rank.
3. Homoscedasticity of the errors .i.e they are expected to have same variance σ^2 in each observation and have no correlation between the observations.

The assumption about the normality of errors is not needed to validate least squares however, if we additionally impose a normality condition on the error terms and make $\boldsymbol{\epsilon} \sim N(0, \sigma^2\mathbf{I})$ we get the maximum likelihood estimator (MLE).

The likelihood of a regression model is defined as the probability of the data given the parameters and inputs,

$$p(\mathbf{y}|\mathbf{w}, \sigma^2, \mathbf{X}) = \prod_{i=1}^{N} N(\mathbf{X}_i^T\mathbf{w}, \sigma^2\mathbf{I}) \tag{3.8}$$

$$= \frac{1}{\sqrt{(2\pi)^N|\sigma^2\mathbf{I}|}} \exp\left\{-\frac{1}{2\sigma^2}(\mathbf{y} - \mathbf{X}_i^T\mathbf{w})^T(\mathbf{y} - \mathbf{X}_i^T\mathbf{w})\right\} \tag{3.9}$$

$$\tag{3.10}$$

It is easy to see from the expression of the likelihood that a maximization is equivalent to minimizing the sum of squared residuals as the latter features in the negative of the exponent. Hence, the MLE and least squares estimator coincide under the assumption of normality of error terms.

1.2 Bayesian and Sparse Bayesian Regression

The MLE estimates often suffer from over-fitting as the parameter estimates are closely tied to the training data alone. The MLE procedure conducts a maximisation and risks optimising the statistic beyond a point where gains in generalisation performance can be obtained. Bayesian linear regression addresses this problem as it entails marginalising (integrating) rather than optimising over all possible choices. Selection of a flat prior reduces Bayesian regression to a problem of maximizing the evidence (marginal likelihood) which is identical to the maximum likelihood estimation problem.

In a standard Bayesian regression model we derive the posterior over the weights using the likelihood (which is the same as above) and a prior (this piece is uniquely Bayesian). We define a Gaussian prior $p(\mathbf{w})$,

$$\mathbf{w} \sim N(\mu_w, \Sigma_w) \tag{3.11}$$

The posterior is then derived as,

$$\begin{aligned} p(\mathbf{w}|\mathbf{y}) &= p(\mathbf{y}|\mathbf{w}, \sigma^2, \mathbf{X})p(\mathbf{w}) && (3.12)\\ &= N(\mathbf{X}_i^T\mathbf{w}, \sigma^2\mathbf{I})N(\mu_w, \Sigma_w) && (3.13)\\ &= N(\mu^*, \Sigma^*) && (3.14) \end{aligned}$$

Since the likelihood and the prior are Gaussian, the posterior is a Gaussian with mean and variance given by,

$$\begin{aligned} \mu^* &= \Sigma^*(\Sigma_w^{-1}\mu_w + \sigma^{-2}\mathbf{X}^T\mathbf{y}) && (3.15)\\ \Sigma^* &= (\Sigma_w^{-1} + \sigma^{-2}\mathbf{X}^T\mathbf{X})^{-1} && (3.16) \end{aligned}$$

If we impose the assumption that the weights $\mathbf{w}$ are distributed independently with a zero mean Gaussian prior $N(0, \Sigma_w^{'})$ where the covariance is a diagonal matrix we get the Sparse Bayes Regression model. The mean and variance of the posterior is the same as above with $\mu_w = 0$.

We don't give the technical specifications here of the derivation as this is the subject of chapter 4.

2 Metrics and Uncertainty

In this section we summarise metrics relating to the most fundamental criteria of model assessment *bias* and *variance* in a regression context.

From a Bayesian perspective, uncertainty in the model is expressed through the posterior distribution of weights/parameters $\mathbf{w}$. However, in a non-Bayesian setting we only extract point estimates for $\mathbf{w}$ based on training on a given dataset $\mathcal{D}$. There are no intrinsic ways to capture uncertainty and hence uncertainty quantification involves simulating (if possible) a large ensemble of datasets $\mathcal{D}_1, \ldots, \mathcal{D}_K$ each of the same size. If it is not possible to generate new datasets it is sometimes sufficient to tweak a given training dataset by sampling with replacement, this is also called bagging or bootstrap aggregation. The different training datasets will give different fits allowing one to estimate uncertainty by taking the average performance over the ensemble of datasets.

The mean squared error is one of the most fundamental tools for assessing model performance in a regression context, it is average deviation of the fitted value from the target output.

$$MSE = E[(y_i - \hat{y}_i)^2] = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{3.17}$$

where $\mathbf{X}_i^T w_i = \hat{y}_i$ denotes the fitted value.

It can be decomposed into bias and variance in the following way:

Let f_i denote the true function values, then $E(y_i) = f_i$, also, note the following short result:

If Z is a random variable,

$$\begin{aligned}
&= E[(Z - \bar{Z})^2] \\
&= E[Z^2 - 2Z\bar{Z} + \bar{Z}^2] \\
&= E[Z^2] - 2\bar{Z}^2 + \bar{Z}^2 \\
&= E[Z^2] - \bar{Z}^2 \\
&\Rightarrow E[Z^2] = E[(Z - \bar{Z})^2] + \bar{Z}^2
\end{aligned}$$

where $\bar{Z}$ denotes the mean value.

The MSE is,

$$= E[(y_i - \hat{y}_i)^2] \tag{3.18}$$
$$= E[(y_i)^2 + (\hat{y}_i)^2 - 2y_i\hat{y}_i] \tag{3.19}$$
$$= E[(\hat{y}_i - \bar{\hat{y}}_i)^2] + \bar{\hat{y}}_i^2 - 2\bar{\hat{y}}_i f_i + E[(y_i - f_i)^2] + f_i^2 \tag{3.20}$$
$$= E[(\hat{y}_i - \bar{\hat{y}}_i)^2] + (\bar{\hat{y}}_i - f_i)^2 + E[(y_i - f_i)^2] \tag{3.21}$$

where,

$$
\begin{aligned}
&(\bar{\hat{y}}_i - f_i)^2 = (\text{ bias })^2 \\
&E[(\hat{y}_i - \bar{\hat{y}}_i)^2] = Var(\hat{y}_i) \\
&E[(y_i - f_i)^2] = E[\epsilon_i^2] = Var(\epsilon_i) = \sigma^2 \text{ (noise variance)}
\end{aligned}
$$

2.1 Frequentist Metrics

In a frequentist world one can compute bias and variance across an ensemble of datasets. In the case of bias, this represents the extent to which average prediction (over all datasets) differs from the true function. In the case of variance, it represents the extent to which fitted values in individual datasets vary from their mean.

1. Bias for a single training run: $E[(\bar{\hat{y}}_i - f_i)]$
2. Bias across datasets: $E_{\mathcal{D}}[E(\bar{\hat{y}}_i - f_i)]$
3. Variance in the fitted values for a single training run: $E[(\hat{y}_i - \bar{\hat{y}}_i)^2]$
4. Variance in the fitted values across datasets: $E_{\mathcal{D}}[(\hat{y}_i - E_{\mathcal{D}}(\bar{\hat{y}}_i))^2]$

2.2 Bayesian Model selection

Let us assume we hava a finite set of models $\{\mathcal{M}_i\}$ to choose from for a dataset $\mathcal{D}$ each with their own parameters θ_i. The value $p(\mathcal{D}|\mathcal{M}_i)$ termed the model evidence summarises the probability that the model could have generated the data. In other words it is the support for the model $\mathcal{M}_i$.

If we have two models $\mathcal{M}_i$ and $\mathcal{M}_j$ with their own parameters θ_i and θ_j, we can define priors over the models $p(\mathcal{M}_i)$ and $p(\mathcal{M}_j)$ and compute the posterior odds given by,

$$
\frac{p(\mathcal{M}_i|\mathcal{D})}{p(\mathcal{M}_j|\mathcal{D})} = \frac{p(\mathcal{M}_i)p(\mathcal{D}|\mathcal{M}_i)}{p(\mathcal{M}_j)p(\mathcal{D}|\mathcal{M}_j)} = \frac{p(\mathcal{M}_i)\int p(\mathcal{D}|\mathcal{M}_i,\theta_i)p(\theta_i|\mathcal{M}_i)d\theta_i}{p(\mathcal{M}_j)\int p(\mathcal{D}|\mathcal{M}_j,\theta_j)p(\theta_j|\mathcal{M}_j)d\theta_j} \tag{3.22}
$$

where we are simply applying the Bayes' rule and marginalising over the parameters θ. The ratio is called the *Bayes factor* in favour of $\mathcal{M}_i$, a simple tool to select between models in an inherently Bayesian way. Notice that if the priors are flat it is just a ratio of the marginal likelihoods which appear in the denominator of the posterior over the parameters θ (see eq. 2.3).

For the rest of the report we narrow our focus to the Bayesian approach to multiple linear regression using linear combination of basis functions. But first we take an in-depth look

at a Bayesian regression procedure called Sparse Bayesian learning. The main difference between the classical Bayesian learning and Sparse Bayesian learning is the choice of a peculiar prior in the latter which causes a lot of the weights to become 0 in the inference procedure thus leading to sparse solutions.

Part II

Chapter 4

Sparse Bayesian Learning

The aim of this section is to deconstruct the Sparse Bayesian Learning algorithm first introduced in [Tip01] and [Bis06]. While the cited papers do not include any derivations of the steps in the inference procedure, this chapter aims to make explicit all the technicalities of the learning procedure and provide derivations where necessary.

1 Introduction

We are given a set of input-target pairs $\{\mathbf{x}_i, t_i\}_{i=1}^N$, $t_i \in \mathbb{R}$ and $\mathbf{x} \in \mathbb{R}^M$ where t_i is a noisy measurement of the output of a function y when its input is $\mathbf{x}_i$. The aim of the 'learning' exercise is to model the dependency of the targets t_i on the inputs $\mathbf{x}_i$.

A typical parametric form for the function $y(\mathbf{x})$ is the generalized linear additive model given by:

$$y(\mathbf{x}, \mathbf{w}) = \mathbf{w}^T\mathbf{x} = \sum_{i=1}^{M} w_i x_i \tag{4.1}$$

where the target is modelled by a linear function of explanatory variables where the parameter vector $\mathbf{w} = (w_1, w_2, \ldots, w_M)^T$ are the coefficients.

If there is a non-linear relationship between $\mathbf{x}$ and y then a popular choice is to express y in the form,

$$y(\mathbf{x}, \mathbf{w}) = \mathbf{w}^T\Phi(\mathbf{x}) \tag{4.2}$$

where $\Phi(\mathbf{x}) = (\phi_1(\mathbf{x}), \ldots, \phi_M(\mathbf{x}))^T$ is called the *design matrix* and each ϕ_i is potentially a non-linear basis function.

The design matrix Φ can be thought of as the following $N \times M$ matrix with columns representing basis functions evaluated at each of the input data points.

$$\Phi = \begin{bmatrix} \phi_1(\mathbf{x}_1) & \phi_2(\mathbf{x}_1) & \dots\dots\dots & \phi_M(\mathbf{x}_1) \\ \phi_1(\mathbf{x}_2) & \phi_2(\mathbf{x}_2) & \dots\dots\dots & \phi_M(\mathbf{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \phi_1(\mathbf{x}_N) & \phi_2(\mathbf{x}_N) & \dots\dots\dots & \phi_M(\mathbf{x}_N) \end{bmatrix}_{N \times M}$$

Another way to express the design matrix is by focussing on the columns and the rows.

$$\Phi = \begin{bmatrix} \vdots & \vdots & & \vdots \\ \vdots & \vdots & & \vdots \\ \phi_1 & \phi_2 & \dots & \phi_M \\ \vdots & \vdots & & \vdots \\ \vdots & \vdots & & \vdots \end{bmatrix} = \begin{bmatrix} \dots & \dots & \Phi(\mathbf{x}_1) & \dots & \dots \\ \dots & \dots & \Phi(\mathbf{x}_2) & \dots & \dots \\ & & \vdots & & \\ & & \vdots & & \\ \dots & \dots & \Phi(\mathbf{x}_M) & \dots & \dots \end{bmatrix} \tag{4.3}$$

Where ϕ_i represents the column vector with basis function ϕ_i evaluated at all the input data points $\mathbf{x_1}, \dots, \mathbf{x}_N$ and $\Phi(\mathbf{x}_i)$ represents the row vector with all basis functions evaluated at a single data point $\mathbf{x}_i$.

The focus is on learning the parameter vector $\mathbf{w} = \{w_i\}_{i=1}^M$ which is used to predict y as a function of the input $\mathbf{x}$.

Bayesian learning applies Bayesian inference to learn models of the form 4.2. A key feature of sparse bayesian learning is that the inferred predictors have relatively few non-zero parameters w_i. These methods when applied to high dimensional input spaces automatically select smaller informative subspaces. In a regression system detailed in 4.2 this translates to selecting a small number of basis functions ϕ_i from the candidates in Φ.

The Relevance Vector Machine is a specific instance of this model which uses *kernels* [1] for basis functions [Tip01]. The general formulation then becomes,

$$y(\mathbf{x}, \mathbf{w}) = \sum_{n=1}^{N} w_n k(\mathbf{x}, \mathbf{x}_n) \tag{4.4}$$

[1] A kernel is a mathematical function that can be applied to two data points and serves as a similarity measure between them. An example of a kernel is the Gaussian kernel given by $k(x, y) = \exp\{\frac{-1}{2\sigma}||x - y||^2\}$

where $\phi_n(\mathbf{x}) = k(\mathbf{x}, \mathbf{x_n})$. $\mathbf{x}_n$ can denote the input data points or any data points in the domain of the input data. It is important to note that in the general Sparse Bayesian learning algorithm there is no condition on the basis functions to be kernels, they can be arbitrary functions.

2 Model Setup

We assume that each target t_i is representative of the true model $y_i = y(\mathbf{x}_i)$ with additive Gaussian noise $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$

$$t_i = y_i + \epsilon_i \tag{4.5}$$

$$\begin{aligned} p(t_i|\mathbf{x}_i, \mathbf{w}, \sigma^2) &\sim \mathcal{N}(y_i, \sigma^2) \\ &= (2\pi\sigma^2)^{-\frac{1}{2}} \exp\left\{\frac{-1}{2\sigma^2}(t_i - y_i)^2\right\} \\ &= (2\pi\sigma^2)^{-\frac{1}{2}} \exp\left\{\frac{-1}{2\sigma^2}(t_i - \mathbf{w}^T\Phi(\mathbf{x}_i))^2\right\} \end{aligned} \tag{4.6}$$

where $\Phi(\mathbf{x}_i) = (\phi_1(\mathbf{x}_i), \ldots, \phi_M(\mathbf{x}_i))^T$. For N training points, vector $\mathbf{t}$ represents all the training targets t_i and the $N \times M$ design matrix Φ is constructed such that the i-th row represents the vector $(\phi_1(\mathbf{x}_i), \ldots, \phi_M(\mathbf{x}_i))$ of all basis functions evaluated at $\mathbf{x}_i$.

Due to the assumption of independence of the targets t_i, the multivariate Gaussian likelihood is constructed by taking the product of the univariate densities.

$$\begin{aligned} p(\mathbf{t}|\mathbf{x}, \mathbf{w}, \sigma^2) &= \prod_{i=1}^{N} \mathcal{N}(\mathbf{w}^T\Phi(\mathbf{x_i}), \sigma^2) \\ &= \prod_{i=1}^{N} (2\pi\sigma^2)^{-\frac{1}{2}} \exp\left\{\frac{-1}{2\sigma^2}(t_i - \mathbf{w}^T\Phi(\mathbf{x}_i))^2\right\} \\ &= (2\pi\sigma^2)^{-\frac{N}{2}} \exp\left\{\frac{-1}{2\sigma^2}(||\mathbf{t} - \Phi\mathbf{w}||)^2\right\} \end{aligned} \tag{4.7}$$

In vector notation the model is given by,

$$\mathbf{t} = \Phi\mathbf{w} + \epsilon \tag{4.8}$$

where $\epsilon \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$ where $\mathbf{I}$ is the $M \times M$ identity matrix.

It is important to note that the multivariate Gaussian likelihood, is viewed as a function of $\mathbf{w}$ rather than $\mathbf{t}$, very often it is confused as a conditional probability density of the data $\mathbf{t}$ given a fixed parameter value $\mathbf{w}$.

We need to introduce a prior distribution over the parameter vector $\mathbf{w}$. Priors can be chosen according to a principle, for instance, choosing a prior from a family that simplifies calculation of the posterior distribution. This can be done in the case of conjugate priors [2]. It is not uncommon to choose a zero-mean Gaussian prior.

The prior probability distribution for a single w_i is given by,

$$p(w_i|\alpha_i) \sim \mathcal{N}(0, \alpha_i^{-1}) \tag{4.9}$$

In eq. 4.9 α_i is the hyperparameter for the parameter w_i. It is called the precision parameter and it is the inverse variance of the weight w_i. It is not uncommon to use the precision parameter to describe the distribution. Precision $\left(\alpha_i = \dfrac{1}{Var(w_i)}\right)$ describes how concentrated the values are around the mean, hence a smaller precision means more spread and higher variance. It is more intuitive to think of how precise a distribution is rather than how imprecise (spread out) which is captured by variance.

Let $\boldsymbol{\alpha}$ denote $(\alpha_1, \ldots, \alpha_M)^T$, then the joint probability distribution of the vector $\mathbf{w}$ is given by,

$$\begin{aligned}
p(\mathbf{w}|\boldsymbol{\alpha}) &= \prod_{i=0}^{M} \mathcal{N}(0, \alpha_i^{-1}) \\
&= (2\pi)^{-M/2} \prod_{i=1}^{M} \frac{1}{\alpha_i^{1/2}} \exp\left\{\frac{-1}{2}\mathbf{w}^T(\mathbf{A}^{-1})^{-1}\mathbf{w}\right\} \\
&= (2\pi)^{-M/2} \frac{1}{|\mathbf{A}^{-1}|^{1/2}} \exp\left\{\frac{-1}{2}\mathbf{w}^T(\mathbf{A}^{-1})^{-1}\mathbf{w}\right\} \\
&= (2\pi)^{-M/2} |\mathbf{A}|^{1/2} \exp\left\{\frac{-1}{2}\mathbf{w}^T\mathbf{A}\mathbf{w}\right\}
\end{aligned} \tag{4.10}$$

(Recall in eq. A.10 the form of the multivariate Gaussian density), Hence, we write the multivariate probability density as, $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \mathbf{A}^{-1})$ where $\mathbf{A} = \begin{bmatrix} \alpha_1 & 0 & \ldots & 0 \\ 0 & \alpha_2 & 0 & \ldots \\ \ddots & \ddots & \ddots & 0 \\ 0 & \ddots & 0 & \alpha_M \end{bmatrix}$

[2] In Bayesian probability theory if the posterior distribution is in the same family as the prior probability distribution, the prior and the posterior are called conjugate distributions, and the prior is called a conjugate prior for the likelihood function.

and

$$\mathbf{A}^{-1} = \begin{bmatrix} \frac{1}{\alpha_1} & 0 & \dots & 0 \\ 0 & \frac{1}{\alpha_2} & 0 & \dots \\ \ddots & \ddots & \ddots & 0 \\ 0 & \ddots & 0 & \frac{1}{\alpha_M} \end{bmatrix}$$

In the model set-up so far, we have three sets of parameters.

1. The weights $\mathbf{w}$.
2. An estimate of the noise variance σ^2.
3. The hyperparameter vector $\boldsymbol{\alpha}$(which contains the hyperparameter of each weight).

In a non-Bayesian setting we would just have the first two as $\mathbf{w}$ would not be modelled as a random variable. Further, $\mathbf{w}$ would be set either by least squares or a maximum likelihood like procedure yielding a point estimate. In a Bayesian setting we take a distinctly different approach, we first define the posterior over all the model unknowns.

The posterior probability over all unknown parameters given the data is expressed as,

$$p(\mathbf{w}, \boldsymbol{\alpha}, \sigma^2|\mathbf{t}) = \frac{p(\mathbf{t}|\mathbf{w}, \boldsymbol{\alpha}, \sigma^2)p(\mathbf{w}, \boldsymbol{\alpha}, \sigma^2)}{p(\mathbf{t})} \tag{4.11}$$

where we apply the Bayes' rule.

The most important observation to make here is that the denominator in the above posterior is not computable directly,

$$p(\mathbf{t}) = \int p(\mathbf{t}|\mathbf{w}, \boldsymbol{\alpha}, \sigma^2)p(\mathbf{w}, \boldsymbol{\alpha}, \sigma^2)d\mathbf{w}d\boldsymbol{\alpha}d\sigma^2 \tag{4.12}$$

The approach we will take is to decompose the posterior into two parts and seek an approximation.

The posterior above is decomposed to,

$$p(\mathbf{w}, \boldsymbol{\alpha}, \sigma^2|\mathbf{t}) = p(\mathbf{w}|\mathbf{t}, \boldsymbol{\alpha}, \sigma^2)p(\boldsymbol{\alpha}, \sigma^2|\mathbf{t}). \tag{4.13}$$

Notice that the weights $\mathbf{w}$ have a dependence on $\mathbf{t}$, $\boldsymbol{\alpha}$ and σ^2.

The first term on the right hand side,

$$p(\mathbf{w}|\mathbf{t}, \boldsymbol{\alpha}, \sigma^2) \tag{4.14}$$

is the posterior over the weights.

The second term on the right hand side,

$$p(\boldsymbol{\alpha}, \sigma^2|\mathbf{t}) \tag{4.15}$$

is the posterior over the hyper-parameters.

2.1 Posterior over the weights $\mathbf{w}$

We apply the Bayes' rule to decompose the posterior over the weights. We can analytically compute this since its denominator (Dr.) $p(\mathbf{t}|\boldsymbol{\alpha}, \sigma^2) = \int p(\mathbf{t}|\mathbf{w}, \sigma^2)p(\mathbf{w}|\boldsymbol{\alpha})d\mathbf{w}$ is a convolution of Gaussian densities and its numerator (Nr.) is a product of two multivariate Gaussian densities.

$$p(\mathbf{w}|\mathbf{t}, \boldsymbol{\alpha}, \sigma^2) = \frac{p(\mathbf{t}|\mathbf{w}, \sigma^2)p(\mathbf{w}|\boldsymbol{\alpha})}{p(\mathbf{t}|\boldsymbol{\alpha}, \sigma^2)} = \frac{p(\mathbf{t}|\mathbf{w}, \sigma^2)p(\mathbf{w}|\boldsymbol{\alpha})}{\int p(\mathbf{t}|\mathbf{w}, \sigma^2)p(\mathbf{w}|\boldsymbol{\alpha})d\mathbf{w}} \tag{4.16}$$

Simplifying the Nr.

$$p(\mathbf{t}|\mathbf{w}, \sigma^2)p(\mathbf{w}|\boldsymbol{\alpha}) = (2\pi\sigma^2)^{-\frac{N}{2}} \exp\left\{\frac{-1}{2\sigma^2}(||\mathbf{t} - \Phi\mathbf{w}||)^2\right\} \times (2\pi)^{-M/2}|\mathbf{A}|^{1/2} \exp\left\{\frac{-1}{2}\mathbf{w}^T\mathbf{A}\mathbf{w}\right\} \tag{4.17}$$

What we have a above is a product of Gaussian densities however one of them has the variable $\mathbf{t}$ and the other is in variable $\mathbf{w}$.

Since we want the posterior over the weights we need to express the final density in terms of $\mathbf{w}$.

Completion of Squares

In this section we try to simplify the quadratic in the exponent of the product and regroup in terms of $\mathbf{w}$. In other words we need to express the quadratic as a square in $\mathbf{w}$ + constants (it will become clear why we do this as we proceed with the proof).

The quadratic in the exponent of the product above is given by,

$$-\frac{1}{2\sigma^2}(\mathbf{t}-\Phi\mathbf{w})^T(\mathbf{t}-\Phi\mathbf{w})-\frac{1}{2}\mathbf{w}^T\mathbf{A}\mathbf{w} \tag{4.18}$$

We attempt to factorize the term:

$$\begin{aligned}
&\Rightarrow \sigma^{-2}(\mathbf{t}-\Phi\mathbf{w})^T(\mathbf{t}-\Phi\mathbf{w})+\mathbf{w}^T\mathbf{A}\mathbf{w}\\
&\Rightarrow \sigma^{-2}(\mathbf{t}^T\mathbf{t}-2\mathbf{w}^T\Phi^T\mathbf{t}+\mathbf{w}^T\Phi^T\Phi\mathbf{w})+\mathbf{w}^T\mathbf{A}\mathbf{w}\\
&\Rightarrow \mathbf{w}^T(\mathbf{A}+\sigma^{-2}\Phi^T\Phi)\mathbf{w}-2\mathbf{w}^T(\sigma^{-2}\Phi^T\mathbf{t})+\sigma^{-2}\mathbf{t}^T\mathbf{t}
\end{aligned} \tag{4.19}$$

The last equation is in the matrix quadratic form,

$$\mathbf{w}^TM\mathbf{w}+\mathbf{w}^Tb+c \tag{4.20}$$

which can be re-written as the sum of a square in $\mathbf{w}$ and a constant.

$$\mathbf{w}^TM\mathbf{w}+\mathbf{w}^Tb+c=(\mathbf{w}-h)^TM(\mathbf{w}-h)+k \tag{4.21}$$

where,

$$\begin{aligned}
h&=-\frac{1}{2}M^{-1}b\\
k&=c-\frac{1}{4}b^TM^{-1}b
\end{aligned} \tag{4.22}$$

We can deduce from this that,

$$\begin{aligned}
h&=-\frac{1}{2}(\mathbf{A}+\sigma^{-2}\Phi^T\Phi)^{-1}(-2\sigma^{-2}\Phi^T\mathbf{t})\\
&=\sigma^{-2}(\mathbf{A}+\sigma^{-2}\Phi^T\Phi)^{-1}\Phi^T\mathbf{t}
\end{aligned} \tag{4.23}$$

If we call $(\mathbf{A}+\sigma^{-2}\Phi^T\Phi)^{-1}$ as Σ, and

$$\Rightarrow \sigma^{-2}(\mathbf{A}+\sigma^{-2}\Phi^T\Phi)^{-1}\Phi^T\mathbf{t} \tag{4.24}$$

$$=\sigma^{-2}\Sigma\Phi^T\mathbf{t} \tag{4.25}$$

$$=\boldsymbol{\mu} \tag{4.26}$$

we can re-write eq 4.21 as,

$$(\mathbf{w} - \boldsymbol{\mu})^T \Sigma^{-1} (\mathbf{w} - \boldsymbol{\mu}) + k \tag{4.27}$$

Where k is a collection of terms free from $\mathbf{w}$.

Deducing *k*

$$\begin{aligned}
k &= c - \frac{1}{4} b^T M^{-1} b && (4.28)\\
&= \sigma^{-2}\mathbf{t}^T\mathbf{t} - \frac{1}{4}(-2\sigma^{-2}\Phi^T\mathbf{t})^T\Sigma(-2\sigma^{-2}\Phi^T\mathbf{t}) && (4.29)\\
&= \sigma^{-2}\mathbf{t}^T\mathbf{t} - (\sigma^{-2}\Phi^T\mathbf{t})^T\Sigma(\sigma^{-2}\Phi^T\mathbf{t}) && (4.30)\\
&= \sigma^{-2}\mathbf{t}^T\mathbf{t} - \sigma^{-4}\mathbf{t}^T\Phi\Sigma\Phi^T\mathbf{t} && (4.31)\\
&= \mathbf{t}^T\left[\sigma^{-2} - \sigma^{-4}\mathbf{t}^T\Phi\Sigma\Phi^T\right]\mathbf{t} && (4.32)\\
&= \mathbf{t}^T\left[\sigma^{-2}(\mathbf{I} - \underbrace{\sigma^{-2}\Phi(\mathbf{A} + \sigma^{-2}\Phi^T\Phi)^{-1}\Phi^T}_{k_*})\right]\mathbf{t} && (4.33)
\end{aligned}$$

Now we use the Woodbury identity from section 3.1 from Appendix A to break down the inverse:

$$\begin{aligned}
&\Rightarrow (\mathbf{A} + \sigma^{-2}\Phi^T\Phi)^{-1} && (4.34)\\
&\Rightarrow \mathbf{A}^{-1} - \mathbf{A}^{-1}\Phi^T(\sigma^2\mathbf{I} + \Phi\mathbf{A}^{-1}\Phi^T)^{-1}\Phi\mathbf{A}^{-1} && (4.35)\\
& && (4.36)
\end{aligned}$$

For clarity we zone into k_* and plug the result of the inverse computed above and simplify:

$$\begin{aligned}
k_* &= \sigma^{-2}\Phi\mathbf{A}^{-1}\Phi^T - \sigma^{-2}\Phi\mathbf{A}^{-1}\Phi^T(\sigma^2\mathbf{I} + \Phi\mathbf{A}^{-1}\Phi^T)^{-1})\Phi\mathbf{A}^{-1}\Phi^T && (4.37)\\
&= \sigma^{-2}\left[\mathbf{I} - \Phi\mathbf{A}^{-1}\Phi^T(\sigma^2\mathbf{I} + \Phi\mathbf{A}^{-1}\Phi^T)^{-1})\right]\Phi\mathbf{A}^{-1}\Phi^T && (4.38)\\
&= (\sigma^2\mathbf{I} + \Phi\mathbf{A}^{-1}\Phi^T)^{-1}\Phi\mathbf{A}^{-1}\Phi^T && (4.39)
\end{aligned}$$

Where we use the identity $(I - (A+B)^{-1}B)A^{-1} = (A+B)^{-1}(A+B-B)A^{-1} = (A+B)^{-1}$

Plugging this simplified form of k_* into the eq. 4.33,

$$= \mathbf{t}^T \left[\sigma^{-2}(\mathbf{I} - (\sigma^2\mathbf{I} + \Phi\mathbf{A}^{-1}\Phi^T)^{-1}\Phi\mathbf{A}^{-1}\Phi^T)\right] \mathbf{t} \tag{4.40}$$

$$= \mathbf{t}^T(\sigma^2\mathbf{I} + \Phi\mathbf{A}^{-1}\Phi^T)^{-1}\mathbf{t} \tag{4.41}$$

Finally, the simplified form of the product of densities we started with in eq. 4.17 yields,

$$p(\mathbf{t}|\mathbf{w}, \sigma^2)p(\mathbf{w}|\boldsymbol{\alpha}) = \frac{1}{\sqrt{(2\pi)^N|\sigma^2\mathbf{I}||\mathbf{A}^{-1}||\Sigma^{-1}|}} \exp\left\{-\frac{1}{2}\mathbf{t}^T(\sigma^2\mathbf{I} + \Phi\mathbf{A}^{-1}\Phi^T)^{-1}\mathbf{t}\right\} \times \tag{4.42}$$

$$\frac{1}{\sqrt{(2\pi)^M|\Sigma|}} \exp\left\{-\frac{1}{2}(\mathbf{w} - \boldsymbol{\mu})^T\Sigma^{-1}(\mathbf{w} - \boldsymbol{\mu})\right\} \tag{4.43}$$

$$\tag{4.44}$$

where we multiply and divide by the factor $|\Sigma|$

$$p(\mathbf{t}|\mathbf{w}, \sigma^2)p(\mathbf{w}|\boldsymbol{\alpha}) = S \times \mathcal{N}(\boldsymbol{\mu}, \Sigma) \tag{4.45}$$

where,

$$\begin{aligned} \Sigma &= (\mathbf{A} + \sigma^{-2}\Phi^T\Phi)^{-1} \\ \boldsymbol{\mu} &= \sigma^{-2}\Sigma\Phi^T\mathbf{t} \end{aligned} \tag{4.46}$$

and,

$$S = \frac{1}{\sqrt{(2\pi)^N|\sigma^2\mathbf{I} + \Phi\mathbf{A}^{-1}\Phi^T|}} \exp\left\{-\frac{1}{2}\mathbf{t}^T(\sigma^2\mathbf{I} + \Phi\mathbf{A}^{-1}\Phi^T)^{-1}\mathbf{t}\right\} \tag{4.47}$$

where in order to get the right normalizing factor we use the result,

$$|\sigma^2\mathbf{I}||\mathbf{A}^{-1}||\Sigma^{-1}| = |\sigma^2\mathbf{I} + \Phi\mathbf{A}^{-1}\Phi^T| \tag{4.48}$$

by a straight forward application of the matrix determinant lemma.

Deduction of the denominator in the posterior

Shifting focus now to the denominator $p(\mathbf{t}|\boldsymbol{\alpha}, \sigma^2) = \int p(\mathbf{t}|\mathbf{w}, \sigma^2)p(\mathbf{w}|\boldsymbol{\alpha})d\mathbf{w}$ where we have simplified the product inside to a great degree in the section above. The denominator simplifies to,

$$\int p(\mathbf{t}|\mathbf{w}, \sigma^2)p(\mathbf{w}|\boldsymbol{\alpha})d\mathbf{w} = \int S \times \mathcal{N}(\boldsymbol{\mu}, \Sigma)d\mathbf{w} \tag{4.49}$$

$$= S \tag{4.50}$$

The expression S is also called the *marginal likelihood* as we are marginalizing out the parameter vector $\mathbf{w}$.

Now that we have simplified the numerator and the denominator the posterior over the weights is given by,

$$p(\mathbf{w}|\mathbf{t}, \boldsymbol{\alpha}, \sigma^2) = \frac{p(\mathbf{t}|\mathbf{w}, \sigma^2)p(\mathbf{w}|\boldsymbol{\alpha})}{p(\mathbf{t}|\boldsymbol{\alpha}, \sigma^2)} = \frac{p(\mathbf{t}|\mathbf{w}, \sigma^2)p(\mathbf{w}|\boldsymbol{\alpha})}{\int p(\mathbf{t}|\mathbf{w}, \sigma^2)p(\mathbf{w}|\boldsymbol{\alpha})d\mathbf{w}} \tag{4.51}$$

$$\tag{4.52}$$

$$= \frac{S \times \mathcal{N}(\boldsymbol{\mu}, \Sigma)}{S} \tag{4.53}$$

$$= \mathcal{N}(\boldsymbol{\mu}, \Sigma) \tag{4.54}$$

In summary, the posterior over the weights is given by,

$$p(\mathbf{w}|\mathbf{t}, \boldsymbol{\alpha}, \sigma^2) = (2\pi)^{-M/2}|\Sigma|^{-1/2} \exp\left\{\frac{1}{2}(\mathbf{w} - \boldsymbol{\mu})^T\Sigma^{-1}(\mathbf{w} - \boldsymbol{\mu})\right\} \tag{4.55}$$

where,

$$\begin{aligned} \Sigma &= (\mathbf{A} + \sigma^{-2}\Phi^T\Phi)^{-1} \\ \boldsymbol{\mu} &= \sigma^{-2}\Sigma\Phi^T\mathbf{t} \end{aligned} \tag{4.56}$$

Note that in order to evaluate $\boldsymbol{\mu}$ and Σ we need to find the hyper-parameters $\boldsymbol{\alpha}$ and σ^2.

2.2 Posterior over the hyper-parameters $\boldsymbol{\alpha}$

The posterior over the hyper-parameters is given by,

$$p(\boldsymbol{\alpha}, \sigma^2|\mathbf{t}) = \frac{p(\mathbf{t}|\boldsymbol{\alpha}, \sigma^2)p(\boldsymbol{\alpha})p(\sigma^2)}{p(\mathbf{t})} \tag{4.57}$$

$$p(\boldsymbol{\alpha}, \sigma^2|\mathbf{t}) \propto p(\mathbf{t}|\boldsymbol{\alpha}, \sigma^2)p(\boldsymbol{\alpha})p(\sigma^2) \tag{4.58}$$

In the case of uniform priors [3] $p(\boldsymbol{\alpha})$ and $p(\sigma^2)$, the maximization of the posterior probability is reduced to maximising:

$$\begin{aligned} p(\mathbf{t}|\boldsymbol{\alpha}, \sigma^2) &= \int p(\mathbf{t}|\mathbf{w}, \sigma^2)p(\mathbf{w}|\boldsymbol{\alpha})d\mathbf{w} \\ &= (2\pi)^{-N/2}|\sigma^2\mathbf{I} + \Phi\mathbf{A}^{-1}\Phi|^{-1/2} \exp\left\{-\frac{1}{2}\mathbf{t}^T(\Phi\mathbf{A}^{-1}\Phi^T + \sigma^2\mathbf{I})\mathbf{t}\right\} \end{aligned} \tag{4.59}$$

We have already encountered this expression in the previous section, it is the marginal likelihood. We seek the hyperparameters, $\boldsymbol{\alpha}$ and σ^2 that maximize the marginal likelihood. Since the marginal likelihood is also called model evidence, the maximization procedure is also called the 'evidence approximation procedure'.

3 Maximizing the marginal likelihood

The maximization in this section is predominantly a re-write of the version in [TF+03]. We include it here for completeness and also because the existing software implementation used to generate the experimental results in chapter 7 follows this version of the algorithm.

The log marginal likelihood (which is more conducive to differentiation) is given by,

$$\ln p(\mathbf{t}|\boldsymbol{\alpha}, \sigma^2) = -\frac{N}{2}\ln(2\pi) - \frac{1}{2}\ln|\sigma^2\mathbf{I} + \Phi\mathbf{A}^{-1}\Phi| - \frac{1}{2}\mathbf{t}^T(\Phi\mathbf{A}^{-1}\Phi^T + \sigma^2\mathbf{I})\mathbf{t} \tag{4.60}$$

Assigning the covariance $\mathbf{C} = \sigma^2\mathbf{I} + \Phi\mathbf{A}^{-1}\Phi^T$,

$$\ln p(\mathbf{t}|\boldsymbol{\alpha}, \sigma^2) = -\frac{1}{2}\left(N\ln(2\pi) + \ln|\mathbf{C}| + \mathbf{t}^T\mathbf{C}^{-1}\mathbf{t}\right) \tag{4.61}$$

We first attempt to re-write equation 4.60 and propose a strategy for the maximisation of the marginal likelihood with respect to the hyperparameters α.

[3] A hyperprior is the prior distribution of the hyperparameter. A uniform hyperprior as the name suggests, is a uniform probability distribution as choice of prior which assigns equal probabilities to all possibilities. Such a prior is also uninformative.

Let,

$$\begin{aligned}\mathcal{L}(\boldsymbol{\alpha}) &= \ln p(\mathbf{t}|\boldsymbol{\alpha},\beta) = \ln \int_{-\infty}^{\infty} p(\mathbf{t}|\mathbf{w},\beta)p(\mathbf{w}|\boldsymbol{\alpha})d\mathbf{w} \\ &= \frac{-1}{2}[N\ln 2\pi + \ln|\mathbf{C}| + \mathbf{t}^T\mathbf{C}^{-1}\mathbf{t}]\end{aligned} \tag{4.62}$$

where $\mathbf{C} = \sigma^{-2}\mathbf{I} + \Phi\mathbf{A}^{-1}\Phi^T$.

In order to analyse the dependence of $\mathbf{C}$ on a single hyperparameter α_i we re-write:

$$\begin{aligned}\mathbf{C} &= \sigma^{-2}\mathbf{I} + \sum_m \alpha_m\phi_m\phi_m^T \\ &= \sigma^{-2}\mathbf{I} + \sum_{m\neq i} \alpha_m^{-1}\phi_m\phi_m^T + \alpha_i^{-1}\phi_i\phi_i^T \\ &= \mathbf{C}_{-i} + \alpha_i^{-1}\phi_i\phi_i^T\end{aligned} \tag{4.63}$$

where $\mathbf{C_{-i}}$ denotes the covariance matrix with the influence of basis vector ϕ_i removed. Further, we establish that,

$$\begin{aligned}|\mathbf{C}| &= |\mathbf{C}_{-i}||1 + \alpha_i^{-1}\phi_i^T\mathbf{C}_{-i}^{-1}\phi_i| \\ \mathbf{C}^{-1} &= \mathbf{C}_{-i}^{-1} - \frac{\mathbf{C}_{-i}^{-1}\phi_i\phi_i^T\mathbf{C}_{-i}^{-1}}{\alpha_i + \phi_i^T\mathbf{C}_{-i}^{-1}\phi_i}\end{aligned} \tag{4.64}$$

The inverse follows from the Sherman-Morrison formula, see section 3.1 of Appendix B.

$\mathcal{L}(\boldsymbol{\alpha})$ can be written as,

$$\begin{aligned}\mathcal{L}(\boldsymbol{\alpha}) =& \frac{-1}{2}[N\ln(2\pi) + \ln|\mathbf{C}_{-i}| + \mathbf{t}^T\mathbf{C}_{-i}^{-1}\mathbf{t} - \ln\alpha_i + \\ & \ln(\alpha_i + \phi_i^T\mathbf{C}_{-i}^{-1}\phi_i) - \frac{(\phi_i^T\mathbf{C}_{-i}^{-1}\mathbf{t})^2}{\alpha_i + \phi_i^T\mathbf{C}_{-i}^{-1}\phi_i}] \\ &= \mathcal{L}(\boldsymbol{\alpha_{-i}}) + \frac{1}{2}\Big[\ln\alpha_i - \ln(\alpha_i + \phi_i^T\mathbf{C}_{-i}^{-1}\phi_i) + \frac{(\phi_i^T\mathbf{C}_{-i}^{-1}\mathbf{t})^2}{\alpha_i + \phi_i^T\mathbf{C}_{-i}^{-1}\phi_i}\Big] \\ &= \mathcal{L}(\boldsymbol{\alpha_{-i}}) + l(\alpha_i)\end{aligned} \tag{4.65}$$

where $\mathcal{L}(\boldsymbol{\alpha_{-i}})$ is the log marginal likelihood with α_i removed from the model. We have now isolated the terms containing α_i in the function $l(\alpha_i)$.

3.1 First derivatives of $\mathcal{L}(\boldsymbol{\alpha})$

Differentiation of $\mathcal{L}(\boldsymbol{\alpha})$ with respect to α_i leads to,

$$\frac{\partial \mathcal{L}(\boldsymbol{\alpha})}{\partial \alpha_i} = \frac{1}{2}\left[\frac{1}{\alpha_i} - \frac{1}{\alpha_i + \phi_i^T \mathbf{C}_{-i}^{-1}\phi_i} - \frac{(\phi_i^T \mathbf{C}_{-i}^{-1}\mathbf{t})^2}{(\alpha_i + \phi_i^T \mathbf{C}_{-i}^{-1}\phi_i)^2}\right] \tag{4.66}$$

Conveniently, the α_is occur explicitly since $\mathbf{C}_{-i}$ is independent of α_i. A further simplification is to re-write using the definitions,

$$\begin{aligned} q_i &= \phi_i^T \mathbf{C}_{-i}^{-1}\mathbf{t} \\ s_i &= \phi_i^T \mathbf{C}_{-i}^{-1}\phi_i \end{aligned} \tag{4.67}$$

$$\frac{\partial \mathcal{L}(\boldsymbol{\alpha})}{\partial \alpha_i} = \frac{(\alpha_i + s_i)^2 - \alpha_i(\alpha_i + s_i) - \alpha_i q_i^2}{2\alpha_i(\alpha_i + s_i)^2} \tag{4.68}$$

$$= \frac{(\alpha_i^2 + s_i^2 + 2\alpha_i s_i) - (\alpha_i^2 + \alpha_i s_i) - \alpha_i q_i^2}{2\alpha_i(\alpha_i + s_i)^2} \tag{4.69}$$

$$= \frac{\alpha_i^{-1}(\alpha_i^2 + s_i^2 + 2\alpha_i s_i) - (\alpha_i + s_i) - q_i^2}{2(\alpha_i + s_i)^2} \tag{4.70}$$

$$= \frac{\alpha_i^{-1} s_i^2 - (q_i^2 - s_i)}{2(\alpha_i + s_i)^2} \tag{4.71}$$

The term q_i can be interpreted as a 'quality' factor for a measure of alignment of basis vector ϕ_i with the error of the model with that vector excluded (it can also be written as $\sigma^{-2}\phi_i^T(\mathbf{t} - \mathbf{y}_{-i})$). The term s_i can be interpreted as a 'sparsity' factor which measures the extent to which basis vector ϕ_i overlaps those already present in the model.

Stationary points occur at both $\alpha_i = \infty$ and at,

$$\alpha_i = \frac{s_i^2}{q_i^2 - s_i} \tag{4.72}$$

The latter is subject to $q_i^2 > s_i$ as α_i being the precision or inverse variance cannot be negative. In the case that $q_i^2 \leq s_i$ then, $\alpha_i = \infty$, this case implies that the corresponding weight $w_i = 0$ and the basis vector ϕ_i is not relevant in the model.

The full characterisation of the log marginal likelihood is given in figure 4.1 where we see that $l(\alpha_i)$ has a single maximum at a finite α_i for $q_i^2 > s_i$ and the maximum occurs at $\alpha_i = \infty$ for $q_i^2 < s_i$.

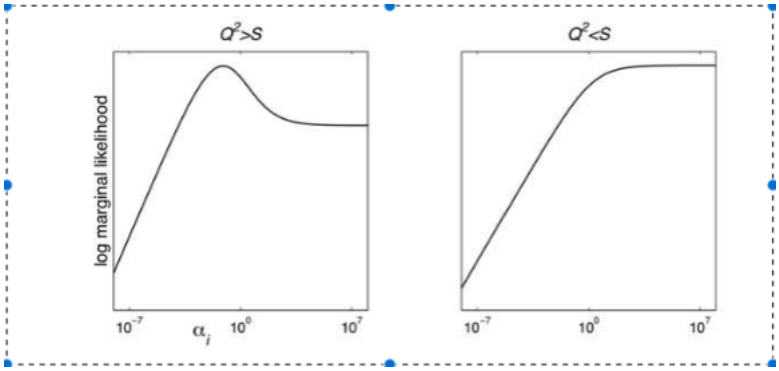


Figure 4.1: Example plot of $l(\alpha_i)$ against α_i on a log scale. (From [TF+03])

3.2 Algorithm

Summarising the facts above, the sequential fast learning algorithm is as follows:

Algorithm 1 Sequential Sparse Bayesian learning

1: Make a suitable choice for basis functions in Φ and initialise the noise variance σ^{-2}.
2: Select a single basis vector ϕ_i and set its corresponding hyperparameter α_i to

$$\alpha_i = \frac{||\phi_i||^2}{||\phi_i^T \mathbf{t}||^2/||\phi_i||^2 - \sigma^2} \tag{4.73}$$

(follows from eq. 4.72) All other α_m are set to ∞.
3: Compute Σ and μ which are scalars initially along with initial values of s_m and q_m for all M bases ϕ_m.
4: Select a candidate basis vector ϕ_i from the set of all M.
5: Compute $\theta_i = q_i^2 - s_i$
6: If $\theta_i > 0$ and $\alpha_i < \infty$ (.i.e ϕ_i is in the model), **re-estimate** α_i.
7: If $\theta_i > 0$ and $\alpha_i = \infty$, **add** ϕ_i to the model with updated α_i.
8: If $\theta_i \leq 0$ and $\alpha_i < \infty$, then **delete** ϕ_i from the model and set $\alpha_i = \infty$.
9: If noise level is fixed then skip this step, if not, update $\sigma^2 = ||\mathbf{t} - \mathbf{y}||^2/(N - M + \sum_m \alpha_m \Sigma_{mn})$ (this expression for the σ^2 has been derived in section 1.2 of Appendix C but is also stated in the original paper [Tip01])
10: Update Σ, μ and all s_m and q_m.
11: If converged terminate, if not go to step 4.

The original algorithm commences with all the M basis functions and updates the hyperparameters iteratively. As a result of the updating procedure some of the basis functions get pruned out and the algorithm accelerates however, the initial few computations still require dealing with the full design matrix of all basis function candidates. In the algorithm described here, we start with an empty model and add basis functions which increase the marginal likelihood and delete basis functions which subsequently become

redundant. The exact update formulas for adding, re-estimating and deleting a basis function are in [TF+03].

3.3 Predictions

In order to come up with a point prediction, the likelihood maximizing $\boldsymbol{\alpha}_{MP}$ are plugged into the expression for mean and covariance of the posterior weight distribution to give a MAP estimate (discussed in chapter 2). It refers to the mode of the posterior probability distribution but since the weights have a Gaussian prior the posterior mean coincides with the mode. We denote the mean of the posterior weight probability distribution as μ_{MAP}.

We use the updated μ and Σ to evaluate the predictive distribution for a new input $\mathbf{x}'$:

$$p(t^{'}|\mathbf{x}', \boldsymbol{\alpha}_{MAP}, \sigma^2_{MAP}) = \int p(t'|\mathbf{w}, \sigma^2)p(\mathbf{w}|\boldsymbol{\alpha}, \sigma^2)d\mathbf{w} \tag{4.74}$$

The expression above states that the posterior predictive distribution is obtained my marginalising the distribution of t' given $\mathbf{w}$ over the posterior weight distribution. This makes sense as the posterior distribution accounts for uncertainty around $\mathbf{w}$. This intrinsic measure of uncertainty is specific to Bayesian model predictions.

Since the quantities inside the integral are Gaussian densities we apply the property of convolution of Gaussians.

The estimate for the mean of the predictive distribution is given by $\Phi(\mathbf{x}')\boldsymbol{\mu}_{MAP}$. (Recall that $\Phi(\mathbf{x}')$ is the row vector of all basis functions evaluated at $\mathbf{x}'$).

The variance is just given by $\sigma^2(\mathbf{x}') = \sigma^2_{MAP} + \Phi(\mathbf{x}')\Sigma\Phi^T(\mathbf{x}')$. The confidence in the prediction is determined by the variance of this distribution.

Therefore, we can write,

$$p(t^{'}|\mathbf{x}', \boldsymbol{\alpha}_{MAP}, \sigma^2_{MAP}) = \mathcal{N}(\Phi(\mathbf{x}')\boldsymbol{\mu}_{MAP}, \sigma^2_{MAP} + \Phi(\mathbf{x}')\Sigma\Phi^T(\mathbf{x}')) \tag{4.75}$$

4 Discussion

One of the most important steps in the algorithm is setting of the the basis functions represented in Φ which needs to be made beforehand, while there is no formal procedure guiding the user to make a good choice of basis functions a host of techniques both heuristic and empirical have been proposed towards the general problem of setting good design matrices in a regression context [GK99] [WM92].

When maxmixing the likelihood with respect to the hyperparameters α_i a significant proportion of them go to infinity and the corresponding weight parameter w_i has a posterior distribution centered around 0. Hence, the basis functions associated with these parameters can be pruned out of the design matrix. This results in a sparse solution. The $\mathbf{x}_i$ corresponding to the remaining non-zero weights are the *relevance* vectors. By the setting of individual hyper-parameters for the weights $\mathbf{w}$ the algorithm gains control over the influence of each basis function in the design matrix giving a more flexible framework. It is this aspect of the algorithm that differentiates it from Bayesian learning in general.

Further, while it is clear through the practical working of the algorithm that sparsity develops in the model, what is not so obvious are the causes for sparsity. Sparsity indicates that a certain dimension of the model space is not relevant for prediction and can be dropped from the design matrix. Experimentation results in the one-dimensional case show that relevance vectors position themselves around the inflection points of the curve. This makes intuitive sense.

There are several interesting questions surrounding sparsity,

1. Are there certain types of model spaces (design matrices Φ) which encourage sparsity?

2. Is the extent of sparsity affected by the choice of basis functions?

3. Is there a way to induce sparsity into models while retaining their generalization properties.

Chapter 5

Analytical Optimization of the Likelihood

In this chapter we conduct an analytical maximisation of the likelihood function $l(\alpha_i)$ we encountered in eq. 4.65.This maximisation is conducted with respect to a single basis function ϕ_i with the respective hyper-parameter α_i.

$$2l(\alpha_i) = \ln \alpha_i - \ln(\alpha_i + \phi_i^T \mathbf{C}_{-i}^{-1} \phi_i) + \frac{(\phi_i^T \mathbf{C}_{-i}^{-1} \mathbf{t})^2}{\alpha_i + \phi_i^T \mathbf{C}_{-i}^{-1} \phi_i} \tag{5.1}$$

$$= \ln \alpha_i - \ln(\alpha_i + \phi_i^T \mathbf{C}_{-i}^{-1} \phi_i) + \frac{\phi_i^T \mathbf{C}_{-i}^{-1} \mathbf{t}\mathbf{t}^T \mathbf{C}_{-1}^{-1} \phi_i}{\alpha_i + \phi_i^T \mathbf{C}_{-i}^{-1} \phi_i} \tag{5.2}$$

$$\tag{5.3}$$

Define,

$$u(\phi_i) = \phi_i^T \mathbf{C}_{-i}^{-1} \phi_i \tag{5.4}$$

$$v(\phi_i) = \phi_i^T \mathbf{C}_{-i}^{-1} \mathbf{t}\mathbf{t}^T \mathbf{C}_{-1}^{-1} \phi_i \tag{5.5}$$

$$\partial u / \partial \phi_i = 2(\mathbf{C}_{-i}^{-1} \phi_i)^T \tag{5.6}$$

$$\partial v / \partial \phi_i = 2(\mathbf{C}_{-i}^{-1} \mathbf{t}\mathbf{t}^T \mathbf{C}_{-i}^{-1} \phi_i)^T \tag{5.7}$$

$$\tag{5.8}$$

Define,

$$g(x) = -\ln(\alpha_i + x) \qquad \Rightarrow \partial g/\partial x = -\frac{1}{\alpha_i + x} \tag{5.9}$$

$$f(x) = (\alpha_i + x)^{-1} \qquad \Rightarrow \partial f/\partial x = -\frac{1}{(\alpha_i + x)^2} \tag{5.10}$$

Re-writing $2l(\alpha_i)$ with these substitutions we get,

$$2l(\alpha_i) = \ln(\alpha_i) + g(u(\phi_i)) + v(\phi_i) f(u(\phi_i)) \tag{5.11}$$

Then we differentiate by chain rule,

$$\frac{\partial}{\partial \phi_i} 2l(\alpha_i) = \frac{\partial g}{\partial x}(u(\phi_i))\frac{\partial u}{\partial \phi_i} + v(\phi_i)\frac{\partial f}{\partial x}(u(\phi_i))\frac{\partial u}{\partial \phi_i} + \frac{\partial v}{\partial \phi_i} f(u(\phi_i)) \tag{5.12}$$

$$\tag{5.13}$$

$$= -\frac{1}{\alpha_i + u(\phi_i)}\frac{\partial u}{\partial \phi_i} + v(\phi_i)\left(-\frac{1}{\alpha_i + u(\phi_i))^2}\right)\frac{\partial u}{\partial \phi_i} + \frac{\partial v}{\partial \phi_i}\frac{1}{\alpha_i + u} \tag{5.14}$$

$$\tag{5.15}$$

$$= -\frac{1}{\alpha_i + \phi_i^T \mathbf{C}_{-i}^{-1}\phi_i} 2(\mathbf{C}_{-i}^{-1}\phi_i)^T - \frac{\phi_i^T \mathbf{C}_{-i}^{-1}\mathbf{t}\mathbf{t}^T\mathbf{C}_{-1}^{-1}\phi_i}{\left(\alpha_i + \phi_i^T \mathbf{C}_{-i}^{-1}\phi_i\right)^2} 2(\mathbf{C}_{-i}^{-1}\phi_i)^T + \tag{5.16}$$

$$\frac{1}{\alpha_i + \phi_i^T \mathbf{C}_{-i}^{-1}\phi_i} 2(\mathbf{C}_{-i}^{-1}\mathbf{t}\mathbf{t}^T\mathbf{C}_{-i}^{-1}\phi_i)^T \tag{5.17}$$

$$\tag{5.18}$$

Finally, equating $2l(\alpha_i)$ to 0 and multiplying both sides by $\frac{1}{2}(\alpha_i + \phi_i^T \mathbf{C}_{-i}^{-1}\phi_i)$

$$(\mathbf{C}_{-i}^{-1}\mathbf{t}\mathbf{t}^T\mathbf{C}_{-i}^{-1}\phi_i) = \left((\mathbf{C}_{-i}^{-1}\phi_i) + \frac{\phi_i^T \mathbf{C}_{-i}^{-1}\mathbf{t}\mathbf{t}^T\mathbf{C}_{-1}^{-1}\phi_i}{\left(\alpha_i + \phi_i^T \mathbf{C}_{-i}^{-1}\phi_i\right)}(\mathbf{C}_{-i}^{-1}\phi_i)\right) \tag{5.19}$$

Note that $\mathbf{C}_{-i}^{-1}$ is symmetric as its a covariance matrix. We cancel out the factor $(\mathbf{C}_{-i}^{-1}$ giving,

$$\mathbf{t}\mathbf{t}^T\mathbf{C}_{-i}^{-1}\phi_i = \underbrace{\left(1 + \frac{\phi_i^T \mathbf{C}_{-i}^{-1}\mathbf{t}\mathbf{t}^T\mathbf{C}_{-1}^{-1}\phi_i}{\alpha_i + \phi_i^T \mathbf{C}_{-i}^{-1}\phi_i}\right)}_{\lambda}\phi_i \tag{5.20}$$

Note that $\mathbf{t}\mathbf{t}^T$ is (outer product) a rank 1 matrix,

$$\mathbf{t}\mathbf{t}^T\mathbf{C}_{-i}^{-1}\phi_i = \lambda\phi_i \tag{5.21}$$
$$\tag{5.22}$$

Define $v_i = \mathbf{C}_{-i}^{-1}\phi_i \Rightarrow \phi_i = \mathbf{C}_{-i}v_i$ then,

$$\mathbf{t}\mathbf{t}^T v_i = \lambda\mathbf{C}_{-i}v_i \tag{5.23}$$

can be represented as a generalized eigenvalue problem:

Assume the Cholesky factorization of $\mathbf{C}_{-i}$ is known (this is a reasonable assumption as $\mathbf{C}_{-i}$ is symmetric, positive semi-definite.)

$$\mathbf{C}_{-i} = LL^T \tag{5.24}$$
$$\mathbf{t}\mathbf{t}^T v_i = \lambda LL^T v_i \tag{5.25}$$
$$\mathbf{t}\mathbf{t}^T(L^T)^{-1}L^T v_i = \lambda LL^T v_i \tag{5.26}$$
$$L^{-1}\mathbf{t}\mathbf{t}^T(L^T)^{-1}L^T v_i = \lambda L^T v_i \tag{5.27}$$
$$(L^{-1}\mathbf{t})(L^{-1}\mathbf{t})^T L^T v_i = \lambda L^T v_i \tag{5.28}$$

$(L^{-1}\mathbf{t})(L^{-1}\mathbf{t})^T$ is an outer product hence rank 1 matrix with only non-zero eigenvector $L^{-1}\mathbf{t}$. Hence,

$$L^T v_i = L^{-1}\mathbf{t} \tag{5.29}$$
$$v_i = (L^T)^{-1}(L^{-1})\mathbf{t} = (LL^T)^{-1}\mathbf{t} = \mathbf{C}_{-i}^{-1}\mathbf{t} \tag{5.30}$$
$$\Rightarrow \phi_i = \mathbf{t} \tag{5.31}$$

The analytic optimisation says that choosing a basis function to be the targets themselves maximises the marginal likelihood (trivial argument) giving absolutely no generalization. Unfortunately, this is not a practically useful result.

Chapter 6

Kernels: A short introduction

The term 'kernel' has several distinct meanings in different mathematical contexts. A full introduction to the topic is beyond the scope of this work.

A kernel is a positive definite function of two inputs x, y which can be scalars or vectors in a Euclidean space. Kernels are said to encode the similarity between two objects. If two vectors x and y are close together, it is reasonable to expect $k(x, y) = \phi(x)^T \phi(y)$ (where ϕ is a feature transformation $\phi : \mathbb{R}^N \to \mathbb{R}^M$, $x \in \mathbb{R}^N$) to be large. Conversely, if $\phi(x)$ and $\phi(y)$ are far apart, say nearly orthogonal to each other - then $k(x, y)$ is likely to be small. A common misconception is that a kernel transforms a feature vector into a high dimensional space, for instance $f(x) = x^2$ is *not* a kernel function.

Given a feature transformation ϕ, kernel functions, $k : \mathbb{R}^N \times \mathbb{R}^N \to \mathbb{R}$ essentially enable computation of dot products between vectors $x, y \in \mathbb{R}^N$ without explicitly transforming them using ϕ into $\mathbb{R}^M$ where $M > N$. Further, the computation of $k(x, y)$ may be inexpensive even though $\phi(x)$ may be expensive to calculate due to its high dimensionality. The usefulness of this is that the algorithm can be made to learn in high dimensional feature space given by ϕ without ever explicitly representing the vectors as $\phi(x)$. This is called the *kernel* trick.

To make it clear with an example, let $x, y \in \mathbb{R}^2$, consider the kernel function,

$$k(x, y) = (1 + x^T y)^2 \text{ where } x, y \in \mathbb{R}^2 \tag{6.1}$$

This can be written as,

$$
\begin{aligned}
k(x,y) &= 1 + 2x^T y + (x^T y)^2 \\
\Rightarrow 1 &+ 2x_1y_1 + 2x_2y_2 + x_1^2y_1^2 + x_2^2y_2^2 + 2x_1x_2y_1y_2 \\
\Rightarrow (1, &\sqrt{2}x_1, \sqrt{2}x_2, x_1^2, x_2^2, \sqrt{2}x_1x_2) \quad \bullet \\
(1, &\sqrt{2}y_1, \sqrt{2}y_2, y_1^2, y_2^2, \sqrt{2}y_1y_2) \\
&\Rightarrow \phi(x)^T \phi y
\end{aligned}
$$

where $\bullet$ is the dot product.

One advantage of kernel functions is that the complexity of the optimisation remains solely dependent on the dimensionality of the input space and not of the feature space. Hence, it is possible to operate in a theoretically infinite dimensional feature space. We will shortly discuss how the Gaussian kernel corresponds to an infinite dimensional feature mapping ϕ.

Broadly speaking, given a candidate function k, we can tell if it could serve as a valid kernel if there exists a transformation ϕ such that,

$$k(x,y) = \phi(x)^T \phi(y) \quad \forall x, y \in \mathbb{R}^N \tag{6.2}$$

Further, kernel functions are symmetric and positive semi-definite.

In practice, a selected group of kernels turn out to be appropriate to use and are widely applicable.

Table 6.1: Popular kernel choices

Type	Function	Parameters
Polynomial	$k(x,y) = (x^T y + \theta)^d$	d, θ
Gaussian (Radial Basis Function)	$k(x,y) = e^{\{\frac{-1}{2\sigma^2}\|\|x-y\|\|^2\}}$	$\gamma = \frac{-1}{2\sigma^2}$
Sigmoid Kernel	$k(x,y) = tanh(\eta xy + \theta)$	η and θ
Spectrum Kernel for strings	Count the number of sub-strings in common	It is a kernel since it is a dot product between vectors of indicators of substrings.

It is worthwhile mentioning that there is no analytical way to choose good kernels for the model space. The most common approach is to try various kernels, adjust their parameters via search and minimize the generalization error.

1 Understanding the RBF Kernel

RBF stands for Radial Basis Functions[1]. The RBF kernel is derived from the Gaussian density by dropping the factors that do not contain any variables. It is given by,

$$k(x, y) = \exp\left\{\frac{-||x - y||^2}{2\sigma^2}\right\} \tag{6.3}$$

for two vectors $x, y \in \mathbb{R}^d$, $||x - y||^2$ is the squared euclidean distance between the two vectors. A simpler definition replacing $\frac{1}{2\sigma^2}$ by γ is given by,

$$k(x, y) = e^{\left(-\gamma||x - y||^2\right)} \tag{6.4}$$

This is a reasonable measure of similarity as when x and y are close, or $||x - y||$ is close to 0 $\Rightarrow k(x, y)$ is close to 1. Conversely, when x and y are far apart, $\Rightarrow k(x, y)$ is close to 0. The value of the RBF kernel conveniently ranges between zero and one.

By using Taylor's expansion $e^x = 1 + x + ... + \frac{1}{k!}x^k$ one can see how e^x is a kernel with an infinite set of features corresponding to polynomial terms, hence its transformation function ϕ maps the original feature space to an infinite dimensional one. The kernel $k(x, y)$ gives a closed form expression for computing the dot product in this higher dimensional space even when there isn't a way to represent the vector directly in this space. The main idea is that even if a vector has infinite dimensions, similarity (expressed as a dot product) between two vectors in this space has a well-defined value. The kernel expresses the dot product through an equation that indicates what the similarity value converges to.

A RBF kernel applied to a vector (a high dimensional data point) is an exponentially decaying function in the input feature space. Its maximum value is achieved at the data point and it decays uniformly in all directions around the vector leading to hyper-spherical contours of the kernel function.

The parameter γ controls the width of the kernel which determines smoothness. In n-dimensions this parameter can be thought of as controlling the hyper-sphere of influence of a vector. Again, the influence of gamma is easier to visualize in 3d. A small gamma implies larger variance (as per eq. 6.4), this leads to relatively flat peaks around the vectors, giving smoother surfaces and requiring few vectors with large spheres of influence.

A large gamma implies lower variance and in 3d this gives a surface of sharp peaks

[1]Functions of this type yield values which depend on the distance between a given vector and the origin or another vector

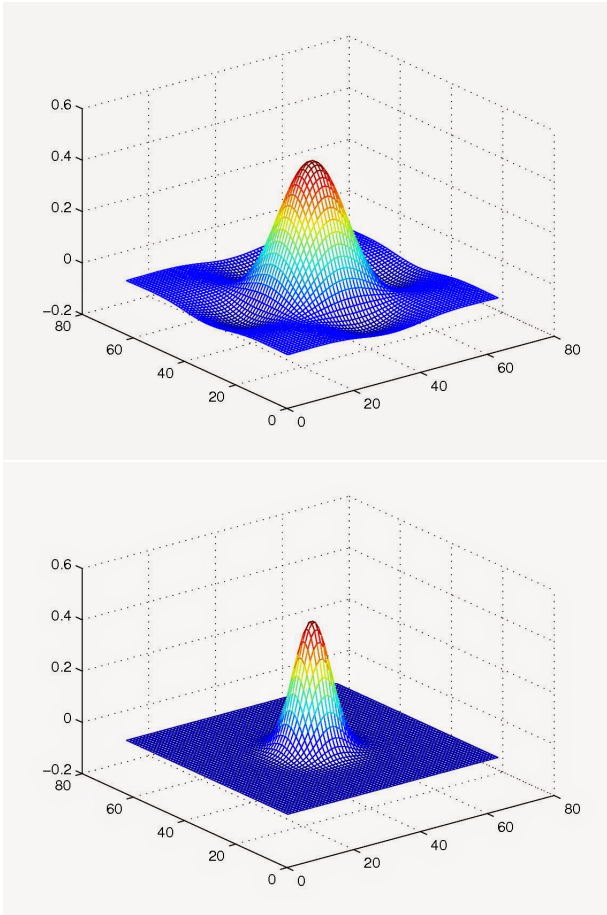


Figure 6.1: These graphs depict the shape of the RBF kernel around a single 2d data point. The top figure depicts a kernel with γ value of 5 (flatter peak, high variance) and the bottom figure depicts a γ value of 10 (sharper peak, lower variance). *Retrieved from [V.K14]*

localizing the influence of each basis function. The graphs in fig. 6.1 depict the influence of two values of gamma, $\gamma = 5, 10$

The kernels localization properties make it a strong choice of kernel for problems which need arbitrarily flexible curves (in a regression context) or boundaries (in a classification context).

Chapter 7

Experiments

All the experiments presented in this section have been conducted with a design matrix Φ with Gaussian kernels as basis functions. Further, the Gaussian kernels have been centered on each data point.

$$\Phi = \begin{bmatrix} k(x_1, x_1) & k(x_2, x_1) & \dots & k(x_n, x_1) \\ k(x_1, x_2) & k(x_2, x_2) & \dots & k(x_n, x_2) \\ \vdots & \vdots & \vdots & \vdots \\ k(x_1, x_n) & k(x_2, x_n) & \dots & k(x_n, x_n) \end{bmatrix} \tag{7.1}$$

where $k(x_i, x_j) = \exp\left\{\frac{-||x_i - x_j||^2}{2\sigma^2}\right\}$, it is easy to see that this matrix has 1 on the diagonals.

1 Visualizing the Basis functions

It is useful to visualise the model space (the term model space is used here to mean the collection of all basis functions), it subtly differs from design matrix as the latter refers to set of all basis functions assembled in a matrix.

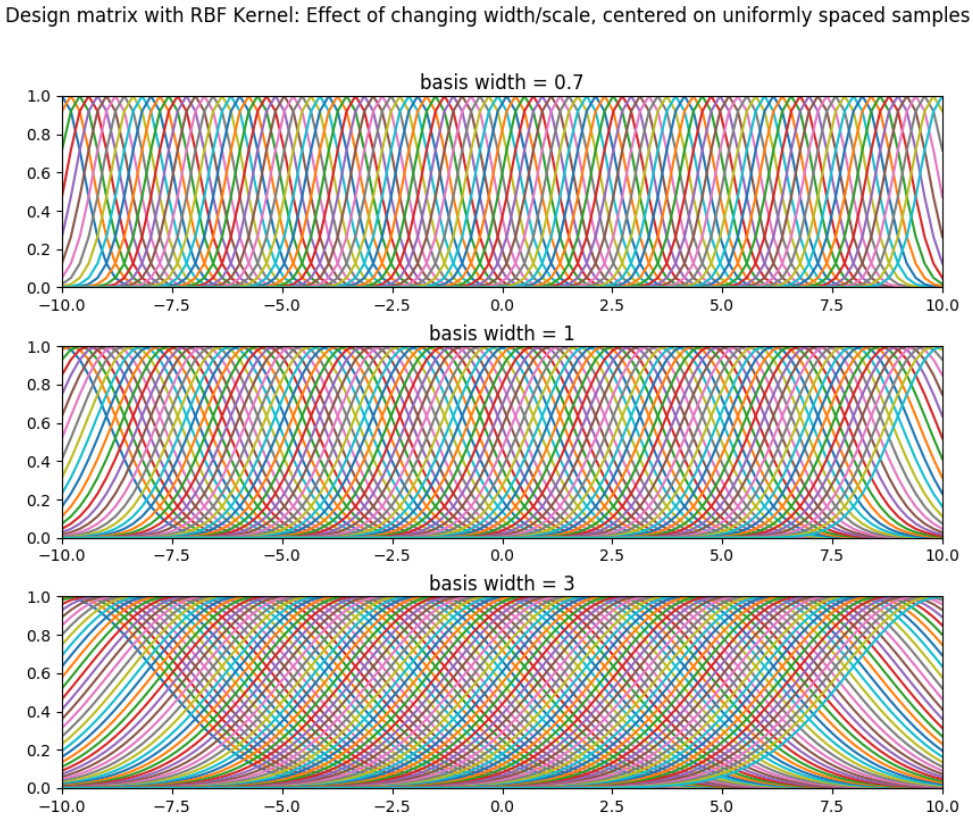


Figure 7.1: This graph plots basis functions each of them centered on N = 100 data points sampled at a fixed interval. As the width σ of the kernel increases uniformly, the Gaussians grow wider with more overlap

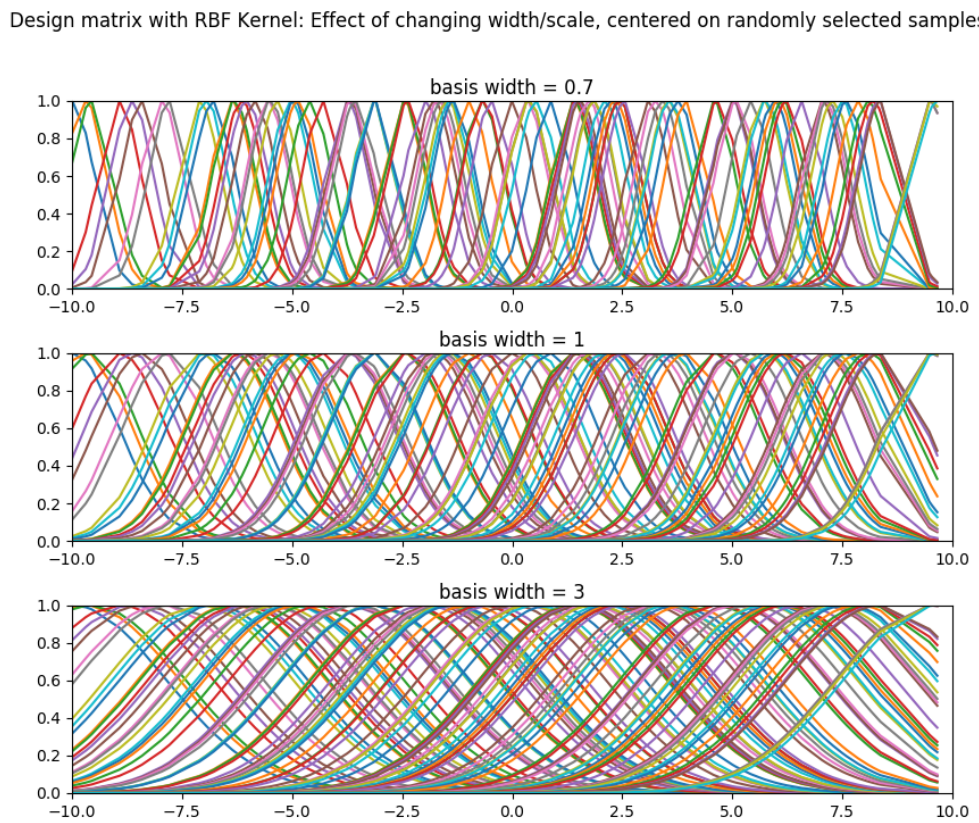


Figure 7.2: This graph plots basis functions, each of them centered on N = 100 data points sampled uniformly in the range (-10,+10). If 2 data points are closer together the basis overlap significantly.

Fig. 7.1 and 7.2 show the localising properties of the Gaussian kernel. Each Gaussian curve denotes a single basis function centered on a data point which corresponds to the peak at 1. A Gaussian kernel centered on a data point does not have any influence on points lying far from it as decays exponentially and uniformly around the mean/center.

For two-dimensional inputs spaced uniformly on a xy-grid, the Gaussian kernels give a surface of bumps with the centers lying on each data point.

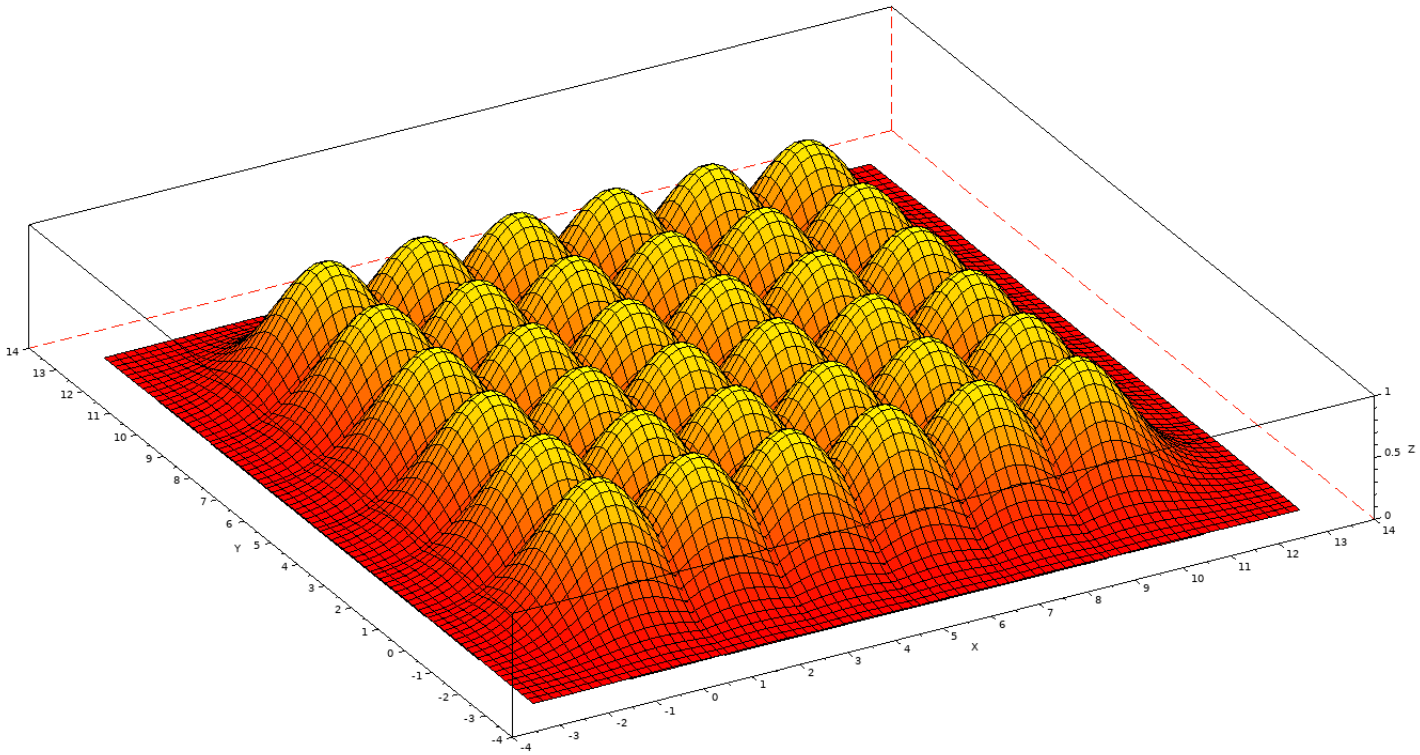


Figure 7.3: Gaussian model space for 2d input data

On the other hand, a basis matrix with polynomial kernels tend to have an effect on data lying far from where they are centered. They are examples of *global* kernels while the RBF kernels are examples of *local* kernels.

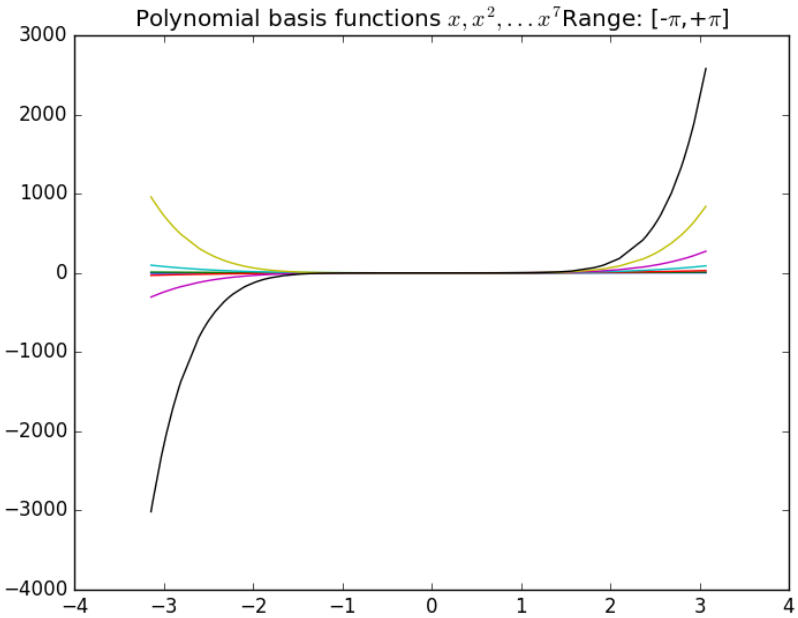


Figure 7.4: The figure shows a range of high order polynomials on data samples in the range $[-\pi, \pi]$. High order polynomials tend to have an overswing effect.

In the experiments we try to predict a one-dimensional generative function given by,

$$y = 0.5\sin(x) + 0.5x - 0.02(x-5)^2 \tag{7.2}$$

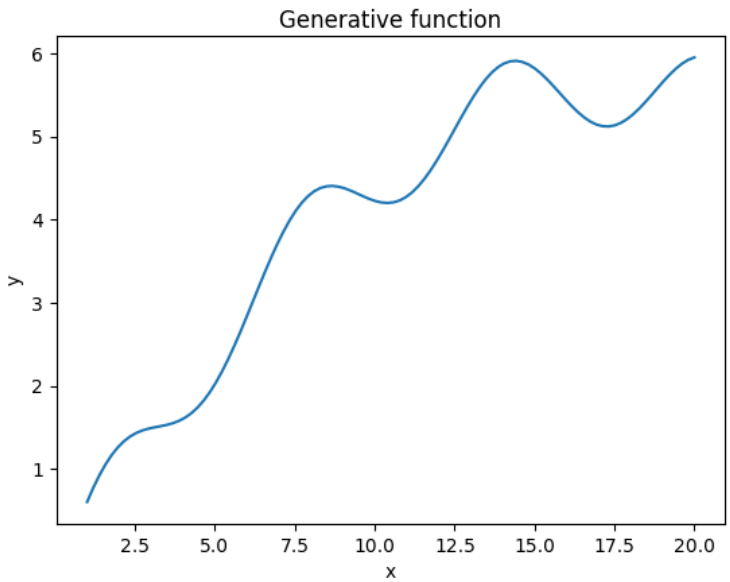


Figure 7.5: A one-dimensional generative function

2 Effect of changing the width and location of the kernels

In the examples below we run the fast version of the algorithm as described in section 3.2. The Gaussian kernels are centered on each data point. The output graphs present a comprehensive picture on the behaviour of the algorithm.

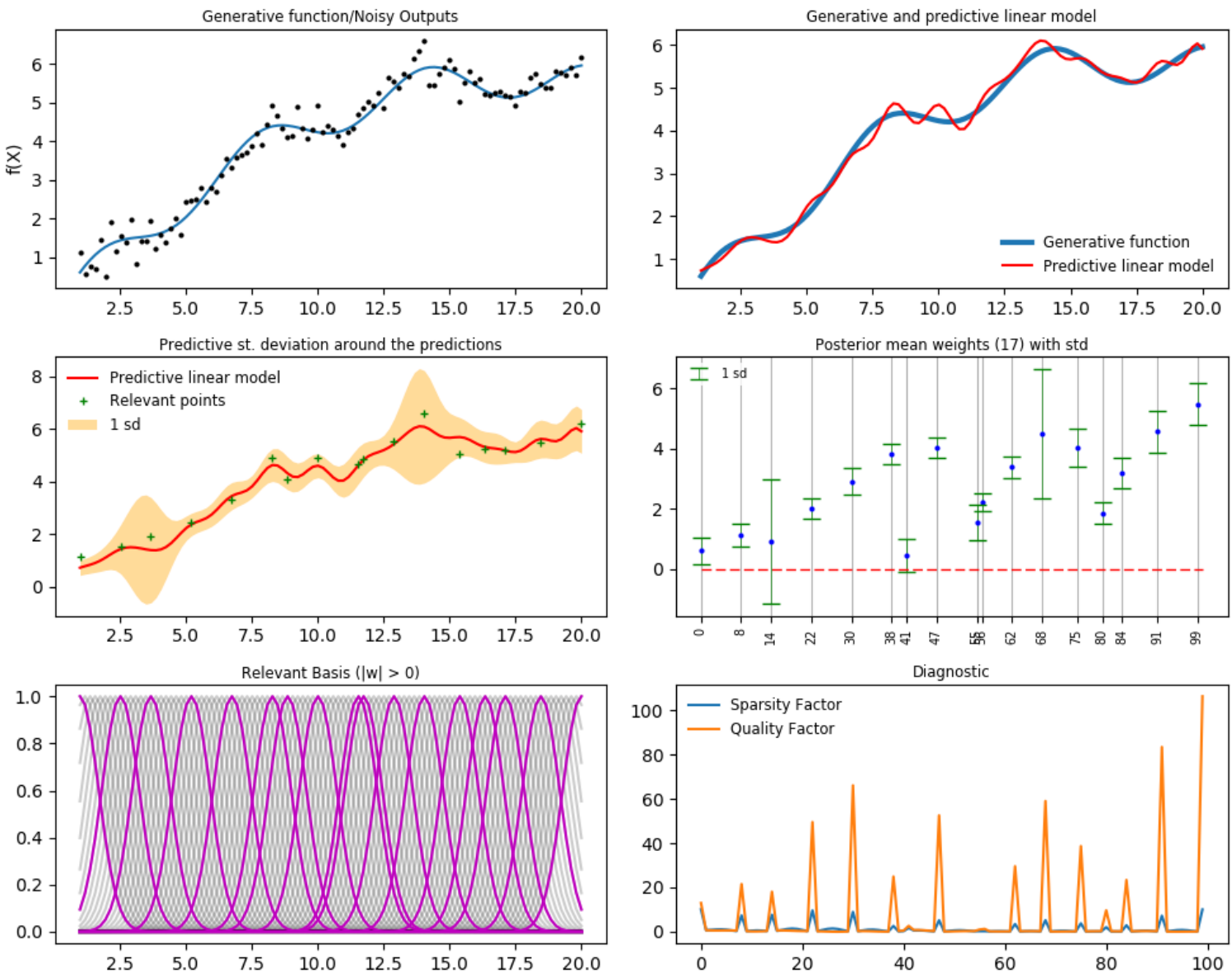


Figure 7.6: The plot shows the fit report of running the sparse Bayesian algorithm on 100 noisy data points of the generative function, the basis width is chosen to be 1, the kernels are centered uniformly with fixed step size. We make some observations,

1. The kernel width of 1 is too narrow for a good fit, the predictive curve is more wriggly that we would desire.
2. We can notice some alignment in the chosen basis functions (overlapping basis functions), we don't do a collinearity check to weed out such overlap but it is a natural extension that can be added to the algorithm.
3. Further we notice that the sparsity and quality factor corresponding to the overlapping basis functions are close to zero. While, the values might be close to 0 but they still satisfy the selection condition of $q_i^2 > s_i$. It might be worth weeding out basis function where their values are close to 0.

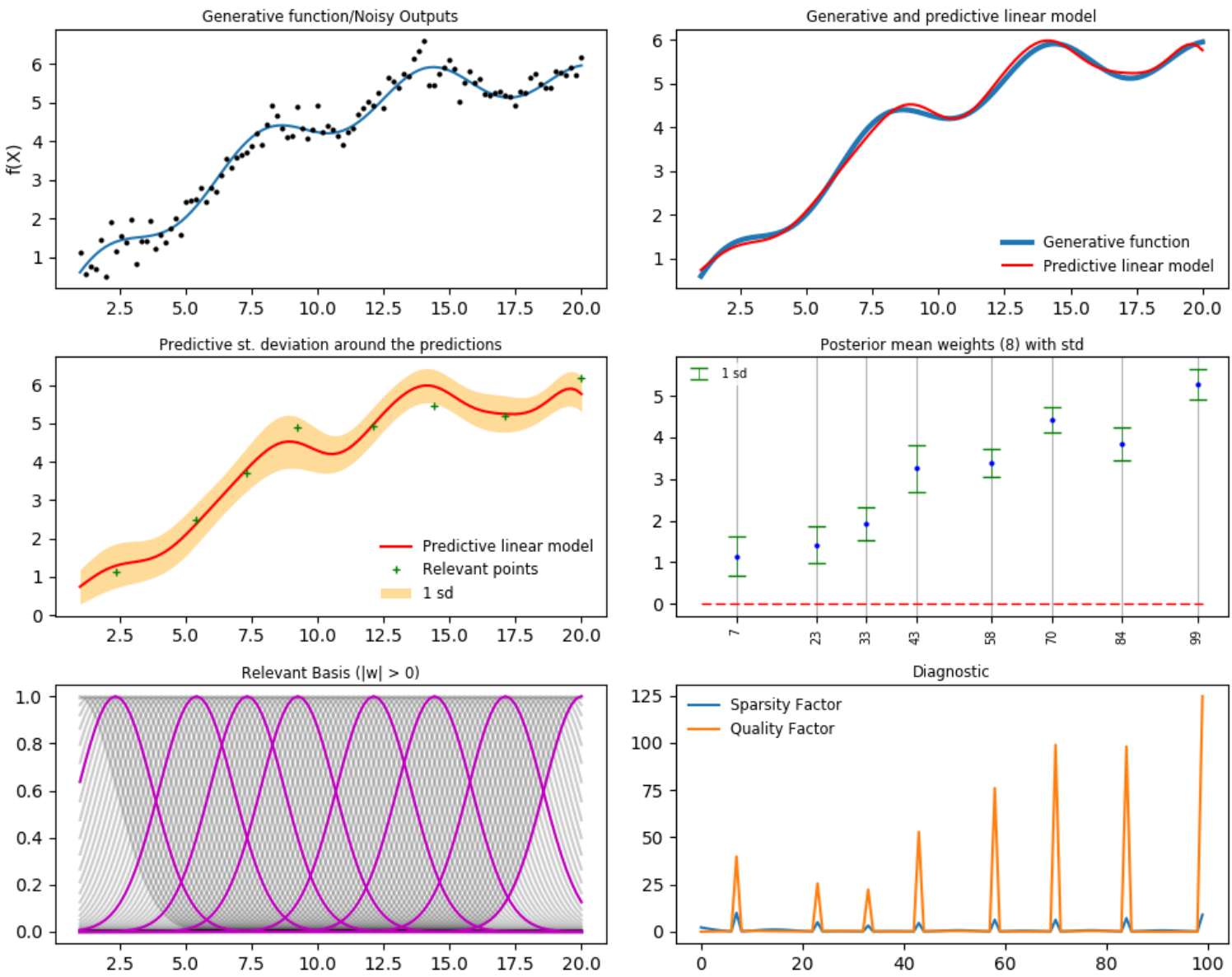


Figure 7.7: The plot shows the fit report of running the sparse Bayesian algorithm on 100 noisy data points of the generative function, the basis width is chosen to be 2, the kernels are centered uniformly with fixed step size. We make some observations,

1. We can notice that the fit here is a lot better than the previous case where the width was chosen to be 1. We see a lot less wriggles and much smoother fit.
2. We also notice that the number of basis vectors corresponding to non-zero weights has gone down from 17 to 8.
3. Convergence was reached in 45 iterations as opposed to 237 in the previous case.
4. We don't encounter overlap in basis functions.

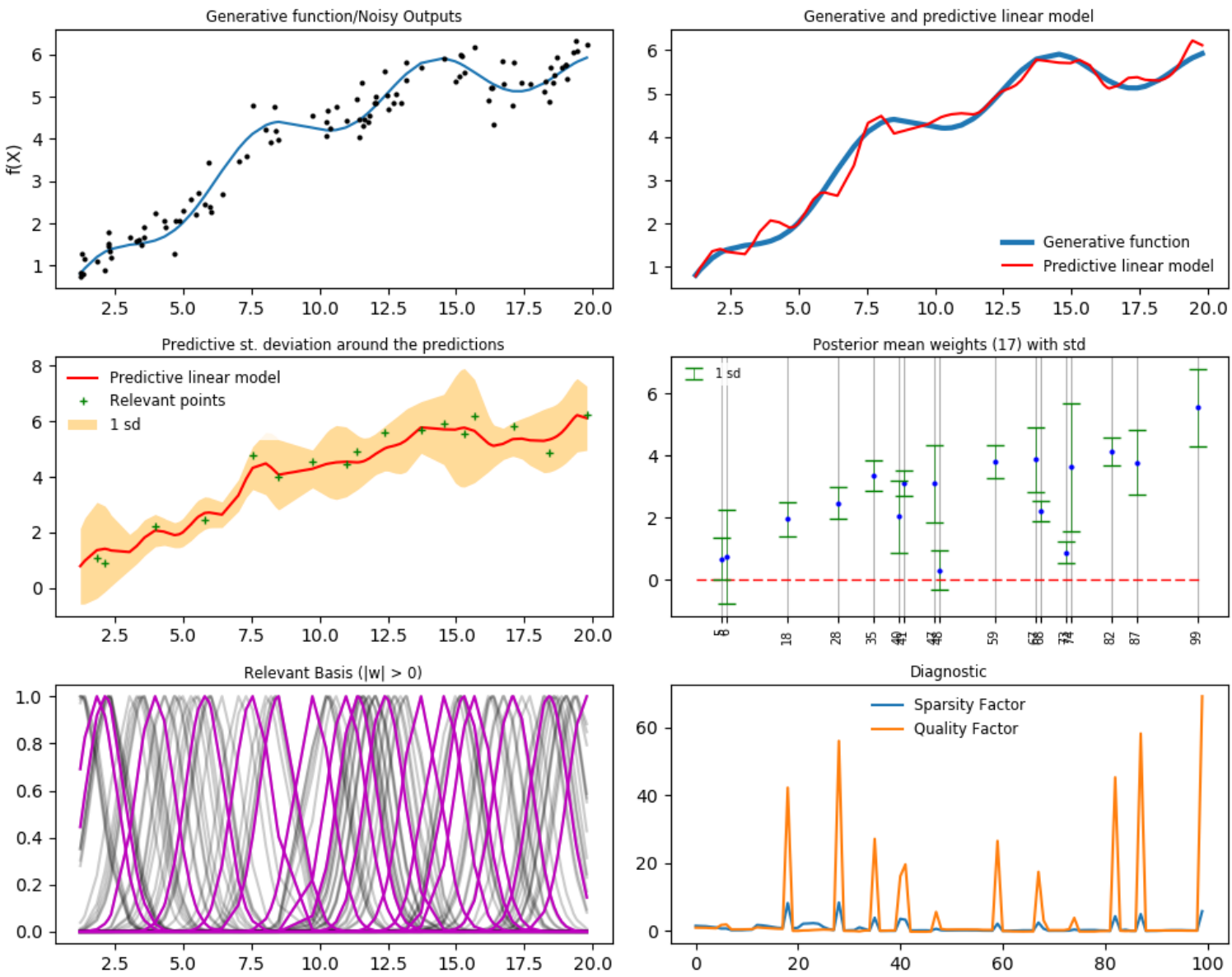


Figure 7.8: The plot shows the fit report of running the sparse Bayesian algorithm on 100 noisy data points of the generative function, the basis width is chosen to be 1, kernels are centered on the actual training data sampled randomly from the range. We make some observations,

1. Data points close together tend to have overlapping basis functions, the algorithm is not able to automatically prune them out.
2. We notice that in some of the cases where aligned basis functions are selected, the weights are quite disparate (this could indicate that the algorithm is trying to cancel out some effect of the overlap.)

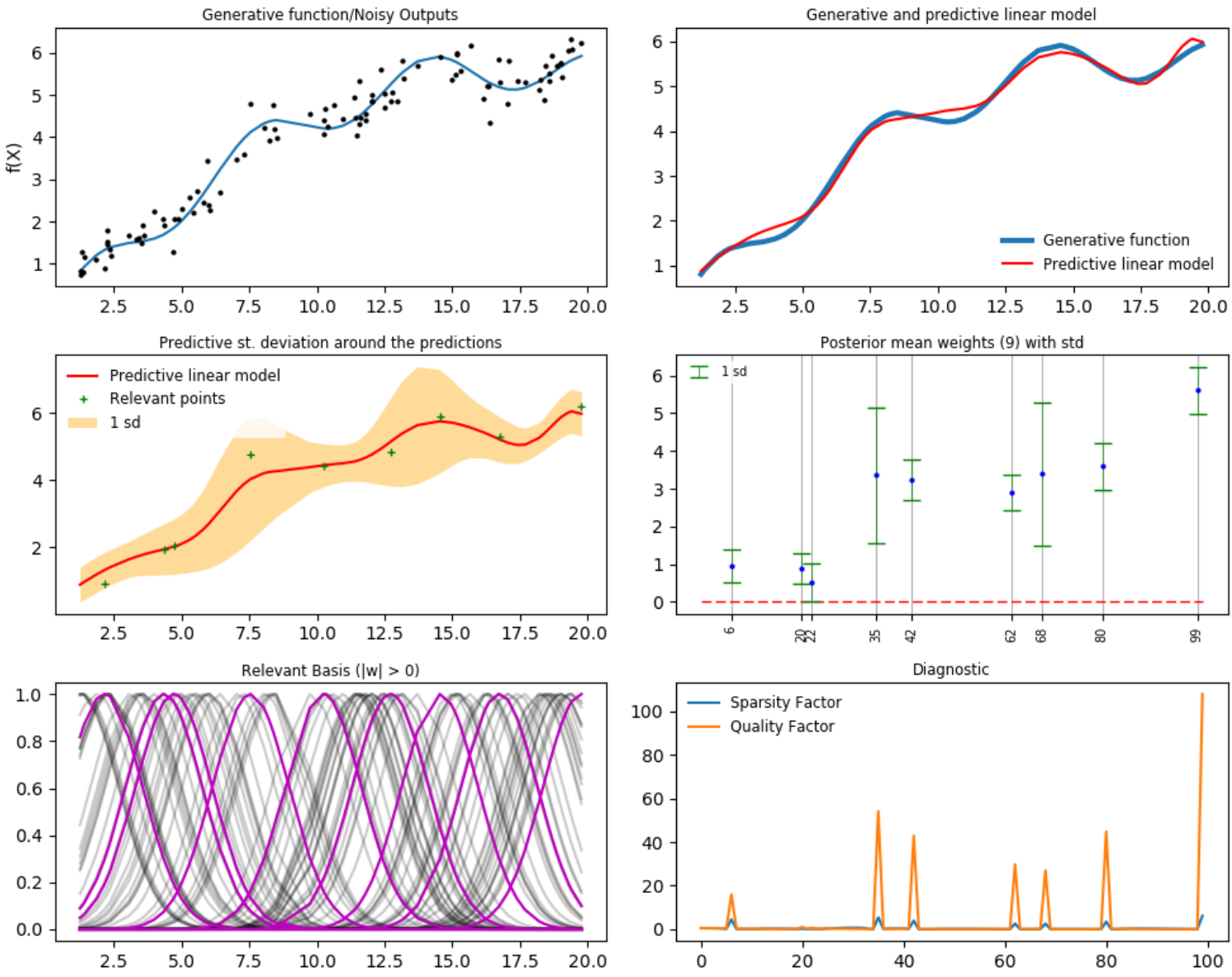


Figure 7.9: The plot shows the fit report of running the sparse Bayesian algorithm on 100 noisy data points of the generative function, the basis width is chosen to be 2, kernels are centered on the actual training data sampled randomly from the range.

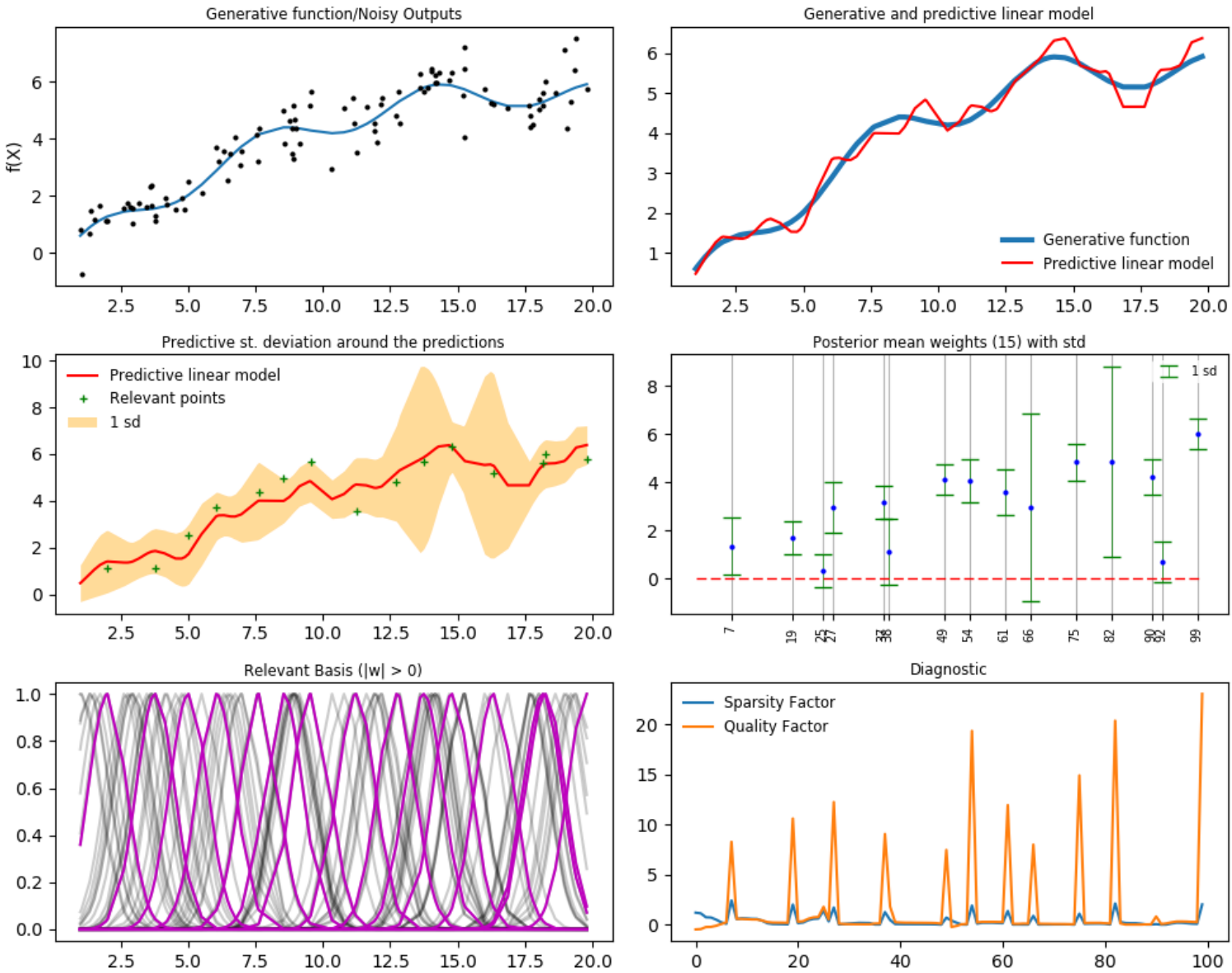


Figure 7.10: The plot shows the fit report of running the sparse Bayesian algorithm on 100 noisy data points of the generative function, the basis width is chosen to be 1, kernels are centered on the actual training data sampled randomly from the range, further, we show the effect of adding additional noise to the data. The fit under a narrow kernel choice of 1 deteriorates further, we also notice wider errors on the predictions.

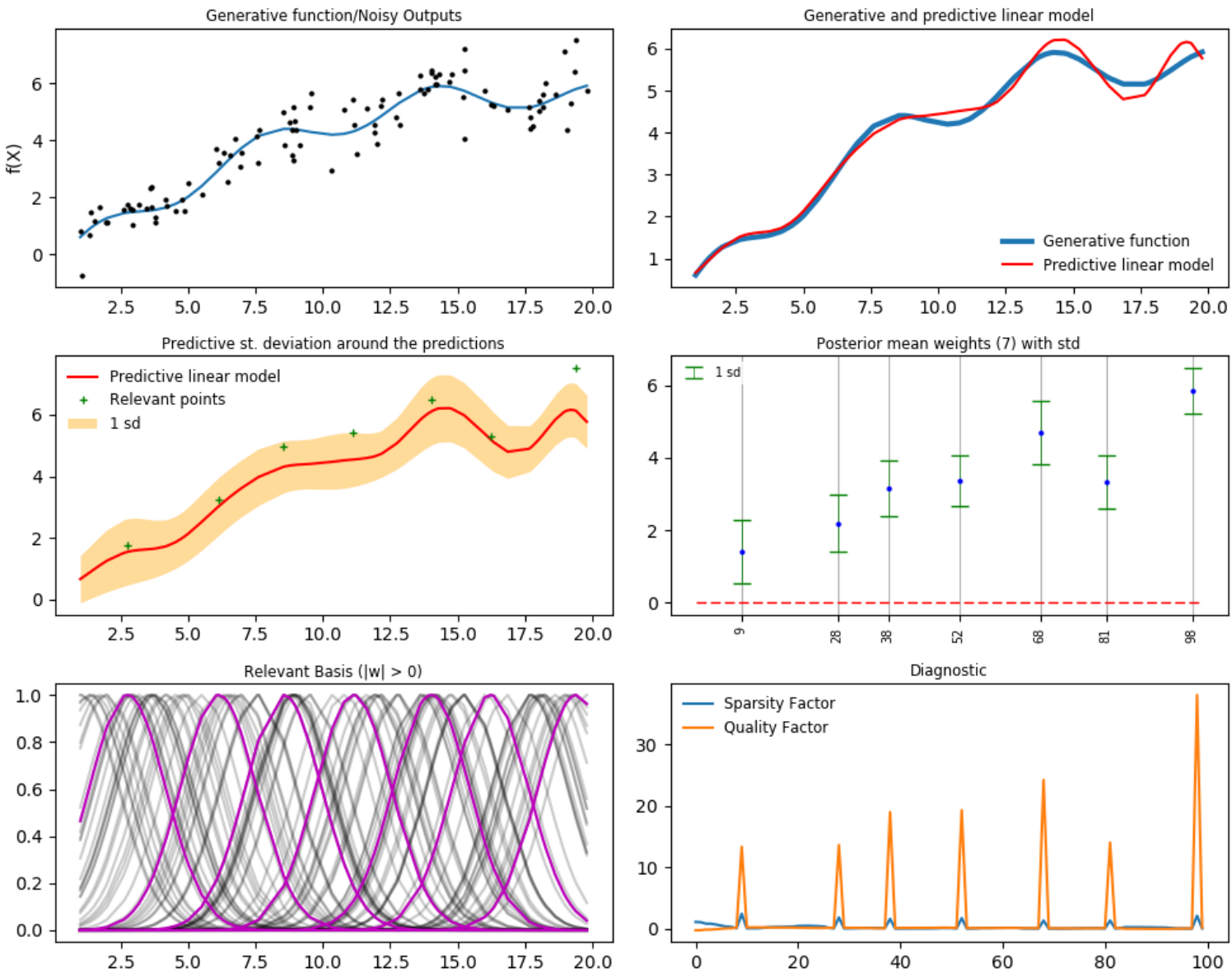


Figure 7.11: The plot shows the fit report of running the sparse Bayesian algorithm on 100 noisy data points of the generative function, the basis width is chosen to be 2, kernels are centered on the actual training data sampled randomly from the range, further, we show the effect of adding additional noise to the data.

3 Summary and Extensions

1. It is evident that both the location and kernel width affect the performance of the algorithm hence, the algorithm can benefit from systematic optimisation.

2. There are two approaches to the optimisation, either to choose a global basis width or try to optimise the width for each of the basis functions. In scenarios where we have a large number of basis functions, the latter approach might run into difficulties.

3. The location of the kernels can be algorithmically chosen and be tied to the spatial properties of the data distribution. For instance one approach could be to convolve the Gaussians for points which appear as clusters and choose stand alone ones to model the outliers.

4. Multicollinearity check would remove basis functions which are closely aligned together, this would also ensure that the algorithm converges faster.

5. An important thing to note (this is not been commented on in the individual plots) is the error bars on the predictive distribution (middle, left figures), we can see that they have the tendency to bulge out around the relevant points (non-zero basis) and narrow down in parts where there is no data. This is very counter-intuitive as we expect the model to be more confident around the data (narrow error bars) and less confident when extrapolating. It turns out that this is an artefact of the algebraic form of the variance of predictions given by,

$$\sigma^2(\mathbf{x}') = \sigma^2_{MAP} + \Phi(\mathbf{x}')\Sigma\Phi^T(\mathbf{x}') \tag{7.3}$$

   σ^2_{MAP} acts as a floor while the contribution of the second term is larger for points closer to the basis function centers (wider error) and smaller for points away from the basis centers (narrow error). This problem can be healed by adopting an alternative, the Gaussian process regression [RQC05].

6. This might be intuitive but the chosen basis functions correspond to the inflection points on the curve. Localised kernels like Gaussians are able to model the kinks in the curve (a polynomial basis might not fair so well.)

The experimentation under Gaussian kernels is a work in progress with a lot more ideas to try and insights to be gained. The goal is to integrate the parameter selection of the kernels as a part of the learning step.

Chapter 8

Research Goals and Timeline

The aim of this thesis is to develop a method for automatically constructing a suitable model space for prediction in a regression context. The term model space refers to a design matrix Φ where each column results from the application of basis functions $\{\phi_i\}_{i=1...M}$ to the set of input data points. It is important to clarify that the term 'basis functions' refers to arbitrary mathematical functions that act on a single input data point while kernel functions refer to a class of functions which act on pairs of input data points and capture a notion of similarity between two input data points. Bayesian Learning does not impose any restrictions on the type of functions that can be used in the design matrix; they can be kernel functions, arbitrary basis functions or a combination of both; the focus of my research would be constructing the design matrix using suitable kernel functions.

1 Automated Model Construction

The performance of multivariate regression is closely tied to the selection of appropriate basis functions for the design matrix. Traditional statistical theory surrounding regression does not provide a theoretically motivated criterion for selection of these functions. Further, the internal parameters of each basis function called hyperparameters need to be optimized keeping in mind several factors, the most fundamental of which is bias-variance trade-off (described in section 4 of Chapter 1). In models with a reasonably small set of parameters the optimization of these is typically achieved through grid-search where several combinations of parameters are systematically tested to optimize a chosen metric. However, in the Bayesian world cross-validation is not always an option as we have a large number of parameters to set. For instance, if our design matrix has kernels of different type then custom hyperparameters for each kernel need to be carefully selected.

This two-fold meta-learning exercise of constructing a model space can be summarized as follows:

1. How do we select good kernel functions for the design matrix?
2. How do we select hyperparameters for the chosen kernel families? Further, this task can be targeted in different ways:
   - By analytically deriving (if tractable) the hyperparameters through maximising the marginal likelihood.
   - By taking a more empirical approach like grid-search (if feasible) and selecting the internal parameters jointly through optimization of performance metrics.
   - By taking a hybrid approach which has elements of both analytical and empirical search for optimal hyperparameter values.

Decisions 1.and 2. are currently based on heuristics, empirical pre-examination of the data, even domain specific knowledge or common sense. For instance, it is not meaningful to apply the RBF kernel to model textual data. The question is if appropriate kernel choices can be learnt from the given training data?

AutoML is a project pioneered at the Engineering department at Cambridge to progressively automate different steps of the machine learning pipeline by making decisions similar to 1. and 2. [BYC13] talks about automating steps in complex computer vision problems.

Sections 1.1 and 1.2 provide some more detail to 1. and 2.

1.1 Kernel Design

Selection of good kernel functions is tightly linked to the performance of a regression system. Hand-crafting kernels is hard as kernel functions must satisfy a set of properties collectively called *Mercer's theorem*. Mercer (1909) showed that a necessary and sufficient condition for a symmetric functions $k(x, z)$ to be a kernel is that it has to be positive definite. This symmetry and positive definiteness of kernels enable them to have some convinient closure properties.

Kernels are closed under operations of addition, multiplication and linear composition with a function. Essentially, if $k_1(x, y)$ and $k_2(x, y)$ are valid kernels on $\mathbb{R}^n$ (i.e. $k : \mathbb{R}^n \times \mathbb{R}^n \Rightarrow \mathbb{R}$) then $k_3(x, y)$ is a valid kernel in each of the following cases:

1. $k_3(x, y) = k_1(x, y) + k_2(x, y)$
2. $k_3(x, y) = k_1(x, y) \times k_2(x, y)$
3. $k_3(x, y) = k_1(\phi(x), \phi(y))$ where ϕ is a real-valued function.

These allow one to construct a rich open-ended class of models to express different structures in the data. In fig. 8.1 we visualise different kernel functions applied to one

dimensional data (each of the plots below depict a single column of a design matrix).

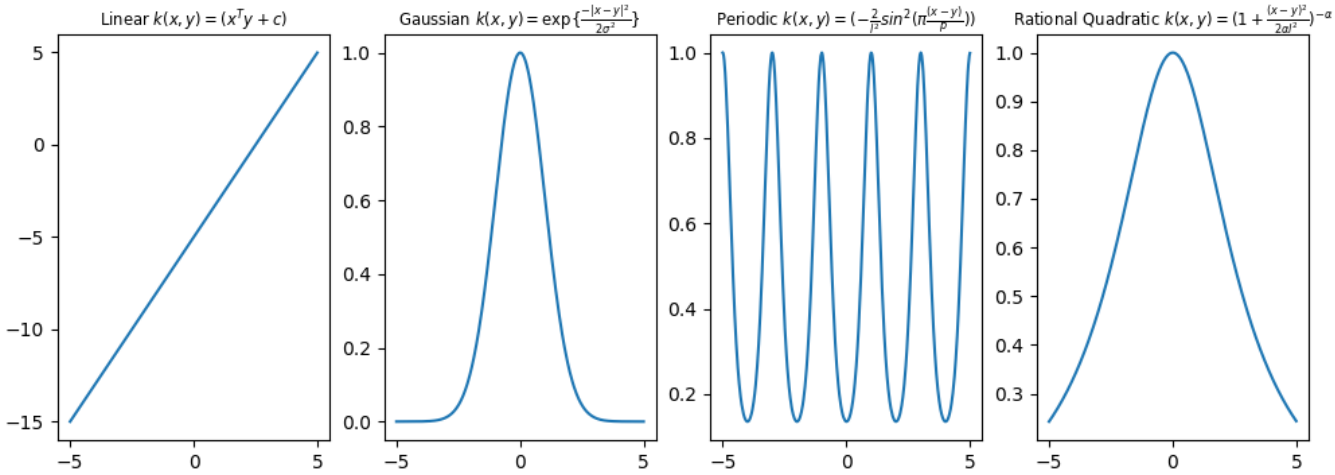


Figure 8.1: A collection of base kernels

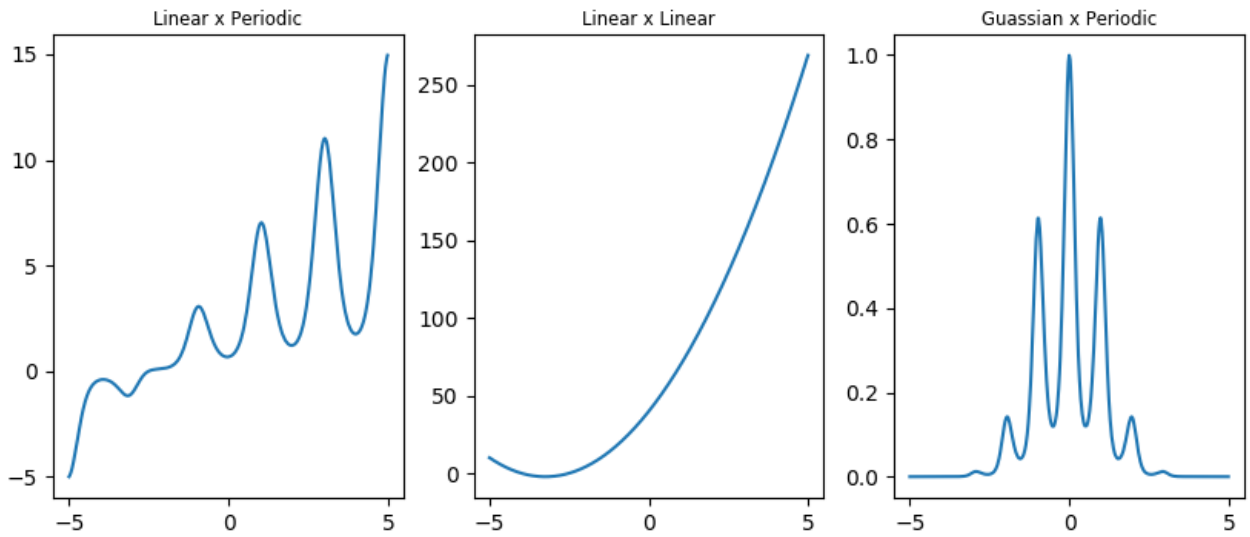


Figure 8.2: Combining Kernels through multiplication to express more complex structures. For instance, multiplying a linear kernel with a periodic kernel is able to model data with a periodic pattern and increasing amplitude. Multiplying linear with linear kernels enables one to construct polynomial kernels and multiplying a Gaussian with a periodic kernel captures behaviour that is locally periodic.

The aim is to automatically identify which class of kernels is suited for a given problem. This question has so far been the remit of human experts who would choose the parametric form of the kernel. There are numerous papers which show that human experts

are capable of manually constructing composite kernels in one or two dimensions. However, in higher dimensions manually designing a kernel is hard to do. The idea here is that a systematic search can consider possibilities that might otherwise be missed.

Here is a simple search recipe.[1] We define a set of base kernels and consider all kernels that can be built by using property 1. and 2. of kernels above. This is easily expressed by a tree.

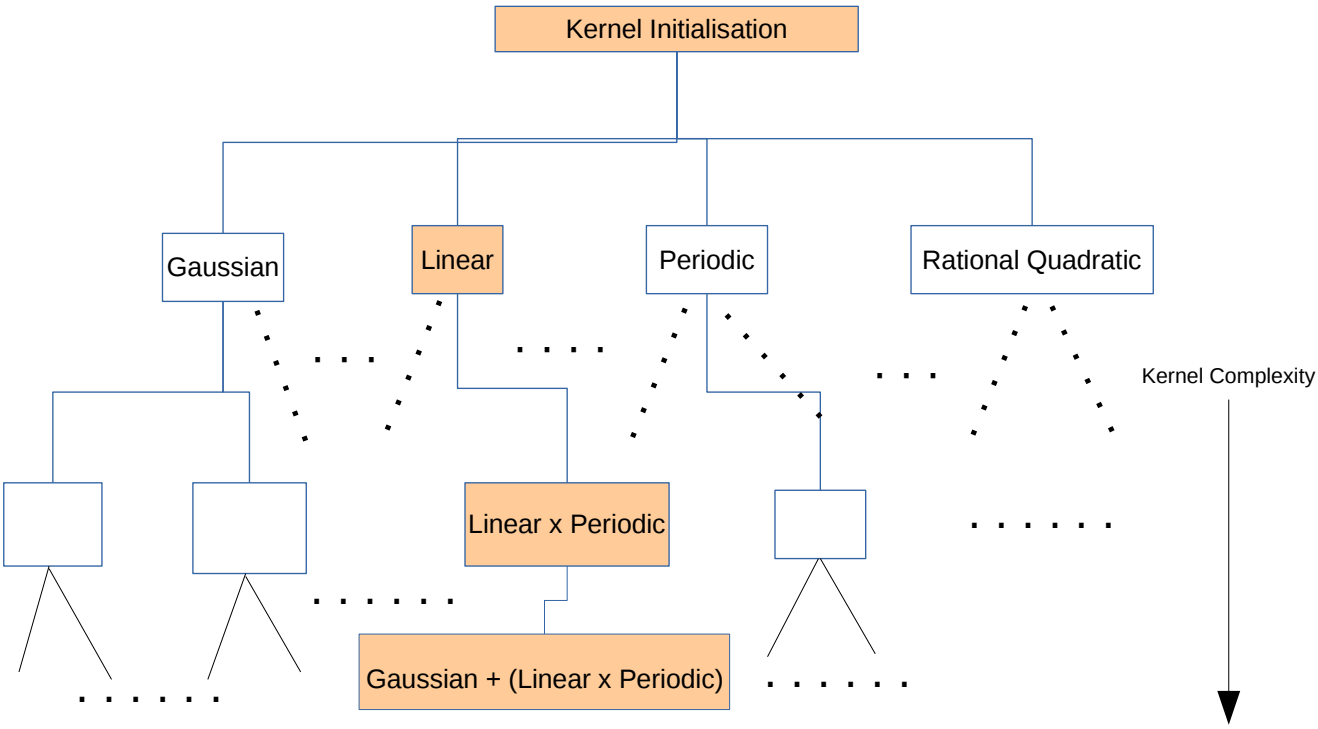


Figure 8.3: In this search tree the highlighted components depict the iterative kernel construction. The design matrix we end up with will have all the base components along with the more complex kernel constructions.

$$\Phi = [\text{ Linear, Periodic, Linear} \times \text{Periodic, Gaussian, Gaussian} + (\text{Linear} \times \text{Periodic})]$$

The primary difficulty is in deciding how to rank kernels in each stage so we can traverse

[1] Duvenaud [Duv14] in his thesis (2014) based on Gaussian processes talks about expressing structure through kernels and constructing composite kernels using a similar iterative search recipe but the main difference here is that instead of constructing one complex kernel to capture the full extent of data structure we can include any number of simpler and complex kernels for our design matrix and let Sparse Bayesian learning pick out the relevant ones.

down the tree. In high-dimensional data one can start with a base kernel which is one dimensional and then consider each dimension incrementally and decide whether to add or multiply to the existing base kernel to capture additional structure. (Adding and multiplying kernels across individual dimensions results in kernels over multi-dimensional inputs). In this way multiple kernels can be chosen for our design matrix.

[Gen01], [DGRC11] offer good summaries on the exercise of recycling new kernels from old ones. [DLG+13] talks about choosing kernel functions for non-parametric regression.

1.2 Kernel Learning

Once we have chosen the families of kernel functions we want to use in our design matrix we need to optimize their hyperparameters. Hyperparameters can either be set before the commencement of the learning step (where we seek to learn the weights) or can be integrated as a part of the learning step. In regression models we are typically trying to learn the weights by using the maximization of the marginal likelihood as the criterion. In Sparse Bayesian learning this happens iteratively and the hyperparameter selection can we weaved into the learning step.

1. Systematic grid search: Under this approach, the kernel families and hyperparameters are set during the pre-processing stage. The design matrix is fixed before the learning step and does not change during learning. In other words, the learning has to be repeated for different settings to decide on the best setting.

2. Adaptive tuning of hyperparameters: In this case, our design matrix is not static during learning and changes as the kernel functions inside them change. The marginal likelihood is chosen as the objective for maximization and gradient methods are used when differentiating does not yield closed form answers.

The two methods above are summarised in fig. 8.6 while fig. 8.4 lays out the taxonomy.

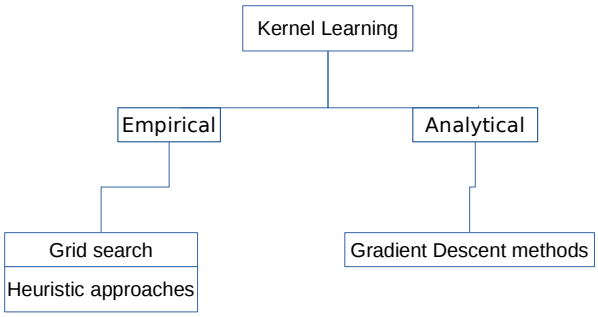


Figure 8.4: Taxonomy of Kernel Learning Approaches

It is important to note that optimizing over kernel hyperparameters is not a convex optimisation problem and there is the problem of getting stuck in local optima. [DLG+13] presents some procedures on how to alleviate this problem.

1.3 Model Selection

A design matrix with kernels can be thought of as a model space. Any system that compares different model spaces does so on the basis of some criteria, like a loss function or a marginal likelihood. A host of different model selection criteria specified in section 2 of chapter 3 select between model spaces after they have been applied to a test set but when searching for a model space at the time of training/learning or pre-training, a different set of criteria might be need to be developed.

2 Software Development

The existing python software which performs the Sparse Bayesian Learning inference procedure has the following features:

1. It supports both one-dimensional and high-dimensional data.
2. A design matrix is chosen and fixed in advance, no new kernels are imagined or constructed either in pre-processing or as part of the learning step.
3. The focus of the learning exercise is $\Rightarrow$ given the parametric structure imposed by the design matrix, infer the best possible weights (the best weights maximize the marginal likelihood).

An illustration of workflow under the existing software system is presented in 8.5.

In order to meet the set research goals I envision developing a software system that would have the following capabilities:

1. Ability to specify an arbitrary number of kernel functions for the design matrix.
2. Ability to specify hyperparameters for each kernel in the design matrix.
3. Ability to auto-tune the hyper-parameters of the basis functions (if turned on): This task lies in the domain of kernel learning described above.
4. Ability to compute and publish a host of model comparison metrics.

Programmatically, this would entail writing an explicit Kernel class. Objects of this class would entail a functional form, hyperparameters, autotuning flag and a host of methods

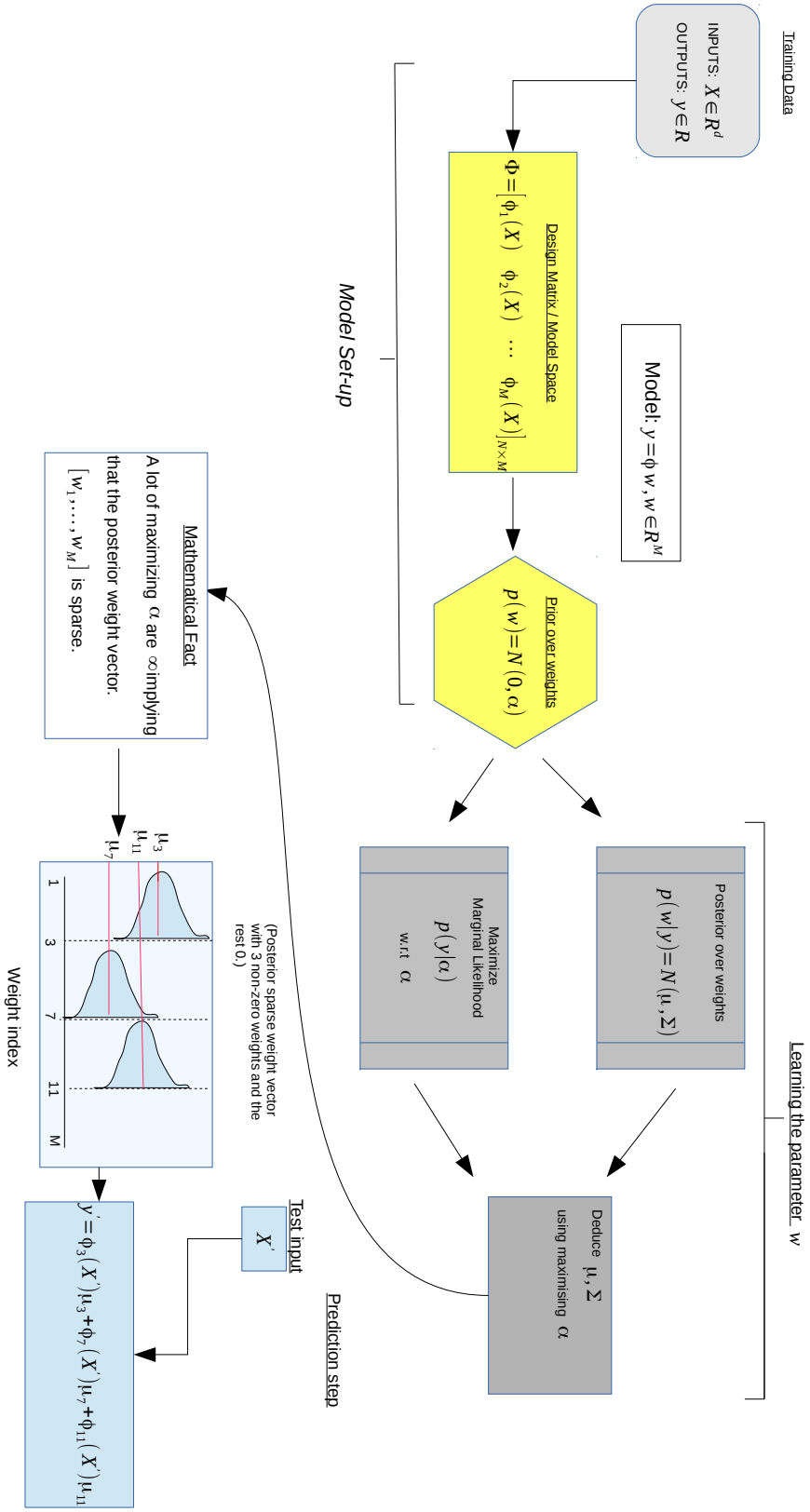


Figure 8.5: Sparse Bayesian Workflow. The yellow boxes define the model set-up steps. The design matrix is chosen and fixed in advance. In the proposed software we want to replace the yellow box for design matrix with meta-learning for kernel construction.

to manipulate and combine kernels. A meta-learning algorithm will need to be developed with two modes of running (see fig. 8.6).

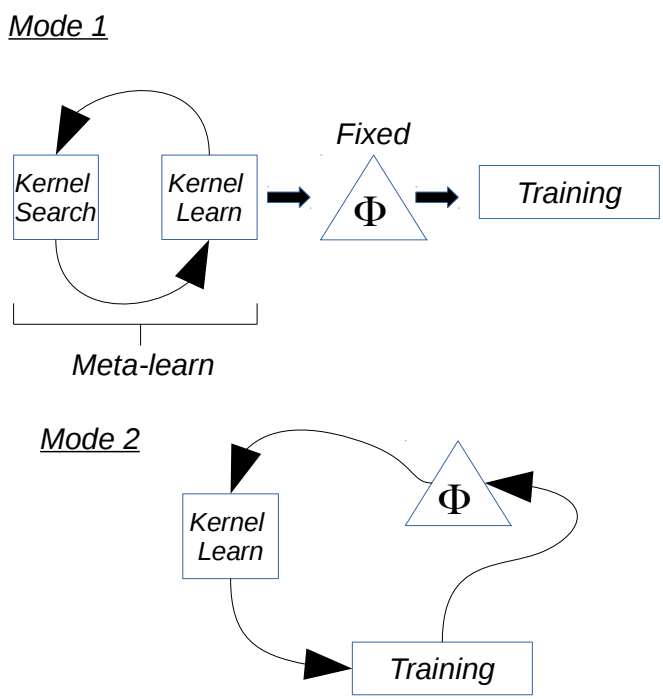


Figure 8.6: Two approaches for Automated Kernel Construction

Ultimately, this software should also support a range of experiments:

1. Testing on data with heteroscedastic noise.
2. Extrapolating outside the training range of the data.
3. High-dimensional data.

3 Parallel Tracks

(The ideas in these sections bear interesting and promising links to the core research problem, however, they will need further research and firming up over the course of my PhD.)

3.1 Extension to other supervised learning paradigms

Kernel selection is a fundamental problem of kernel based learning algorithms. A methodology aimed at capturing structure from data by detecting suitable kernel functions is readily applicable to several supervised machine learning systems where it is incumbent on the user to specify a choice of kernel function with values for hyperparameters.

1. Non-Parametric Linear Regression: A methodology to select good kernel functions for the design matrix can be used as a pre-processing step for Bayesian Linear Regression.

2. Gaussian Processes: A Gaussian process (GP) model is initilised by specifying a covariance function. Since all kernel functions are positive definite they can be interpreted as covariance functions. Different kernel functions capture particular types of similarity in the data. Hence, the covariance function (in a GP context) is really a kernel function that acts on pairs of data points and controls the shape, smoothness, periodicity and stationarity aspects of the function we are trying to model. A method to detect suitable kernels can be readily deployed to select the right covariance function for GP modelling. [ADWW17] provides an empirical metrics based technique for automatic kernel selection.

3. Support Vector Machines: The effectiveness of the SVM depends on the selection of the kernel and its parameters. These choices are predominantly heuristic and left as user-defined inputs in most software packages that provide an SVM implementation, hence a systematic kernel construction method that can be integrated into the SVM training process could prove useful and novel. [ASM06], [DL17] are some papers which explore this idea.

3.2 Non-Parametric Density Estimation in HEP

One of the fundamental inference problems in high energy physics is density estimation where the input data are typically one-dimensional (for instance, mass measurements of a particle). The aim is to find a univariate probability density $p(x)$ in a non-parametric way.

There is a host of research on kernel density estimation in the realm of high energy physics, [Cra01], [And15], [NP13] and even more which focus on the problem of hyperparameter optimization (like width selection for Gaussian kernels) once a parametric kernel form for the density is guessed. Further, the meta-inference problem of selecting the hyperparameters like width settles on a single width leading to a mono-width kernel density estimate. There is no reason that kernels within densities cannot have multiple widths, this leads to a more flexible and powerful class of methods called *'Variable/Adaptive Kernel density estimation'*. In principle, kernel density estimation techniques can be embedded with a kernel design and learning procedure which suggests a composite kernel and then optimizes its hyperparameters. The key question is if the performance gains

of such learning tools merits the added complexity. This can only be answered after an in-depth study of the problem.

4 Timeplan

The timeplan below is an overview of the more detailed workplan in the gantt chart below.

Time Schedule	Research Tasks
Jan - Sep 2017	1. Background Study. 2. Software Testing. 3. Initial Experiments. 4. Consolidation for Annual Report.
Oct - Dec 2017	1. In-depth Literature Review. 2. Current Software Improvements. 3. Start installing a framework for Kernel class.
Jan - June 2018	1. In-depth literature review. 2. Kernel Design methods and Code development. 3. Kernel Learning methods and Code development.
July - Dec 2018	1. Working out the math (where necessary). 2. Bayesian Model selection methods. 3. Algorithm testing on synthetic data, one and high-dimensional.
Jan - June 2019	1. Computational improvements and further testing. 2. Scope out the potential for kernel learning within density estimation. 3. Comparison with similar techniques in literature.
July - Dec 2019	1. Consolidation of Results. 2. Commence thesis writing (preliminary chapters).
Jan - June 2020	1. Consolidation of all parallel tracks of research. 2. Thesis Writing (advanced chapters).

Table 8.1: Overview of Workplan

Workplan

Pipeline	Details	2017				2018											2019				2020	
		Jan-Mar	Mar-Jun	Jul-Sep	Oct-Dec	Jan	Feb	Mar	April	May	June	July	Aug	Sept	Oct	Nov	Dec	Jan-Mar	Apr-Jun	Jul-Sep	Oct-Dec	Jan-Mar	Apr-Jun
		Q1	Q2	Q3	Q4	Q1			Q2			Q3			Q4			Q1	Q2	Q3	Q4	Q1	Q2
0 Background Study																							
	0.1 Bayesian Inference: The Basics.																						
	0.2 Bayesian Inference for parameters in Regression.																						
	0.3 Experiment/Code examples of Bayesian Linear Regression.																						
	0.4 Sparse Bayesian Learning (SBL) (Paper by Tipping).																						
	0.5 Derivations of SBL Inference procedure (first principles).																						
	0.6 Code/Experiment examples of SBL.																						
	0.7 Python software for visualizing a design matrix.																						
	0.8 Consolidate content for Annual Report.																						
1 Software Development																							
	1.1 Detect alignment between basis functions.																						
	1.2 Faster convergence by looking at sparsity/quality factors.																						
	1.3 Fixing confidence bounds for SBL predictions.																						
	1.4 Kernel class to support hand-crafting of kernels.																						
	1.4 Methods to support combining kernels.																						
	1.5 Training with automatic kernel selection.																						
	1.6 Training with automatic hyper-parameter optimization.																						
	1.7 Class to support computation of metrics to evaluate different kernel designs.																						
	1.8 Computational improvements & Code documentation																						
2 Research & Experimentation																							
	2.1 Review related literature on automatic kernel construction.																						
	2.2 Research on data structure representation using different kernels.																						
	2.3 Explore viability of grid-search techniques.																						
	2.4 Explore analytical techniques for hyperparameter optimization.																						
	2.5 Mathematical proofs where necessary to demonstrate effectiveness of proposed meta-algorithm.																						
	2.6 Research on Bayesian model selection methodologies.																						
3 Parallel Tracks																							
	3.1 Discussions with Chris on getting hands on experimental HEP data.																						
	3.2 Test meta-learning for density estimation in HEP.																						
	3.3 Gaussian Process kernel selection.																						
	3.4 SVM Kernel Selection.																						
	3.5 NP Regression Kernel Selection.																						
5 Thesis Writing																							
	4.1 Background Chapters																						
	4.2 Derivations and Proofs (of Bayesian inference)																						
	4.3 Literature Review																						
	4.4 Proposed methodology																						
	4.5 Mathematics of proposed methods																						
	4.6 Kernel Learning for Density Estimation in HEP																						
	4.7 Consolidation & Results																						

Page 1

Appendix A

The Gaussian Density

The Gaussian density [1] plays an important role in Bayesian learning and as such in probability theory and all of statistics on the whole. Its ubiquity can be attributed to the central limit theorem[2] which enables methods that work for normal distributions to be applied to data coming from non-normal distributions. It is the de-facto distribution choice when we know very little about the underlying distribution from a sample of data. It is fully determined by its first two moments, the mean and variance. The Gaussian density is attractive due to a bunch of closure properties in relation to sums, products, convolutions, marginals and conditionals.

1 Univariate Case

The probability density function (pdf) of a normally distributed random variable $x \sim N(\mu, \sigma^2)$ with mean μ and variance σ^2 is given by:

$$f(x|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \tag{A.1}$$

The normal pdf is characterized by an exponentiated quadratic[3] term which gives it the bell shape around the mean. Further, it is symmetric, unimodal and infinitely differentiable. Its mean, median and mode all coincide.

It is important to note that density of a normal variable decays to (almost) zero within 3 standard deviations from the mean. Hence, it might not be a good model for data with

[1] The word density here indicates a probability density function

[2] The central limit theorem states that the mean of samples drawn independently from any distribution converge in distribution to the normal distribution.

[3] In a lot of literature this term is loosely but wrongly called 'squared exponential'. Neil Lawrence says that this should instead be called, 'exponentiated quadractic'

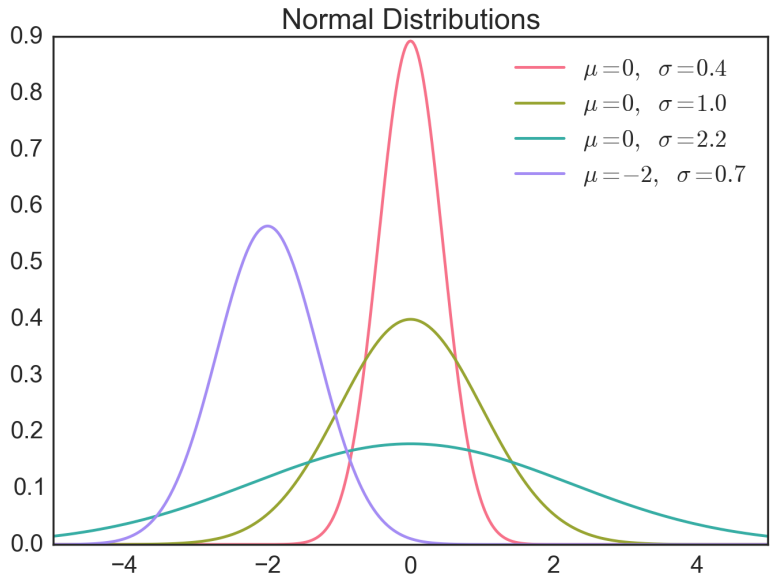


Figure A.1: The univariate Gaussian density for different values of μ and σ. The standard deviation σ controls the spread of the distribution around the mean.

a high fraction of outliers, in this case more heavy tailed distributions like the Student-t might be more suitable.

It is sometimes convenient, especially in Bayesian learning to use the *precision* instead of variance as the parameter controlling the width of the Gaussian. The precision γ is defined as the reciprocal of the variance, $\frac{1}{\sigma^2}$. The pdf then becomes,

$$f(x|\mu,\gamma) = \sqrt{\frac{\gamma}{2\pi}} exp\left(-\frac{\gamma(x-\mu)^2}{2}\right) \tag{A.2}$$

Intuitively, the precision tells us how concentrated the values are around the mean. Hence, a large precision implies a more peaked distribution (low variance) and vice-versa.

1.1 Properties

1. The sum of two **independent** Gaussian random variables (univariate) $X \sim N(\mu_x, \sigma_x^2)$ and $Y \sim N(\mu_y, \sigma_y^2)$, where $Z = X + Y$ is normally distributed with $Z \sim N(\mu_x + \mu_y, \sigma_x^2 + \sigma_y^2)$. In the case X and Y are not independent then their variances are not additive due to the correlation, the variance is then given by $\sigma_z^2 = \sigma_x^2 + \sigma_y^2 + 2\rho\sigma_x\sigma_y$ where ρ is the correlation coefficient.

   By extension, the sum of K mutually independent random variables $(X_1, \ldots, X_k)$

with means $\mu_1, \ldots, \mu_K$ and variances $\sigma_1^2, \ldots, \sigma_K^2$ is a random variable ($W = \sum_{i=1}^{K} X_i$) with $W \sim N\left(\sum_{i=1}^{K} \mu_i, \sum_{i=1}^{K} \sigma_i^2\right)$.

The proof in the multivariate case is enclosed in Appendix B sections 1 and 2.

2. An affine transformation of a Gaussian random variable $X \sim N(\mu, \sigma^2)$ is Gaussian. If $W = aX + b$,

$$W \sim N(a\mu + b, a^2\sigma^2) \tag{A.3}$$

as $E(W) = aE(X) + E(b) = a\mu + b$ and $Var(W) = a^2 Var(X) + Var(b) = a^2\sigma^2$.

3. Following on from this, any linear combination of independent Gaussian random variables $X_1, \ldots, X_k$ with means $\mu_1, \ldots, \mu_k$ and variances $\sigma_1^2, \ldots, \sigma_k^2$ where the resulting random variable has the form,

$$Z = \sum_{i=1}^{k} a_i X_i \tag{A.4}$$

where $a_1, \ldots, a_k$ are arbitrary scalar constants is given by,

$$Z \sim N(\sum_{i=1}^{k} a_i \mu_i, \sum_{i=1}^{k} a_i^2 \sigma_i^2) \tag{A.5}$$

Property 2 and 3 above use the linearity of expectation, the property that the variance of a constant is 0 and variance of a scaled variable is the square of the constant times the variance of the variable.

4. The product of two Gaussian random variables X and Y is not Gaussian however, the product of the pdfs of two Gaussian random variables is proportional to a Gaussian pdf (or is equal to a scaled Gaussian pdf). If $f_x \sim N(\mu_x, \sigma_x^2)$ and $f_y \sim N(\mu_y, \sigma_y^2)$ denote the pdfs of X and Y then,

$$f_x f_y = \frac{S_{xy}}{\sqrt{2\pi}\sigma_{xy}} exp\left(-\frac{(x - \mu_{xy})^2}{2\sigma_{xy}^2}\right) \tag{A.6}$$

where S_{xy} is a scaling factor and,

$$\begin{aligned}\mu_{xy} &= \frac{\mu_x\sigma_y^2 + \mu_y\sigma_x^2}{\sigma_x^2 + \sigma_y^2} = \frac{\mu_x}{\sigma_x^2} + \frac{\mu_y}{\sigma_y^2} \\ \sigma_{xy}^2 &= \frac{\sigma_x^2\sigma_y^2}{\sigma_x^2 + \sigma_y^2} = \left(\frac{1}{\sigma_x^2} + \frac{1}{\sigma_y^2}\right)^{-1}\end{aligned} \tag{A.7}$$

Appendix B, section 2.1 and 2.2 gives the necessary proof in the multivariate case. It mainly involves expanding the sum of quadratics in the exponent, collecting the terms and completing the square. The scaling factor S_{xy} is itself a Gaussian function (not pdf as there is no rv involved) involving μ_x, μ_y and variance $\sigma_x^2 + \sigma_y^2$ giving,

$$S_{xy} = \frac{1}{\sqrt{2\pi(\sigma_x^2 + \sigma_y^2)}} \exp\left(\frac{(\mu_x - \mu_y)^2}{2(\sigma_x^2 + \sigma_y^2)}\right) \tag{A.8}$$

5. The convolution of two univariate Gaussian densities,

$$\begin{aligned}f(x) &= \frac{1}{\sqrt{2\pi\sigma_f^2}} \exp\left(-\frac{(x-\mu_f)^2}{2\sigma_f^2}\right) \\ g(x) &= \frac{1}{\sqrt{2\pi\sigma_g^2}} \exp\left(-\frac{(x-\mu_g)^2}{2\sigma_g^2}\right)\end{aligned} \tag{A.9}$$

   defined as $(f*g)(z) = \int\limits_{-\infty}^{\infty} f(x)g(z-x)dx$ is a Gaussian density with mean $\mu_f + \mu_g$ and variance $\sigma_f^2 + \sigma_g^2$.

6. If $X \sim N(\mu, \sigma^2)$ then $Z = \dfrac{X-\mu}{\sigma}$ has a standard normal distribution, $Z \sim N(0,1)$.

The density for a single random variable as depicted in eq. A.1 is called the univariate Gaussian density. The multivariate Gaussian density is the generalisation of the univariate density to higher dimensions.

2 Multivariate Case

A random vector $\boldsymbol{X} \in \mathbb{R}^n$ is normally distributed if any linear combination of its components (random variables) is univariate Gaussian .i.e $a^T\boldsymbol{X} = \sum_{i=1}^n a_i X_i$ is Gaussian $\forall a \in \mathbb{R}^n$.

In terms of notation, $\boldsymbol{X} \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ denotes that the n-dimensional random vector $\boldsymbol{X}$ has a multivariate Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$.

The mean vector $\boldsymbol{\mu}$ is $[E(X_1), \ldots, E(X_n)]^T$ and the $n \times n$ covariance matrix $\boldsymbol{\Sigma}$ is $E[(\boldsymbol{X} - \boldsymbol{\Sigma})(\boldsymbol{X} - \boldsymbol{\Sigma})^T] = E[(\boldsymbol{X}\boldsymbol{X}^{\boldsymbol{T}})] - \boldsymbol{\mu}\boldsymbol{\mu}^T$.

The pdf of a multivariate Gaussian (MVN) is,

$$f(\boldsymbol{X}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{\sqrt{(2\pi)^n|\boldsymbol{\Sigma}|}} \exp\left(-\frac{1}{2}(\boldsymbol{X} - \boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-\mathbf{1}}(\boldsymbol{X} - \boldsymbol{\mu})\right) \tag{A.10}$$

where $|\boldsymbol{\Sigma}|$ is the determinant of the covariance matrix.

The pdf makes intuitive sense as when $n = 1$ we get the univariate Gaussian pdf because $\boldsymbol{\Sigma}$ for one variable is just a 1×1 matrix with the variance σ^2.

The factor in the exponent is a quadratic form in the vector variable $\boldsymbol{X}$. The covariance matrix is always positive definite implying that the inverse is positive definite, hence, for any vector $\boldsymbol{X} \neq \boldsymbol{\mu}$

$$\begin{aligned} (\boldsymbol{X} - \boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-\mathbf{1}}(\boldsymbol{X} - \boldsymbol{\mu}) &> 0 \\ -\frac{1}{2}(\boldsymbol{X} - \boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-\mathbf{1}}(\boldsymbol{X} - \boldsymbol{\mu}) &< 0 \end{aligned} \tag{A.11}$$

This gives the density the shape of a downward opening quadratic bowl. The normalization factor ensures that,

$$\frac{1}{\sqrt{(2\pi)^n|\boldsymbol{\Sigma}|}} \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\ldots\int_{-\infty}^{\infty} \exp\left(-\frac{1}{2}(\boldsymbol{X} - \boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{X} - \boldsymbol{\mu})\right) dX_1 dX_2 \ldots dX_n = 1 \tag{A.12}$$

where $X_1 \ldots X_n$ are the components of the vector $\boldsymbol{X}$.

In the non-degenerate case, the covariance matrix $\boldsymbol{\Sigma}$ has full rank. When $\boldsymbol{\Sigma}$ is not full rank ($\boldsymbol{\Sigma}^{-1}$ does not exist) then the MVN is degenerate and does not have a density.

It is important to note that a set of univariate normals $X_1, \ldots X_n$ does not ensure that they are jointly normal .i.e. assembling them together in a random vector $\boldsymbol{X} = [X_1, \ldots, X_n]$ does not make $\boldsymbol{X}$ multivariate normal. However, if a set of Gaussian random variables is mutually independent then the random vector formed by them is multivariate normal. Such a construction would yield multivariate normals with diagonal covariance matrix (with zeroes everywhere except the diagonal). The independence of the components in a MVN is an example case of a multivariate normal random vector. It is an important one as it highlights the relationship with the univariate normal. Multivariate normal random vectors typically have non-zero off-diagonal entries.

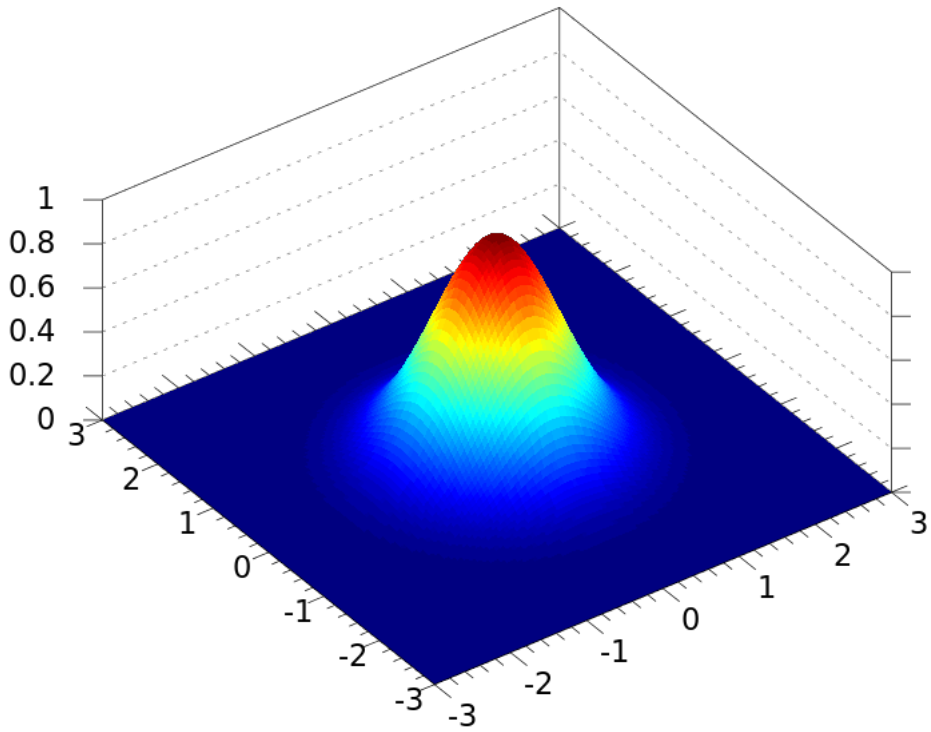


Figure A.2: A bivariate Gaussian density

A set of random variables $X_1, \ldots, X_n$ are mutually independent with $X_i \in N(\mu_i, \sigma_i^2)$ if and only if $\boldsymbol{X} = (X_1, \ldots, X_n) \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ where $\boldsymbol{\Sigma}$ is a diagonal matrix with entries σ_i^2 on the diagonal. Another way to say this is that if $\boldsymbol{X}$ is multivariate Gaussian then, X_i, X_j are independent if and only if the $Cov(X_i, X_j) = 0$. This characterization of independence only applies to Gaussian random variables.

To summarize, marginal normality and mutual independence (diagonal covariance matrix) yield multivariate normality, the inverse is not true.

The joint density of a set of independent random variables is a product of their densities.

$$f(X_1, \ldots, X_n) = f(X_1), \ldots, f(X_n) \tag{A.13}$$

It is easy to prove this, we do it below for the two variable case:

$$\begin{aligned} &\Rightarrow f(X_1, X_2) = f(X_1|X_2)f(X_2) \\ &\Rightarrow f(X_1|X_2) = f(X_1) \text{ (independent events)} \\ &\Rightarrow f(X_1, X_2) = f(X_1)f(X_2) \end{aligned} \tag{A.14}$$

The top line in fig. A.3 shows bi-variate Gaussian densities of a 2-dimensional variable

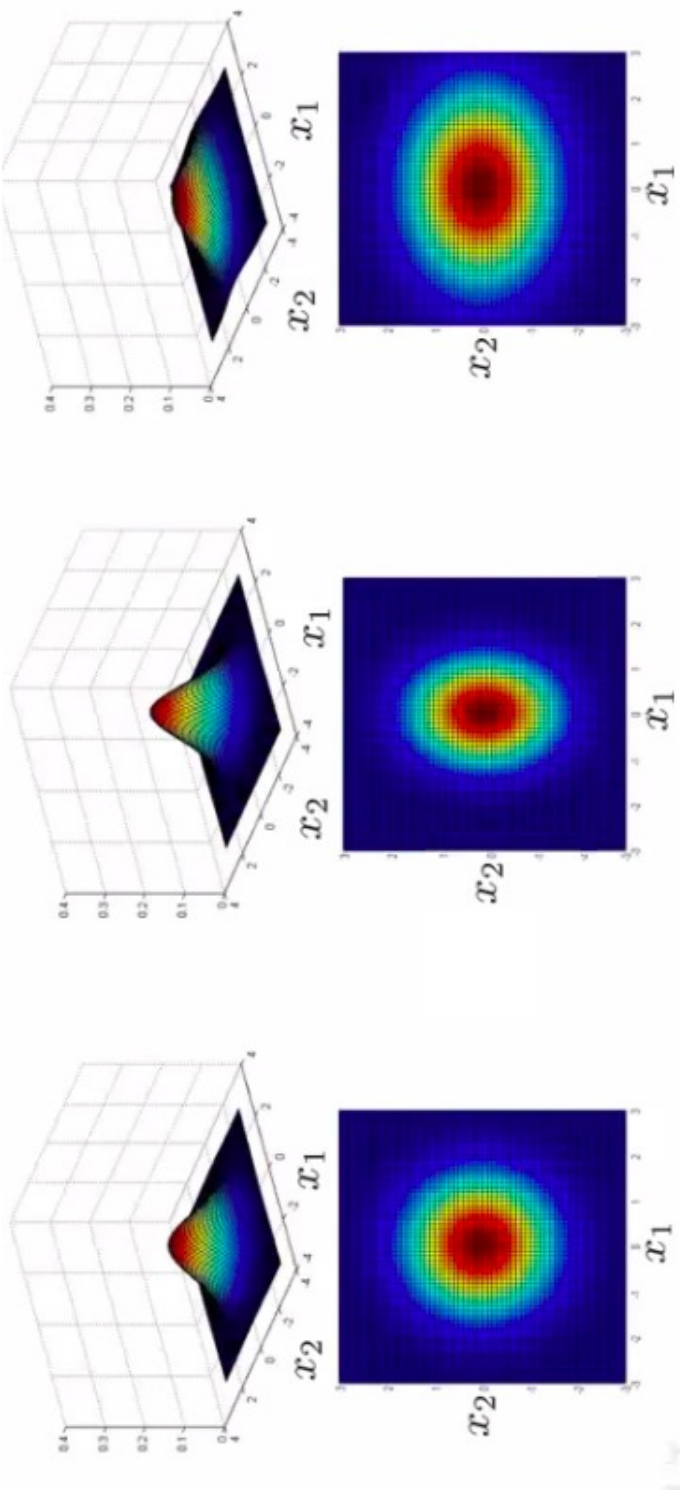


Figure A.3: The figures show the bi-variate Gaussian densities for 3 different diagonal covariance matrices Σ_1, Σ_2, Σ_3 respectively.

$\boldsymbol{x} = (x_1, x_2)$. The bottom line shows the corresponding contour graph. One thing to note from the contour graph is that the ellipses are axes-aligned, this implies that the $Cov(x_1, x_2)$ is 0.The covariance matrix (from left to right) for the three figures on the plot is given by,

$$\Sigma_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{A.15}$$

$$\Sigma_2 = \begin{bmatrix} 0.6 & 0 \\ 0 & 0.8 \end{bmatrix} \tag{A.16}$$

$$\Sigma_3 = \begin{bmatrix} 4 & 0 \\ 0 & 2 \end{bmatrix} \tag{A.17}$$

The effect of different variances on the diagonal entries is reflected in the dispersion of the ellipses.

Further, when the covariance between x_1 and x_2 is non-zero, we get ellipses which are rotated (the center is fixed, as it is governed by the mean) to reflect this. In fig. A.4, the covariance matrices are given by,

$$\Sigma_1' = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{A.18}$$

$$\Sigma_2' = \begin{bmatrix} 1 & -0.5 \\ -0.5 & 1 \end{bmatrix} \tag{A.19}$$

$$\Sigma_3' = \begin{bmatrix} 1 & -0.8 \\ -0.8 & 1 \end{bmatrix} \tag{A.20}$$

When the covariance is negative, we get rotated ellipses that show the negative relationship between the two variables. The magnitude of the covariance governs the diagonal spread of the ellipse. A covariance of -0.8 gives ellipses which are much narrower (stronger relationship) than the ones for -0.5.

The rotation angle of the ellipse is also deduced from the covariance matrix. It is the angle between the x-axis and the first eigenvector (corresponding to the largest eigenvalue) of the covariance matrix.

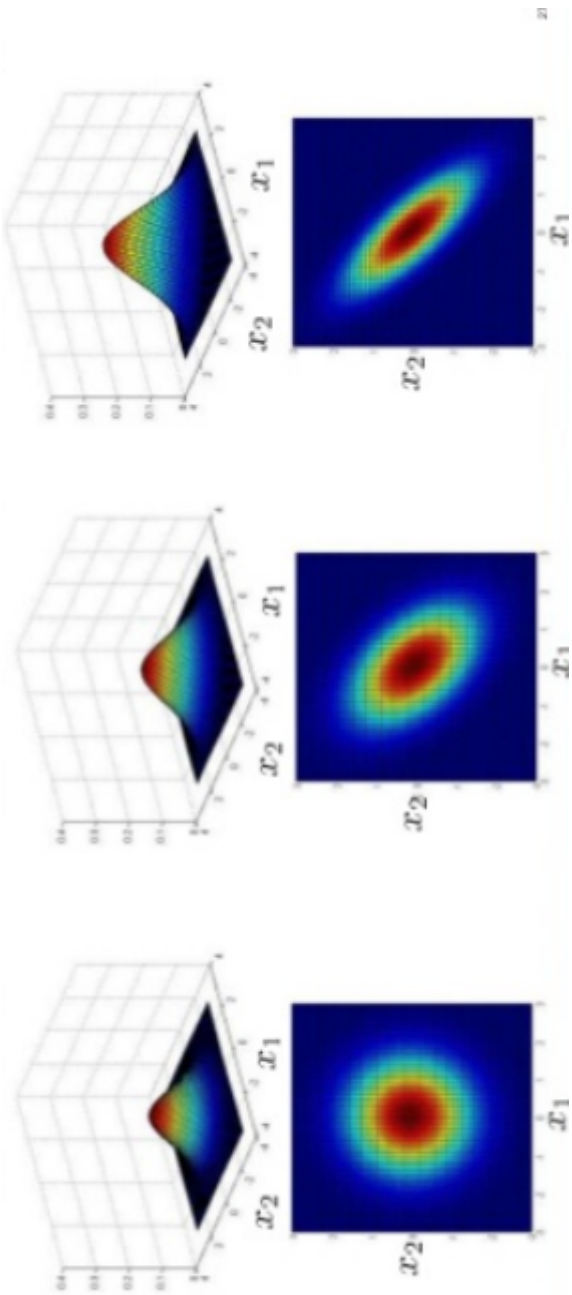


Figure A.4: The figures show the bi-variate Gaussian densities for 3 different covariance matrices Σ_1', Σ_2', Σ_3' respectively.

2.1 Properties

1. The sum of two independent Gaussian random vectors $\boldsymbol{X} \sim N(\mu_x, \Sigma_x)$ and $\boldsymbol{Y} \sim N(\mu_y, \Sigma_y)$, where $\boldsymbol{X}, \boldsymbol{Y}, \mu_x, \mu_y \in \mathbb{R}^n$ and Σ_x, Σ_y are $n \times n$ matrices, the random vector $\boldsymbol{Z} = \boldsymbol{X} + \boldsymbol{Y}$ is multivariate Gaussian with density $\boldsymbol{Z} \sim N(\mu_x + \mu_y, \Sigma_x + \Sigma_y)$.

   As in the univariate case, the proof of this entails showing that the convolution of the densities gives the density of the resulting random vector $\boldsymbol{Z}$. This is shown in Appendix B sections 1 and 2.

   By extension, the sum of K mutually independent random vectors $(\boldsymbol{X_1}, \ldots, \boldsymbol{X_k})$ with mean vectors $\mu_1, \ldots, \mu_K$ and covariance matrices $\Sigma_1, \ldots, \Sigma_K$ is a random vector $(\boldsymbol{W} = \sum_{i=1}^{K} \boldsymbol{X}_i)$ with $\boldsymbol{W} \sim N\left(\sum_{i=1}^{K} \mu_i, \sum_{i=1}^{K} \Sigma_i\right)$.

2. An affine transformation of a Gaussian random vector $\boldsymbol{X} \sim N(\mu, \Sigma)$ is Gaussian distributed. If $\boldsymbol{W} = A\boldsymbol{X} + B$, where A is a full-rank matrix and B is a random vector.

$$\boldsymbol{W} \sim N(A\mu + B, A\Sigma A^T) \tag{A.21}$$

3. Following on from this, any linear combination of independent Gaussian random vectors $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_k$ with means $\mu_1, \ldots, \mu_k$ and covariance matrices $\Sigma_1, \ldots, \Sigma_k$ where the resulting random vector has the form,

$$\boldsymbol{Z} = \sum_{i=1}^{k} A_i X_i \tag{A.22}$$

   where $A_1, \ldots, A_k$ are full-rank matrices of compatible size is given by,

$$\boldsymbol{Z} \sim N\left(\sum_{i=1}^{k} A_i \mu_i, \sum_{i=1}^{k} A_i \Sigma_i A_i^T\right) \tag{A.23}$$

4. The product of two multivariate Gaussian random variables is not multivariate Gaussian distributed. However, the product of the densities of two multivariate gaussian variables is a scaled normal density.

$$N(\mu_1, \Sigma_1) N(\mu_2, \Sigma_2) = S N(\mu_3, \Sigma_3) \tag{A.24}$$

   where,

$$\begin{aligned}
\Sigma_3 &= (\Sigma_1^{-1} + \Sigma_2^{-1})^{-1} \\
\mu_3 &= \Sigma_3(\Sigma_1^{-1}\mu_1 + \Sigma_2^{-1}\mu_2) \\
S &= \frac{1}{\sqrt{(2\pi)^n|\Sigma_1 + \Sigma_2|}} \exp\left(-\frac{1}{2}(\mu_1 - \mu_2)^T(\Sigma_1 + \Sigma_2)^{-1}(\mu_1 - \mu_2)\right)
\end{aligned} \tag{A.25}$$

(proof in Appendix B.)

5. If $\boldsymbol{X} \sim N(\mu, \Sigma)$ is multivariate normal, then $\boldsymbol{Z} = M\boldsymbol{X}$ where M is a *whitening* matrix with the property that $M^T M = \Sigma^{-1}$ gives the random vector $\boldsymbol{Z}$ with unit diagonal variance.

2.2 Covariance Matrix

The covariance between two random variables is the expected value of the product of their deviations from their means (individual expected values).

$$Cov[X, Y] = E[(X - E(X))(Y - E(Y))] \tag{A.26}$$

By using the linearity of the expectation operation, it simplifies to

$$E[XY] - E[X]E[Y] \tag{A.27}$$

When X and Y are random n-vectors, there are $n \times n$ covariance terms that can be calculated $Cov(X_i, Y_j)$ and $i, j \in 1, \ldots, n$. The terms assembled in a $n \times n$ matrix is the covariance matrix.

The covariance operation is commutative, $Cov(X_i, Y_j) = Cov(Y_j, X_i)$, this makes the covariance matrix symmetric. Further, $Cov(X, X) = Var(X)$, this can be seen from the definition of the covariance.

Let Σ denote the covariance matrix $Cov(X, Y)$

$$\Sigma = \begin{bmatrix} Cov(X_1, X_1) & Cov(X_1, X_2) & \ldots & Cov(X_1, X_n) \\ Cov(X_2, X_1) & Cov(X_2, X_2) & \ldots & Cov(X_1, X_n) \\ \vdots & \vdots & \vdots & \vdots \\ Cov(X_n, X_1) & Cov(X_n, X_2) & \ldots & Cov(X_n, X_n) \end{bmatrix} \tag{A.28}$$

The covariance matrix is vital to understanding the multivariate Gaussian distributions as the entries of covariance matrix Σ capture the covariance between the different components of X. The covariance matrix is often called the *variance-covariance* matrix since the diagonal terms are in fact variances.

The covariance matrix Σ is typically a positive semi-definite[4] matrix with full rank (the rank is the dimension of X). If the covariance matrix is singular, the corresponding distribution has no density. The inverse of the covariance matrix Σ^{-1} is the precision matrix. The diagonal terms of the precision matrix are just the reciprocal of the variance, the ith diagonal term in Σ^{-1} is just $\frac{1}{Var(X_i)}$.

A covariance matrix can be graphed using a grid plot where each cell (ij) represents the covariance value of the components $Cov(X_i, X_j)$

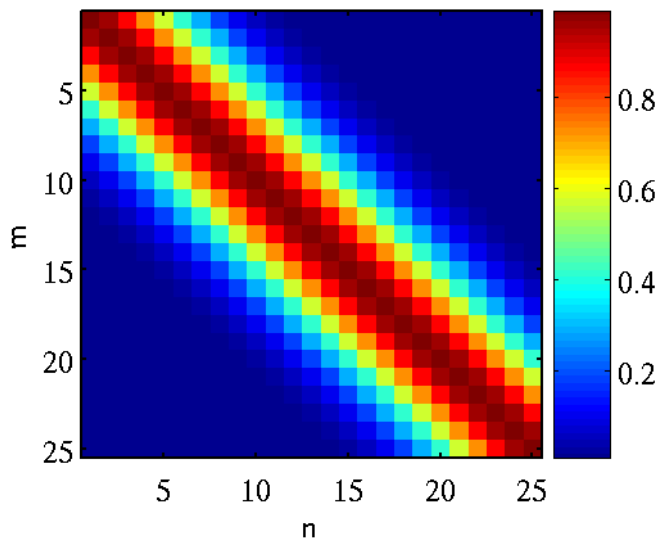


Figure A.5: Covariance matrix of a 25-dimensional Gaussian random vector

It is important to distinguish between the concepts of covariance of two random variables which is an attribute or a property of the joint distribution of the random variables and the *sample covariance* which serves as an estimate and is computed using the data.

If we have N observations of two variables X and Y with sample means, $\bar{X}$ and $\bar{Y}$. The sample covariance is given by,

$$\frac{1}{N-1}\sum_{i=1}^{N}(X_i - \bar{X})(Y_i - \bar{Y}) \tag{A.29}$$

In parameter estimation we usually do not know the mean and the covariance of the random variables, the sample mean and sample covariance serve as unbiased estimates of the population mean and covariance. If the variables are Gaussian, the estimates of the mean and the covariance characterize the complete distribution/density.

[4]A matrix M is said to be positive semi-definite if $a^T M a \geq 0 \forall$ non-zero column vectors a.

2.3 Marginalisation and Conditioning

Marginalisation

Let X_1 and X_2 be multivariate normal random vectors which also are jointly normal .i.e. $X = [X_1, X_2]$ has a multivariate normal distribution with pdf,

$$p\left(X_1, X_2\right) = \frac{1}{\sqrt{(2\pi)^{m+n}|\Sigma|}} \exp\left(-\frac{1}{2}\begin{bmatrix} X_1 - \mu_1 \\ X_2 - \mu_2 \end{bmatrix}^T \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix}^{-1} \begin{bmatrix} X_1 - \mu_1 \\ X_2 - \mu_2 \end{bmatrix}\right) \tag{A.30}$$

where m and n are just the dimensions of X_1 and X_2.

The marginal distribution is just obtained by integrating out the other variable.

$$\begin{aligned} p(X_1) &= \int p(X_1, X_2) dX_2 \\ p(X_2) &= \int p(X_1, X_2) dX_1 \end{aligned} \tag{A.31}$$

(This is the continuous analog of the sum rule of probability)

The marginal distributions of X_1 and X_2 are just,

$$\begin{aligned} X_1 &\sim N(\mu_1, \Sigma_{11}) \\ X_2 &\sim N(\mu_2, \Sigma_{22}) \end{aligned} \tag{A.32}$$

Before we embark on the conditional Gaussian distribution we need to acquaint with an important property of matrices called *block diagonalization*.

Partitioning Matrices and the Schur complement

Consider a partitioned matrix M,

$$M = \begin{bmatrix} E & F \\ G & H \end{bmatrix} \tag{A.33}$$

where we assume that both E and H are invertible.

To diagonalize M we need to target F and G. To make F vanish we need to pre-multiply with row blocks I and $-FH^{-1}$. Similarly for G we need to post-multiply with column blocks I and $-GH^{-1}$.

If we plug the required blocks above into a pre- and post- matrix leaving the other entries to be identity we get,

$$\begin{bmatrix} I & -FH^{-1} \\ 0 & I \end{bmatrix} \begin{bmatrix} E & F \\ G & H \end{bmatrix} \begin{bmatrix} I & 0 \\ -H^{-1}G & I \end{bmatrix} = \begin{bmatrix} E - FH^{-1}G & 0 \\ 0 & H \end{bmatrix} \tag{A.34}$$

The *Schur* complement of the matrix M with respect to H is defined as, M/H is the term $E - FH^{-1}G$.

Now take the inverse on both sides, notice that the expression is of the form $W = XYZ$ where taking an inverse yields, $W^{-1} = Z^{-1}Y^{-1}X^{-1} \Rightarrow ZW^{-1}X = Y^{-1}$.

$$\begin{aligned} \begin{bmatrix} E & F \\ G & H \end{bmatrix}^{-1} &= \begin{bmatrix} I & 0 \\ -H^{-1}G & I \end{bmatrix} \begin{bmatrix} (M/H)^{-1} & 0 \\ 0 & H^{-1} \end{bmatrix} \begin{bmatrix} I & -FH^{-1} \\ 0 & I \end{bmatrix} \\ &= \begin{bmatrix} (M/H)^{-1} & -(M/H)^{-1}FH^{-1} \\ -H^{-1}G(M/H)^{-1} & H^{-1} + H^{-1}G(M/H)^{-1}FH^{-1} \end{bmatrix} \end{aligned} \tag{A.35}$$

The main takeaway from the last equation is that we express the inverse of a block partitioned matrix in terms of the inverses of its blocks.

Also, the determinant operation can be applied to eq. A.34.

$$|M| = |M/H||H| \tag{A.36}$$

Conditioning

We are interested in the conditional distribution $p(X_1|X_2)$ ($p(X_2|X_1)$ can be obtained by symmetry arguments). The joint distribution of $X = [X_1, X_2]$ has a covariance matrix Σ which can be partitioned, $\begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix}$

$$\Sigma^{-1} = \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix}^{-1} = \begin{bmatrix} (\Sigma/\Sigma_{22})^{-1} & -(\Sigma/\Sigma_{22})^{-1}\Sigma_{12}\Sigma_{22}^{-1} \\ -\Sigma_{22}^{-1}\Sigma_{21}(\Sigma/\Sigma_{22})^{-1} & \Sigma_{22}^{-1} + \Sigma_{22}^{-1}\Sigma_{21}(\Sigma/\Sigma_{22})^{-1}\Sigma_{12}\Sigma_{22}^{-1} \end{bmatrix} \tag{A.37}$$

The quadratic form in the exponent of the joint distribution of X can be re-written as,

$$= \begin{bmatrix} X_1 - \mu_1 \\ X_2 - \mu_2 \end{bmatrix}^T \begin{bmatrix} (\Sigma/\Sigma_{22})^{-1} & -(\Sigma/\Sigma_{22})^{-1}\Sigma_{12}\Sigma_{22}^{-1} \\ -\Sigma_{22}^{-1}\Sigma_{21}(\Sigma/\Sigma_{22})^{-1} & \Sigma_{22}^{-1} + \Sigma_{22}^{-1}\Sigma_{21}(\Sigma/\Sigma_{22})^{-1}\Sigma_{12}\Sigma_{22}^{-1} \end{bmatrix} \begin{bmatrix} X_1 - \mu_1 \\ X_2 - \mu_2 \end{bmatrix}$$

$$= \begin{bmatrix} (X_1 - \mu_1)^T(\Sigma/\Sigma_{22})^{-1} - (X_2 - \mu_2)^T\Sigma_{22}^{-1}\Sigma_{21}(\Sigma/\Sigma_{22})^{-1} \\ -(X_1 - \mu_1)^T(\Sigma/\Sigma_{22})^{-1}\Sigma_{12}\Sigma_{22}^{-1} + (X_2 - \mu_2)^T\Sigma_{22}^{-1} + (X_2 - \mu_2)^T\Sigma_{22}^{-1}\Sigma_{21}(\Sigma/\Sigma_{22})^{-1}\Sigma_{12}\Sigma_{22}^{-1} \end{bmatrix}^T \times \begin{bmatrix} X_1 - \mu_1 \\ X_2 - \mu_2 \end{bmatrix}$$

$$\begin{aligned} &= (X_1 - \mu_1)^T(\Sigma/\Sigma_{22})^{-1}(X_1 - \mu_1) \\ &- (X_2 - \mu_2)^T\Sigma_{22}^{-1}\Sigma_{21}(\Sigma/\Sigma_{22})^{-1}(X_1 - \mu_1) \\ &- (X_1 - \mu_1)^T(\Sigma/\Sigma_{22})^{-1}\Sigma_{12}\Sigma_{22}^{-1}(X_2 - \mu_2) + (X_2 - \mu_2)^T\Sigma_{22}^{-1}(X_2 - \mu_2) \\ &+ (X_2 - \mu_2)^T\Sigma_{22}^{-1}\Sigma_{21}(\Sigma/\Sigma_{22})^{-1}\Sigma_{12}\Sigma_{22}^{-1}(X_2 - \mu_2) \end{aligned}$$

$$\begin{aligned} &= (X_1 - \underbrace{\mu_1 - \Sigma_{12}\Sigma_{22}^{-1}(X_2 - \mu_2)}_{\mu_3})^T(\Sigma/\Sigma_{22})^{-1}(X_1 - \mu_1 - \Sigma_{12}\Sigma_{22}^{-1}(X_2 - \mu_2)) \\ &+ (X_2 - \mu_2)^T\Sigma_{22}^{-1}(X_2 - \mu_2) \end{aligned} \tag{A.38}$$

Also,

$$|\Sigma| = |\Sigma/\Sigma_{22}||\Sigma_{22}| \tag{A.39}$$

Using all of the above, we express $p(X_1, X_2)$ as a product of two terms,

$$\begin{aligned} p(X_1, X_2) = \frac{1}{\sqrt{2\pi|\Sigma/\Sigma_{22}|}} &\exp(-\frac{1}{2}(X_1 - \mu_3)^T(\Sigma/\Sigma_{22})^{-1}(X_1 - \mu_3)) \\ &\times \frac{1}{\sqrt{2\pi|\Sigma_{22}|}} \exp(-\frac{1}{2}(X_2 - \mu_2)^T\Sigma_{22}^{-1}(X_2 - \mu_2)) \end{aligned} \tag{A.40}$$

Given that,

$$p(X_1, X_2) = p(X_1|X_2)p(X_2) \tag{A.41}$$

It must be that,

$$p(X_1|X_2) = \frac{1}{\sqrt{2\pi|\Sigma/\Sigma_{22}|}} \exp(-\frac{1}{2}(X_1 - \mu_3)^T(\Sigma/\Sigma_{22})^{-1}(X_1 - \mu_3)) \tag{A.42}$$

Hence, we have

$$X_1|X_2 \sim N(\mu_3, (\Sigma/\Sigma_{22})) \tag{A.43}$$

where,

$$\begin{aligned} \mu_3 &= \mu_1 - \Sigma_{12}\Sigma_{22}^{-1}(X_2 - \mu_2) \\ (\Sigma/\Sigma_{22}) &= \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21} \end{aligned} \tag{A.44}$$

2.4 Sampling from a MVN

In practicality, most programming environments implement a univariate and multivariate Gaussian sampler however, it is instructive to see how the multivariate samples are generated from the univariate ones.

Algorithm 2 Sampling from a Multivariate Gaussian Distribution

Goal: To generate samples from a multivariate Gaussian $X \sim N(\mu, \Sigma)$ where X has dimension d.

1: Compute the Cholesky decomposition of $\Sigma = LL^T$ where L is a lower triangular matrix and L^T is upper triangular.

2: Sample d times from a standard normal distribution to generate samples $s \sim N(0, 1)$ to form random vector $\mathbf{s}$.

3: Compute $\mathbf{x} = \mu + L\mathbf{s}$.

4: $\mathbf{x}$ has the desired distribution as, $E(\mathbf{x}) = \mu$ and $Var(\mathbf{x}) = LL^T$ due to the affine transformation property of multivariate Gaussian variables.

Appendix B

Proofs involving the Gaussian density

1 Sum of two independent random variables

The density of the sum of two independent random variables is a convolution of their densities. This means if X and Y are two random variables with densities $f(x)$ and $g(x)$ then, the random variable $Z = X + Y$ has the density,

$$(f * g)(z) = \int_{-\infty}^{\infty} f(x)g(z-x)dx \tag{B.1}$$

In order to provide intuition for this, we appeal to the discrete case where let X and Y be two independent discrete random variables with distributions (probability mass function or pmf) $m_1(x)$ and $m_2(x)$. We want to compute the pmf for $Z = X + Y$. The pmf of a discrete random variable gives the probability of Z taking on any specific value say s. For each such assignment $Z = s$, X and Y take on discrete values such that, if $X = k$ (where k is an arbitrary integer) then $Z = s$ if and only if $Y = s - k$. So the event $Z = s$ is the union of disjoint events $P(X = k)$ and $P(Y = s - k)$[GS12],

$$P(Z = s) = \sum_{k=-\infty}^{\infty} P(X = k) \times P(Y = s - k) \tag{B.2}$$

Eq. B.2 motivates the following definition in the discrete case.

Given discrete random variables X and Y with probability mass functions $m_1(x)$ and $m_2(x)$, the convolution gives a third probability mass function $m_3(x)$,

$$m_3(x) = \sum_{k} m_1(k) \times m_2(x-k) \tag{B.3}$$

where $m_3(x)$ is the probability mass function of the random variable Z.

We can see that eq. B.3 is just the discrete analog of the convolution function in eq. B.1. This implies that the convolution of two probability densities say, $p_1(x)$ and $p_2(x)$ of two continuous variables X and Y gives a third density $p_3(x)$ of the random variable $X + Y$.

$$p_3(z) = \int p_1(x)p_2(z - x)dx \tag{B.4}$$

In sections 2.2 and 2.3 we will consider two multivariate normal random variables $X \sim N(\mu_a, A)$ and $Y \sim N(\mu_b, B)$ with densities $f(x)$ and $g(x)$ and show that the result of the convolution is actually a probability density of the form,

$$Z \sim N(\mu_a + \mu_b, A + B) \tag{B.5}$$

2 Convolution of functions

The convolution of two functions $f(x)$ and $h(x)$ denoted as $g(x) = (f * h)(x)$ is a mathematical operation that entails taking the integral of a point wise multiplication operation between $f(x)$, and the function $h(x)$ as it is translated and shifted across $f(x)$. Mathematically,

$$(f * h)(x) = \int_{-\infty}^{\infty} f(s)h(x - s)ds \tag{B.6}$$

where s is just the dummy variable of integration.

Intuitively, it can be thought about as a measure of overlap (not area[1]) between $f(x)$ and the translated version of function $h(x) \to h(x - s)$ as it slides across $f(x)$. The result of the convolution of two one-dimensional functions is another one-dimensional function.

[1]The intuition with the area of overlap is only valid when f(x) is in interval [0,1]

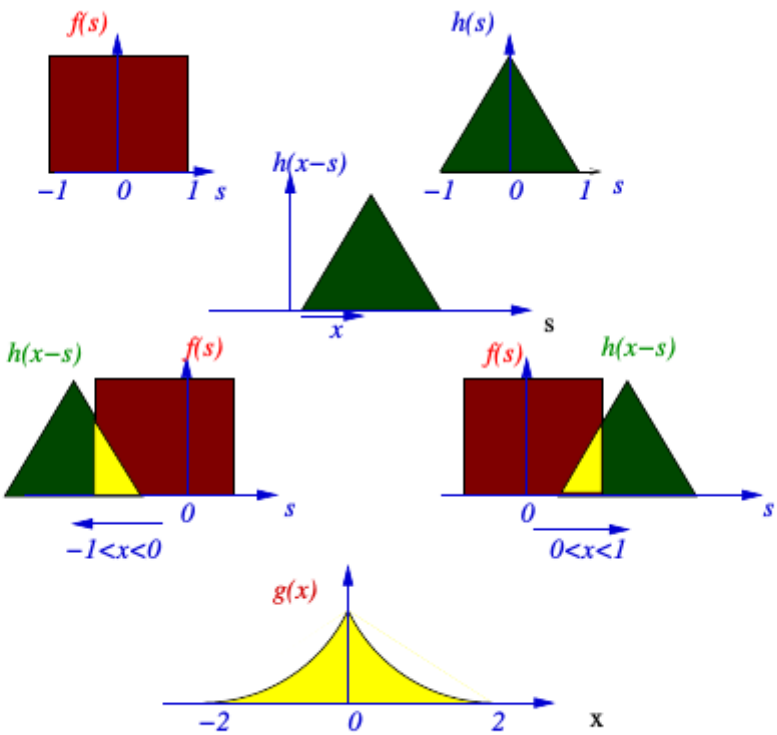


Figure B.1: Convolution of two one-dimensional functions $g(x) = (f * h)(x) = \int_{-\infty}^{\infty} f(s)h(x-s)ds$

2.1 Quadratic Form Factorization

If x, μ_a and μ_b are vectors of dimension m, A and B are symmetric matrices of order m, additionally, their sum $(A+B)$ is non-singular, then we consider the quadratic form we encounter in the exponent of normal densities.

$$\begin{aligned} &\Rightarrow (x-\mu_a)^T A^{-1}(x-\mu_a) + (x-\mu_b)^T B^{-1}(x-\mu_b) \\ &\Rightarrow x^T(A^{-1}+B^{-1})x - 2x^T(A^{-1}\mu_a + B^{-1}\mu_b) + \mu_a^T A^{-1}\mu_a + \mu_b^T B^{-1}\mu_b \end{aligned} \tag{B.7}$$

The last equation is in the matrix quadratic form,

$$x^T M x + x^T b + c \tag{B.8}$$

where,

$$\begin{aligned} M &= A^{-1} + B^{-1} \text{ (coefficient of } x) \\ b &= -2(A^{-1}\mu_a + B^{-1}\mu_b) \text{ (coefficient of } x^T) \\ c &= \mu_a^T A^{-1}\mu_a + \mu_b^T B^{-1}\mu_b \text{constant} \end{aligned} \tag{B.9}$$

The matrix quadratic form can be re-written as the sum of a square in x and a constant.

$$x^T M x + x^T b + c = (x-h)^T M (x-h) + k \tag{B.10}$$

where,

$$\begin{aligned} h &= -\frac{1}{2} M^{-1} b \\ k &= c - \frac{1}{4} b^T M^{-1} b \end{aligned} \tag{B.11}$$

Now we re-write eq. B.7 in the sum of squares form,

First, we derive h and k.

$$\begin{aligned} h &= (A^{-1} + B^{-1})^{-1}(A^{-1}\mu_a + B^{-1}\mu_b) \\ k &= \mu_a^T A^{-1}\mu_a + \mu_b^T B^{-1}\mu_b - (A^{-1}\mu_a + B^{-1}\mu_b)^T (A^{-1} + B^{-1})^{-1}(A^{-1}\mu_a + B^{-1}\mu_b) \end{aligned} \tag{B.12}$$

Next we simplify k further.

Let $k = k1 - k2$ where,

$$\begin{aligned} k1 &= \mu_a^T A^{-1}\mu_a + \mu_b^T B^{-1}\mu_b \\ k2 &= (A^{-1}\mu_a + B^{-1}\mu_b)^T (A^{-1} + B^{-1})^{-1}(A^{-1}\mu_a + B^{-1}\mu_b) \end{aligned} \tag{B.13}$$

Simplifying $k2$ first, we expand the product to get 4 separate terms, for ease of writing we substitute $\Sigma_c = (A^{-1} + B^{-1})^{-1}$

$$k2 = \underbrace{\mu_a^T A^{-1}\Sigma_c A^{-1}\mu_a}_{k_{21}} + \underbrace{\mu_a^T A^{-1}\Sigma_c B^{-1}\mu_b}_{k_{22}} + \underbrace{\mu_b^T B^{-1}\Sigma_c A^{-1}\mu_a}_{k_{23}} + \underbrace{\mu_b^T B^{-1}\Sigma_c B^{-1}\mu_b}_{k_{24}} \tag{B.14}$$

Since Σ_c, A, B are all symmetric matrices k_{22} and k_{23} are the same (as one is a transpose of the other), we can group them together,

$$k2 = \underbrace{\mu_a^T A^{-1}\Sigma_c A^{-1}\mu_a}_{k_{21}} + \underbrace{2\mu_a^T A^{-1}\Sigma_c B^{-1}\mu_b}_{k_{22}} + \underbrace{\mu_b^T B^{-1}\Sigma_c B^{-1}\mu_b}_{k_{23}} \tag{B.15}$$

We zone in to the Σ_c term and use the matrix identity $(A^{-1}+B^{-1})^{-1} = A(A+B)^{-1}B = B(A+B)^{-1}A$ to simplify each of the terms in $k2$

$$\begin{aligned} k_{21} &= \mu_a^T A^{-1}\Sigma_c A^{-1}\mu_a = \mu_a^T A^{-1}A(A+B)^{-1}BA^{-1}\mu_a \\ k_{22} &= 2\mu_a^T A^{-1}\Sigma_c B^{-1}\mu_b = 2\mu_a^T A^{-1}A(A+B)^{-1}BB^{-1}\mu_b \\ k_{23} &= \mu_b^T B^{-1}\Sigma_c B^{-1}\mu_b = \mu_b^T B^{-1}B(A+B)^{-1}AB^{-1}\mu_b \end{aligned} \tag{B.16}$$

A lot of the inverses cancel giving,

$$\begin{aligned} k_{21} &= \mu_a^T (A+B)^{-1}BA^{-1}\mu_a \\ k_{22} &= 2\mu_a^T (A+B)^{-1}\mu_b \\ k_{23} &= \mu_b^T (A+B)^{-1}AB^{-1}\mu_b \end{aligned} \tag{B.17}$$

$$k = k1 - (k_{21} + k_{22} + k_{23}) \tag{B.18}$$

$$\begin{aligned} k = \mu_a^T A^{-1}\mu_a + \mu_b^T B^{-1}\mu_b \\ - \mu_a^T (A+B)^{-1}BA^{-1}\mu_a \\ - 2\mu_a^T (A+B)^{-1}\mu_b \\ - \mu_b^T (A+B)^{-1}AB^{-1}\mu_b \end{aligned} \tag{B.19}$$

Grouping terms together,

$$\begin{aligned} k = \mu_a^T (I - (A+B)^{-1}B)A^{-1}\mu_a \\ - 2\mu_a^T (A+B)^{-1}\mu_b \\ + \mu_b^T (I - (A+B)^{-1}A)B^{-1}\mu_b \end{aligned} \tag{B.20}$$

where I is the identity matrix.

However,

$$\begin{aligned}(I-(A+B)^{-1}B)A^{-1} &= (A+B)^{-1}(A+B-B)A^{-1} = (A+B)^{-1} \\ (I-(A+B)^{-1}A)B^{-1} &= (A+B)^{-1}(A+B-A)B^{-1} = (A+B)^{-1}\end{aligned} \tag{B.21}$$

Hence, the final equation for k is,

$$\begin{aligned}k &= \mu_a^T(A+B)^{-1}\mu_a - 2\mu_a^T(A+B)^{-1}\mu_b + \mu_b^T(A+B)^{-1}\mu_b \\ &= (\mu_a-\mu_b)^T(A+B)^{-1}(\mu_a-\mu_b)\end{aligned} \tag{B.22}$$

Recall that h is,

$$h = (A^{-1}+B^{-1})^{-1}(A^{-1}\mu_a + B^{-1}\mu_b) \tag{B.23}$$

Substituting h and k to express the quadratic form in eq. B.7 as a sum of a square and a constant term.

$$(x-h)^T(A^{-1}+B^{-1})^{-1}(x-h) + (\mu_a-\mu_b)^T(A+B)^{-1}(\mu_a-\mu_b) \tag{B.24}$$

Letting,

$$\begin{aligned}\Sigma_c &= (A^{-1}+B^{-1})^{-1} \\ \mu_c &= h = \Sigma_c(A^{-1}\mu_a + B^{-1}\mu_b) \\ C &= (A+B)^{-1}\end{aligned} \tag{B.25}$$

The final form yields,

$$(x-\mu_c)^T\Sigma_c(x-\mu_c) + (\mu_a-\mu_b)^TC(\mu_a-\mu_b) \tag{B.26}$$

2.2 Integral simplification

This section computes the integral $\int f(x)g(x)dx$ where $f(x) \sim N(\mu_a, A)$ and $g(x) \sim N(\mu_b, B)$ are multivariate Gaussian densities, x is p-dimensional.

Writing it out, we have.

$$
\begin{aligned}
&= \int f(x)g(x)dx \\
&= \int \frac{1}{\sqrt{(2\pi)^p|A|}} \exp\left(-\frac{1}{2}(x-\mu_a)^T A^{-1}(x-\mu_a)\right) \frac{1}{\sqrt{(2\pi)^p|B|}} \exp\left(-\frac{1}{2}(x-\mu_b)^T B^{-1}(x-\mu_b)\right) dx \\
&= \frac{1}{\sqrt{(2\pi)^p|A|}} \frac{1}{\sqrt{(2\pi)^p|B|}} \int \exp\left(-\frac{1}{2}(x-\mu_a)^T A^{-1}(x-\mu_a) + (x-\mu_b)^T B^{-1}(x-\mu_b)\right) dx \\
&= \frac{1}{\sqrt{(2\pi)^p|A||B|}} \frac{1}{\sqrt{(2\pi)^p}} \int \exp\left(-\frac{1}{2}(x-\mu_c)^T (A^{-1}+B^{-1})(x-\mu_c) + (\mu_a-\mu_b)^T C(\mu_a-\mu_b))\right) dx \\
&\text{Using result from 2.1; multiply and divide by } \sqrt{|(A^{-1}+B^{-1})^{-1}|} \\
&= \frac{\sqrt{|(A^{-1}+B^{-1})^{-1}|}}{\sqrt{(2\pi)^p|A||B|}} \exp\left(-\frac{1}{2}(\mu_a-\mu_b)^T C(\mu_a-\mu_b)\right) \times \\
&\qquad \int \frac{1}{\sqrt{(2\pi)^p|(A^{-1}+B^{-1})^{-1}|}} \exp\left(-\frac{1}{2}(x-\mu_c)^T (A^{-1}+B^{-1})(x-\mu_c)\right) dx \\
&= \frac{\sqrt{|(A^{-1}+B^{-1})^{-1}|}}{\sqrt{(2\pi)^p|A||B|}} \exp\left(-\frac{1}{2}(\mu_a-\mu_b)^T C(\mu_a-\mu_b)\right) \\
&= \frac{1}{\sqrt{(2\pi)^p|A||B|A^{-1}+B^{-1}|}} \exp\left(-\frac{1}{2}(\mu_a-\mu_b)^T C(\mu_a-\mu_b)\right) \\
&= \frac{1}{\sqrt{(2\pi)^p|ABA^{-1}+A|}} \exp\left(-\frac{1}{2}(\mu_a-\mu_b)^T (A+B)^{-1}(\mu_a-\mu_b)\right) \\
&= \frac{1}{\sqrt{(2\pi)^p|A(B+A)A^{-1}|}} \exp\left(-\frac{1}{2}(\mu_a-\mu_b)^T (A+B)^{-1}(\mu_a-\mu_b)\right) \\
&= \frac{1}{\sqrt{(2\pi)^p|(B+A)|}} \exp\left(-\frac{1}{2}(\mu_a-\mu_b)^T (A+B)^{-1}(\mu_a-\mu_b)\right)
\end{aligned}
\tag{B.27}
$$

It is worth noting that since A and B are symmetric positive definite matrices, so is their sum $(A+B)$.

$$
x^T(A+B)x = x^T Ax + x^T Bx > 0 \tag{B.28}
$$

2.3 Convolution Result

$$
\begin{aligned}
f * g &= \int f(x)g(z-x)dx \\
&= \int \frac{1}{\sqrt{(2\pi)^p|A|}} \exp\left(-\frac{1}{2}(x-\mu_a)^T A^{-1}(x-\mu_a)\right) \times \\
&\qquad \frac{1}{\sqrt{(2\pi)^p|B|}} \exp\left(-\frac{1}{2}(z-x-\mu_b)^T B^{-1}(z-x-\mu_b)\right) dx \\
&\text{Using the result from 2.2} \\
&= \frac{1}{\sqrt{(2\pi)^p|A+B|}} \exp\left(-\frac{1}{2}((z-(\mu_a+\mu_b))^T (A+B)^{-1}(z-(\mu_a+\mu_b))\right) \\
&\sim N(\mu_a+\mu_b, A+B)
\end{aligned}
\tag{B.29}
$$

3 Matrix identities

Let matrices A, B and C be $n \times n$, non-singular matrices unless specified otherwise. A_{ij} refers to the element in the $(ij)^{th}$ position in the matrix A and $|A|$ denotes the determinant.

3.1 Inverse

1. $A^{-1} = \frac{1}{|A|} \times adjugate(A)$
2. $(AB)^{-1} = B^{-1}A^{-1}$
3. $(ABC)^{-1} = C^{-1}B^{-1}A^{-1}$
4. $(A^T)^{-1} = (A^{-1})^T$
5. $(A^{-1}+B^{-1})^{-1} = A(A+B)^{-1}B = B(A+B)^{-1}A = A - A(A+B)^{-1}A$
6. $(A+UCV)^{-1} = A^{-1} - A^{-1}U(C^{-1}+VA^{-1}U)^{-1}VA^{-1}$ where matrices U and V are of compatible sizes is the Woodbury matrix identity. This identity is also called the *matrix inversion lemma*.
7. $(A+uv^T)^{-1} = A^{-1} - \frac{A^{-1}uv^TA^{-1}}{1+v^TA^{-1}u}$ where u and v are n-vectors and $1+v^TA^{-1}u \neq 0$ is the *Sherman-Morrison formula*.
8. $(A+B)^{-1} = A^{-1} - A^{-1}(A^{-1}+B^{-1})^{-1}A^{-1}$ (application of the Woodbury identity.)

3.2 Trace

1. $Tr(A) = \sum_i A_{ii} = \sum_i \lambda_i, \lambda_i = eigenvalue(A)$
2. $Tr(A + B) = Tr(A) + Tr(B)$

3.3 Determinants

1. $|A| = \prod_i \lambda_i, \lambda_i = eigenvalue(A)$
2. $|cA| = c^n A$
3. $|A^T| = |A|$
4. $|AB| = |A||B|$
5. $|BAB^{-1}| = |A|$
6. $|A^n| = |A|^n$
7. $|A+uv^T| = (1+v^T A^{-1} u)|A|$ where u and v are n-vectors is the *matrix determinant lemma*.

3.4 Expectations

Let $\boldsymbol{x}$ be a random n-vector with mean μ, $Var[\bullet]$ and $Cov[\bullet]$ are shorthand for variance and covariance.

1. $E(A\boldsymbol{x} + B) = AE(\boldsymbol{x}) + B = A\mu + B$
2. $Var[A\boldsymbol{x}] = AVar[\boldsymbol{x}]A^T$
3. $Cov[A\boldsymbol{x}, B\boldsymbol{y}] = ACov[\boldsymbol{x}, \boldsymbol{y}]B$ where $\boldsymbol{y}$ is a generic random n-vector.

3.5 Derivatives

1. Jacobi's formula expresses the derivative of a determinant of a matrix in terms of its inverse and determinant,

$$\frac{\partial |A(t)|}{\partial t} = |A(t)| \times Tr\left(A(t)^{-1}\frac{\partial A(t)}{\partial t}\right)$$

2. $\dfrac{\partial |A|}{\partial A} = |A|(A^{-1})^T$

3. $\dfrac{\partial log|A|}{\partial t} = Tr(A^{-1}\dfrac{\partial A}{\partial t})$

4. $\dfrac{\partial A(t)^{-1}}{\partial t} = -A(t)^{-1}\dfrac{\partial dA(t)}{\partial t}A(t)^{-1}$

5. If $\boldsymbol{x}$ and $\boldsymbol{a}$ are n-vectors then,

$$\frac{\partial \boldsymbol{x}^T\boldsymbol{a}}{\partial \boldsymbol{x}} = \boldsymbol{a}$$

6. If $\boldsymbol{y} = A\boldsymbol{x}$ where $\boldsymbol{x}$ and $\boldsymbol{y}$ are n-vectors then,

$$\frac{\partial \boldsymbol{y}}{\partial \boldsymbol{x}} = A$$

7. If $\boldsymbol{z} = \boldsymbol{y}^T A\boldsymbol{x}$ where $\boldsymbol{x}$, $\boldsymbol{y}$ and $\boldsymbol{z}$ are n-vectors then,

$$\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{x}} = \boldsymbol{y}^T A$$

8. If $\boldsymbol{z} = \boldsymbol{x}^T A\boldsymbol{x}$ (quadratic form) where $\boldsymbol{x}$ and $\boldsymbol{z}$ are n-vectors then,

$$\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{x}} = \boldsymbol{x}^T(A + A^T)$$

9. If further $\boldsymbol{x}$ depends on some vector $\boldsymbol{k}$ then,

$$\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{x}} = \boldsymbol{x}^T(A + A^T)\frac{\partial \boldsymbol{x}}{\partial \boldsymbol{k}}$$

10. If α is a scalar parameter then,

$$\frac{\partial A}{\partial \alpha} = \begin{bmatrix} \frac{\partial A_{11}}{\partial \alpha} & \frac{A_{12}}{\partial \alpha} & \cdots & \frac{A_{1n}}{\partial \alpha} \\ \frac{A_{21}}{\partial \alpha} & \frac{A_{22}}{\partial \alpha} & \cdots & \frac{A_{2n}}{\partial \alpha} \\ \vdots & \vdots & & \vdots \\ \frac{A_{n1}}{\partial \alpha} & \frac{A_{n2}}{\partial \alpha} & \cdots & \frac{A_{nn}}{\partial \alpha} \end{bmatrix}_{n\times n}$$

11. If $\boldsymbol{k}$ is a m-vector then,

$$\frac{\partial A}{\partial \boldsymbol{k}} = \begin{bmatrix} \frac{\partial A_{11}}{\partial \boldsymbol{k}} & \frac{A_{12}}{\partial \boldsymbol{k}} & \cdots & \frac{A_{1n}}{\partial \boldsymbol{k}} \\ \frac{A_{21}}{\partial \boldsymbol{k}} & \frac{A_{22}}{\partial \boldsymbol{k}} & \cdots & \frac{A_{2n}}{\partial \boldsymbol{k}} \\ \vdots & \vdots & & \vdots \\ \frac{A_{n1}}{\partial \boldsymbol{k}} & \frac{A_{n2}}{\partial \boldsymbol{k}} & \cdots & \frac{A_{nn}}{\partial \boldsymbol{k}} \end{bmatrix}_{n \times n}$$

where each element $\frac{\partial A_{ij}}{\partial \boldsymbol{k}}$ is a scalar by vector differentiation, which results in a m-vector $[\frac{\partial A_{ij}}{\partial k_1}, \ldots, \frac{\partial A_{ij}}{\partial k_m}]$

Much of the information here is extracted directly from [PP+08]. For an exhaustive list of matrix identities the reader is directed to this worthy reference. The ones which have been stated in this section are the identities which were used and referred to in the derivations in previous sections of this report.

Appendix C

Derivations in Sparse Bayesian Learning

1 Deriving the Marginal Likelihood from first principles

In this section we derive the marginal likelihood term $p(\mathbf{t}|\boldsymbol{\alpha}, \sigma^2) = \int p(\mathbf{t}|\mathbf{w}, \sigma^2)p(\mathbf{w}|\boldsymbol{\alpha})d\mathbf{w}$ from first principles by expanding the individual densities and performing the integration.

$$\begin{aligned}\int p(\mathbf{t}|\mathbf{w}, \sigma^2)p(\mathbf{w}|\boldsymbol{\alpha}) &= \int \left(2\pi\sigma^2\right)^{-N/2} \exp\left\{\frac{-1}{2\sigma^2}||\mathbf{t} - \Phi\mathbf{w}||^2\right\} (2\pi)^{-M/2} \prod_{i=1}^{M} \alpha_i^{1/2} \exp\left\{\frac{-1}{2}\mathbf{w}^T\mathbf{A}\mathbf{w}\right\} d\mathbf{w} \\ &= \left(2\pi\sigma^2\right)^{-N/2} (2\pi)^{-M/2} \prod_{i=1}^{M} \alpha_i^{1/2} \int \exp - \left\{\frac{1}{2\sigma^2}||\mathbf{t} - \Phi\mathbf{w}||^2 + \frac{1}{2}\mathbf{w}^T\mathbf{A}\mathbf{w}\right\} d\mathbf{w}\end{aligned} \tag{C.1}$$

The simplification entails handling the term in the exponent:

$$E(\mathbf{w}) = \frac{1}{2\sigma^2}||\mathbf{t} - \Phi\mathbf{w}||^2 + \frac{1}{2}\mathbf{w}^T\mathbf{A}\mathbf{w} \tag{C.2}$$

Expanding $E(\mathbf{w})$ gives,

$$
\begin{aligned}
E(\mathbf{w}) &= \frac{1}{2\sigma^2}(\mathbf{t}^T\mathbf{t} - 2\mathbf{w}^T\Phi^T\mathbf{t} + \mathbf{w}^T\Phi^T\Phi\mathbf{w}) + \frac{1}{2}\mathbf{w}^T\mathbf{A}\mathbf{w} \\
&= \frac{1}{2\sigma^2}\left(\mathbf{t}^T\mathbf{t} - 2\mathbf{w}^T\Phi^T\mathbf{t} + \mathbf{w}^T\Phi^T\Phi\mathbf{w} + \mathbf{w}^T\mathbf{A}\mathbf{w}\right) \\
&= \frac{1}{2}\left\{\sigma^{-2}\mathbf{t}^T\mathbf{t} - 2\sigma^{-2}\mathbf{w}^T\Phi^T\mathbf{t} + \mathbf{w}^T(\mathbf{A} + \sigma^{-2}\Phi^T\Phi)\mathbf{w}\right\}
\end{aligned} \tag{C.3}
$$

Using the formula for completion of square in the matrix case depicted below,

$$
x^TAx + x^Tb + c = (x-h)^TA(x-h) + k \tag{C.4}
$$

where $h = -\frac{1}{2}A^{-1}b$, $k = c - \frac{1}{4}b^TA^{-1}b$ and A is symmetric $\&$ invertible.

we can simplify eq. C.3 further.

Assigning,

$$
\begin{aligned}
x &= \mathbf{w} \\
A &= (\mathbf{A} + \sigma^{-2}\Phi^T\Phi) \\
b &= -2\sigma^{-2}\Phi^T\mathbf{t} \\
c &= \sigma^{-2}\mathbf{t}^T\mathbf{t} \\
h &= (\mathbf{A} + \sigma^{-2}\Phi^T\Phi)^{-1}\sigma^{-2}\Phi^T\mathbf{t} \\
k &= \sigma^{-2}\mathbf{t}^T\mathbf{t} - (\sigma^{-2}\Phi^T\mathbf{t})^T(\mathbf{A} + \sigma^{-2}\Phi^T\Phi)^{-1}(\sigma^{-2}\Phi^T\mathbf{t})
\end{aligned} \tag{C.5}
$$

and substituting the term $(\mathbf{A} + \sigma^{-2}\Phi^T\Phi)^{-1}$ as Σ and $\sigma^{-2}\Sigma\Phi^T\mathbf{t}$ as $\boldsymbol{\mu}$ we re-write eq. C.3 as,

$$
\begin{aligned}
E(\mathbf{w}) &= \frac{1}{2}\left\{(\mathbf{w}-\boldsymbol{\mu})^T\Sigma^{-1}(\mathbf{w}-\boldsymbol{\mu}) + \sigma^{-2}\mathbf{t}^T\mathbf{t} - (\sigma^{-2}\mathbf{t}^T\Phi^T\Sigma^T\Sigma^{-1}\Sigma\sigma^{-2}\Phi^T\mathbf{t})\right\} \\
&= \frac{1}{2}\left\{(\mathbf{w}-\boldsymbol{\mu})^T\Sigma^{-1}(\mathbf{w}-\boldsymbol{\mu}) + \sigma^{-2}\mathbf{t}^T\mathbf{t} - \boldsymbol{\mu}^T\Sigma^{-1}\boldsymbol{\mu}\right\} \\
&= \frac{1}{2}(\mathbf{w}-\boldsymbol{\mu})^T\Sigma^{-1}(\mathbf{w}-\boldsymbol{\mu}) + E(\mathbf{t})
\end{aligned} \tag{C.6}
$$

where $E(\mathbf{t}) = \frac{1}{2}(\sigma^{-2}\mathbf{t}^T\mathbf{t} - \boldsymbol{\mu}^T\Sigma^{-1}\boldsymbol{\mu})$

The integral from eq. C.1 simplifies to

$$\begin{aligned}
&\Rightarrow \left(2\pi\sigma^2\right)^{-N/2} (2\pi)^{-M/2} \prod_{i=1}^{M} \alpha_i^{1/2} \exp(E(\mathbf{t})) \int \exp - \left\{\frac{1}{2}(\mathbf{w}-\boldsymbol{\mu})^T \Sigma^{-1} (\mathbf{w}-\boldsymbol{\mu})\right\} d\mathbf{w} \\
&\Rightarrow \left(2\pi\sigma^2\right)^{-N/2} |\Sigma|^{1/2} \prod_{i=1}^{M} \alpha_i^{1/2} \exp(-E(\mathbf{t}))
\end{aligned} \tag{C.7}$$

where we multiply and divide by $|\Sigma|$ to create a pdf in $\mathbf{w}$.

$$\frac{1}{(2\pi)^{M/2}|\Sigma|^{1/2}} \int \exp\left\{-\frac{1}{2}(\mathbf{w}-\boldsymbol{\mu})^T \Sigma^{-1} (\mathbf{w}-\boldsymbol{\mu})\right\} d\mathbf{w} = 1$$

Thus the final form of the marginal likelihood is given by,

$$\left(2\pi\sigma^2\right)^{-N/2} |\Sigma|^{1/2} \prod_{i=1}^{M} \alpha_i^{1/2} \exp(-E(\mathbf{t})) \tag{C.8}$$

where,

$$\boldsymbol{\mu} = \sigma^{-2}\Sigma\Phi^T\mathbf{t} \tag{C.9}$$

$$\Sigma = (\mathbf{A} + \sigma^{-2}\Phi^T\Phi)^{-1} \tag{C.10}$$

1.1 Differentiating w.r.t α

It is often conducive while differentiating to work with the log of the marginal likelihood.

$$\ln p(\mathbf{t}|\boldsymbol{\alpha}, \sigma^2) = -\frac{N}{2}\ln(\sigma^2) - \frac{N}{2}\ln(2\pi) + \frac{1}{2}\ln|\Sigma| - E(\mathbf{t}) + \frac{1}{2}\sum_{i=1}^{M}\ln\alpha_i \tag{C.11}$$

In equation C.11 we can see that the third, fourth and fifth terms have dependencies on α_i.

Tackling each term individually,

$$\frac{d}{d\alpha_i}\left[\frac{1}{2}\sum_{i=1}^{M}\ln\alpha_i\right] = \frac{1}{2\alpha_i} \tag{C.12}$$

$$\frac{d}{d\alpha_i}\left[\frac{1}{2}\ln|\Sigma|\right] = \frac{1}{2|\Sigma|}\frac{d}{d\alpha_i}(|\Sigma|) \tag{C.13}$$

Further,

$$\frac{d}{d\alpha_i}(|\Sigma|) = |\Sigma| \times Tr\Big(\Sigma^{-1} \times \frac{d}{d\alpha_i}\Sigma\Big) \tag{C.14}$$

where we use Jacobi's formula from appendix section 3.5.

We know that $\Sigma = (\mathbf{A} + \sigma^{-2}\Phi^T\Phi)^{-1}$, for neatness let $(\mathbf{A} + \sigma^{-2}\Phi^T\Phi) = K$ so then $\Sigma = K^{-1}$

Using the result on derivatives of inverse matrices from appendix section 3.5 we get,

$$\frac{d}{d\alpha_i}\Sigma = \frac{d}{d\alpha_i}K^{-1} = -K^{-1}\frac{dK}{d\alpha_i}K^{-1} \tag{C.15}$$

Hence,

$$\begin{aligned}
\frac{d}{d\alpha_i}(|\Sigma|) &= |\Sigma| \times Tr\Big(K \times -K^{-1}\frac{dK}{d\alpha_i}K^{-1}\Big) \\
&= -|\Sigma| \times Tr\left(\frac{dK}{d\alpha_i}K^{-1}\right) \\
&= -|\Sigma| \times Tr\left(\frac{d}{d\alpha_i}(\mathbf{A} + \sigma^{-2}\Phi^T\Phi) \times (\mathbf{A} + \sigma^{-2}\Phi^T\Phi)^{-1}\right) \\
&= -|\Sigma| \times Tr\left(\frac{d\mathbf{A}}{d\alpha_i} \times (\mathbf{A} + \sigma^{-2}\Phi^T\Phi)^{-1}\right)
\end{aligned} \tag{C.16}$$

Now,

$$(\mathbf{A} + \sigma^{-2}\Phi^T\Phi)^{-1} = \Sigma$$

and

$$\frac{d\mathbf{A}}{d\alpha_i} = \begin{bmatrix} 0 & 0 & 0 & & 0 \\ 0 & \ddots & \ddots & & 0 \\ 0 & \ddots & 1(=\frac{\partial\alpha_i}{\partial\alpha_i}) & \ddots & \\ & \ddots & & \ddots & 0 \\ 0 & 0 & & 0 & 0 \end{bmatrix}_{M\times M} \tag{C.17}$$

with α_i in the i-th row and i-th column.

Hence, enabling to re-write;

$$\frac{d}{d\alpha_i}(|\Sigma|) = -|\Sigma| \times Tr\left(\begin{bmatrix} 0 & 0 & 0 & & 0 \\ 0 & \ddots & \ddots & & 0 \\ 0 & \ddots & \Sigma_{ii} & \ddots & \\ & \ddots & & \ddots & 0 \\ 0 & 0 & & 0 & 0 \end{bmatrix}_{M\times M}\right) \tag{C.18}$$

$$\frac{d}{d\alpha_i}(|\Sigma|) = -|\Sigma| \times \Sigma_{ii}$$

Substituting this result back into eq. C.13 we have,

$$\frac{d}{d\alpha_i}\left[\frac{1}{2}\ln|\Sigma|\right] = \frac{1}{2|\Sigma|}\frac{d}{d\alpha_i}(|\Sigma|) = \frac{-1}{2}\Sigma_{ii} \tag{C.19}$$

Now for the last part we need to differentiate $-E(\mathbf{t})$ where,

$$-E(\mathbf{t}) = \frac{-1}{2}\left\{\sigma^{-2}\mathbf{t}^T\mathbf{t} - \boldsymbol{\mu}^T\Sigma^{-1}\boldsymbol{\mu}\right\} \tag{C.20}$$

$$\begin{aligned}
\frac{d}{d\alpha_i}(-E(\mathbf{t})) &= \frac{1}{2}\left\{\frac{d}{d\alpha_i}(\boldsymbol{\mu}^T\Sigma^{-1}\boldsymbol{\mu})\right\} \\
&= \frac{1}{2}\left\{\frac{d}{d\alpha_i}((\sigma^{-2}\Sigma\Phi^T\mathbf{t})^T\Sigma^{-1}(\sigma^{-2}\Sigma\Phi^T\mathbf{t}))\right\} \\
&= \frac{1}{2}\left\{\frac{d}{d\alpha_i}(\sigma^{-4}\mathbf{t}^T\Phi\Sigma^T\Sigma^{-1}\Sigma\Phi^T\mathbf{t})\right\} \\
&= \frac{1}{2}\left\{\frac{d}{d\alpha_i}(\sigma^{-4}\mathbf{t}^T\Phi\Sigma\Phi^T\mathbf{t}) \text{ as } \Sigma^T\Sigma^{-1} = \mathbf{I}\right\} \\
&= \frac{1}{2}\left\{\frac{d}{d\alpha_i}(\sigma^{-4}\mathbf{t}^T\Phi(\mathbf{A} + \sigma^{-2}\Phi^T\Phi)^{-1}\Phi^T\mathbf{t})\right\} \\
&= \frac{1}{2}\left\{\sigma^{-4}\mathbf{t}^T\Phi\frac{d}{d\alpha_i}(\mathbf{A} + \sigma^{-2}\Phi^T\Phi)^{-1}\Phi^T\mathbf{t}\right\}
\end{aligned} \tag{C.21}$$

We already know $\frac{d}{d\alpha_i} = -\Sigma\frac{d\mathbf{A}}{d\alpha_i}\Sigma$,

Plugging this result back into the last equation of C.21,

$$\begin{aligned}\frac{d}{d\alpha_i}(-E(\mathbf{t})) &= \frac{1}{2}\left\{-(\sigma^{-2}\mathbf{t}^T\Phi\Sigma^T)\frac{d\mathbf{A}}{d\alpha_i}(\sigma^{-2}\Sigma\Phi^T\mathbf{t})\right\} \\ &= \frac{1}{2}\left\{-\boldsymbol{\mu}^T\frac{d\mathbf{A}}{d\alpha_i}\boldsymbol{\mu}\right\}\end{aligned} \tag{C.22}$$

Substituting the result of $\frac{d\mathbf{A}}{d\alpha_i}$ from equation C.17,

$$\frac{d}{d\alpha_i}(-E(\mathbf{t})) = \frac{-1}{2}\mu_i^2 \tag{C.23}$$

Finally, putting all the terms in eq. C.12, C.19 and C.23 together, we have the derivative of the log marginal likelihood w.r.t α_i:

$$\begin{aligned}\frac{d}{d\alpha_i}\ln p(\mathbf{t}|\boldsymbol{\alpha},\sigma^2) &= \frac{1}{2\alpha_i} - \frac{1}{2}\Sigma_{ii} - \frac{1}{2}\mu_i^2 = 0 \\ &\Rightarrow \alpha_i = \frac{1-\alpha_i\Sigma_{ii}}{\mu_i^2}\end{aligned} \tag{C.24}$$

In equation C.24, Σ_{ii} is the (i,i)-th element in the posterior weight covariance matrix, since it is on the diagonal it is the variance of w_i. μ_i is the i-th element of the posterior mean weight vector $\boldsymbol{\mu}$.

Substituting, $\gamma_i = 1 - \alpha_i\Sigma_{ii}$ we can express the above equation as,

$$\alpha_i = \frac{\gamma_i}{\mu_i^2} \tag{C.25}$$

1.2 Differentiating w.r.t σ^2

We now need to differentiate with respect to σ^2 and set the derivative to 0. Recall that, the log marginal likelihood is given by:

$$\ln p(\mathbf{t}|\boldsymbol{\alpha},\sigma^2) = -\frac{N}{2}\ln(\sigma^2) - \frac{N}{2}\ln(2\pi) + \frac{1}{2}\ln|\Sigma| - E(\mathbf{t}) + \frac{1}{2}\sum_{i=1}^{M}\ln\alpha_i \tag{C.26}$$

Notice that σ^2 appears in the denominator in Σ as $\Sigma = (\mathbf{A} + \sigma^{-2}\Phi^T\Phi)^{-1}$. For mathematical convenience we assign $\beta^{-1} = \sigma^2$ and differentiate w.r.t β.

Substituting β in the log marginal likelihood,

$$\ln p(\mathbf{t}|\boldsymbol{\alpha}, \beta) = \frac{N}{2}\ln(\beta) - \frac{N}{2}\ln(2\pi) + \frac{1}{2}\ln|\Sigma| - E(\mathbf{t}) + \frac{1}{2}\sum_{i=1}^{M}\ln\alpha_i \tag{C.27}$$

Only the first, third and fourth terms in eq. C.26 have dependency on σ^2.
Tackling each term individually,

$$\frac{d}{d\beta}\left[\frac{N}{2}\ln\beta\right] = \frac{N}{2\beta} \tag{C.28}$$

$$\begin{aligned}
\frac{d}{d\beta}E(\mathbf{t}) &= \frac{1}{2}\mathbf{t}^T\mathbf{t} - \frac{1}{2}\left[\frac{d}{d\beta}(\boldsymbol{\mu}^T\Sigma^{-1}\boldsymbol{\mu})\right] \\
&= \frac{1}{2}\mathbf{t}^T\mathbf{t} - \frac{1}{2}\left[\frac{d}{d\beta}((\beta\Sigma\Phi^T\mathbf{t})^T\Sigma^{-1}(\beta\Sigma\Phi^T\mathbf{t}))\right] \\
&= \frac{1}{2}\mathbf{t}^T\mathbf{t} - \frac{1}{2}\left[\frac{d}{d\beta}(\beta^2\mathbf{t}^T\Phi\Sigma^T\Sigma^{-1}\Sigma\Phi^T\mathbf{t})\right] \\
&= \frac{1}{2}\mathbf{t}^T\mathbf{t} - \frac{1}{2}\left[\frac{d}{d\beta}(\beta^2\mathbf{t}^T\Phi\Sigma\Phi^T\mathbf{t})\right] \text{ as } \Sigma^T\Sigma^{-1} = \mathbf{I}
\end{aligned} \tag{C.29}$$

Given that Σ has a dependency on β we apply the product rule,

$$\frac{d}{d\beta}(\beta^2\mathbf{t}^T\Phi\Sigma\Phi^T\mathbf{t}) = 2\beta\mathbf{t}^T\Phi\Sigma\Phi^T\mathbf{t} + \beta^2\mathbf{t}^T\Phi\frac{d}{d\beta}(\mathbf{A} + \beta\Phi^T\Phi)^{-1}\Phi^T\mathbf{t} \tag{C.30}$$

Applying the rule for the derivative of an inverse of a matrix and letting $(\mathbf{A} + \beta\Phi^T\Phi) = \Sigma^{-1} = K$,

$$\begin{aligned}
\frac{d}{d\beta}(\mathbf{A} + \beta\Phi^T\Phi)^{-1} = \frac{d}{d\beta}K^{-1} &= -K^{-1}\frac{dK}{d\beta}K^{-1} \\
&= -\Sigma\Phi^T\Phi\Sigma
\end{aligned} \tag{C.31}$$

Hence,

$$\begin{aligned}\frac{d}{d\beta}(\beta^2\mathbf{t}^T\Phi\Sigma\Phi^T\mathbf{t}) &= 2\beta\mathbf{t}^T\Phi\Sigma\Phi^T\mathbf{t} - \beta^2\mathbf{t}^T\Phi\Sigma\Phi^T\Phi\Sigma\Phi^T\mathbf{t} \\ &= 2(\beta\mathbf{t}^T\Phi\Sigma^T)\Phi^T\mathbf{t} - (\beta\mathbf{t}^T\Phi\Sigma^T)\Phi^T\Phi(\beta\Sigma\Phi^T\mathbf{t}) \\ &= 2\boldsymbol{\mu}^T\Phi^T\mathbf{t} - \boldsymbol{\mu}^T\Phi^T\Phi\boldsymbol{\mu}\end{aligned} \tag{C.32}$$

where we use the symmetry of Σ.

Plugging this back into eq. C.29 gives,

$$\begin{aligned}\frac{d}{d\sigma^2}E(\mathbf{t}) &= \frac{1}{2}\left\{\mathbf{t}^T\mathbf{t} - 2\boldsymbol{\mu}^T\Phi^T\mathbf{t} + \boldsymbol{\mu}^T\Phi^T\Phi\boldsymbol{\mu}\right\} \\ &= \frac{1}{2}||\mathbf{t} - \Phi\boldsymbol{\mu}||^2\end{aligned} \tag{C.33}$$

Note, this is the squared difference between the targets and their predicted values.

For the last part we need,

$$\begin{aligned}&\frac{d}{d\beta}\Big(\frac{1}{2}\ln|\Sigma|\Big) \\ &= \frac{1}{2|\Sigma|}\frac{d}{d\beta}(|\Sigma|)\end{aligned} \tag{C.34}$$

$$\frac{d}{d\beta}(|\Sigma|) = Tr\Big(|\Sigma| \times \Sigma^{-1} \times \frac{d}{d\beta}(\Sigma)\Big) \tag{C.35}$$

We already worked out $\frac{d}{d\beta}(\Sigma)$ in eq.C.31.

Plugging this result back into C.35 we get,

$$\begin{aligned}\frac{d}{d\beta}(|\Sigma|) &= Tr\Big(|\Sigma| \times \Sigma^{-1} \times (-\Sigma\Phi^T\Phi\Sigma)\Big) \\ &= -|\Sigma| \times Tr\Big(\Phi^T\Phi\Sigma\Big)\end{aligned} \tag{C.36}$$

Finally, plugging this back into C.34,

$$\frac{d}{d\beta}\Big(\frac{1}{2}\ln|\Sigma|\Big) = \frac{-1}{2}Tr(\Phi^T\Phi\Sigma) \tag{C.37}$$

Hence,

$$\frac{d}{d\beta}\ln p(\mathbf{t}|\boldsymbol{\alpha},\beta) = \frac{1}{2}\left(\frac{N}{\beta} - ||\mathbf{t}-\Phi\boldsymbol{\mu}||^2 - Tr\left[\Phi^T\Phi\Sigma\right]\right) = 0 \tag{C.38}$$

Simplifying the argument of the trace operator $Tr[\bullet]$,

$$\begin{aligned}\Phi^T\Phi\Sigma &= \Phi^T\Phi\Sigma + \beta^{-1}\mathbf{A}\Sigma - \beta^{-1}\mathbf{A}\Sigma \\ &= \beta^{-1}(\Phi^T\Phi\beta + \mathbf{A})\Sigma - \beta^{-1}\mathbf{A}\Sigma \\ &= \beta^{-1}(\Phi^T\Phi\beta + \mathbf{A})(\mathbf{A} + \beta\Phi^T\Phi)^{-1} - \beta^{-1}\mathbf{A}\Sigma \\ &= \beta^{-1}(\mathbf{I} - \mathbf{A}\Sigma)\end{aligned} \tag{C.39}$$

Substituting this back into equation C.38

$$\begin{aligned}&\frac{1}{2}(\frac{N}{\beta} - ||\mathbf{t}-\Phi\boldsymbol{\mu}||^2 - Tr\left[\frac{\mathbf{I}-\mathbf{A}\Sigma}{\beta}\right]) \\ &\Rightarrow \frac{N}{\beta} - Tr\left[\frac{\mathbf{I}-\mathbf{A}\Sigma}{\beta}\right] = ||\mathbf{t}-\Phi\boldsymbol{\mu}||^2 \\ &\Rightarrow \frac{1}{\beta}\left(N - Tr\left[\mathbf{I}-\mathbf{A}\Sigma\right]\right) = ||\mathbf{t}-\Phi\boldsymbol{\mu}||^2 \\ &\Rightarrow \frac{1}{\beta} = \frac{||\mathbf{t}-\Phi\boldsymbol{\mu}||^2}{N - Tr\left[\mathbf{I}-\mathbf{A}\Sigma\right]} \\ &\Rightarrow \beta = \frac{N - \sum_i \gamma_i}{||\mathbf{t}-\Phi\boldsymbol{\mu}||^2}\end{aligned} \tag{C.40}$$

1.3 Summary

where $Tr\left[\mathbf{I}-\mathbf{A}\Sigma\right] = \sum_i \gamma_i$.

It comes to light from the expressions of the maximising α_i's (eq. C.24) and β (eq. C.40) that the values cannot be obtained in closed form as they disadvantageously appear on the right hand side. Thus, iterative re-estimation is used. For instance, in the case of α_i the next, α_i^{new} is predicted by,

$$\alpha_i^{new} = \frac{1 - \alpha_i^{current}\Sigma_{ii}}{\mu_i^2} \tag{C.41}$$

where the current values of $\boldsymbol{\alpha}$ and β are used.

Similarly, for β, the re-estimation equation yields,

$$\beta^{new} = \frac{N - \sum_i \gamma_i^{current}}{||\mathbf{t} - \Phi\boldsymbol{\mu}||^2} \tag{C.42}$$

where the current values of $\boldsymbol{\alpha}$ and β are used in computing the statistics $\boldsymbol{\mu}$, Σ and γ_i on the right hand side of eq. The learning algorithm entails the repeated computation of eq. C.41 and eq. C.42.

One approach to find the α_i and β uses an Expectation- Maximization approach of setting them first to initial values, computing $\boldsymbol{\mu}$ and Σ and then using these to re-calculate $\boldsymbol{\alpha}$ and β and repeating this process until a convergence criteria is met.

1.4 Slow version of the Algorithm

The algorithm has an iterative flavour and involves repeatedly estimating the hyper-parameters $\boldsymbol{\alpha}$ and β until a stopping criterion is triggered. The explicit steps are as follows:

Algorithm 3 Slow Maximization of the marginal likelihood in SBL

1: Make a suitable choice for basis functions in Φ.
2: Establish a suitable convergence criterion for $\boldsymbol{\alpha}$ and β. For example, a threshold value for change δ_{thresh} between one iteration and the next $\delta = \sum_{i=1} \alpha_i^{n+1} - \alpha_i^n$ (the subscript i indicate iteration numbers) so that re-estimation will stop when $\delta < \delta_{thresh}$.
3: **Select** a threshold value α_{thresh} which when triggered by any α_i it is assumed that it is tending to ∞.
4: **Choose** starting values for $\boldsymbol{\alpha}$ and β.
5: **Calculate** $\boldsymbol{\mu} = \beta\boldsymbol{\Sigma}\boldsymbol{\Phi}^{\boldsymbol{T}}\boldsymbol{t}$ and $\Sigma = (\mathbf{A} + \beta\boldsymbol{\Phi}^{\boldsymbol{T}}\boldsymbol{\Phi})^{-1}$
6: **Update** $\alpha_i = \dfrac{\gamma_i}{\mu_i^2}$ and $\beta = \dfrac{N - \Sigma_i\gamma_i}{||\mathbf{t} - \boldsymbol{\Phi}\boldsymbol{m}||^2}$
7: Prune the α_i and corresponding basis function where $\alpha_i > \alpha_{Thresh}$.
8: **Repeat** steps (5)-(7) until the convergence criterion is met.

A fast marginal likelihood algorithm was proposed by Faul and Tipping in 2003 [TF+03]. Maximisation procedure under the fast algorithm is the theme of chapter 4.

Bibliography

[ADWW17] Anis Ben Abdessalem, Nikolaos Dervilis, David J Wagg, and Keith Worden. Automatic kernel selection for gaussian processes regression with approximate bayesian computation and sequential monte carlo. *Frontiers in Built Environment*, 3:52, 2017.

[And15] Lucio Anderlini. Density estimation trees in high energy physics. *arXiv preprint arXiv:1502.00932*, 2015.

[ASM06] Shawkat Ali and Kate A Smith-Miles. A meta-learning approach to automatic kernel selection for support vector machines. *Neurocomputing*, 70(1):173–186, 2006.

[Bis06] Christopher M Bishop. Pattern recognition. *Machine Learning*, 128, 2006.

[Bou] Chris Boucher. Wolframs demonstration project. http://demonstrations.wolfram.com.

[BYC13] James Bergstra, Daniel Yamins, and David Cox. Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. In *International Conference on Machine Learning*, pages 115–123, 2013.

[Cra01] Kyle Cranmer. Kernel estimation in high-energy physics. this work was partially supported by a graduate research fellowship from the national science foundation and us department of energy grant de-fg0295-er40896. *Computer Physics Communications*, 136(3):198–207, 2001.

[DGRC11] Nicolas Durrande, David Ginsbourger, Olivier Roustant, and Laurent Carraro. Additive covariance kernels for high-dimensional gaussian process modeling. *arXiv preprint arXiv:1111.6233*, 2011.

[DL17] Lizhong Ding and Shizhong Liao. An approximate approach to automatic kernel selection. *IEEE transactions on cybernetics*, 47(3):554–565, 2017.

[DLG+13] David Duvenaud, James Robert Lloyd, Roger Grosse, Joshua B Tenenbaum, and Zoubin Ghahramani. Structure discovery in nonparametric regression through compositional kernel search. *arXiv preprint arXiv:1302.4922*, 2013.

[Duv14] David Duvenaud. *Automatic model construction with Gaussian processes*. PhD thesis, University of Cambridge, 2014.

[Gen01] Marc G Genton. Classes of kernels for machine learning: a statistics perspective. *Journal of machine learning research*, 2(Dec):299–312, 2001.

[GK99] Blaž Gotovac and Vedrana Kozulić. On a selection of basis functions in numerical analyses of engineering problems. *International Journal for Engineering Modelling*, 12(1-4):25–41, 1999.

[GS12] Charles Miller Grinstead and James Laurie Snell. *Introduction to probability*. American Mathematical Soc., 2012.

[Mac92a] David JC MacKay. Bayesian interpolation. *Neural computation*, 4(3):415–447, 1992.

[Mac92b] David JC MacKay. *Bayesian methods for adaptive models*. PhD thesis, California Institute of Technology, 1992.

[Nad64] Elizbar A Nadaraya. On estimating regression. *Theory of Probability & Its Applications*, 9(1):141–142, 1964.

[NP13] Ilya Narsky and Frank C Porter. *Statistical analysis techniques in particle physics: Fits, density estimation and supervised learning*. John Wiley & Sons, 2013.

[P+99] John Platt et al. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10(3):61–74, 1999.

[PP+08] Kaare Brandt Petersen, Michael Syskind Pedersen, et al. The matrix cookbook. *Technical University of Denmark*, 7:15, 2008.

[RQC05] Carl Edward Rasmussen and Joaquin Quinonero-Candela. Healing the relevance vector machine through augmentation. In *Proceedings of the 22nd international conference on Machine learning*, pages 689–696. ACM, 2005.

[TF+03] Michael E Tipping, Anita C Faul, et al. Fast marginal likelihood maximisation for sparse bayesian models. In *AISTATS*, 2003.

[Tip01] Michael E Tipping. Sparse bayesian learning and the relevance vector machine. *Journal of machine learning research*, 1(Jun):211–244, 2001.

[Van14] Jake VanderPlas. Frequentism and bayesianism: a python-driven primer. *arXiv preprint arXiv:1411.5018*, 2014.

[V.K14] Mikhail V.Konnik. A short intro to radial basis functions. `http://www.mvkonnik.info/2014/07/a-short-intro-to-radial-basis-functions.html`, July 2014.

[WM92] L-X Wang and Jerry M Mendel. Fuzzy basis functions, universal approximation, and orthogonal least-squares learning. *IEEE transactions on Neural Networks*, 3(5):807–814, 1992.