An Introduction to Probabilistic Machine Learning

Machine Learning (190.012/13)

by Univ.-Prof. Dr. Elmar Rueckert

Disclaimer
The lecture notes in this book are the first part of a series of lecture notes on robot learning and AI. This part provides an introduction to probabilistic machine learning. A second part on Reinforcement Learning is under development. Details to the accompanying courses can be found on ai-lab.science or cps.unileoben.ac.at.

The authors of the contributions.
Nils Rottmann, University of Luebeck 2018 to 2021. Rabia Demirci, University of Luebeck 2020 to 2021.

Colophon
This document was typeset with the help of KOMA-Script and LaTeX using the kaobook class.

AI is 'the science and engineering of making intelligent machines, especially intelligent computer programs. It is related to the similar task of using computers to understand human intelligence, but AI does not have to confine itself to methods that are biologically observable.'

– John McCarthy

Das einzig Unendliche auf unserer Erde ist das unerschöpfliche Wissen, das es zu schaffen gilt.

– Elmar Rueckert

Preface

This book presents fundamental theories, algorithms and concepts of probabilistic machine learning techniques. It is written for experienced undergraduate or first semester graduate students. The content will be expanded during the lecture series of Univ.-Prof. Dr. Elmar Rueckert.

Organization of this book. The book is organized in five chapters that are *An Introduction t6 Probability Theory*, *Linear Probabilistic Regression*, *Nonlinear Probabilistic Regression*, *Probabilistic Inference*, and *Probabilistic Optimization*.

Each chapter starts with listing the *Learning Objectives*, followed by a *Theory or Methods* section. The chapter concludes with a series of mathematical or programming *Exercises*.

Recommended Books. This book is a brief introduction to numerous basic and advanced machine learning concepts. Parts of the presented materials builds on the following books which are great sources for detailed information.

- Daphne Koller, Nir Friedman. Probabilistic Graphical Models: Principles and Techniques. ISBN 978-0-262-01319-2
- Christopher M. Bishop. Pattern Recognition and Machine Learning. Springer (2006). ISBN 978-0-387-31073-2.
- David Barber. Bayesian Reasoning and Machine Learning, Cambridge University Press (2012). ISBN 978-0-521-51814-7.
- Kevin P. Murphy. Machine Learning: A Probabilistic Perspective. ISBN 978-0-262-01802-9

Contents

Contents vi

Introduction to Machine Learning and Probability Theory 1

1 Introduction to Machine Learning 2
1.1 Types of Machine Learning . . . 2

2 Introduction to Probability Theory 4
2.1 Frequentist or Bayesian View . . . 4
2.2 Definition of Random Variables . . . 4
2.3 Discrete Random Variables . . . 5
2.4 Fundamental Rules . . . 5
2.5 Fundamental Discrete Distributions . . . 6
2.6 Fundamental Continuous Distributions . . . 8
2.7 Information Theory . . . 9
2.8 Exercises . . . 10

Probabilistic Regression 13

3 Linear Probabilistic Regression 14
3.1 Linear Feature Regression . . . 14
3.2 Basis function models . . . 15
3.3 Logisitic Regression . . . 16
3.4 Maximum Likelihood . . . 16
3.5 Maximum A-Posteriori . . . 18
3.6 Full Bayesian Inference . . . 19
3.7 Predictive Distribution . . . 20
3.8 Exercises . . . 21

4 Nonlinear Probabilistic Regression 22
4.1 Gaussian Processes . . . 22
4.2 GP with built-in GMRF . . . 24
4.3 Exercises . . . 26

Probabilistic Inference 27

5 Markov Models 28
5.1 Markov Chains . . . 28
5.2 Markov Random Fields . . . 29
5.3 Gaussian Markov Random Fields . . . 29
5.4 Variogram . . . 30
5.5 Exercises . . . 30

6 Probabilistic Time Series Models 32
6.1 Time Series Data . . . 32

6.2 Single Time Step Model 33
6.3 Multi-Time Step Models 34
6.4 Trajectory Prediction or Completion 35
6.5 Exercises 36

Appendix **37**

A Notation **38**
A.1 Numbers and Arrays 38
A.2 Sets and Graphs 38
A.3 Indexing 39
A.4 Calculus 39
A.5 Probability and Information Theory 40
A.6 Functions 40
A.7 The Greek Alphabet 41

B Mathematical background and derivations **42**
B.1 Derivation of the Least Squares Solution 42

Bibliography **44**

Alphabetical Index **45**

List of Figures

1.1 Types of Machine Learning 2

2.1 Venn Diagramm of fruit preferences 5
2.2 PDFs of Beta Distributions 8

3.1 Perceptron Neuron Model 14
3.2 Overview of Probabilistic Linear Regression Methods 21

4.1 Multivariate Gaussian Distribution 22
4.2 Gaussian Process 23

5.1 Directed Markov Chain 28
5.2 Markov Graph 28
5.3 Neighbouring System 29
5.4 Data Arrangement 30

6.1 Illustration of Radial Basis Functions 33
6.2 Probabilisitic Trajectory Prediction Example 36

Introduction to Machine Learning and Probability Theory

Introduction to Machine Learning

1

In this section, machine learning fundamentals are discussed. In Figure 1.1, the input/output relations and the usage of teacher signals for training the models is contrasted for these three fundamental learning types.

1.1 Types of Machine Learning 2

1.1 Types of Machine Learning

Three main types of learning can be differentiated by inspecting how the model parameters are updated, and how model inputs and outputs are used.

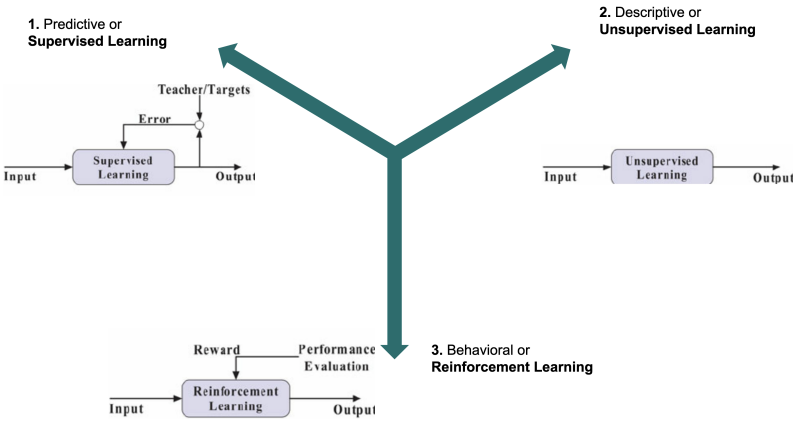

Figure 1.1: The figure shows the three main types of machine learning approaches.

Supervised Learning

In supervised learning, a target value or output sample exists for each input sample. This type of learning is also called **predictive learning**.

Classical regression and classification techniques fall into that category. For concrete examples, please read the chapter on linear regression (Chapter 3).

The key idea of supervised learning is to minimize an objective, e.g., the Euclidean distance between the inputs and given targets.

Unsupervised Learning

In unsupervised learning, only input samples are given and the goal is to analyze or compress the data. This type of learning is also called **descriptive learning**.

Typical examples of unsupervised learning are *clustering algorithms* such the *k-means* algorithm. Other typical examples of unsupervised learning approaches are **feature learning algorithms** such as *autoencoder*. An

autoencoder is a neural network that aims at reproducing the inputs. The network consists of an encoder and decoder. The encoder compresses the data, which corresponds to feature learning. The decoder tries to reproduce the (uncompressed/original) inputs based on the output of the encoder (the typically low-dimensional feature representation). For more details, please have a look at [1].

[1]: Goodfellow et al. (2016), *Deep Learning*

The key idea of unsupervised learning is to extract representative features, compress the data or identify similarities.

Reinforcement Learning

Reinforcement Learning is optimizing a model based on a sparse reward signal. Unlike in supervised learning (where the teaching signal is given for every input), such a sparse reward is only given occasionally. The sparse reward signal is used to interpret a sequence of past actions of a behaving agent.

Reinforcement learning is also called **trial and error** or **behavorial learning**. An agent perceives the world, computes an action based on its learned policy (the model) and might receive a reward. The challenges lie in the perception of the world, the modeling of the policy and the sparse reward. For more details, we refer to the book by Sutton and Barto on reinforcement learning [2].

[2]: Sutton et al. (2018), *Reinforcement learning: An introduction*

The key idea of reinforcement learning is to maximize a cumulative reward.

Introduction to Probability Theory

2

In this chapter we will discuss basic concepts of probability theory and how they can be applied. We start with the concept of discrete random variables and their fundamental rules.

2.1 Frequentist or Bayesian View 4
2.2 Definition of Random Variables 4
2.3 Discrete Random Variables 5
2.4 Fundamental Rules 5
2.5 Fundamental Discrete Distributions 6
2.6 Fundamental Continuous Distributions 8
2.7 Information Theory 9
2.8 Exercises 10

2.1 Frequentist or Bayesian View

When talking about probabilities, two different and often competing perspectives exist. First, in the *Frequentist* interpretation, a probability denotes the number of occurrences of certain events, e.g., if we flip a coin many times we expect it to land heads or tails about half of the time.

In the *Bayesian* interpretation we quantify the uncertainty of events happening. For example in the above coin flipping experiment, we believe that the coin is equally likely to land heads or tails.

It is important to understand the differences of these two conceptual views. Throughout this book we will apply the *Bayesian* interpretation of probabilities as it can be applied to problem instances with rare events, allows to integrate prior information in a principled way and can be used to solve complex inference tasks.

2.2 Definition of Random Variables

Probability space. Let Ω define a set of possible states of the world. For example $\Omega = \{\text{heads}, \text{tails}\}$ or $\Omega = \{H, T\}$ of a coin.

An event is a subspace of Ω. For example, consider tossing two coins and you are interested in the event that both land on *heads*, i.e., $A = \{HH\}$.

Random variables. Assume a function that maps from Ω to real numbers $\mathbb{R}$. We can formalize this mapping mathematically for a random variable X with $X : \Omega \rightarrow \mathbb{R}$. **Such a function is called random variable or variable**.

Definition 2.2.1 *A random variable is a function that maps from Ω to real numbers $\mathbb{R}$, i.e., $X : \Omega \rightarrow \mathbb{R}$.*

Indicator function. Using the above definitions we can formalize experimental conditions. For example, let x denote a value of X like that a coin lands on *heads*. Further, let $\{X = x\}$ denote the **event** of this scenario. This event is defined as $\{w \in \Omega : X(w) = x\}$, where w denotes the individual or trial in our experiment. Moreover, the set of all possible values of the random variable (RV) X is denoted by Ω_X.

2.3 Discrete Random Variables

Let $\mathfrak{X}$ denote a *finite* or *countable infinite* set of discrete random values. We define the probablity $p(x)$ as a short hand for the probability that the random variable X takes the value x, i.e., $p(X = x)$. Here $p(x)$ denotes the *probability mass function* (pmf) that satisfies the properties,

$$0 \leq p(x) \leq 1 \text{ and } \sum_{x\ in \mathfrak{X}} p(x) = 1. \tag{2.1}$$

Binary Random Variables

A binary random variable takes the values of either 'true' or 'false'. We denote binary random variables by $p(A)$ which is the short hand for $p(A = 1)$. Contrary, $p(\overline{A})$ means that event A will be 'false', or $p(A = 0)$. From Equation 2.1 we can infer $p(\overline{A}) = 1 - p(A)$.

2.4 Fundamental Rules

In the following, basic fundamental rules for calculations with probability distributions are discussed. A common visual representation for these rules are *Venn diagrams* [3]. An example of fruit preferences of 40 children is shown in Figure 2.1.

[3]: Ruskey et al. (1997), 'A survey of Venn diagrams'

Compliment of an event. Let $p(A^c)$ define the compliment of $p(A)$ with respect to the probability space Ω. Thus A^c is the event of all states not in A. Note that for binary random variables $p(\overline{A})$ is used as notation. Sometimes the notation $P(A^{'})$ is used in related literature.

The Union of two events is defined as $p(A \cup B) = p(A) + p(B) - p(A \cap B)$. If both random variables are mutually exclusive, i.e., when both events cannot occur at the same time, then the union of two events simplifies to $p(A \cup B) = p(A) + p(B)$.

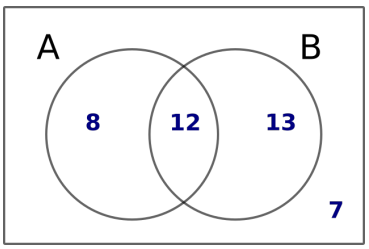

Figure 2.1: The figure shows an example of a Venn Diagram representing fruit preferences of 40 children. The *Union* of children that like apples (A) or bananas (B) is $p(A \cup B) = 33/40$.

The Intersection of two events is defined as $p(A \cap B)$ and denotes events where both random variables take the same value. For example, when tossing two coins and both land on heads.

The Joint probability of two events as

$$p(A, B) = p(A \cap B) = p(A|B)p(B).$$

It is also called the product rule and is of fundamental importance to understand conditional probabilities and the *Bayes* rule which we will introduce next.

The Marginal (distribution) is defined as

$$p(A) = \sum_b p(A|B = b)p(B = b).$$

We are summing over all possible values of the random variable B.

The Conditional probability is defined as $p(A|B)$ and can be computed from the joint probability $p(A|B) = p(A, B)/p(B)$.

The Bayes Rule From $p(A, B) = p(A|B)p(B) = p(B|A)p(A)$, the Bayes rule can be derived, i.e.,

$$p(A|B) = \frac{p(B|A)p(A)}{p(B)}. \tag{2.2}$$

Often the denominator is expressed through applying the *Marginal rule* above which leads to

$$p(A|B) = \frac{p(B|A)p(A)}{\sum_a p(B|A = a)p(A = a)}. \tag{2.3}$$

Remark 2.4.1 In the literature, the term *likelihood* denotes the term $p(B|A)$ and often refers to the probability that our observed data B could have been generated by some model A. The result of the Bayes rule is called *posterior* and can be used to answer the most important question of *how well our model for A explains the observed data B*.

Note that through applying the Bayes rule we can answer this question in an elegant way. However, we assume that our model can *generate* artificial data, thus we assume a *generative model*. Moreover, we assume that we know the prior distribution $p(A)$, which is one of the most significant points of critics of the *frequentist* community.

2.5 Fundamental Discrete Distributions

In this section, we discuss some fundamental distributions that are defined on **discrete state spaces**, both finite and countable finite.

Bernoulli and Binomial distributions

In Section 2.3, we introduced a binary random variable $p(A)$ that takes the values of either 'true' ($A = 1$) or false ($A = 0$). To model this RV as parameterized distribution with the parameter μ, we denote the probability of $A = 1$ by

$$p(A = 1|\mu) = \mu,$$

where $0 \leq \mu \leq 1$. From Equation 2.1 it follows that $p(A = 0|\mu) = 1 - \mu$. The probability distribution over A can be written as parameterized distribution as

$$\text{Bern}(A|\mu) = \mu^A \, (1 - \mu)^{(1-A)}. \tag{2.4}$$

This distribution is known as *Bernoulli* distribution. For a fair coin, $\mu = 0.5$.

Now suppose we toss such a coin n times. Typically, we are interested in how often a particular event occurs, e.g., k times the coin lands on heads. The total number of possible combinations of choosing k items from n can be computes as

$$\binom{n}{k} := \frac{n!}{(n-k)!\, k!},$$

which is the definition (note the := symbol) of the *biominal coefficient*. Using the definition of the probability of $A = 1$ in Equation 2.4, we can derive the parametric distribution of observing k times heads of tossing a coin n times,

$$\text{Bin}(k|n, \mu) := \binom{n}{k} \mu^k \, (1 - \mu)^{(n-k)}.$$

Remark 2.5.1 This parametric distribution is known as the *Binomial* distribution. Note that the mean of this distribution is $n\mu$ and the variance is given by $n\mu(1 - \mu)$.

Multinomial distributions

Let K denote dimensionality of the probability space Ω (defined in Section 2.2). For *binary* random variables $K = 2$. For discrete random variables with more than two possible states or categories ($K > 2$), we define the parameterized multinomial distribution with

$$\text{Mu}(\boldsymbol{x}|n, \boldsymbol{\mu}) := \binom{n}{x_1 \ldots x_K} \prod_{j=1}^{K} \mu_j^{x_j}.$$

The multinomial coefficient denotes the number of ways of partitioning n objects into K groups of sizes $x_1, \ldots, x_K$ and is given by

$$\binom{n}{x_1 \ldots x_K} := \frac{n!}{x_1!\, x_2! \ldots x_K!}.$$

Note that the sum of the group sizes has to be equal to the total number of objects in the set, i.e., $\sum_{j=1}^{K} x_j = n$.

Multinoulli distributions model the special case where $n = 1$. In this case, the vector $\boldsymbol{x}$ is a binary vector where only one of its elements is equal to one and all others are zero. Such an encoding is known as *1-of-K encoding* or *one-hot* encoding. Suppose rolling a $K = 6$ sided dice. Potential states

are $\boldsymbol{x} = [1,0,0,0,0,0]$, $\boldsymbol{x} = [0,0,1,0,0,0]$ or $\boldsymbol{x} = [0,0,0,0,0,1]$. The *Multinoulli* distribution is defined as

$$\text{Mu}(\boldsymbol{x}|1,\boldsymbol{\mu}) := \prod_{j=1}^{K} \mu_j^{x_j},$$

with the constraints $\sum_{j=1}^{K} \mu_j = 1$ and $\mu_j \geq 0 \;\; \forall j \in [0,K]$.

2.6 Fundamental Continuous Distributions

Many real-world problems require continuous variables for modeling. In this section, we will only briefly review the distributions that will be used in the subsequent chapters. For more details and other continuous distributions, we recommend reading *Chapter II on Probability Distributions* in the book of Christopher M. Bishop [4] or *Chapter II on Probability* in the book of Kevin P. Murphy [5].

[4]: Bishop (2006), *Pattern recognition and machine learning*

[5]: Murphy (2012), *Machine learning: a probabilistic perspective*

Beta distributions

The *Beta* distribution has support over the interval $[0,1]$ and is defined as

$$\text{Beta}(x|a,b) := B(a,b)\; x^{(a-1)}\; (1-x)^{(b-1)}, \tag{2.5}$$

where a, b and x are scalars and $B(a,b)$ denotes the **Beta function**

$$\text{B}(a,b) := \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)}.$$

The beta function makes use of the gamma function, which is defined as the integral over $\Gamma(y) = \int_0^\infty u^{(y-1)}\; \exp^{-u}\; du$.

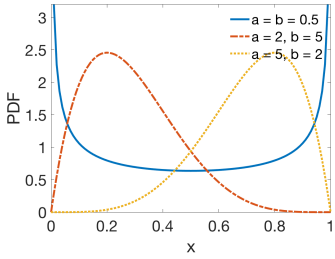

Figure 2.2: The figure shows three example PDFs of the beta distribution with different parameters a and b.

Gaussian or Normal distributions

The *Gaussian* distribution is also known as *normal* distribution and is defined for scalar variables as

$$\mathcal{N}(x|\mu,\sigma) := \frac{1}{(2\pi\sigma^2)^{1/2}} \exp^{\{-\frac{1}{2\sigma^2}\;(x-\mu)^2\}}.$$

The variable μ denotes the mean of the distribution and σ the variance. Note that σ^{-1} denotes the **precision** and is often used to simply the calculus with Gaussian distributions.

Multivariate Gaussians and Conditional distributions

For vectors $\boldsymbol{x} \in \mathbb{R}^{Dx1}$, the Gaussian distribution is defined as

$$\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) := \frac{1}{(2\pi)^{D/2}|\Sigma|^{1/2}} \exp^{\{-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\}}, \tag{2.6}$$

where $|\Sigma|$ denotes the *determinant* of the *covariance matrix* $\boldsymbol{\Sigma} \in \mathbb{R}^{DxD}$.

In Section 2.4, we defined the rule for conditional probabilities. This fundamental rule is an important property of multivariate Gaussians and used for inference. Suppose the state vector $\boldsymbol{x}$ of your random variable is composed of two row vectors, i.e., $\boldsymbol{x} = [\boldsymbol{x}_a^T, \boldsymbol{x}_b^T]^T$.

Remark 2.6.1 The random variable $\boldsymbol{x}_b$ might denote the ball position and $\boldsymbol{x}_a$ the position of the goalkeeper in a sports game. Using the rule of conditional probabilities, we can **infer the most likely goalkeeper position** $\boldsymbol{x}_a$, given a certain ball location (potentially a few milliseconds after the ball kick event, such that the keeper has still time to react).

Thus we are interested in a conditional probability $p(\boldsymbol{x}_a|\boldsymbol{x}_b; \boldsymbol{\mu}, \boldsymbol{\Sigma})$. Note that we assume that we have learned a model from observations, e.g., pairs of ball and keeper positions. This model is represented by $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ and as we assume that these parameters are known. We highlight that difference to the given variable $\boldsymbol{x}_a$ with a semicolon (';').

The solution is given by

$$p(\boldsymbol{x}_a|\boldsymbol{x}_b; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \mathcal{N}(\boldsymbol{x}_a|\boldsymbol{\mu}_{a|b}, \boldsymbol{\Sigma}_{a|b}), \tag{2.7}$$

with

$$\boldsymbol{\mu}_{a|b} = \boldsymbol{\mu}_a + \boldsymbol{\Sigma}_{ab}\boldsymbol{\Sigma}_{bb}^{-1}(\boldsymbol{x}_b - \boldsymbol{\mu}_b) \text{ and} \tag{2.8}$$

$$\boldsymbol{\Sigma}_{a|b} = \boldsymbol{\Sigma}_{aa} - \boldsymbol{\Sigma}_{ab}\boldsymbol{\Sigma}_{bb}^{-1}\boldsymbol{\Sigma}_{ba}, \tag{2.9}$$

where we assumed $\boldsymbol{\mu} = [\boldsymbol{\mu}_a^T, \boldsymbol{\mu}_b^T]^T$ and $\boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma}_{aa} & \boldsymbol{\Sigma}_{ab} \\ \boldsymbol{\Sigma}_{ba} & \boldsymbol{\Sigma}_{bb} \end{bmatrix}$.

For more details, we refer to Chapter 2.3, page 87 in the text book of Christopher M. Bishop [4].

2.7 Information Theory

In this section, we review some of the most important concepts in information theory.

Entropy

The entropy of a random variable x is given by

$$\mathrm{H}(x) := -\sum_{x\in\Omega} p(x)\log p(x).$$

Kullback-Leibler Divergence or Relative Entropy

In computer science, we are often interested in comparing distributions. Suppose, we have a *true data distribution* $p(x)$ and we developed a model that is represented by the *model distribution* $q(x)$. To quantify the quality of our model distribution, we would like to evaluate the 'distance' or 'similarity' to the true data distribution. A common distance measure is the **Kullback-Leibler Divergence** or **relative entropy**,

$$\mathrm{KL}(p||q) := -\int p(x)\log\frac{q(x)}{p(x)}dx.$$

Remark 2.7.1 Note that the KL-divergence is not symmetric, i.e.,

$$\mathrm{KL}(q||p) \neq \mathrm{KL}(p||q).$$

Two common approaches to generate symetric measures are

$$\begin{aligned}\mathrm{KL}_{s_1}(q||p) &:= \mathrm{KL}(q||p)+\mathrm{KL}(p||q) \quad \text{or}\\ \mathrm{KL}_{s_2}(q||p) &:= \min\{\mathrm{KL}(q||p),\mathrm{KL}(p||q)\}.\end{aligned}$$

Mutual Information

Another approach to identify the similarities between two distributions is the **mutual information**. Suppose we have a joint distribution over two random variables $p(x,y)$. If both variables are conditionally independent (see Section 2.4 on fundamental rules of probabilities), then $p(x,y) = p(x)p(y)$. This hypothesis can be validated using the KL-divergence measure,

$$\begin{aligned}\mathrm{KL}(p(x,y)||p(x)p(y)) &= -\int\int p(x,y)\log\frac{p(x)p(y)}{p(x,y)}dxdy\\ &:= \mathrm{I}(x,y),\end{aligned}$$

which is called the **mutual information**.

2.8 Exercises

Frequentist or Bayesian View

(a) Explain in your own words the difference between the Frequentist and Bayesian view when talking about probabilities. Give an example.

(b) Assume you are playing with a stranger a game where the winner is determined by a coin flip. The coin was brought by our opponent and you are not sure if the coin is loaded, thus not with $p(\text{head} = h) = p(\text{tail} = t) = 0.5$. Use both, the Bayesian View as well as the Frequentist approach to make a statement about the fairness of the coin if the number of tossed heads is $h = 7$ and the number of tossed tails $t = 3$ ($h = 70$, $t = 30$). For the Bayesian View, we use a range of $p(h) = 0.45$ to $p(h) = 0.55$ to define a fair coin and for the Frequentist approach we use a confidence of 0.95%.

advanced exercise

Discrete Random Variables

(c) Let $\mathfrak{X} = \{\text{rainy}, \text{sunny}, \text{cloudy}\}$ be a set of discrete random values with $p(\text{rainy}) = 0.3$ and $p(\text{cloudy}) = 0.5$. Determine the probability $p(\text{sunny})$.

(d) Assume you are playing with a stranger a game where the winner is determined by a coin flip. Assume the coin flip is fair, thus $p(\text{head} = h) = p(\text{tail} = t) = 0.5$. You win, if you are tossing $1, 2, 3, \ldots, n$ heads in a row. How high has to be your profit if the bet is 1 Euro to be a fair game. A fair game means thereby that the profit expectation for both sides is zero. Determine the profit values for $1, 2, 3$ times head in a row and state a general formula for the case of n times head in a row.

Fundamental Rules

(e) Let $\mathfrak{X} = \{\text{rainy}, \text{sunny}, \text{cloudy}\}$ be a set of discrete random values with $p(\text{rainy} = r) = 0.2$, $p(\text{cloudy} = c) = 0.4$ and $p(\text{sunny} = s) = 0.4$. Calculate the Union of $\{\text{rainy}, \text{sunny}\}$. Assume further you take with probability $p(\text{umbrella} = u) = 0.4$ an umbrella with you when you go out. Calculate the Joint probability $p(r, u)$.

(f) Again let $\mathfrak{X} = \{\text{rainy}, \text{sunny}, \text{cloudy}\}$ be a set of discrete random values with $p(\text{rainy} = r) = 0.2$, $p(\text{cloudy} = c) = 0.4$ and $p(\text{sunny} = s) = 0.4$. Now, with conditional probabilities $p(\text{umbrella} = u|r) = 1$, $p(u|c) = 0.7$, $p(u|s) = 0$ you take an umbrella with you when you go out. Calculate the probability $p(u)$.

(g) Imagine you are going to a blood donation. For security reasons a blood probe for every participant is checked for AIDS. The control of your probe using a standard AIDS test gives back a positive result. A standard AIDS test is 99.9% sensitive and 99.7% specific. Thus, it gives back a positive result with a probability of 99.9% for infected persons and a negative result with a probability of 99.7% for non-infected persons. About 0.1% of the german population are infected by AIDS. Calculate the probability that you are really infected with AIDs using the Bayes' Theorem.

Continuous Random Variables

(h) Experience shows that the time between two calls to a call center is approximately exponentially distributed to a certain parameter λ such that the probability density function (pdf) is

$$f_\lambda(t) = \lambda \exp(-\lambda t), \tag{2.10}$$

where $t \geq 0$. The parameter λ corresponds to the average number of calls per time unit and is assumed to be $\lambda = 0.5$. Calculate the

probability that the next call is received $1-2$ time units after the previous one.

Fundamental Discrete Distributions

(i) At the University of Luebeck there is a Applied Robotics class. The class consists of 100 persons where 60 are Europeans (A), 35 Asians (B) and 5 Americans (C). Every day one of the students is randomly chosen to clean the board. What is the probability that 1 European, 3 Asians and 1 American are chosen within one week for cleaning the board?

Fundamental Continuous Distributions

(j) An industrial company wants to analyze its manufacturing line. Therefore, they inspect the probability of the produced items to be defective. From past experience the analyst expects this probability to be equal to 4%. To quantify his uncertainty he attached a standard deviation of 2%. Furthermore, the analyst decides to use a beta distribution to model his uncertainty. How does he have to set the two parameters α, β in order to match his expected values?

(k) Proof that

$$\boldsymbol{\mu}_{a|b} = \boldsymbol{\mu}_a + \boldsymbol{\Sigma}_{ab}\boldsymbol{\Sigma}_{bb}^{-1}(\boldsymbol{x}_b - \boldsymbol{\mu}_b) \text{ and} \tag{2.11}$$

$$\boldsymbol{\Sigma}_{a|b} = \boldsymbol{\Sigma}_{aa} - \boldsymbol{\Sigma}_{ab}\boldsymbol{\Sigma}_{bb}^{-1}\boldsymbol{\Sigma}_{ba}, \tag{2.12}$$

advanced exercise

with $\boldsymbol{\mu} = [\boldsymbol{\mu}_a^T, \boldsymbol{\mu}_b^T]^T$ and $\boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma}_{aa} & \boldsymbol{\Sigma}_{ab} \\ \boldsymbol{\Sigma}_{ba} & \boldsymbol{\Sigma}_{bb} \end{bmatrix}$.

Information Theory

(l) State the four requirements/axioms which lead to the development of the entropy

$$H(x) = -\sum p(x)\log(p(x)). \tag{2.13}$$

advanced exercise

Give for every requirement an example.

Probabilistic Regression

Linear Probabilistic Regression 3

In this chapter we discuss probabilistic basic linear regression methods. We assume unknown functions of the form $\boldsymbol{y} = f(\boldsymbol{x})$. Per definition we will always assume column vectors, if not explicitly stated differently. Thus for the linear regression considered in this chapter, the inputs are denoted by the vector $\boldsymbol{x} \in \mathbb{R}^D$ and scalar outputs $y \in \mathbb{R}^1$. Note that for multi-dimensional outputs *multiple independent regression models* or more complex models like *multivariate Gaussian Processes* [6] may be used.

3.1 Linear Feature Regression . 14
3.2 Basis function models 15
3.3 Logisitic Regression 16
3.4 Maximum Likelihood 16
3.5 Maximum A-Posteriori . . . 18
3.6 Full Bayesian Inference . . . 19
3.7 Predictive Distribution . . . 20
3.8 Exercises 21

3.1 Linear Feature Regression

A simple linear regression model can be constructed by

$$\begin{aligned} y &= w_1 + w_2 x_1 + w_3 x_2 + \cdots + w_{D+1} x_D, \\ y &= \boldsymbol{x}^T \boldsymbol{w}. \end{aligned} \tag{3.1}$$

The first parameter w_1 is often called the *bias term*, the *offset* or the *intercept*. All unknown parameters are denoted by the vector $\boldsymbol{w} = [w_1, w_2, \ldots, W_{D+1}]^T$. The input vector is given by $\boldsymbol{x} = [1, x_1, x_2, \ldots, x_D]^T$, where we added a preceding 1 to the inputs.

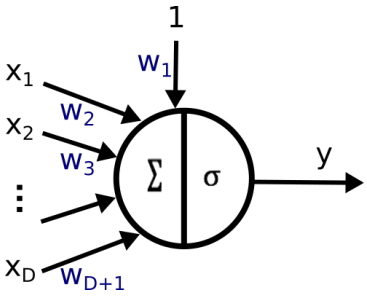

Figure 3.1: The perceptron neuron model computes implements $y = \sigma(w_1 + w_2 x_1 + \cdots + w_{D+1} x_D)$, where the symbol σ denotes the activation function in this artificial neuron model.

The perceptron neuron model. It is important to note that there is a direct relationship to the *perceptron* neuron model, which is shown in Figure 3.1. To model non-linear functions with perceptions, an *activation function* is used, i.e., $y = \sigma'(\boldsymbol{x}^T \boldsymbol{w})$. If a *sigmoid activation* function is used, the regression model implements **logistic regression** which is discussed below.

For a detailed theoretical discussion and a solid introduction we recommend reading the book of Hertz, Krogh and Plamer [7].

[7]: Hertz (2018), *Introduction to the theory of neural computation*

Feature or basis function models. A major drawback of the model in Equation 3.1 is that the number of parameters $\boldsymbol{w}$ is equal to the input dimension plus one offset term, i.e., $D + 1$. Thus for high dimensional inputs and a small number of input samples, the model will overfit to the data. Or in other words, the system of equations is underrepresented.

In practice, a basis function extension of Equation 3.1 is used.

$$y = \phi(\boldsymbol{x})^T \boldsymbol{w}, \tag{3.2}$$

where the function $\boldsymbol{\phi}$ implements a *non-linear* mapping from the D-dimensional input space to a M-dimensional feature space, i.e, $\boldsymbol{\phi}(\boldsymbol{x}) : \mathbb{R}^D \rightarrow \mathbb{R}^M$. However, it is still a *linear regression model*.

Feature transformation with $\boldsymbol{\phi}(\boldsymbol{x}) : \mathbb{R}^D \rightarrow \mathbb{R}^M$

3.2 Basis function models

We will briefly discuss some of the most common basis function models which we will also use in subsequent chapters.

Polynomial basis functions. A polynomial regression model of the order $M-1$ is defined as

$$y = w_1 + w_2 x + w_3 x^2 + w_4 x^3 + \cdots + w_M x^{M-1},$$

where x denotes a scalar variable. Extensions to D-dimensional inputs $\boldsymbol{x} \in \mathbb{R}^D$ may use $D \times M$ parameters w and the derivation is straight forward. The polynomial regression model using the definition in Equation 3.2 can be written as

$$\begin{aligned} y &= \sum_{i=1}^{M} x^{i-1} w_i, \\ y &= \boldsymbol{\phi}(\boldsymbol{x})^T \boldsymbol{w}, \end{aligned}$$

with the *feature vector* $\boldsymbol{\phi}(\boldsymbol{x}) = [1, x, x^2, x^3, \ldots, x^{M-1}]^T$.

Gaussian basis functions. We consider M dimensional basis function models where $\boldsymbol{\phi}(\boldsymbol{x}) = [\phi_1(\boldsymbol{x}), \phi_2(\boldsymbol{x}), \ldots, \phi_M(\boldsymbol{x})]^T$. Each of the individual ϕ_i maps the D-dimensional inputs to a scalar, i.e., $\phi_i(\boldsymbol{x}) : \mathbb{R}^D \rightarrow \mathbb{R}^1$.

For scalar inputs (as in the polynomial fit example above), Gaussian basis function models are defined as

$$\phi_i(x) = \exp^{\{-\frac{1}{2s^2}(x-\mu_i)^2\}},$$

and for vector inputs, multinomial Gaussians are used, where

$$\boldsymbol{\phi}_i(\boldsymbol{x}) = \exp^{\{-\frac{1}{2}\,(\boldsymbol{x}-\boldsymbol{\mu}_i)^T\,\boldsymbol{\Sigma}_i^{-1}\,(\boldsymbol{x}-\boldsymbol{\mu}_i)\}}. \tag{3.3}$$

Note that the model covariance $\boldsymbol{\Sigma}_i$ might be equal for all dimensions $i = 1 \ldots M$.

Sigmoidal basis functions. Another commonly used feature transformation uses the sigmoid function which *maps all real number from* $-\infty$ *to* $+\infty$ *to the interval* $[0, 1]$. The sigmoid function is given by

$$\sigma'(a) = \frac{1}{1 + \exp^{\{-a\}}}. \tag{3.4}$$

The *sigmoidal basis functions* are defined as

$$\phi_i(\boldsymbol{x}) = \sigma'(\frac{\boldsymbol{x} - \boldsymbol{\mu}_i}{\boldsymbol{s}}),$$

where $\boldsymbol{s}$ denotes a *bandwidth* or scaling factor.

3.3 Logisitic Regression

Logistic regression exploits the sigmoid function to map all inputs to the interval $[0, 1]$. By defining a threshold within this interval, a classifier can by build.

Logistic linear regression is defined by

$$y = \sigma'(\boldsymbol{x}^T \boldsymbol{w}),$$

where σ' denotes the sigmoid function in Equation 3.4. Note that the function $\phi(\boldsymbol{x})$ in Equation 3.2 may be in the simple linear model the identity function $\phi(\boldsymbol{x}) = \boldsymbol{x}$.

In a two-class classification problem, the posterior probability of class C_1 is given by

$$p(C_1|\phi(\boldsymbol{x})) = \sigma'(\phi(\boldsymbol{x})^T \boldsymbol{w}),$$

with $p(C_2|\phi(\boldsymbol{x})) = 1 - p(C_1|\phi(\boldsymbol{x}))$, according to the property of random variables in Equation 2.1.

A convenient description of this binary classification problem is based on the *Bernoulli* distribution that we defined in Subsection 2.5. We can define a generative probabilistic model for the binary random variable $y \in 0, 1$ as follows

$$p(y|\boldsymbol{x}, \boldsymbol{w}) = \text{Bern}(y|\sigma'(\phi(\boldsymbol{x})^T \boldsymbol{w})),$$

where the mean of the *Bernoulli* distribution is given by $\boldsymbol{\mu} = \sigma'(\phi(\boldsymbol{x})^T \boldsymbol{w})$.

If we threshold the output probability, we can induce a decision rule. For some new input x^* the predicted class is computed by

$$y^*(x^*) = 1 \iff p(y = 1|x^*, \boldsymbol{w}) > \lambda.$$

The threshold is denoted by λ and we assume the the model parameters $\boldsymbol{w}$ were learned from a training dataset, which we will discuss next.

3.4 Maximum Likelihood

To learn the unknown model parameters we assume a given dataset $D = \{\boldsymbol{X}, \boldsymbol{y}\}$ with n data samples. The D-dimensional inputs are used in the input matrix $\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n] \in \mathbb{R}^{D \times n}$ and output vectors $\boldsymbol{y} \in \mathbb{R}^n$.

Datasets $D = \{\boldsymbol{X}, \boldsymbol{y}\}$ with $\boldsymbol{X} \in \mathbb{R}^{D \times n}$ and $\boldsymbol{y} \in \mathbb{R}^n$

We assume Gaussian additative noise on the outputs where

$$y = \boldsymbol{\phi}(\boldsymbol{x})^T \boldsymbol{w} + \epsilon \text{ with,} \tag{3.5}$$
$$\epsilon \sim \mathcal{N}(0, \sigma). \tag{3.6}$$

The linear basis function model in Equation 3.2 was extended by additative Gaussian noise.

Under the Gaussian noise assumption, we can construct a probabilistic generative model to model a single output y of the dataset D by

$$p(y|\boldsymbol{x}, \boldsymbol{w}) = \mathcal{N}(y|\boldsymbol{\phi}(\boldsymbol{x})^T \boldsymbol{w}, \sigma^2), \tag{3.7}$$

where both vectors, $\boldsymbol{w}$ and $\boldsymbol{\phi}(\boldsymbol{x})$ are M-dimensional column vectors.

Remark 3.4.1 Assuming **identically and independently distributed (i.i.d.) data samples**, the probability of generating the dataset D is given by the *factorial* distribution

$$\begin{aligned} p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{w}) &= \prod_{i=1}^{n} \mathcal{N}(y_i|\boldsymbol{\phi}(\boldsymbol{x}_i)^T \boldsymbol{w}, \sigma^2), & (3.8) \\ &= \mathcal{N}(\boldsymbol{y}|\boldsymbol{A}\boldsymbol{w}, \sigma^2 \boldsymbol{I}), & (3.9) \end{aligned}$$

which the matrix $\boldsymbol{A} = [\boldsymbol{\phi}(\boldsymbol{x}_1), \boldsymbol{\phi}(\boldsymbol{x}_2), \ldots, \boldsymbol{\phi}(\boldsymbol{x}_n)]^T \in \mathbb{R}^{n \times M}$.

The factorial simplifies to a Gaussian distribution due to the **i.i.d. data samples assumption** and the product rule of two **independent** Gaussians where

$$\mathcal{N}(a|\mu_1, \sigma_1)\, \mathcal{N}(b|\mu_2, \sigma_2) = \mathcal{N}\left(\begin{bmatrix} a \\ b \end{bmatrix} | \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}, \begin{bmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{bmatrix} \right).$$

Model Learning. We want to find the optimal parameters $\boldsymbol{w}^*$ that maximizes the likelihood, i.e.,

$$\boldsymbol{w}^* = \arg\max_{\boldsymbol{w}} p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{w}).$$

Instead of directly evaluating the likelihood, it is mathematically more convenient to maximize the log of the likelihood (for Gaussians). Thus, the maximum of the log likelihood can be computed as

$$\begin{aligned} \boldsymbol{w}^*_{ML} &= \arg\max_{\boldsymbol{w}} \{\log\, \mathcal{N}(\boldsymbol{y}|\boldsymbol{A}\boldsymbol{w}, \sigma^2 \boldsymbol{I})\}, \\ &= \arg\max_{\boldsymbol{w}} \{-c - \frac{1}{2\sigma^2}(\boldsymbol{y} - \boldsymbol{A}\boldsymbol{w})^T (\boldsymbol{y} - \boldsymbol{A}\boldsymbol{w})\}, \end{aligned}$$

where we used the definition of a multivariate Gaussian distribution in Equation 2.6. The constant is given by $c = \log(2\pi\sigma^2)^{D/2}$ and is independent of the unknown model parameter w. It will therefore by not considered in the following derivations.

Definition 3.4.1 *The maximization problem above denoted as maximum likelihood is equivalent to the minimization of the Least squares objective J_{LS}, i.e.,*

$$\begin{aligned} \boldsymbol{w}^*_{LS} &= \arg\min_{\boldsymbol{w}} J_{LS}, \\ &= \arg\min_{\boldsymbol{w}} \{\frac{1}{2\sigma^2}(\boldsymbol{y} - \boldsymbol{A}\boldsymbol{w})^T (\boldsymbol{y} - \boldsymbol{A}\boldsymbol{w})\}, \\ &= (\boldsymbol{A}^T \boldsymbol{A})^{-1} \boldsymbol{A}^T \boldsymbol{y}. & (3.10) \end{aligned}$$

To obtain the least squares regression result, we computed the derivative of the objective with respect to the unknown parameters $\boldsymbol{w}$ and set it to zero, i.e.,

$$\frac{\partial J_{LS}}{\partial \boldsymbol{w}} = \frac{\partial}{\partial \boldsymbol{w}}\{1/2\sigma^{-2}(\boldsymbol{y}-\boldsymbol{A}\boldsymbol{w})^T(\boldsymbol{y}-\boldsymbol{A}\boldsymbol{w})\} = 0. \tag{3.11}$$

A detailed derivation of the least squares result can be found in the Appendix B in Section B.1.

Remark 3.4.2 Major drawbacks of the *maximum likelihood solution* are that it may overfit to the data or the matrix $\boldsymbol{A}^T\boldsymbol{A}$ may be singular for small numbers of data samples. Or the matrix may not be invertible if not all dimensions of the matrix have been sufficiently explored.

Ridge Regression. In statistics a common solution is called *ridge regression* which adds the quadratic regularization term $\frac{\lambda}{2}(\boldsymbol{w}^T\boldsymbol{w})$ to the objective J_{LS}. Adding the quadratic loss results in the *regularized least squares* solution

$$\boldsymbol{w}^*_{LS'} = (\boldsymbol{A}^T\boldsymbol{A} + \lambda\boldsymbol{I})^{-1}\boldsymbol{A}^T\boldsymbol{y}. \tag{3.12}$$

3.5 Maximum A-Posteriori

An alternative approach to implement robust linear regression is to put a prior on the unknown parameters $\boldsymbol{w}$. A common prior is a Gaussian with zero mean and precision λ^{-1}. Note that the precision is the inverse of the variance and used for mathematical convenience. We denote the prior by

$$p(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}|0, \lambda^{-1}\boldsymbol{I}). \tag{3.13}$$

To obtain the optimal parameter vector $\boldsymbol{w}^*$ we apply the Bayes rule in Equation 2.2 to compute the posterior

$$p(\boldsymbol{w}|\boldsymbol{X},\boldsymbol{y}) = \frac{p(\boldsymbol{y}|\boldsymbol{X},\boldsymbol{w})p(\boldsymbol{w})}{p(\boldsymbol{y}|\boldsymbol{X})} \propto p(\boldsymbol{y}|\boldsymbol{X},\boldsymbol{w})p(\boldsymbol{w}). \tag{3.14}$$

The *maximum a-posteriori* solution can be obtained through

$$\begin{aligned}
\boldsymbol{w}^*_{MAP} &= \arg\max_{\boldsymbol{w}}\{\log p(\boldsymbol{y}|\boldsymbol{X},\boldsymbol{w}) + \log p(\boldsymbol{w})\},\\
&= \arg\max_{\boldsymbol{w}}\{\log \mathcal{N}(\boldsymbol{y}|\boldsymbol{A}\boldsymbol{w},\sigma^2\boldsymbol{I}) + \log \mathcal{N}(\boldsymbol{w}|0,\lambda^{-1}\boldsymbol{I})\},\\
&= \arg\max_{\boldsymbol{w}}\{-c_1 - \frac{1}{2\sigma^2}(\boldsymbol{y}-\boldsymbol{A}\boldsymbol{w})^T(\boldsymbol{y}-\boldsymbol{A}\boldsymbol{w}) - c_2 - \frac{1}{2}\lambda\boldsymbol{w}^T\boldsymbol{w}\},\\
&= \arg\min_{\boldsymbol{w}}\{\frac{1}{2\sigma^2}(\boldsymbol{y}-\boldsymbol{A}\boldsymbol{w})^T(\boldsymbol{y}-\boldsymbol{A}\boldsymbol{w}) + \frac{1}{2}\lambda\boldsymbol{w}^T\boldsymbol{w}\},\\
&= (\boldsymbol{A}^T\boldsymbol{A} + \sigma^2\lambda\boldsymbol{I})^{-1}\boldsymbol{A}^T\boldsymbol{y}.
\end{aligned} \tag{3.15}$$

Note that the constants c_1 and c_2 are the normalizing factors in the Gaussian multinomial distribution in Equation 2.6 and are independent

of $\boldsymbol{w}$. The final MAP result directly follows from the derivations in Section B.1 in the Appendix.

Remark 3.5.1 A notable difference to **ridge regression** in Equation 3.12 is the different regularization term when computing the inverse $(\boldsymbol{A}^T\boldsymbol{A} + \sigma^2\lambda\boldsymbol{I})^{-1}$. Ridge regression uses λ as regularization parameter which is a result of the additionally introduced cost term $\frac{\lambda}{2}(\boldsymbol{w}^T\boldsymbol{w})$.

The *maximum a-posteriori* solution additionally applies a *data dependent* regularization parameter σ that is a result of the regression model definition of $\mathcal{N}(y|\boldsymbol{\phi}(\boldsymbol{x})^T\boldsymbol{w}, \sigma^2)$ in Equation 3.7. This data dependent regularization strategy increases the regularization with 'larger' observation noise.

3.6 Full Bayesian Inference

Consider the application of the Bayes rule to compute the posterior over the unknown model parameters $\boldsymbol{w}$ in Equation 3.14. To derive the MAP solution we approximated this posterior by the product of the likelihood and the prior over the parameters $\boldsymbol{w}$. In a full Bayesian approach we will solve this product numerically through considering a conjugate prior of the form

$$p(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}|\boldsymbol{m}_0, \boldsymbol{S}_0).$$

The product of the likelihood and the prior in 3.14 is computed as

$$\begin{aligned} p(\boldsymbol{w}|\boldsymbol{X}, \boldsymbol{y}) &\propto p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{w})p(\boldsymbol{w}), \\ &\propto \mathcal{N}(\boldsymbol{y}|\boldsymbol{A}\boldsymbol{w}, \sigma^2\boldsymbol{I})\mathcal{N}(\boldsymbol{w}|\boldsymbol{m}_0, \boldsymbol{S}_0), \\ &= \mathcal{N}(\boldsymbol{w}|\boldsymbol{V}\boldsymbol{S}_0^{-1}\boldsymbol{m}_0 + \sigma^{-2}\boldsymbol{V}\boldsymbol{A}^T\boldsymbol{y}, \boldsymbol{V}), \end{aligned} \tag{3.16}$$

where the posterior covariance is given by $\boldsymbol{V} = \sigma^2(\sigma^2\boldsymbol{S}_0^{-1} + \boldsymbol{A}^T\boldsymbol{A})^{-1}$. Note that to obtain this result we made use of the Gaussian marginal and conditional distributions. For more details on this derivation we refer to [4].

[4]: Bishop (2006), *Pattern recognition and machine learning*

To better visualize the difference to *ridge regression* in Equation 3.12 and to the *maximum a-posteriori* solution in Equation 3.15, we assume a zero mean, scalar variance prior as in Equation 3.13 with $\boldsymbol{m}_0 = 0$ and $\boldsymbol{S}_0 = \lambda^{-1}\boldsymbol{I}$.

$$\begin{aligned} p(\boldsymbol{w}|\boldsymbol{X}, \boldsymbol{y}) &= \mathcal{N}(\boldsymbol{w}|(\lambda\boldsymbol{I} + \sigma^{-2}\boldsymbol{A}^T\boldsymbol{A})^{-1}\sigma^{-2}\boldsymbol{A}^T\boldsymbol{y}, (\lambda\boldsymbol{I} + \sigma^{-2}\boldsymbol{A}^T\boldsymbol{A})^{-1}), \\ &= \mathcal{N}(\boldsymbol{w}|\sigma^2(\boldsymbol{A}^T\boldsymbol{A} + \sigma^2\lambda\boldsymbol{I})^{-1}\sigma^{-2}\boldsymbol{A}^T\boldsymbol{y}, (\lambda\boldsymbol{I} + \sigma^{-2}\boldsymbol{A}^T\boldsymbol{A})^{-1}), \\ &= \mathcal{N}(\boldsymbol{w}|\boldsymbol{w}^*_{MAP}, (\lambda\boldsymbol{I} + \sigma^{-2}\boldsymbol{A}^T\boldsymbol{A})^{-1}), \\ &= \mathcal{N}(\boldsymbol{w}|\boldsymbol{\mu}_{w|y}, \boldsymbol{\Sigma}_{w|y}). \end{aligned} \tag{3.17}$$

where $\boldsymbol{\mu}_{w|y} = \boldsymbol{w}^*_{MAP}$ with

$$\boldsymbol{\mu}_{w|y} = (\boldsymbol{A}^T\boldsymbol{A} + \sigma^2\lambda\boldsymbol{I})^{-1}\boldsymbol{A}^T\boldsymbol{y}, \tag{3.18}$$

$$\boldsymbol{\Sigma}_{w|y} = (\lambda\boldsymbol{I} + \sigma^{-2}\boldsymbol{A}^T\boldsymbol{A})^{-1}. \tag{3.19}$$

Remark 3.6.1 When using a zero mean prior with a scalar variance, the *mean* of the full Bayesian solution is equivalent to the *maximum a-posteriori* solution. However, in the full Bayesian approach, we additionally compute an estimate of the parameter uncertainty. This uncertainty estimate will become important when computing predictions.

The parameter mean is denoted by $\boldsymbol{\mu}_{w|y}$ in Equation 3.18. The parameter uncertainty $\boldsymbol{\Sigma}_{w|y}$, defined in Equation 3.19, is important for computing predictions which we will discuss next.

3.7 Predictive Distribution

In practice we are often interested in making predictions about potential outputs y^* given some test input sample $\boldsymbol{x}*$. Such predictions can be computed by evaluating the *predictive posterior distribution* defined by

$$\begin{aligned} p(y^*|\boldsymbol{x}^*, \boldsymbol{X}, \boldsymbol{y}) &= \int p(y^*|\boldsymbol{x}^*, \boldsymbol{w}) p(\boldsymbol{w}|\boldsymbol{X}, \boldsymbol{y}) d\boldsymbol{w}. \\ &= \int \mathcal{N}(y^*|\boldsymbol{\phi}(\boldsymbol{x}^*)^T \boldsymbol{w}, \sigma_y^2) \mathcal{N}(\boldsymbol{w}|\boldsymbol{\mu}_{w|y}, \boldsymbol{\Sigma}_{w|y}) d\boldsymbol{w}. \end{aligned}$$

Definition 3.7.1 *The convolution of the two Gaussian distributions can be solved through applying the Gaussian identity [8]*

$$\int \mathcal{N}(\boldsymbol{y}|\boldsymbol{F}\boldsymbol{x}, \sigma_y^2 \boldsymbol{I}) \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_x, \boldsymbol{\Sigma}_x) d\boldsymbol{x} = \mathcal{N}(\boldsymbol{y}|\boldsymbol{F}\boldsymbol{\mu}_x, \sigma_y^2 \boldsymbol{I} + \boldsymbol{F}\boldsymbol{\Sigma}_x \boldsymbol{F}^T). \quad (3.20)$$

Note that this convolution is solved through first computing the joint distribution of the product and thereafter the marginal.

When using parameter posterior in Equation 3.17 and the likelihood in Equation 3.8 we obtain the predictive distribution of

$$p(y^*|\boldsymbol{x}^*, \boldsymbol{X}, \boldsymbol{w}) = \mathcal{N}\left(y^*|\boldsymbol{\phi}(\boldsymbol{x}^*)^T \boldsymbol{\mu}_{w|y},\ \sigma_y^2 + \boldsymbol{\phi}(\boldsymbol{x}^*)^T \boldsymbol{\Sigma}_{w|y} \boldsymbol{\phi}(\boldsymbol{x}^*)\right), \quad (3.21)$$

where the mean $\boldsymbol{\mu}_{w|y}$ and the covariance $\boldsymbol{\Sigma}_{w|y}$ of the parameter posterior are given in Equations 3.18 and 3.19.

Remark 3.7.1 The variance of the predictive distribution depends on two terms. First on the test data sample noise model by σ_y and on the model uncertainty denoted by $\boldsymbol{\phi}(\boldsymbol{x}^*)^T \boldsymbol{\Sigma}_{w|y} \boldsymbol{\phi}(\boldsymbol{x}^*)$. Note that the second term translates into a variance that depends on how 'close' the new test sample is to the training data used to learn the model in Equation 3.17. For large training datasets this term converges to zero and the σ_y dominates the uncertainty estimate.

We conclude this chapter with an overview over the discussed linear regression methods in Figure 3.2.

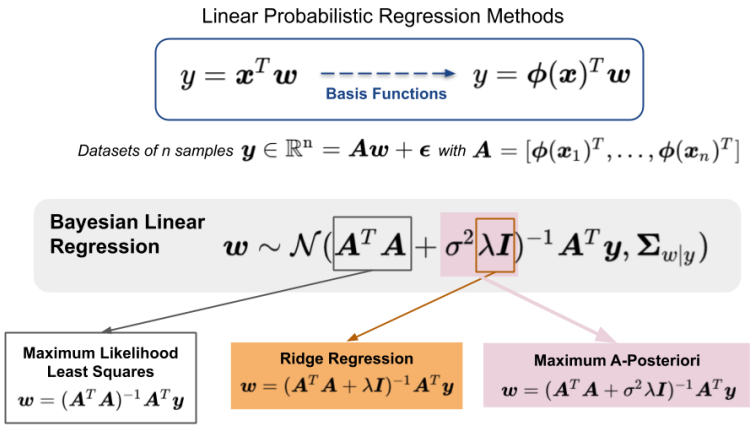

Figure 3.2: This overview of the discussed probabilistic linear regression methods which are all special cases of Bayesian Linear Regression.

3.8 Exercises

Maximum Likelihood

(a) Consider an online learning scenario where data samples arrive in sequence. Derive an online least least squares approach that applies updates of the form $w^{\tau+1} = w^{\tau} - \eta \Delta w$. The symbol η denotes the learning rate, τ the learning iteration and Δw denotes the weight update. The approach is also known as the *least-mean-squares* approach.

(b) Extend the linear basis function model to multi-dimensional or multi-variate outputs, where the matrix $W = [w_1, \ldots, w_{D_y}] \in \mathbb{R}^{M \times D_y}$ denotes the unknown model parameters and D_y the dimension of the outputs. Derive the *maximum likelihood* solution and show that each column in W is computed through the least squares regression in Equation 3.10.

Ridge Regression

(c) Derive the maximum likelihood solution with an objective that implements the additional term $\frac{\lambda}{2}(w^T w)$.

(d) Derive regularized online updates for $w^{\tau+1}$ as in the sequential model learning task in Exercise (a).

Maximum A-posteriori

(e) Derive the MAP solution following the principles in Section B.1 in the Appendix.

(f) Demonstrate in a synthetic dataset with large induced noise how the prior influences the model predictions based on the number of data samples. Evaluate the corresponding mean squared error with respect to the training dataset size.

Full Bayesian Regression

(g) Compute the predictive posterior for the prior distribution $p(w) = \mathcal{N}(w|m_0, S_0)$.

(h) Show that Equation 3.21 is a special case of the resulting model from Exercise (g).

Nonlinear Probabilistic Regression

4

4.1 Gaussian Processes 22
4.2 GP with built-in GMRF . . 24
4.3 Exercises 26

4.1 Gaussian Processes

In this section, we will briefly explain and derive Gaussian Processes. For more details we refer to [9].

Multivariate Conditional Distribution

In order to understand Gaussian Processes first consider a multivariate normal distribution as introduced in section 2.6 with $\boldsymbol{x}' = \{x'_1, \ldots, x'_k\}$

$$f(\boldsymbol{x}'|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{\sqrt{(2\pi)^k |\boldsymbol{\Sigma}|}} \exp\left(-\frac{1}{2}(\boldsymbol{x}' - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}' - \boldsymbol{\mu})\right), \tag{4.1}$$

which can be written as

$$\boldsymbol{x}' \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}). \tag{4.2}$$

Figure 4.1 shows such a normal distribution. By partition the Gaussian random vector $\boldsymbol{x}'$ into $\boldsymbol{x}$ and $\boldsymbol{y}$, where both are jointly Gaussian random vectors, the term Equation 4.2 becomes

$$\begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \boldsymbol{\mu}_x \\ \boldsymbol{\mu}_y \end{bmatrix}, \begin{bmatrix} \boldsymbol{A} & \boldsymbol{C} \\ \boldsymbol{C}^\top & \boldsymbol{B} \end{bmatrix}\right). \tag{4.3}$$

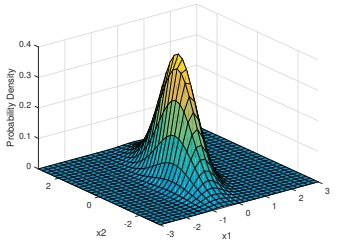

Figure 4.1: The figure shows a two dimensional Gaussian Distribution created using Matlab with $\boldsymbol{\mu}$ = [0 0] and $\boldsymbol{\Sigma}$ = [0.2 0.2; 0.2 1.0].

The marginal distribution of $\boldsymbol{x}$ is

$$\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}_x, \boldsymbol{A}), \tag{4.4}$$

and the conditional distribution of $\boldsymbol{x}$ given $\boldsymbol{y}$ is

$$\boldsymbol{x}|\boldsymbol{y} \sim \mathcal{N}(\boldsymbol{\mu}_x + \boldsymbol{C}\boldsymbol{B}^{-1}(\boldsymbol{y} - \boldsymbol{\mu}_y), \boldsymbol{A} - \boldsymbol{C}\boldsymbol{B}^{-1}\boldsymbol{C}^\top).$$

Thus the conditional expectation (denoted by the symbol $\mathbb{E}$) and the covariance matrix (denoted by the symbol Σ) can be written as

$$\begin{aligned} \mathbb{E}(\boldsymbol{x}|\boldsymbol{y}) &= \boldsymbol{\mu}_x + \boldsymbol{C}\boldsymbol{B}^{-1}(\boldsymbol{y} - \boldsymbol{\mu}_y), \\ \Sigma_{\boldsymbol{x}|\boldsymbol{y}} &= \boldsymbol{A} - \boldsymbol{C}\boldsymbol{B}^{-1}\boldsymbol{C}^\top. \end{aligned} \tag{4.5}$$

Derivation of Gaussian Processes

Gaussian Processes are probabilistic models which can be used to estimate, on the basis of known data, a mean and a variance for an unknown data point. They use the relation given in Equation 4.5. Consider a training set $T = \{(x_i, y_i)\} = (X, \boldsymbol{y})$ where $\boldsymbol{x}_i$ denotes an input vector of dimension D and y_i denotes a scalar output. Here, X contains all the input data and $\boldsymbol{y}$ all the output (target) data of the training set. In order to determine expectation values for $\boldsymbol{y}_*$, the values $\boldsymbol{y}$ at the states $X = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$

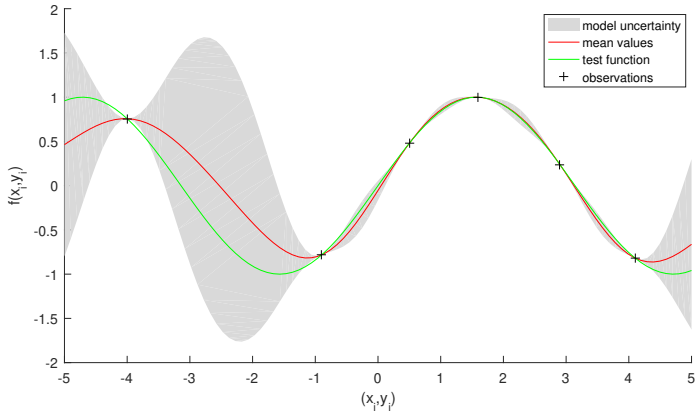

Figure 4.2: Example for a one dimensional Gaussian Process. The test function is a sinus; Six observation points have been used to determine an estimation result (mean values) and the model uncertainty (variance).

have to be known. Rewriting equation Equation 4.3 with the predefined training set yields

$$\begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{y}_* \end{bmatrix} \sim N\left(\begin{bmatrix} \boldsymbol{\mu}(X) \\ \boldsymbol{\mu}(X_*) \end{bmatrix}, \begin{bmatrix} \boldsymbol{K}(X,X) & \boldsymbol{K}(X,X_*) \\ \boldsymbol{K}(X_*,X) & \boldsymbol{K}(X_*,X_*) \end{bmatrix} \right), \tag{4.6}$$

where $\boldsymbol{\mu}(\cdot)$ are the expectation values given a certain input set and $\boldsymbol{K}(\cdot,\cdot)$ a covariance matrix determined through two input sets. The expectation values and the variance of the unknown values $\boldsymbol{y}_*$ can be derived with Equation 4.5, as

$$\begin{aligned} \mathbb{E}(\boldsymbol{y}_*|\boldsymbol{y},X,X_*) &= \boldsymbol{\mu}_* + \boldsymbol{K}_*^\top \boldsymbol{K}^{-1}(\boldsymbol{y}-\boldsymbol{\mu}) \,, \\ \Sigma_{\boldsymbol{y}_*|\boldsymbol{y},X,X_*} &= \boldsymbol{K}_{**} - \boldsymbol{K}_*^\top \boldsymbol{K}^{-1}\boldsymbol{K}_*, \end{aligned} \tag{4.7}$$

where $\boldsymbol{\mu}(X) = \boldsymbol{\mu}$, $\boldsymbol{\mu}(X_*) = \boldsymbol{\mu}_*$, $\boldsymbol{K}(X,X) = \boldsymbol{K}$, $\boldsymbol{K}(X,X_*) = \boldsymbol{K}_*$ and $\boldsymbol{K}(X_*,X_*) = \boldsymbol{K}_{**}$. The mean function $\boldsymbol{\mu}(\cdot)$ can be generated using proper information of the target values y_i given the input $\boldsymbol{x}_i$ or, if non such information is available, can be set to zero. The covariance function (kernel function) defines nearness or similarity and the covariance matrix $\boldsymbol{K}$ has to be positive definite. Possible types of such kernels are

- $k = k(\boldsymbol{x}-\boldsymbol{x}')$ (stationary, invariant to translation in the input space).
- $k = k(||\boldsymbol{x}-\boldsymbol{x}'||)$ (isotropic, invariant to all rigid motions).
- $k = k(\boldsymbol{x}\cdot\boldsymbol{x}')$ (dot product, invariant to rotation).

A commonly used kernel function is the squared-exponential covariance function $k_{\text{se}}(\tau) = \sigma_f^2 \exp\left(-\frac{\tau^2}{2l^2}\right)$ with $\tau = ||\boldsymbol{x}-\boldsymbol{x}'||$. The variables σ_f and l are hyperparameters. This kernel is smooth because the function is infinitely differentiable. The properties of a kernel around $\mathbf{0}$ determine the smoothness of the stationary process. A one dimensional example of a Gaussian Process is shown in Figure 4.2.

Noisy Observations

Are the given data, thus the training set $T = \{(\boldsymbol{x}_i, y_i)\} = (X, \boldsymbol{y})$, corrupted with white noise, e.g. $y_i = z(\boldsymbol{x}_i) + \epsilon_i$ with $\epsilon_i \sim N(0, \sigma_n^2)$, it can be taken into account by adding $\sigma_n^2\ \boldsymbol{I}$ to the covariance matrix $\boldsymbol{K}$. The matrix $\boldsymbol{I}$ describes an identity matrix with convenient size. Equation Equation 4.6

becomes

$$\begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{y}_* \end{bmatrix} \sim N\left(\begin{bmatrix} \boldsymbol{\mu}(\mathrm{X}) \\ \boldsymbol{\mu}(\mathrm{X}_*) \end{bmatrix}, \begin{bmatrix} \boldsymbol{K} + \sigma_n^2 \boldsymbol{I} & \boldsymbol{K}_* \\ \boldsymbol{K}_*^\top & \boldsymbol{K}_{**} \end{bmatrix} \right) ,$$

with

$$\begin{aligned} \mathbb{E}(\boldsymbol{y}_* | \boldsymbol{y}, X, X_*) &= \boldsymbol{\mu}_* + \boldsymbol{K}_*^\top (\boldsymbol{K} + \sigma_n^2 \boldsymbol{I})^{-1} (\boldsymbol{y} - \boldsymbol{\mu}) , \\ \Sigma_{\boldsymbol{y}_* | \boldsymbol{y}, X, X_*} &= \boldsymbol{K}_{**} - \boldsymbol{K}_*^\top (\boldsymbol{K} + \sigma_n^2 \boldsymbol{I})^{-1} \boldsymbol{K}_* . \end{aligned} \tag{4.8}$$

Meta Learning for Parameter Adjustment

Assuming zero mean ($\boldsymbol{\mu}(\cdot) = 0$), in equation Equation 4.8 only the parameters of the kernel function $\boldsymbol{\theta}$ are unknown. Either one can choose these parameters by trial and error or use Bayesian Optimization to find optimal parameters for a certain problem. The aim of Bayesian Optimization is to find parameters $\boldsymbol{\theta}$ which maximize equation Equation 4.1. The logarithm of the likelihood,

$$\log(f(\boldsymbol{y}|X, \boldsymbol{\theta})) = -\frac{1}{2} \boldsymbol{y}^\top \boldsymbol{K}^{-1} \boldsymbol{y} - \frac{1}{2} \log||\boldsymbol{K}|| - \frac{n}{2} \log(2\pi) ,$$

is used for maximization. In order to use optimization methods, such as gradient descent, the derivation of the above log likelihood has to be computed

$$\frac{\partial \log(f(\boldsymbol{y}|X, \boldsymbol{\theta}))}{\partial \theta_j} = \frac{1}{2} \mathrm{tr}\left((\boldsymbol{\alpha}\boldsymbol{\alpha}^\top - \boldsymbol{K}^{-1}) \frac{\partial \boldsymbol{K}}{\partial \theta_j} \right) ,$$

with $\boldsymbol{\alpha} = \boldsymbol{K}^{-1} \boldsymbol{y}$ and n the number of data points in the training set, is required.

4.2 GP with built-in GMRF

This section deals with the construction of a Gaussian Process with built-in Gaussian Markov Random Field. This method can lead to an immense reduction in computational effort compared to Gaussian Processes but leads to worse estimation results. Nevertheless, the reduction in computational effort makes this method interesting for the utilization in mobile robotic applications, such as drones. A detailed overview of the theory for Gaussian Processes with built-in Gaussian Markov Random Fields can be found in [10].

Spatial Field

First, consider a Gaussian Markov Random Field with respect to a graph $S = (V, E)$ and $F = \{f(\boldsymbol{p}_1), \ldots, f(\boldsymbol{p}_m)\}^\top \sim \mathcal{N}(0, \boldsymbol{Q}^{-1})$. As shown $Q_{ij} \neq 0 \Leftrightarrow \{i, j\} \in E$ (Equation 5.3) or, in other words, $f(\boldsymbol{p}_i)$ and $f(\boldsymbol{p}_j)$ are not conditionally independent.

A spatial field can be modelled as

$$z(\boldsymbol{x}) = \mu(\boldsymbol{x}) + \sum_{j=1}^{m} \lambda(\boldsymbol{x}, \boldsymbol{p}_j) f(\boldsymbol{p}_j) ,$$

which is a Gaussian Process with a built-in Gaussian Markov Random Field with a weighting function $\lambda(\cdot,\cdot)$. The weighting function can be seen as a similarity function (e.g. squared exponential kernel). The vertices of the graph S, $\{\boldsymbol{p}_1,\ldots,\boldsymbol{p}_m\}$, are called generating points. The number and the position of the generating points can be determined using a model selection criterion and the log likelihood method.
The advantage of such an approach is, that there are many tuning knobs, such as a different number of generating points or a different structure of the precision matrix. Thus, the model can be adjusted to different problems using this tuning knobs.

In order to develop an estimation scheme using the given spatial field, assume there are observations $\boldsymbol{y} = (y_1,\ldots,y_n)^\top$ at points $\boldsymbol{x}_1,\ldots,\boldsymbol{x}_n$ given. The observations are noisy with $y_i = z(\boldsymbol{x}_i)+\epsilon_i$ with $\epsilon_i \sim N(0,\sigma_w^2)$, thus an independent and identically distributed Gaussian white noise. The covariance matrix for $\boldsymbol{y}$ is

$$\boldsymbol{C} = \boldsymbol{\Lambda}\boldsymbol{Q}^{-1}\boldsymbol{\Lambda}^\top + \sigma_w^2\boldsymbol{I}\ , \tag{4.9}$$

and the covariance between $\boldsymbol{y}$ and $z_0(\boldsymbol{s})$, where $\boldsymbol{s}$ is a certain point of interest, is

$$\boldsymbol{k} = \boldsymbol{\Lambda}\boldsymbol{Q}^{-1}\boldsymbol{\lambda}. \tag{4.10}$$

Here $\boldsymbol{\Lambda} \in \mathbb{R}^{n\times m}$ is a matrix given by $(\boldsymbol{\Lambda})_{ij} = \lambda(\boldsymbol{x}_i,\boldsymbol{p}_j)$ and $\boldsymbol{\lambda} \in \mathbb{R}^{m\times 1}$ is a vector given by $(\boldsymbol{\lambda})_i = \lambda(\boldsymbol{p}_i,\boldsymbol{s})$. A proof can be found in [10, p. 79, 80].

With the foregoing conclusions, it is possible to make a prediction by using the terms for Gaussian Process Regression as shown in section Equation 4.1. By inserting the covariance matrices from Equation 4.9 and Equation 4.10 into Equation 4.7, where $\boldsymbol{K} = \boldsymbol{C}$ and $\boldsymbol{K}_* = \boldsymbol{k}$,

$$\begin{aligned}\mathbb{E}(z_0|\boldsymbol{y}) &= \mu(\boldsymbol{s}) + (\boldsymbol{\Lambda}\boldsymbol{Q}^{-1}\boldsymbol{\lambda})^\top(\boldsymbol{\Lambda}\boldsymbol{Q}^{-1}\boldsymbol{\Lambda}^\top + \sigma_w^2\boldsymbol{I})^{-1}(\boldsymbol{y}-\boldsymbol{\mu}(X))\ ,\\ \Sigma_{z_0|\boldsymbol{y}} &= \boldsymbol{\lambda}^\top\boldsymbol{Q}^{-1}\boldsymbol{\lambda} - (\boldsymbol{\Lambda}\boldsymbol{Q}^{-1}\boldsymbol{\lambda})^\top(\boldsymbol{\Lambda}\boldsymbol{Q}^{-1}\boldsymbol{\Lambda}^\top + \sigma_w^2\boldsymbol{I})^{-1}(\boldsymbol{\Lambda}\boldsymbol{Q}^{-1}\boldsymbol{\lambda})\ ,\end{aligned} \tag{4.11}$$

can be derived. Using the *Woodbury Matrix Identity* (see Marc Toussaint's Gaussian identities [8]), the Equations 4.11 transform into

$$\begin{aligned}\mathbb{E}(z_0|\boldsymbol{y}) &= \mu(\boldsymbol{s}) + \boldsymbol{\lambda}^\top\hat{\boldsymbol{Q}}^{-1}\hat{\boldsymbol{y}}\ ,\\ \Sigma_{z_0|\boldsymbol{y}} &= \boldsymbol{\lambda}^\top\hat{\boldsymbol{Q}}^{-1}\boldsymbol{\lambda}\ ,\end{aligned}$$

with

$$\begin{aligned}\hat{\boldsymbol{Q}} &= \boldsymbol{Q} + \sigma_w^{-2}\boldsymbol{\Lambda}^\top\boldsymbol{\Lambda}\ ,\\ \hat{\boldsymbol{y}} &= \sigma_w^{-2}\boldsymbol{\Lambda}^\top(\boldsymbol{y}-\boldsymbol{\mu}(X))\ .\end{aligned}$$

A proof can be found in [10, p. 81–82]. Comparing the estimation method with the Gaussian Processes presented in section 4.1, a difference in the matrices which have to be inverted can be seen. For the Gaussian Process a matrix $\boldsymbol{K} \in \mathbb{R}^{n\times n}$ has to be inverted, whereas for the GP with built-in GMRF a matrix $\boldsymbol{Q} \in \mathbb{R}^{m\times m}$ has to be inverted. The computational complexity, without calculating the inverse, grows linearly with the number of observations n. In combination with the computational effort for the matrix inversion it can be shown that Gaussian Processes have a computational complexity of $\mathcal{O}(n^3)$, whereas GP with built-in GMRF have a computational complexity of $\mathcal{O}(nm^2)$. In most cases the number of generating points m will be chosen at the beginning of the process and

then remains constant. On the other hand the number of observations n may rise with time. Thus, the computational complexity may grow fast for Gaussian Processes considering the factor n^3.

4.3 Exercises

Gaussian Processes

(a) Briefly describe the role that Kernel functions play in modelling Gaussian processes. Why are the Kernel functions used in Gaussian Processes required to be positive definite?

(b) For $x \in [0, 5]$, compute the Covariance function of a Gaussian Process using three different Kernels:

$$k1(x1, x2) = \exp\left(-\frac{||x_1 - x_2||^2}{2}\right) ,$$

$$k1(x1, x2) = \left(x_1^\top x_2 + 2\right) ,$$

$$k1(x1, x2) = \exp\left(-|x_1 - x_2|\right) ,$$

Then, sample a set of 10 different latent functions f according to the Gaussian process prior:

$$f \sim \mathcal{N}(0, K) ,$$

for all three different Kernel/Covariance Matrices and plot them. What are the differences?

GP with built-in GMRF

(c) A precision matrix required for the GP with built-in GMRF has to be positive definite, thus $\boldsymbol{Q} > 0$. State the definition for positive definite matrices. Further proof, that if a Matrix $\boldsymbol{A}$ is symmetrical and strictly diagonal dominant and all diagonal elements of $\boldsymbol{A}$ are positive, then $\boldsymbol{A}$ is positive definite.

(d) Assume you have a regular lattice with a distance between the vertices of 1. Design precision matrices for taking into account

  (a) neighbouring vertices with distance $r \leq 1$.
  (b) neighbouring vertices with distance $r \leq 2$.

(e) Proof for $\sigma_w = 0$ that the GP with built-in GMRF transform into

$$\mathbb{E}(z_0|\boldsymbol{y}) = \mu(\boldsymbol{s}) + \boldsymbol{\lambda}^\top(\boldsymbol{\Lambda}^\top\boldsymbol{\Lambda})^{-1}\boldsymbol{\Lambda}^\top(\boldsymbol{y} - \boldsymbol{\mu}(X)) ,$$
$$\Sigma_{z_0|\boldsymbol{y}} = 0 .$$

Probabilistic Inference

Markov Models 5

This section on Markov models was written by *Nils Rottmann*. It discusses Markov Chains, Random Fields, and Gaussian Markov Random Fields. Further details to Markov Random Fields can be found in [11] and [12]. Gaussian Markov Random Fields are discussed in detail in [13].

5.1 Markov Chains 28
5.2 Markov Random Fields . . . 29
5.3 Gaussian Markov Random Fields 29
5.4 Variogram 30
5.5 Exercises 30

5.1 Markov Chains

A *Markov chain* is a representation of a sequence of variables $(x_1, x_2, \dots)$ which are specified by the conditionals $p(x_i|x_{i-1}, \dots, x_1)$. A simple *Markov chain* example models weather forecasts which can be either sunny or rainy, such that $x_i \in L = \{\text{sunny}, \text{rainy}\}$. The weather on day i can be influenced by the weather of many foregoing days but, for the simplest case, it would be only directly related to the weather of day $i-1$. Such a *Markov chain* is shown in Figure 5.1. The underlying assumption is called a first-order Markov assumption

$$p(x_i|x_{i-1}, \dots, x_1) = p(x_i|x_{i-1}).$$

In the same way a n-th-order Markov assumption can be described as

$$p(x_i|x_{i-1}, \dots, x_1) = p(x_i|x_{i-1}, \dots, x_{i-n}).$$

Of course, even in the first-order *Markov chain* all foregoing variables are

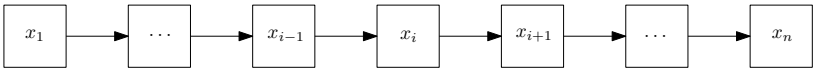

Figure 5.1: The figure shows a directed *Markov chain* where the foregoing events have an influence on the following events.

linked implicitly with x_i. As for our example, the conditional probabilities can be described as

$$P = \begin{bmatrix} p(\text{sunny}|\text{sunny}) & p(\text{rainy}|\text{sunny}) \\ p(\text{sunny}|\text{rainy}) & p(\text{rainy}|\text{rainy}) \end{bmatrix} = \begin{bmatrix} 0.9 & 0.1 \\ 0.5 & 0.5 \end{bmatrix},$$

which can also be represented in a graph as shown in Figure 5.2. Continuing with the example and assuming the weather on day one to be sunny gives $X_1 = [p(\text{sunny}) \quad p(\text{rainy})] = [1 \quad 0]$. Calculating the conditional probabilities for day two leads to

$$X_2 = X_1 P = \begin{bmatrix} 0.9 & 0.1 \end{bmatrix},$$

and for day $1 + n$ to

$$X_{1+n} = X_1 \, P^n \, .$$

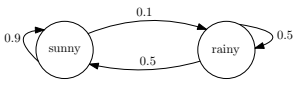

Figure 5.2: A Markov Graph with a first-order Markov assumption. The conditional probabilities are written on the arrows.

5.2 Markov Random Fields

A Markov Random Field is described by an undirected graph $S = (V, E)$, where $V = \{1, \dots, n\}$ are the vertices and E the edges of the graph, in which the vertices and edges are related via a neighbouring system

$$N = \{N_i | \forall i \in V\}.$$

Here N_i is a set of vertices neighbouring i. The properties for such a neighbouring system are

(1) $i \notin N_i$,
(2) $i \in N_j \leftrightarrow j \in N_i$.

For a regular lattice such a neighbouring system could be described as

$$N_i = \{j \in V | d(x_j, x_i) \leq r; j \neq i\},$$

where x_j and x_i are the positions of the vertices j and i and $d(\cdot, \cdot)$ a distance measure. In Figure 5.3 examples of such a neighbouring system are shown. Other (irregular) neighbouring systems are possible.

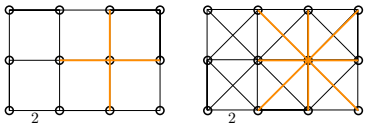

Figure 5.3: Two systems with a regular lattice and $r = 2$ (left) and $r = 3$ (right). The yellow lines lead to the neighbours of the central point. They represent the edges. The distance between two nearest points, vertical and horizontal, is 2.

A *Markov Random Field* can be defined as a family of random variables $F = \{F_i, \dots, F_n\}$, which are defined on the set of vertices V. Each variable F_i takes a value $f_i \in L$, where L is a label set (e.g. $L = \{\text{sunny}, \text{rainy}\}$) and the probability of F taking the value f_i is

$$p(F_i = f_i) = p(f_i).$$

For a Markov Random Field on V the following two properties have to hold:

(1) $p(f) > 0 \quad \forall f \in F$, (Positivity)
(2) $p(f_i | f_{V-\{i\}}) = p(f_i | N_i)$. (Markovianity)

The positivity is assumed for technical reasons and normally no problem in practice. The *Markovianity* depicts that, given all neighbours to i, the non-neighbours do not have any influence on the probability of i. In other words, labels without edges in between have to be conditionally independent, thus the equation

$$p(x_i, x_j | \boldsymbol{x}_{-\{ij\}}) = p(x_i | \boldsymbol{x}_{-\{ij\}}) p(x_j | \boldsymbol{x}_{-\{ij\}}) ,$$

has to be fulfilled if x_i and x_j are no neighbours.

5.3 Gaussian Markov Random Fields

A *Gaussian Markov Random Field* has the same structure as a *Markov Random Field* adding one restriction, such that the probability distribution of $F = \{F_i, \dots, F_n\}$ is normally distributed with mean $\boldsymbol{\mu}$ and a covariance matrix $\boldsymbol{\Sigma}$. The conditional independence between pairwise unconnected vertices, thus the information of the graph S, can be found in the parameters of the normal distribution $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$, thus $p(x_i) \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. Since the mean does not have any influence on the conditional independence properties of $\boldsymbol{F}$ only the covariance matrix $\boldsymbol{\Sigma}$ remains. Considering the inverse covariance

matrix instead of the covariance matrix, the so called precision matrix $\boldsymbol{Q} = \boldsymbol{\Sigma}^{-1}$, it turns out that

$$p(x_i, x_j|\boldsymbol{x}_{-\{ij\}}) = p(x_i|\boldsymbol{x}_{-\{ij\}})p(x_j|\boldsymbol{x}_{-\{ij\}}) \quad \Longleftrightarrow \quad \boldsymbol{Q}_{ij} = 0.$$

A proof of this property can be found in [13, p. 22]. These important property of Gaussian Markov Random Fields makes the precision matrix sparse, which leads to an immense reduction of computational effort.

5.4 Variogram

The *variogram* is used in spatial statistics to describe the degree of spatial dependence of a spatial field. Thus, in case of an estimation problem, it can give information about which dimensions have a great influence on the estimation process. Such information can give an idea how to set the hyperparameters of a certain covariance function and which type of covariance function should be chosen.

For simplicity only the two dimensional field is considered. Assume a vector $\boldsymbol{h}$ which has a length $||\boldsymbol{h}|| = l$ and a direction α (angle). Given n pairs of data separated by $\boldsymbol{h}$, the experimental *semi-variogram* can be calculated for the distance l and angle α by

$$\gamma(\boldsymbol{h}) = \gamma(l, \alpha) = \frac{1}{2n} \sum_{i=1}^{n} (y(\boldsymbol{x}_i + \boldsymbol{h}) - y(\boldsymbol{x}_i))^2 ,$$

where $y(\boldsymbol{x})$ is the value at position $\boldsymbol{x}$.
This is the expectation for the squared increment of the values between locations $\boldsymbol{x}$ and $\boldsymbol{z}$:

$$\gamma(\boldsymbol{x}, \boldsymbol{z}) = \mathbb{E}\left[(y(\boldsymbol{x}) - y(\boldsymbol{z}))^2\right].$$

Thus, the lower the value of the *variogram*, the higher is the spatial dependence between points in $\boldsymbol{h}$ direction.

The methods used for the construction of the *variogram* depend highly on the structure of the given data. In this thesis, only aligned data within a regularly spaced grid are considered. Therefore, four main directions, α_1, α_2, α_3 and α_4, with four main distances, l_1, l_2, l_3 and l_4 are used (see Figure 5.4).

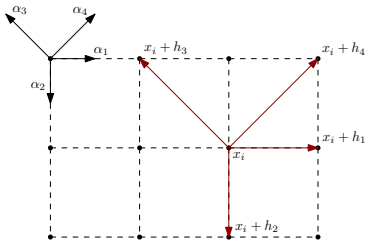

Figure 5.4: The figure shows how the data is arranged by calculating the experimental *semi-variogram*.

5.5 Exercises

(a) A Markov Model shall be used for the weather forecast. Therefore the three states rainy, sunny and cloudy are defined. The transition propabilities for the state transition from state s_0 to state s_1 in one time step can be found in the table below. Sketch the complete Markov Model (inclusive self-return) and calculate the transfer matrix $\boldsymbol{P}$ which can be used to transfer the state $\boldsymbol{x}_i$ into the state $\boldsymbol{x}_{i+1}$ for $\boldsymbol{x} = [\text{rainy}, \text{sunny}, \text{cloudy}]^\top$.

$$\boldsymbol{x}_{i+1} = \boldsymbol{P}\boldsymbol{x}_i$$

Using the transfer matrix $\boldsymbol{P}$ calculate the steady state of the system, thus the state for $i \to \infty$.

s_0	s_1	$p(s_1\|s_0)$
rainy	sunny	0.5
rainy	cloudy	0.1
sunny	rainy	0.2
sunny	cloudy	0.5
cloudy	rainy	0.8
cloudy	sunny	0.1

(b) Proof that

$$p(x_i, x_j|\boldsymbol{x}_{-\{ij\}}) = p(x_i|\boldsymbol{x}_{-\{ij\}})p(x_j|\boldsymbol{x}_{-\{ij\}}) \quad \Longleftrightarrow \quad \boldsymbol{Q}_{ij} = 0.$$

with $\boldsymbol{Q} = \boldsymbol{\Sigma}^{-1}$ being the precision matrix for a GMRF.

Probabilistic Time Series Models | 6

6.1 Time Series Data 32
6.2 Single Time Step Model . . 33
6.3 Multi-Time Step Models . . 34
6.4 Trajectory Prediction or Completion 35
6.5 Exercises 36

In this chapter, we will introduce a probabilistic time series model that is based on a linear feature regression (see Section 3.1).

Definition 6.0.1 *Time series are defined as a series of data points indexed or listed in time order. We distinguish between two categories of analysis approaches: (i) time-domain models and (ii) frequency-domain models. In this chapter, we will discuss a time-domain model and for frequency domain approaches we refer to spectral or wavelet analyses methods discussed in [14].*

Another important feature of a time series model is how correlations in multi-dimensional datasets are represented. In decoupled approaches, independent models are learned for each dimensions. As an example consider independent Gaussian Processes (see Section 4.1) for the position of a quadcopter. Here for each dimension of Cartesian coordinates in x-y-z individual GPs are trained.

In contrast in coupled time series models, the correlation between multiple input dimensions is captured by the model. This coupling can be exploited to predict for example time series data of the x-coordinate give the y-coordinate in a quadcopter flight. Or this powerful feature can be used to predict human arm motions solely from observing the motion of the shoulders [15]. For exampling in sports like boxing, humans can anticipate future actions from observing only a few milliseconds of shoulder movements.

In the following, we will introduce such a coupled time series model. However, for simplicity, we will initially only consider a one-dimensional dataset and will generalize the model to multi-dimensional datasets at the end of this chapter.

6.1 Time Series Data

We consider data of the form $D = \{y_{1,1}, y_{2,1}, \ldots, y_{T,n}\}$ where T denotes the number of time steps and n the number of demonstrations or *trajectories*. We denote a trajectory by $\boldsymbol{\tau}$, which is given by a sequence of observations y, i.e., $\boldsymbol{\tau}_j = [y_{1,j}, \ldots, y_{T,j}]$. For the moment we assume scalar observations ($y \in \mathbb{R}^1$) and that all trajactories $j = 1 \ldots n$ have the length T.

Our goal is to build a probabilistic time series model from these n trajectories denoted by $p(\boldsymbol{\tau})$.

6.2 Single Time Step Model

For a single time step, we define a Gaussian linear feature model of the form

$$p(y_t|\boldsymbol{w}) = \mathcal{N}(y_t|\boldsymbol{\phi}_t^T \boldsymbol{w}, \sigma_y^2) \ , \tag{6.1}$$

where $\boldsymbol{\phi}_t$ denotes non-linear basis functions for the time step $t \in [1, T]$. The feature vector $\boldsymbol{w}$ will be learned for example through least squares regression and σ_y denotes the standard deviation of the observations. This noise parameter will be used later to balance how much we trust new observations compared to a learned model.

Remark 6.2.1 The same model was used for *static* data in Equation 3.7. Note that we assume here additive Gaussian noise with zero mean (see Equations 3.5).

Radial Basis Functions. We consider *radial basis functions* that implement the function: $\boldsymbol{\phi}_t : \mathbb{R}^1 \rightarrow \mathbb{R}^M$. *Radial basis functions* are Gaussians basis functions as in Section3.2, however, here the exponential functions are **arranged in time**. An illustration of $M = 5$ radial basis functions is shown in Figure 6.1.

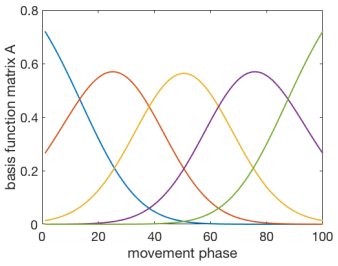

Figure 6.1: Shown are five *normalized* basis functions arranged in the movement phase interval $[0, 1]$.

Let i denote the i-th of the M basis functions, where

$$\phi_t^i = \exp^{\{-\frac{1}{2h}(z_t - c_i)^2\}} \ .$$

For the linear regression model in Equation 6.1, we compute the normalized radial basis function vector:

$$\boldsymbol{\phi}_t = \frac{1}{\sum_{i=1}^M \phi_t^i} [\phi_t^1, \phi_t^2, \ldots, \phi_t^M] \ . \tag{6.2}$$

The *bandwidth* of the Gaussian is denoted by h. In practice, it is often computed by

$$h = \frac{1}{2}(c_{i+1} - c_i)^2 \ ,$$

which automatically adapts the *bandwidth* to different basis function numbers M.

The Movement Phase, is denoted by the variable z_t and implements a mapping from the discrete time steps $t \in [1, T]$ to a movement phase where $z_t \in [0, 1]$. In the linear case,

$$z_t = \frac{1}{T-1}(t-1) \ .$$

The movement phase can be scaled by a multiplicative factor to any movement duration regardless of the model representations, e.g., to 5 seconds or 5 minutes. More complex movement phase implementations might exploit the sigmoid function which *maps all real number from* $-\infty$ *to* $+\infty$ *to the interval* $[0, 1]$ (see Equation 3.4).

6.3 Multi-Time Step Models

For multiple time steps, we assume that the observations in $\boldsymbol{\tau}_j = [y_{1,j}, \ldots, y_{T,j}]$ are **i.i.d.**. Note that a similar assumption was made when we derived the Bayesian linear regression model in Equation 3.8.

We define the distribution of the trajectory τ by

$$\begin{aligned} p(\boldsymbol{\tau}_j|\boldsymbol{w}) &= \prod_{t=1}^{T} p(y_{t,j}|\boldsymbol{w}) \,, \\ &= \prod_{t=1}^{T} \mathcal{N}(y_{t,j}|\boldsymbol{\phi}_t^T \boldsymbol{w}, \sigma_y^2) \,, \\ &= \mathcal{N}(\boldsymbol{\tau}_j|\boldsymbol{A}\boldsymbol{w}, \sigma_y^2 \boldsymbol{I}) \,. \end{aligned}$$

The matrix $\boldsymbol{A}$ is given by

$$\boldsymbol{A} = [\boldsymbol{\phi}_1, \boldsymbol{\phi}_2, \ldots, \boldsymbol{\phi}_T]^T \in \mathbb{R}^{T \times M} \,. \tag{6.3}$$

Model Learning. The unknown parameters $\boldsymbol{w}^j$ can be learned for example in the simplest case through least squares regression. We derived this result for static datasets in Equation 3.10. For our time series model, $\boldsymbol{w}_{LS}^j = (\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{\tau}_j$.

The distribution $p(\boldsymbol{\tau}|\boldsymbol{w})$ (without index j) is a conditional probability distribution that follows from the Bayes rule in Section 2.4, i.e., $p(A|B) = p(A,B)/p(B)$. Our goal is to model a distribution over trajectories $p(\tau)$ which can be obtained by computing the joint distribution $p(\boldsymbol{\tau}, \boldsymbol{w})$ and marginalizing over the parameters $\boldsymbol{w}$. Thus,

$$p(\boldsymbol{\tau}) = \int p(\boldsymbol{\tau}|\boldsymbol{w}) p(\boldsymbol{w}) d\boldsymbol{w}.$$

Note that the model is represented by the prior distribution $p(\boldsymbol{w})$.

Remark 6.3.1 For Gaussian distributions this integral can be solved analytically where

$$\begin{aligned} p(\boldsymbol{\tau}) &= \int \mathcal{N}(\boldsymbol{\tau}|\boldsymbol{A}\boldsymbol{w}, \sigma_y^2 \boldsymbol{I}) \, \mathcal{N}(\boldsymbol{w}|\boldsymbol{\mu}_{w|y}, \boldsymbol{\Sigma}_{w|y}) \, d\boldsymbol{w}, & (6.4) \\ &= \mathcal{N}(\boldsymbol{\tau} \mid \boldsymbol{A}\boldsymbol{\mu}_{w|y}, \; \sigma_y^2 \boldsymbol{I} + \boldsymbol{A}\boldsymbol{\Sigma}_{w|y}\boldsymbol{A}^T). & (6.5) \end{aligned}$$

The model prior $\mathcal{N}(\boldsymbol{w}|\boldsymbol{\mu}_{w|y}, \boldsymbol{\Sigma}_{w|y})$ can be computed from n demonstrations or trajectories with

$$\begin{aligned} \boldsymbol{\mu}_{w|y} &= \frac{1}{n} \sum_{j=1}^{n} \boldsymbol{w}^j \quad \text{and} \,, \\ \boldsymbol{\Sigma}_{w|y} &= \frac{1}{n-1} \sum_{j=1}^{n} (\boldsymbol{w}^j - \boldsymbol{\mu}_{w|y})(\boldsymbol{w}^j - \boldsymbol{\mu}_{w|y})^T \,. \end{aligned}$$

An important benefit of this probabilistic time series model is that we can tradeoff the importance of a learned prior model compared to new

observations. We will now derive probabilistic operations for *trajectory prediction* or **completion**.

6.4 Trajectory Prediction or Completion

Given some new observations denoted by $\boldsymbol{o} = [o_{t_1}, o_{t_2}, \dots]$, our goal is to predict a trajectory $\boldsymbol{\tau}^o$ that completes the missing observations. An example is shown in Figure 6.2.

The algorithm implements the following four steps:

1. Compute the basis function matrix at the time steps of the observations with $\boldsymbol{A}^o = [\boldsymbol{\phi}_1, \boldsymbol{\phi}_2, \dots]^T$.
2. Compute the parameter posterior $p(\boldsymbol{w}^o) := \mathcal{N}(\boldsymbol{w}^o | \boldsymbol{\mu}_{w|o}, \boldsymbol{\Sigma}_{w|o})$ using

$$\begin{aligned} \boldsymbol{\mu}_{w|o} &= \boldsymbol{\mu}_{w|y} + \boldsymbol{L}(\boldsymbol{o} - \boldsymbol{A}^o \boldsymbol{\mu}_{w|y}) \text{ and }, \\ \boldsymbol{\Sigma}_{w|o} &= \boldsymbol{\Sigma}_{w|y} - \boldsymbol{L}\, \boldsymbol{A}^o \boldsymbol{\Sigma}_{w|y} \text{ and }, \\ \boldsymbol{L} &= \boldsymbol{\Sigma}_{w|y} \boldsymbol{A}^{oT} (\sigma_o \boldsymbol{I} + \boldsymbol{A}^o \boldsymbol{\Sigma}_{w|y} \boldsymbol{A}^{oT})^{-1} . \end{aligned}$$

3. Predict or complete the observation trajectory through

$$p(\boldsymbol{\tau}^o) = N(\boldsymbol{\tau}^o \mid \boldsymbol{A}^o \boldsymbol{\mu}_{w|o},\ \sigma_y^2 \boldsymbol{I} + \boldsymbol{A}^o \boldsymbol{\Sigma}_{w|o} \boldsymbol{A}^{oT}).$$

In Step 2, the *parameter posterior* given the observation *likelihood* and the *prior* is computed using the Bayes rule, i.e.,

$$\begin{aligned} p(\boldsymbol{w}^o) &\propto \mathcal{N}(\boldsymbol{o} | \boldsymbol{A}^o \boldsymbol{w}, \sigma_o \boldsymbol{I}) \mathcal{N}(\boldsymbol{w} | \boldsymbol{\mu}_{w|y}, \boldsymbol{\Sigma}_{w|y}), \\ &:= \mathcal{N}(\boldsymbol{w}^o | \boldsymbol{\mu}_{w|o}, \boldsymbol{\Sigma}_{w|o}) . \end{aligned}$$

In Step 3, the marginal is computed using Equation 6.4.

Note that the observation time steps $t_1, t_2, \dots$ just need to be within the interval $[1, T]$ and can be used to model missing data, e.g., $t_1 = 10$, $t_2 = 50$.

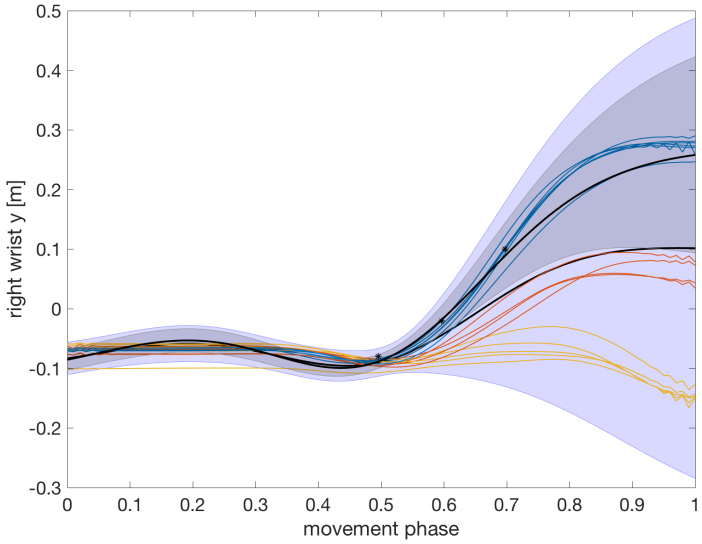

Figure 6.2: Shown are human right arm reaching motions to three different targets denoted by the colored lines. The blue shaded area and the solid black line denote the learn prior distribution. **Given three observations** denoted by the black dots, a trajectory is predicted. This prediction passes through the observations when σ_o is small and is close the the prior for large values.

6.5 Exercises

The goal of this exercise is to extend the probabilistic time series model in Section 6 to multi-dimensional data. Consider a D-dimensional trajectory of the form $\boldsymbol{\tau} = [y_{1,1}, y_{2,1}, \ldots, y_{T,1}, y_{1,2}, y_{2,2}, \ldots, y_{T,D}]$. Such a $\mathbb{R}^{TD}$ dimensional vector can be generated with

$$\boldsymbol{\tau} = \boldsymbol{A}\boldsymbol{w} + \boldsymbol{\epsilon},$$

where ϵ denotes the noise term. The basis function matrix has the dimension $\boldsymbol{A} \in \mathbb{R}^{TD \times MD}$ and the feature vector $\boldsymbol{w} \in \mathbb{R}^{MD}$ contains of $M \cdot D$ unknown parameters.

(a) Define and implement $\boldsymbol{A} \in \mathbb{R}^{TD \times MD}$ for D-dimensional data by extending the basis function $\boldsymbol{A}$ in Equation 6.3 for D-dimensional systems.
(b) Implement the probabilistic trajectory model in Equation 6.4 in MATLAB for the $D = 2$ dimensional dataset.
(c) Visualize the learned prior $p(\boldsymbol{w})$. Are the left wrist and right wrist in the prior correlated?
(d) Predict the complete left wrist trajectory from observing the three right wrist positions provided in the MATLAB script. Visualize the result for both wrists.
(e) How does the prediction change with an increasing observation uncertainty?

Appendix

Notation A

We follow the *International Conference on Learning Representations* (ICLR) attempt to a standardized notation based on the textbook, *Deep Learning* [1] available at ICLR Notation.

[1]: Goodfellow et al. (2016), *Deep Learning*

A.1 Numbers and Arrays

a	A scalar (integer or real)
$\boldsymbol{a}$	A vector
$\boldsymbol{A}$	A matrix
$\mathbf{A}$	A tensor
$\boldsymbol{I}_n$	Identity matrix with n rows and n columns
$\boldsymbol{I}$	Identity matrix with dimensionality implied by context
$\boldsymbol{e}^{(i)}$	Standard basis vector $[0, \dots, 0, 1, 0, \dots, 0]$ with a 1 at position i
$\text{diag}(\boldsymbol{a})$	A square, diagonal matrix with diagonal entries given by $\boldsymbol{a}$
a	A scalar random variable
$\mathbf{a}$	A vector-valued random variable
$\mathbf{A}$	A matrix-valued random variable

A.2 Sets and Graphs

$\mathbb{A}$	A set
$\mathbb{R}$	The set of real numbers
$\{0, 1\}$	The set containing 0 and 1
$\{0, 1, \dots, n\}$	The set of all integers between 0 and n
$[a, b]$	The real interval including a and b
$(a, b]$	The real interval excluding a but including b
$\mathbb{A} \backslash \mathbb{B}$	Set subtraction, i.e., the set containing the elements of $\mathbb{A}$ that are not in $\mathbb{B}$
$\mathcal{G}$	A graph
$Pa_{\mathcal{G}}(\mathrm{x}_i)$	The parents of x_i in $\mathcal{G}$

A.3 Indexing

a_i	Element i of vector $\boldsymbol{a}$, with indexing starting at 1
a_{-i}	All elements of vector $\boldsymbol{a}$ except for element i
$A_{i,j}$	Element i, j of matrix $\boldsymbol{A}$
$\boldsymbol{A}_{i,:}$	Row i of matrix $\boldsymbol{A}$
$\boldsymbol{A}_{:,i}$	Column i of matrix $\boldsymbol{A}$
$\mathsf{A}_{i,j,k}$	Element (i, j, k) of a 3-D tensor $\mathbf{A}$
$\mathbf{A}_{:,:,i}$	2-D slice of a 3-D tensor
a_i	Element i of the random vector $\mathbf{a}$

A.4 Calculus

$\frac{dy}{dx}$	Derivative of y with respect to x
$\frac{\partial y}{\partial x}$	Partial derivative of y with respect to x
$\nabla_{\boldsymbol{x}} y$	Gradient of y with respect to $\boldsymbol{x}$
$\nabla_{\boldsymbol{X}} y$	Matrix derivatives of y with respect to $\boldsymbol{X}$
$\nabla_{\mathbf{X}} y$	Tensor containing derivatives of y with respect to $\mathbf{X}$
$\frac{\partial f}{\partial \boldsymbol{x}}$	Jacobian matrix $\boldsymbol{J} \in \mathbb{R}^{m \times n}$ of $f : \mathbb{R}^n \to \mathbb{R}^m$
$\nabla_{\boldsymbol{x}}^2 f(\boldsymbol{x})$ or $\boldsymbol{H}(f)(\boldsymbol{x})$	The Hessian matrix of f at input point $\boldsymbol{x}$
$\int f(\boldsymbol{x}) d\boldsymbol{x}$	Definite integral over the entire domain of $\boldsymbol{x}$
$\int_{\mathbb{S}} f(\boldsymbol{x}) d\boldsymbol{x}$	Definite integral with respect to $\boldsymbol{x}$ over the set $\mathbb{S}$

A.5 Probability and Information Theory

$P(\mathrm{a})$	A probability distribution over a discrete variable
$p(\mathrm{a})$	A probability distribution over a continuous variable, or over a variable whose type has not been specified
$\mathrm{a} \sim P$	Random variable a has distribution P
$\mathbb{E}_{\mathrm{x}\sim P}[f(x)]$ or $\mathbb{E}f(x)$	Expectation of $f(x)$ with respect to $P(\mathrm{x})$
$\mathrm{Var}(f(x))$	Variance of $f(x)$ under $P(\mathrm{x})$
Σ or $\mathrm{Cov}(f(x), g(x))$	Covariance of $f(x)$ and $g(x)$ under $P(\mathrm{x})$
$H(\mathrm{x})$	Shannon entropy of the random variable x
$D_{\mathrm{KL}}(P\|Q)$	Kullback-Leibler divergence of P and Q
$\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$	Gaussian distribution over $\boldsymbol{x}$ with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$

A.6 Functions

$f : \mathbb{A} \to \mathbb{B}$	The function f with domain $\mathbb{A}$ and range $\mathbb{B}$
$f \circ g$	Composition of the functions f and g
$f(\boldsymbol{x}; \boldsymbol{\theta})$	A function of $\boldsymbol{x}$ parametrized by $\boldsymbol{\theta}$. (Sometimes we write $f(\boldsymbol{x})$ and omit the argument $\boldsymbol{\theta}$ to lighten notation)
$\log x$	Natural logarithm of x
$\sigma(x)$	Logistic sigmoid, $\dfrac{1}{1+\exp(-x)}$
$\zeta(x)$	Softplus, $\log(1+\exp(x))$
$\|\|\boldsymbol{x}\|\|_p$	L^p norm of $\boldsymbol{x}$
$\|\|\boldsymbol{x}\|\|$	L^2 norm of $\boldsymbol{x}$
x^+	Positive part of x, i.e., $\max(0, x)$
$\mathbf{1}_{\mathrm{condition}}$	is 1 if the condition is true, 0 otherwise

A.7 The Greek Alphabet

Character	Name	Character	Name
α	alpha *AL-fuh*	ν	nu *NEW*
β	beta *BAY-tuh*	ξ, Ξ	xi *KSIGH*
γ, Γ	gamma *GAM-muh*	o	omicron *OM-uh-CRON*
δ, Δ	delta *DEL-tuh*	π, Π	pi *PIE*
ϵ	epsilon *EP-suh-lon*	ρ	rho *ROW*
ζ	zeta *ZAY-tuh*	σ, Σ	sigma *SIG-muh*
η	eta *AY-tuh*	τ	tau *TOW (as in cow)*
θ, Θ	theta *THAY-tuh*	υ, Υ	upsilon *OOP-suh-LON*
ι	iota *eye-OH-tuh*	ϕ, Φ	phi *FEE, or FI (as in hi)*
κ	kappa *KAP-uh*	χ	chi *KI (as in hi)*
λ, Λ	lambda *LAM-duh*	ψ, Ψ	psi *SIGH, or PSIGH*
μ	mu *MEW*	ω, Ω	omega *oh-MAY-guh*

Capitals shown are the ones that differ from Roman capitals.

Mathematical background and derivations

B

In this section, we discuss some basic concepts that will be used in the course. For many of the derivations we make use of *matrix calculus rules*. A comprehensive discussion such rules can be found in *The Matrix Cookbook* by [16].

[16]: Petersen et al. (2008), 'The matrix cookbook'

Simple derivation of a matrix pseudo inverse. Let's assume a simple system of equations given by, a system matrix $\boldsymbol{A} \in \mathbb{R}^{n \times m}$, a *unknown* model vector $\boldsymbol{x} \in \mathbb{R}^m$ and a vector $\boldsymbol{b} \in \mathbb{R}^m$.

For a symmetric matrix $\boldsymbol{A}$ with n equals m, the unknown vector can be compute as $\boldsymbol{x} = \boldsymbol{A}^{-1}\boldsymbol{b}$.

For the common case where $n \neq m$,

$$\begin{aligned} \boldsymbol{A}\boldsymbol{x} &= \boldsymbol{b} && \mid \boldsymbol{A}^T, \\ \boldsymbol{A}^T\boldsymbol{A}\boldsymbol{x} &= \boldsymbol{A}^T\boldsymbol{b} && \mid (\boldsymbol{A}^T\boldsymbol{A})^{-1}, \\ \boldsymbol{x} &= (\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{b}, \\ \boldsymbol{x} &= \boldsymbol{A}^{\dagger}\boldsymbol{b}. \end{aligned}$$

The upper script † (with the latex command *dagger*) denotes the **pseudo inverse**,

$$\boldsymbol{A}^{\dagger} = (\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{b}. \tag{B.1}$$

B.1 Derivation of the Least Squares Solution

Here we provide a detailed derivation of the least squares objective in Equation 3.11. Note that the dimensions of the vectors are $\boldsymbol{y} \in \mathbb{R}^n$, $\boldsymbol{A} \in \mathbb{R}^{n \times M}$ and $\boldsymbol{w} \in \mathbb{R}^M$ is a column vector.

which is given by

$$\begin{aligned} \frac{\partial J_{LS}}{\partial \boldsymbol{w}} &= \frac{\partial}{\partial \boldsymbol{w}}\{1/2\sigma^{-2}(\boldsymbol{y} - \boldsymbol{A}\boldsymbol{w})^T(\boldsymbol{y} - \boldsymbol{A}\boldsymbol{w})\}, \\ &= \frac{\partial}{\partial \boldsymbol{w}}\{1/2\sigma^{-2}(\boldsymbol{y}^T - \boldsymbol{w}^T\boldsymbol{A}^T)(\boldsymbol{y} - \boldsymbol{A}\boldsymbol{w})\}, \\ &= 1/2\sigma^{-2}\frac{\partial}{\partial \boldsymbol{w}}\{\boldsymbol{y}^T\boldsymbol{y} - \boldsymbol{w}^T\boldsymbol{A}^T\boldsymbol{y} - \boldsymbol{y}^T\boldsymbol{A}\boldsymbol{w} + \boldsymbol{w}^T\boldsymbol{A}^T\boldsymbol{A}\boldsymbol{w}\}, \\ &= 1/2\sigma^{-2}\frac{\partial}{\partial \boldsymbol{w}}\{\boldsymbol{y}^T\boldsymbol{y} - (\boldsymbol{y}^T\boldsymbol{A}\boldsymbol{w})^T - \boldsymbol{y}^T\boldsymbol{A}\boldsymbol{w} + \boldsymbol{w}^T\boldsymbol{A}^T\boldsymbol{A}\boldsymbol{w}\}, \\ &= 1/2\sigma^{-2}\frac{\partial}{\partial \boldsymbol{w}}\{\boldsymbol{y}^T\boldsymbol{y} - 2\boldsymbol{y}^T\boldsymbol{A}\boldsymbol{w} + \boldsymbol{A}^T\boldsymbol{w}\boldsymbol{w}^T\boldsymbol{A}\}, \end{aligned}$$

since $\boldsymbol{y}^T\boldsymbol{A}\boldsymbol{w}$ is a *scalar* (with the dimensions $1 \times n$ times $n \times M$ times $M \times 1$), it holds that and $\boldsymbol{y}^T\boldsymbol{A}\boldsymbol{w} = (\boldsymbol{y}^T\boldsymbol{A}\boldsymbol{w})^T$. The partial derivative can be computed as

$$\frac{\partial J_{LS}}{\partial \boldsymbol{w}} = 1/2\sigma^{-2}\{-2\boldsymbol{y}^T\boldsymbol{A} + 2\boldsymbol{w}^T\boldsymbol{A}^T\boldsymbol{A}\},$$

where we made use of the matrix derivatives $\partial/\partial \boldsymbol{w}\{\boldsymbol{y}^T\boldsymbol{A}\boldsymbol{w}\} = \boldsymbol{y}^T\boldsymbol{A}$ and $\partial/\partial \boldsymbol{w}\{\boldsymbol{w}^T\boldsymbol{A}^T\boldsymbol{A}\boldsymbol{w}\} = 2\boldsymbol{w}^T\boldsymbol{A}^T\boldsymbol{A}$.

To compute the least squares minimum, we zero the partial derivative result and solve for the parameter vector $\boldsymbol{w}$, i.e.,

$$\begin{aligned}
\frac{\partial J_{LS}}{\partial \boldsymbol{w}} &= 0, \\
1/2\sigma^{-2}\{-2\boldsymbol{y}^T\boldsymbol{A} + 2\boldsymbol{w}^T\boldsymbol{A}^T\boldsymbol{A}\} &= 0 && \mid \sigma^2, \\
-\boldsymbol{y}^T\boldsymbol{A} + \boldsymbol{w}^T\boldsymbol{A}^T\boldsymbol{A} &= 0 && \mid^T, \\
-\boldsymbol{A}^T\boldsymbol{y} + \boldsymbol{A}^T\boldsymbol{A}\boldsymbol{w} &= 0^T && \mid (\boldsymbol{A}^T\boldsymbol{A})^{-1}, \\
\boldsymbol{w} &= (\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{y}.
\end{aligned}$$

Note that we assume here that $\boldsymbol{A}^T\boldsymbol{A}$ has full rank and is invertible. The final result computes the pseudo inverse $\boldsymbol{A}^\dagger = (\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T$ as in the example above.

Bibliography

Here are the references in citation order.

[1] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. http://www.deeplearningbook.org. MIT Press, 2016 (cited on pages 3, 38).

[2] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018 (cited on page 3).

[3] Frank Ruskey and Mark Weston. 'A survey of Venn diagrams'. In: *Electronic Journal of Combinatorics* 4 (1997), p. 3 (cited on page 5).

[4] Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006 (cited on pages 8, 9, 19).

[5] Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012 (cited on page 8).

[6] Matthias Seeger. 'Gaussian processes for machine learning'. In: *International journal of neural systems* 14.02 (2004), pp. 69–106 (cited on page 14).

[7] John A Hertz. *Introduction to the theory of neural computation*. CRC Press, 2018 (cited on page 14).

[8] Marc Toussaint. 'Lecture notes: Gaussian identities'. In: *a A* 1 (2011). https://ipvs.informatik.uni-stuttgart.de/mlr/marc/notes/gaussians.pdf, p. 2 (cited on pages 20, 25).

[9] Christopher KI Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*. Vol. 2. 3. MIT press Cambridge, MA, 2006 (cited on page 22).

[10] Yunfei Xu et al. *Bayesian Prediction and Adaptive Sampling Algorithms for Mobile Sensor Networks: Online Environmental Field Reconstruction in Space and Time*. Springer, 2015 (cited on pages 24, 25).

[11] Andrew Blake, Pushmeet Kohli, and Carsten Rother. *Markov Random Fields for vision and image processing*. MIT Press, 2011 (cited on page 28).

[12] Stan Z Li. *Markov random field modeling in image analysis*. Springer Science & Business Media, 2009 (cited on page 28).

[13] Havard Rue and Leonhard Held. *Gaussian Markov Random Fields: theory and applications*. CRC Press, 2005 (cited on pages 28, 30).

[14] William WS Wei. 'Time series analysis'. In: *The Oxford Handbook of Quantitative Methods in Psychology: Vol. 2*. 2006 (cited on page 32).

[15] Elmar Rueckert et al. 'Probabilistic Movement Models Show that Postural Control Precedes and Predicts Volitional Motor Control'. In: *Nature Publishing Group: Scientific Reports* 6.28455 (June 2, 2016). DOI: 10.1038/srep28455. published (cited on page 32).

[16] Kaare Brandt Petersen, Michael Syskind Pedersen, et al. 'The matrix cookbook'. In: *Technical University of Denmark* 7.15 (2008), p. 510 (cited on page 42).

Alphabetical Index

preface, v