Technical University of Munich

Department of Mathematics

Interventional Causal Structure Learning With Gaussian Process Regression

Master's Thesis

by

Johannes Michael Schmitt
Matriculation Number: - - -
Study Program: Mathematics in Data Science

Supervisor and Examiner: Prof. Dr. Mathias Drton

Date of Submission (electronically): 15^{th} of October 2020

Titel (Deutsch): Interventionelles Lernen kausaler Strukturen mit Gauß-Prozess-Regression

Hiermit erkläre ich, dass ich die Masterarbeit selbstständig angefertigt und nur die angegebenen Quellen verwendet habe.

München, den 15. Oktober 2020

Johannes Schmitt

Abstract

We review the SCM(Structural Causal Model)-view of causal modeling, which features functional dependencies between direct causes and direct effects. In our framework these functions are modeled in the Gaussian Process Regression setting in order to capture nonlinear functions and employ a flexible, non-parametric regression method that works well in the small data setting, which arises when admitting interventions that are assumed to be scarce. Further, we explore how these interventions can be utilized to extract the causal structure while simultaneously learning the functional dependencies.

Acknowledgments

I would like to thank my supervisor Prof. Drton for his interest, support and valuable input. Furthermore, I would like to thank my parents for their encouragement and my housemates for the positive atmosphere that helped a lot to endure these strange times of uncertainty.

Contents

1 Introduction

The very recent paper of von Kügelgen et al. (2019) - parts of which inspired this thesis - explores the problem of causal discovery, i.e. learning an underlying causal graph structure along with functional dependencies between variables as indicated by the graph structure. The setting they chose will also become our environment to investigate approaches of achieving the goal of causal discovery: we consider a

- small number of variables
- small number of data points, i.e. no "Big Data"
- situation, where interventions, which in practice amount to real experiments, are possible

Additionally, we prefer a setting of few assumptions on functional dependencies between, or distributions of variables or at the very least different assumptions than those common in the prevailing literature.

In this context, von Kügelgen et al. (2019)[pp. 1] mention the possible application scenario of experimental scientific discovery. In general, science or any other effort to understand the inner workings of processes that yield data, which we can observe, is a prime candidate for utilizing the concepts and methods of statistical causality (see Chapter 2).

As Gaussian Process Regression is a compelling technique for the purpose of modeling functional dependencies under the conditions we listed above, we decide to examine this method (see Chapter 3). The idea that this technique fits our objective particularly well is evidenced for example by efforts of Duvenaud (2014) to construct a machinery of automatic model construction with Gaussian Processes going in the direction of a vision labelled "Automatic Statistician" Duvenaud (2014)[p. 6], i.e. the idea of feeding data accompanied with minimal a priori knowledge to such an automatic process that in turn yields advanced statistical information, e.g. about regression problems.

In Chapter 4 we combine results from the previous chapters in order to explore ways to estimate the causal structure in conjuncture with addressing the question of estimating functional dependencies between variables. Lastly, Chapter 5 presents simulations of these devices.

1.1 Notation And Basic Definitions

We adapt the style of Peters et al. (2017).

Abbreviations	
p.*, pp.*	page *, page * and following [may be relative to just the cited article when in a collection]
ch.	chapter
wrt.	"with respect to"
iff	"if and only if"
General	
$:=$	"defined as"
$\forall$	"for all"
$\exists$	"it exists"
$\mathbb{R}, \mathbb{R}_+, \mathbb{N}$	real values, positive real values, natural numbers
x	Value in $\mathbb{R}$
$\mathbf{x}$, $\mathbf{x}_i$	Vector in $\mathbb{R}^d$, for some $d \in \mathbb{N}$, and i^{th}-value of the vector
$\lvert \cdot \rvert$	Absolute value for numeric arguments, cardinality for sets
$\mathbb{1}_{\mathcal{S}}$	Indicator function (0-1) for set $\mathcal{S}$
I	Identity matrix
det	Determinant of a matrix

Probability	
$(\Omega, \mathcal{A}, \mathbb{P})$	General background **probability space** with Ω, set of all outcomes ω, further, **σ-field** $\mathcal{A}$ as collection of possible events and **probability measure** $\mathbb{P}$, which is also the joint distribution of all variables considered
X	**Random Variable**, i.e. function: $\Omega \to \mathbb{R}$
$\mathbb{P}_X$	**Push-forward measure** of X
$p(x)$	**Density** of random variable X
$\mathbf{X}$	**Random Vector**, i.e. vector of random variables, access to i^{th} coordinate via $\mathbf{X}_i$
$p(\mathbf{x}), p(x_1, ..., x_n)$	**Density** of n random vector $\mathbf{X}$, **Joint Density** of random variables $X_1, ..., X_n$, identical for $\mathbf{X} = (X_1, ..., X_n)$
$\bigotimes_{i=1}^{n} \mathbb{P}_X$	**Product measure** of n iid. instances of random variable X
$\mathbb{E}[X]$	**Expectation** of random variable X
$\mathbb{V}[X]$	**Variance** of random variable X
$\mathbb{E}[X\|Y]$	**Conditional expectation** of random variable X, given $\sigma(Y)$
$\mathbb{V}[X\|Y]$	**Conditional variance** of random variable X, given $\sigma(Y)$
$\mathbb{P}[\mathcal{S}\|Y]$	**Conditional probability** of set $\mathcal{S} \subseteq \Omega$, given $\sigma(Y)$
$p(x\|Y = y)$	**Conditional density** of random variable X, given $Y = y$, shortened by $p(x\|y)$
$X \perp\!\!\!\perp Y$	X is **independent** of Y
$X \perp\!\!\!\perp Y\|Z$	X is **conditionally independent** of Y **given** Z
$\sim$	"with distribution"
iid.	"independent identically distributed"
cdf	cumulative distribution function
$\mathcal{N}(\mu, \sigma^2)$	(Density of) normal distribution (univariate) with mean μ and variance σ^2
$\mathcal{N}(\mathbf{x}\|\boldsymbol{\mu}, \Sigma)$	Density (as function of $\mathbf{x}$) of normal distribution (multivariate) with mean vector $\boldsymbol{\mu}$ and covariance matrix Σ

Graph Theory	
$\mathcal{G} = (\mathbf{V}, E)$	**Graph** $\mathcal{G}$ with the set of **nodes/vertices** $\mathbf{V} = \{v_1, ..., v_d\}$ and the set of **edges** $E \subset \mathbf{V} \times \mathbf{V}$ from start to end node, where we exclude edges with the same start and end node, i.e. (v_i, v_i); two nodes v_i, v_j are *adjacent* if $(v_i, v_j) \in E$ or $(v_j, v_i) \in E$; $\mathcal{G}$ is *fully connected* if all pairs of nodes are adjacent; we call an edge *undirected* if we have both $(v_i, v_j) \in E$ and $(v_j, v_i) \in E$; if an edge is not undirected it is *undirected*, which is represented with an arrow $v_i \to v_j$; if all edges in $\mathcal{G}$ are directed, we also call $\mathcal{G}$ *directed*; we obtain the *skeleton* of a graph if we replace every directed edge in the graph with an undirected one
$\mathcal{G}_1 \leq \mathcal{G}_2$	Graph $\mathcal{G}_1$ is a **subgraph** of $\mathcal{G}_2$ if $\mathbf{V}_1 = \mathbf{V}_2$ and $E_1 \subseteq E_2$; if additionally $E_1 \neq E_2$, then $\mathcal{G}_1$ is a *proper subgraph*
$v_{i_1}, ...v_{i_m}$	A sequence of (at least two distinct) nodes with an edge between v_{i_k} and $v_{i_{k+1}}$ for $k = 1, ..., m-1$ is called **path**; if all edges are directed then the path is also *directed*; a *partially directed acyclic graph* (PDAG) has no *directed cycles*, i.e. no two nodes v_i, v_j, with directed paths from v_i to v_j and vice versa; if additionally all edges are directed we speak of a *directed acyclic graph* (DAG)
$\mathbf{S}_i^{\mathcal{G}}$, $\mathbf{s}_i^{\mathcal{G}}$	Special set of nodes v_i are are also identified with the corresponding random variables X_i and lower case for set of its realizations:
$\mathbf{PA}_i^{\mathcal{G}}$, $\mathbf{pa}_i^{\mathcal{G}}$	Set of **parents** of node v_i, i.e. vertices $v_j \in \mathbf{V}$, such that $(v_j, v_i) \in E$; a node without parents is called *source*, all sources of a graph $\mathcal{G}$ are collected in $\mathbf{SO}_{\mathcal{G}}$, all non-sources or *dependants* in $\mathbf{DP}_{\mathcal{G}} = \mathbf{V} \setminus \mathbf{SO}_{\mathcal{G}}$
$\mathbf{CH}_i^{\mathcal{G}}$, $\mathbf{ch}_i^{\mathcal{G}}$	Set of **children** of node v_i, i.e. vertices $v_j \in \mathbf{V}$, such that $(v_i, v_j) \in E$; an *immortality/v-structure* is a collection of three nodes, such that one is a child of the other two, which are not adjacent themselves; a node without children is called *sink*
$\mathbf{AN}_i^{\mathcal{G}}$, $\mathbf{an}_i^{\mathcal{G}}$	Set of **ancestors** of node v_i, i.e. vertices $v_j \in \mathbf{V}$, such that there exists a directed path from v_j to v_i
$\mathbf{DE}_i^{\mathcal{G}}$, $\mathbf{de}_i^{\mathcal{G}}$	Set of **descendants** of node v_i, i.e. vertices $v_j \in \mathbf{V}$, such that there exists a directed path from v_i to v_j; a permutation π is called *topological/causal ordering*, if $\pi(i) < \pi(j)$ for $v_j \in \mathbf{DE}_i^{\mathcal{G}}$
$\mathbf{ND}_i^{\mathcal{G}}$, $\mathbf{nd}_i^{\mathcal{G}}$	Set of **non-descendants** of node v_i, i.e. vertices $v_j \in \mathbf{V} \setminus \left(\mathbf{DE}_i^{\mathcal{G}} \cup \{v_i\}\right)$
$\mathbf{A} \perp\!\!\!\perp_{\mathcal{G}} \mathbf{B} \vert \mathbf{S}$	Set of nodes $\mathbf{A}$ and set of nodes $\mathbf{B}$ are **d-separated** by set of nodes $\mathbf{S}$ in $\mathcal{G}$, details can be found in Section 2.2

1.2 Standard Background Framework

We will now describe a background framework, which we will utilize throughout this thesis and extend if needed (or partially adapt as in Chapter 3) (see e.g. Schmidt (2011), Shorack (2017)).

Starting with the background probability space $(\Omega, \mathcal{A}, \mathbb{P})$, we consider d univariate random variables (i.e. taking values in $\mathbb{R}$) collected in a random vector $\mathbf{X} = (X_1, ...X_d)$. We have for the entire vector $\mathbf{X} \sim \mathbb{P}_{\mathbf{X}}$ with joint density $p(\mathbf{x}) = p(x_1, ..., x_n)$ wrt. the Lebesgue measure in d dimensions (i.e. we assume $\mathbb{P}_{\mathbf{X}} << \lambda^d$ in order to obtain $p(\mathbf{x})$ as the Radon Nikodym derivative). Analogously, we have for the marginal distributions $\mathbb{P}_{X_i}$ marginal densities wrt. the Lebesgue measure λ. Densities in this work are always defined to be wrt. the Lebesgue measure. When dealing with conditional densities we assume the existence with the marginal density always taking non-zero values.

For a given random variable X we denote the n iid. copies describing samples $\sim \mathbb{P}_X$ as random variables, by $X^{(j)}$ with $j \in \{1, ..., n\}$. For the collection of such *random samples* for all considered variables X_i we have

$$((X_1^{(j)}, ..., X_d^{(j)}), j = 1, ..., n) \sim \bigotimes_{j=1}^{n} \mathbb{P}_{\mathbf{X}} \tag{1.1}$$

which ultimately needs to be extended in Chapter 4.

2 Foundations Of Statistical Causality

As it is central to the human experience of reasoning about the real world, the notion of causality fascinates many disciplines of scientific inquiry. With the inductive character of the scientific method prescribing the collection and interpretation of data, the task of formalizing the idea of causality naturally becomes a statistical one. Fortunately, the language of probability, as Pearl (2009)[pp. 1] argues, is suited quite well for the purpose of understanding causality, since we are dealing with noisy observations, uncertainty and exceptions in real world applications.

This is not say that a theory of causality as proposed by statistics can solve all mysteries around the concept or even validate its existence, which has been contested multiple times, for example in the context of physics (see e.g. DAriano (2018)). We rather seek to provide a machinery for the use in data analysis that extends the usual framework of detecting dependencies in the sense of associations to detecting the directions of the dependencies. These are loaded with an interpretation of causality, i.e. they have a causal direction, which becomes a meaningful concept in the context of a specific causal model (see later).

2.1 The Starting Point And First General Assumptions

For this section see Lopez-Paz (2016)[ch. 6] and Peters et al. (2017)[ch. 1,2].

To go beyond modeling associations from observational data, i.e. data sampled from the joint distribution $p(x_1, ..., x_d)$, we first need to recognize the symmetry embedded in the concept of statistical (in)dependency: if a variable is (in)dependent on/of another, then the statement also holds vice versa. Both directions can be modeled for the purpose of prediction, yet the structure of the data-generating process is still not identifiable. In order to differentiate between *cause* and *effect*, i.e. to model the generation of data in a causal model, we need to reflect the asymmetry between cause and effect that characterizes the causality: the cause generates the effect, but not vice versa (with respect to a distinct causation; for cyclic structures we can have such causations in both directions).

We further understand causality modeled among a certain number of measured variables as an abstraction of more complicated underlying processes: for a given causal dependency (e.g. an physical one) it is often possible to "zoom in" and determine intermediate causes and effects. The notion of time is influential for causality as for example "[...] physics incorporates causality into its basic laws by excluding causation from future to past" Peters et al. (2017)[p. 26]. However, the statistical approach to causal modeling in a further effort of abstraction merely implicitly addresses time by introducing a hierarchy with the causal

directions. We assume in this setting a simple form of hierarchy by restricting ourselves to acyclic causal structures, even if cyclic causal structures are investigated in various papers (see e.g. Mooij et al. (2011)).

There are additional general assumptions about causality that lead to justifications for model assumptions and causal discovery techniques, notably the *independence of mechanisms*, that is the premise, that "[...] [t]he causal generative process of a systems variables is composed of autonomous modules that do not inform or influence each other" Peters et al. (2017)[p. 19], which is understood as a characteristic feature. Since the universal applicability is debatable, as Peters et al. (2017)[pp. 19] note themselves, we won't refer to it in our setting.

2.2 Establishing A Link Between Distribution And Graph

For this section see Peters et al. (2017)[ch. 6], Peters (2012)[ch. 2].

As we want to implement the intuitions expressed in the previous section in a formal way, we recognize the persistent themes of asymmetry and hierarchy, which directs our attention to the language of graphs.

More specifically, in our case we consider *directed acyclic graphs* (DAGs), with the *standard setting*, $\mathbf{X} = (X_1, ...X_d)$. The random variables X_i correspond to individual nodes of a graph $\mathcal{G} = (\mathbf{V}, \mathbf{E})$. For notational convenience, we can directly identify sets of vertices with sets of random variables, as done for example in Definition 2.2 or Theorem 2.3. More on basic definitions from graph theory can be found in Section 1.1. We will now give two further definitions in that realm:

We call a path between $v_{i_1}, ...v_{i_m}$ **blocked** by a set $\mathbf{S} \subseteq \mathbf{V}$, with $v_{i_1}, v_{i_m} \notin \mathbf{S}$ if $\exists\, v_{i_k}$, such that

i) either $v_{i_k} \in \mathbf{S}$ and the neighboring nodes in the path $v_{i_{k-1}}$ and $v_{i_{k+1}}$ are not both parents of v_{i_k},
ii) or $(\{v_{i_k}\} \cup \mathbf{DE}_{i_k}) \cap \mathbf{S} = \emptyset$ and both $v_{i_{k-1}}$ and $v_{i_{k+1}}$ are parents of v_{i_k},

which lays the groundwork for

Definition 2.1 (**d-separation**).
*We call two disjoint sets $\boldsymbol{A}, \boldsymbol{B} \subseteq \boldsymbol{V}$ **d-separated** by a third set $\boldsymbol{S} \subseteq \boldsymbol{V}$ (disjoint from $\boldsymbol{A}$ and $\boldsymbol{B}$) in a graph $\mathcal{G}$ denoted by $\boldsymbol{A} \perp\!\!\!\perp_{\mathcal{G}} \boldsymbol{B} | \boldsymbol{S}$, if every path between nodes in $\boldsymbol{A}$ and $\boldsymbol{B}$ is blocked by $\boldsymbol{S}$.*

This definition yields the central criterion of graphs, that is associated to a property of the joint probability distribution, i.e. conditional independence among (sets) of its coordinates, via the following definitions:

Definition 2.2 (**Global Markov Property**).
A joint distribution $\mathbb{P}_{\boldsymbol{X}}$ is said to have the ***global Markov property*** *with respect to a DAG $\mathcal{G}$ if $\boldsymbol{A} \perp\!\!\!\perp_{\mathcal{G}} \boldsymbol{B} | \boldsymbol{S} \Rightarrow \boldsymbol{A} \perp\!\!\!\perp \boldsymbol{B} | \boldsymbol{S}$ for all disjoint $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{S} \subseteq \boldsymbol{V}$.*

This feature can also be represented in two other ways:

Theorem 2.3 (**Markov property equivalences**).
The global Markov property of a joint distribution $\mathbb{P}_{\boldsymbol{X}}$ with respect to a graph $\mathcal{G}$ is equivalent to the following two characterizations:

i) the ***local Markov property****: $X_i \perp\!\!\!\perp \boldsymbol{ND}_i^{\mathcal{G}} \mid \boldsymbol{PA}_i^{\mathcal{G}}$ for all $i = 1, ..., d$*

ii) the ***Markov factorization property****:*

$$p(x_1, ..., x_d) = \prod_{i=1}^{d} p(x_i | \, \boldsymbol{pa}_i^{\mathcal{G}}) \tag{2.1}$$

where we call $p(x_i | \, \boldsymbol{pa}_i^{\mathcal{G}})$ the ***causal Markov kernels****.*

The second characterization requires the joint distribution to have a density, which is entailed in our standard setting. The proof is referenced in Peters et al. (2017)[p. 101]. The Markov factorization will play a major role in the discussions of this thesis as it allows access to localized building blocks of the total joint distribution, thus breaking it up into more manageable parts.

Next, we define the converse of the global Markov property:

Definition 2.4 (**Faithfulness**).
A joint distribution $\mathbb{P}_{\boldsymbol{X}}$ is said to be ***faithful*** *to a DAG $\mathcal{G}$ if $\boldsymbol{A} \perp\!\!\!\perp_{\mathcal{G}} \boldsymbol{B} | \boldsymbol{S} \Leftarrow \boldsymbol{A} \perp\!\!\!\perp \boldsymbol{B} | \boldsymbol{S}$ for all disjoint $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{S} \subseteq \boldsymbol{V}$.*

Naturally, if both any Markov property (we will from now on just say: the Markov property) and faithfulness hold for a distribution with respect to a graph, any d-separation of (sets of) nodes in the graph can be associated uniquely with the conditional independence statement of the corresponding (sets of) random variables.

A further characteristic that extends the Markov property is the following:

Definition 2.5 (**Causal Minimality**).
A joint distribution $\mathbb{P}_{\boldsymbol{X}}$ is said to satisfy ***causal minimality*** *with respect to a DAG $\mathcal{G}$, if it is Markov with respect to $\mathcal{G}$, but not any proper subgraph of $\mathcal{G}$.*

The Markov property together with faithfulness implies causal minimality (for a proof see e.g. Peters et al. (2017)[p. 108]).

Given a graph, i.e. an abstract representation of the causal structure, in order to examine all distributions that are Markov with respect to that graph, we collect them in a set $\mathcal{M}(\mathcal{G}) := \{\mathbb{P}_{\mathbf{X}} : \mathbb{P}_{\mathbf{X}} \text{ Markov wrt. } \mathcal{G}\}$, which becomes an attribute of graphs:

Definition 2.6 (**Markov Equivalence**).
*Two graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ are called **Markov equivalent**, if $\mathcal{M}(\mathcal{G}_1) = \mathcal{M}(\mathcal{G}_2)$. The set $\{\tilde{\mathcal{G}} : \mathcal{M}(\tilde{\mathcal{G}}) = \mathcal{M}(\mathcal{G})\}$ is the **Markov equivalence class** of the graph $\mathcal{G}$.*

The Markov equivalence class of a graph $\mathcal{G} = (\mathbf{V}, E)$ can be represented by a completed PDAG: $\text{CPDAG}(\mathcal{G}) = (\mathbf{V}, \tilde{E})$ with edge set $\tilde{E}$ that contains an edge (i, j) if and only if any member of the Markov equivalence class does. More precisely we have

Lemma 2.7 (**Test of Markov equivalence in the graph**).
Two DAGs are Markov equivalent if and only if they have the same skeleton and same immortalities (v-structures).

The proof is referenced in Peters et al. (2017)[p. 102].

2.3 The Structural Causal Model (SCM)

For this section see Peters et al. (2017)[ch. 6, 7], Peters (2012)[ch. 2], Bongers et al. (2016)[pp. 4].

After connecting the representation of variables in probabilistic form (i.e. the joint distribution) with the representation in graphical form describing the causal structure, we now examine another form of representation: *structural equations*. As these turn out to be quite rich in the information, we directly define a causal model with them:

Definition 2.8 (**The Structural Causal Model**).
*A **structural causal model (SCM)** $\mathfrak{S} = (\boldsymbol{S}, \mathbb{P}_{\boldsymbol{N}})$ consisting of a collection of d **structural equations** $\boldsymbol{S} = (S_1, ...S_d)$ that is assignments with $x_i, n_i \in \mathbb{R}$, $\boldsymbol{pa}_i \in \mathbb{R}^{|\boldsymbol{PA}_i|}$, $i \in \{1, ..., d\}$ of the form*

$$\begin{aligned} S_i : \quad & x_i := f_i(\boldsymbol{pa}_i, n_i) \\ & X_i :=_{as} f_i(\boldsymbol{PA}_i, N_i) \end{aligned} \tag{2.2}$$

*where the inputs $\boldsymbol{PA}_i \subseteq \{X_1, ..., X_d\} \setminus \{X_i\}$ are the **parents** of X_i, which can be identified with the corresponding parents $\boldsymbol{PA}_i^{\mathcal{G}}$ in the **associated graph** representation $\mathcal{G}$: $\boldsymbol{PA}_i = \boldsymbol{PA}_i^{\mathcal{G}}$. The graph $\mathcal{G}$ contains a directed edge from each parent (a **direct cause**) to the respective variable that is the function of its parents (a direct effect) of its direct causes. As we assume acyclicity we have that $\mathcal{G}$ is a DAG.*
Further we have a joint distribution $\mathbb{P}_{\boldsymbol{N}}$ over all noise variables $(N_1, ..., N_d) = \boldsymbol{N}$ which are required to be jointly independent.

The independence of the noise variables can be viewed as a manifestation of the assumption of **causal sufficiency**, i.e. the absence of hidden variables, which can lead to different results when using the causal structure for the purpose of inference (see e.g. Heinze-Deml et al. (2018)[p. 5]).

We now address the question of what the previous definition means for the observational distribution, that is the full unaltered joint distribution of a **solution** of the structural

equations: $\mathbf{X}$, a random vector with almost sure (as.) equalities $X_i = f_i(\mathbf{PA}_i, N_i)$ as (2.2) defines.

Theorem 2.9 (**Observational Distribution From SCM**).
A SCM $\mathfrak{S}$ entails a unique distribution over a given solution $\boldsymbol{X}$ (and thus the other as. solutions) which we will denote by $\mathbb{P}_{\boldsymbol{X}}^{\mathfrak{S}}$ (or with context just $\mathbb{P}_{\boldsymbol{X}}$).

Proof. As we exclude cycles in the SCM, we can for any given solution $\mathbf{X}$ recursively substitute the functional assignments into each other leading to a unique substitution procedure for each variable X_i, which yields a unique function g_i of the noises $\mathbf{N} = (N_1, ..., N_d)$:

$$X_i = g_i((N_k)_{k \in \mathbf{AN}_i}) \tag{2.3}$$

where the equality holds almost surely and thus in distribution and $\mathbf{AN}_i$ are the ancestors of X_i in the SCM. As $\mathbb{P}_{\mathbf{X}}$ is determined by the joint cdf $\mathrm{F}_{\mathbf{X}}(\mathbf{x}) = \mathbb{P}(X_1 \leq x_1, ..., X_d \leq x_d)$ for $\mathbf{x} = (x_1, ..., x_d) \in \mathbb{R}^d$, we see with (2.3) that

$$\mathrm{F}_{\mathbf{X}}(\mathbf{x}) = \mathbb{P}\left(\bigcap_{i=1}^{d} \{g_i((N_k)_{k \in \mathbf{AN}_i}) \leq x_i\}\right) = \mathbb{P}(\mathbf{N} \in A) = \mathbb{P}_{\mathbf{N}}(A) \tag{2.4}$$

where $A = \{(n_1, ..., n_d) \in \mathbb{R}^d : \forall i \in \{1, ..., d\}\ g_i((n_k)_{k \in \mathbf{AN}_i}) \leq x_i\}$. Since the noise terms $\mathbf{N}$ have a unique distribution $\mathbb{P}_{\mathbf{N}}$, it follows, that they induce a unique joint distribution $\mathbb{P}_{\mathbf{X}}^{\mathfrak{S}}$ among the X_i, independent of the particular solution for a fixed noise distribution. □

The proof of Theorem 2.9 prescribes how we sample from $\mathbb{P}_{\mathbf{X}}^{\mathfrak{S}}$: we use a draw from $\mathbb{P}_{\mathbf{N}}$ to obtain the realizations of the X_i via the structural assignments f_i starting with the sources in the order the assignments specify. In this way the f_i have a direction attached to them, which is why we call them assignments. Furthermore, we will mostly omit the distinction between different solutions from future discussions and pick any concrete random vector to represent the "almost sure" class of solutions.

Next, we discuss the existence of SCM's given a joint distribution $\mathbb{P}_{\mathbf{X}}$:

Theorem 2.10 (**SCM From Observational Distribution**).
Assuming the joint distribution $\mathbb{P}_{\boldsymbol{X}}$ has a density (wrt. Lebesgue measure as always assumed) and is Markov with respect to $\mathcal{G}$, there exists a SCM $\mathfrak{S}$ with associated graph $\mathcal{G}$ that entails $\mathbb{P}_{\boldsymbol{X}}$, i.e. $\mathbb{P}_{\boldsymbol{X}}^{\mathfrak{S}} = \mathbb{P}_{\boldsymbol{X}}$.

For a proof see Peters et al. (2017)[p. 230].

With this result we can conclude that given an observational distribution (with density) there is at least a SCM with a *complete graph* (any pair of nodes is connected) as associated graph, as $\mathbb{P}_{\mathbf{X}}$ is trivially Markov with respect to the complete graph. With other words: SCMs can generate any given observational distribution.

If, on the other hand, we start with a SCM, we automatically have

Theorem 2.11 (**SCM Implies Markov**).
For a SCM $\mathfrak{S}$ with induced $\mathbb{P}_{\boldsymbol{X}}^{\mathfrak{S}}$ and associated graph $\mathcal{G}$, we have that $\mathbb{P}_{\boldsymbol{X}}^{\mathfrak{S}}$ is Markov with respect to $\mathcal{G}$.

A proof is referenced in Peters et al. (2017)[p. 105].

Whenever we encounter two SCMs $\mathfrak{S}^1 = (\mathbf{S}^1, \mathbb{P}_{\mathbf{N}})$ and $\mathfrak{S}^2 = (\mathbf{S}^2, \mathbb{P}_{\mathbf{N}})$(with assignments f_i^1 and f_i^2 respectively) for modeling the causal structure of $\mathbf{X}$ such that

$$f_i^1(\mathbf{pa}_i, n_i) = f_i^2(\mathbf{pa}_i^*, n_i) \tag{2.5}$$

$\forall i, \mathbf{pa}_i, n_i$ with $p(n_i) > 0$, where $\mathbf{PA}_i^* \subsetneq \mathbf{PA}_i$ and thus $\mathbf{pa}_i^*$ a projection of $\mathbf{pa}_i$, we choose the latter SCM with the *minimal* representation. This principle is called **structural minimality** of SCMs.

Modeling interventions, the central concept of our approach to causality, requires us to deviate from the usual observational framework of ordinary joint distributions.

Definition 2.12 (**Interventional distribution**).
*For the SCM $\mathfrak{S} = (\boldsymbol{S}, \mathbb{P}_{\boldsymbol{N}})$ we speak of an **interventional SCM** $\tilde{\mathfrak{S}}$ when replacing at least one assignment, say for X_k, in $\mathfrak{S}$ by*

$$X_k :=_{as} \tilde{f}_k(\tilde{\boldsymbol{PA}}_k, \tilde{N}_k) \tag{2.6}$$

*where we require any $\tilde{N}_k$ to be jointly independent of the original noises $\boldsymbol{N}$. Modifications $\tilde{\boldsymbol{PA}}_k$ of the parents are only allowed if the associated graph of $\tilde{\mathfrak{S}}$ is still acyclic. In this case, we say that we have intervened on X_k. The induced distribution of the new SCM $\mathbb{P}_{\boldsymbol{X}}^{\tilde{\mathfrak{S}}}$, that is the **interventional distribution**, is denoted*

$$\mathbb{P}_{\boldsymbol{X}}^{\mathfrak{S};\mathrm{do}\left(X_k := \tilde{f}_k(\tilde{\boldsymbol{PA}}_k, \tilde{N}_k)\right)} := \mathbb{P}_{\boldsymbol{X}}^{\tilde{\mathfrak{S}}} \tag{2.7}$$

*An **atomic** intervention consists of altered assignments of the form $\mathrm{do}(X_k := a)$ with $a \in \mathbb{R}$. If we don't intervene with deterministic assignments, i.e. the marginal of X_k has positive variance, we call the intervention **stochastic**.*

Intervening and conditioning are two different concepts: If we consider for example a SCM $\mathfrak{S}$ with two variables

$$\begin{aligned} X_1 &:= N_1 \sim \mathcal{N}(0,1) \\ X_2 &:= X_1 + N_2,\ N_2 \sim \mathcal{N}(0, 0.1) \end{aligned} \tag{2.8}$$

we have for intervening on X_2 with $\mathrm{do}(X_2 := a)$ $(a \in \mathbb{R})$ that the marginal distribution of X_1 reads

$$p^{\mathfrak{S},\mathrm{do}(X_2:=a)}(x_1) = p^{\mathfrak{S}}(x_1) \neq p^{\mathfrak{S}}(x_1 | X_2 = a) \tag{2.9}$$

With our interventional framework, we can now examine meaningful formalizations of the essential notion of causality between two variables:

Definition 2.13 (**Total Causal Effect**).
For a SCM $\mathfrak{S}$ we say there is a ***total causal effect*** *from random variables X to Y if $X \not\perp\!\!\!\perp Y$ in $\mathbb{P}_{\boldsymbol{X}}^{\mathfrak{S};\mathrm{do}\left(X:=\tilde{N}_X\right)}$ for some random variable $\tilde{N}_X$*

Luckily, this definition is equivalent to others, one could think of:

Theorem 2.14 (**Total Causal Effect - Equivalencies**).
Given a SCM $\mathfrak{S}$ we have the equivalence of the following statements:

i) There is a total causal effect from X to Y
ii) There are $x_1, x_2 \in \mathbb{R}$ such that for the marginals of Y $\mathbb{P}_Y^{\mathfrak{S};\mathrm{do}(X:=x_1)} \neq \mathbb{P}_Y^{\mathfrak{S};\mathrm{do}(X:=x_2)}$
iii) There is $x \in \mathbb{R}$ such that for the marginals of Y $\mathbb{P}_Y^{\mathfrak{S};\mathrm{do}(X:=x)} \neq \mathbb{P}_Y^{\mathfrak{S}}$
iv) We have $X \not\perp\!\!\!\perp Y$ in $\mathbb{P}_{\boldsymbol{X}}^{\mathfrak{S};\mathrm{do}\left(X:=\tilde{N}_X\right)}$ for any random variable $\tilde{N}_X$ with a full support distribution

A proof is given in Peters et al. (2017)[pp. 226].

For our definition of a causal effect we can state the following about the relation of this concept to the associated graph:

Theorem 2.15 (**Relation - Total Causal Effect to Graph**).
Given a SCM $\mathfrak{S}$ with associated graph $\mathcal{G}$ we have:

i) If there is no directed path from X to Y, then there is no total causal effect from X to Y.
ii) There are cases, where there exists a directed path from X to Y, but no total causal effect from X to Y.

For a proof see Peters et al. (2017)[pp. 228].

The notion of total causal effect is realized in the construction of **randomized trials**, where we consider a target variable G and the randomization of a treatment variable T by intervening on it with an independent variable $T := \tilde{N}_T$ (and thus removing any hidden common cause of T and G). If we then observe a dependence between T and G, we conclude by Definition 2.13 that there is a total causal effect from treatment T to target G.

2.4 The Graphical Model (GM) And Its Connection To The SCM

For this section see Peters et al. (2017)[ch. 6], Peters (2012)[ch. 2].

Often times in the relevant literature about causality not the SCM but (variants of) another model are used:

Definition 2.16 (**The Graphical Model**).
A ***graphical model (GM)*** *$\mathfrak{G} = (\mathcal{G}, \mathbb{P}_{\boldsymbol{X}})$ consists of a DAG $\mathcal{G}$ and a joint distribution $\mathbb{P}_{\boldsymbol{X}}$ over the nodes/variables $\boldsymbol{X}$, where $\mathbb{P}_{\boldsymbol{X}}$ has the Markov property with respect to $\mathcal{G}$.*

Assuming the existence of a joint density, Theorem 2.10 states that for a given GM $\mathfrak{G} = (\mathcal{G}, \mathbb{P}_{\mathbf{X}})$ there exists at least one SCM $\mathfrak{S} = (\mathbf{S}, \mathbb{P}_{\mathbf{N}})$, such that $\mathcal{G}$ is the associated graph of $\mathfrak{S}$, which entails $\mathbb{P}_{\mathbf{X}}$. On the other hand, given a SCM we automatically obtain a unique GM by Definition 2.8 and Theorem 2.9. We conclude that every SCM can also be seen as a GM, since that is the coarser model.

As the Markov property is required in the definition of a GM $\mathfrak{G} = (\mathcal{G}, \mathbb{P}_{\mathbf{X}})$, we arrive under the assumption of a strictly positive, continuous joint density $p(x_1, ..., x_d)$ at the following situation: We have as a consequence of the Markov factorization property, that

$$p(x_1, ..., x_d) = \prod_{i=1}^{d} p(x_i | \, \mathbf{pa}_i^{\mathcal{G}}) \tag{2.10}$$

where the Markov kernels $p(x_i | \, \mathbf{pa}_i^{\mathcal{G}})$ are all well-defined (conditional) densities, i.e. $\forall i = 1, ..., d$ and given $\mathbf{pa}_i^{\mathcal{G}} \in \mathbb{R}^{|\mathbf{PA}_i|}$:

$$\int_{\mathbb{R}} p(x_i | \, \mathbf{pa}_i^{\mathcal{G}}) \, dx_i = 1 \tag{2.11}$$

Interventions in this model lead to an altered factorization. If we have interventions of the form $\text{do}\left(X_k := \tilde{N}_k\right)$, the joint density of the interventional distribution, denoted by $\mathbb{P}_{\mathbf{X}}^{\text{do}\left(X_k := \tilde{N}_k\right)}$, becomes

$$p^{\text{do}\left(X_k := \tilde{N}_k\right)}(x_1, ..., x_d) = \prod_{i \neq k} p(x_i | \, \mathbf{pa}_i^{\mathcal{G}}) \times p(\tilde{n}_k) \tag{2.12}$$

where $p(\tilde{n}_k)$ is the density of the independent intervention noise variable $\tilde{N}_k$.

For the interventional case, in order to access marginal, joint (i.e. the full interventional) or conditional distributions other than the Markov kernels and thus answer "causal questions", we refer to Peters et al. (2017)[ch. 6.6, 6.7]. Here, several ways to obtain such distributions of interest from observational marginals and intervention noise distributions are presented, namely working with *adjustment sets* and Pearls *do-calculus*. Central is the assumption of a known, fixed causal structure and the fact that the Markov kernels of the variables that are not intervened on, $p(x_i | \, \mathbf{pa}_i^{\mathcal{G}})$, $i \neq k$ in (2.12), don't change, which is known e.g. as **truncated factorization** or the **manipulation theorem**.

Possessing such a "causal inference machinery", which is also explored by (Pearl, 2009), for a fixed causal structure hence really motivates the pursuit of learning the latter.

2.5 Causal Structure Learning

For this section see Peters et al. (2017)[ch. 7], and e.g. Heinze-Deml et al. (2018) for a recent survey.

We can falsify a given causal structure (i.e. a model that is solely GM or SCM) by recognizing that the entailed observational and interventional distributions do not agree with data from the underlying data generating process (see Peters et al. (2017)[ch. 6.8]).

In order to actually learn the causal structure, multiple approaches have been proposed over the last decades, which include the large group of *independence-based* methods utilizing tests of conditional independencies and the connection via the results of Section 2.2 to the underlying graph. Another class can be described as *score-based* methods, which employ a scoring criterion in order to find the best scoring causal structure, which can effectively be equivalent to performing conditional independence tests (see e.g. Nandy et al. (2018)[p. 3163]). In this thesis, we will operate in the latter group of algorithms.

These techniques have various input conditions with respect to the type of data: e.g. algorithms for purely observational vs. those for observational and interventional data. Again, the latter setting is relevant for our considerations.

3 Gaussian Process Regression

In order to choose a flexible method to model the functions in the SCM setting we go beyond the standard assumptions of e.g. linear dependencies, which are often found in typical approaches (see e.g. references of Section 2.5), by employing Gaussian process regression (GPR). We will now introduce this slightly unconventional regression technique, which requires us to deal with Bayesian methods as GPR is a Bayesian non-parametric procedure.

For this chapter we adapt the notation of Rasmussen and Williams (2006): Assume that we have *input* (or predictor) random variables X_i with $i = 1, ..., D$ collected in random vector $\mathbf{X}$ and one *output* (or response) random variable Y, taking values in $\mathbb{R}^D$ and $\mathbb{R}$. We call $\bar{\mathbf{x}}$ the $D \times n$ *design matrix* gathering the real-valued realizations of $\mathbf{X}$ from n iid. samples from $\mathbb{P}_{\mathbf{X},Y}$, which also yield $\bar{y}$, the real-valued n-dimensional vector of realizations of Y. Further, we denote the collection of realizations of $\mathbf{X}$ and $\mathbf{y}$ in density/likelihood functions by writing the matrix and vector $\bar{\mathbf{x}}$ and $\bar{y}$ respectively.

3.1 Basis Functions As Starting Point

For this section see Rasmussen and Williams (2006)[ch. 2, 4], Steinwart and Christmann (2008)[pp. 111], Kanagawa et al. (2018)[p. 7].

We consider a Bayesian regression model with additive Gaussian noise $\epsilon \sim \mathcal{N}(0, \sigma^2)$, $\sigma^2 \in \mathbb{R}_+$, of the form

$$y = f(\mathbf{x}) + \epsilon, \quad f(\mathbf{x}) = \phi(\mathbf{x})^T \mathbf{w} \tag{3.1}$$

with $\mathbf{x} \in \mathbb{R}^D$, $y \in \mathbb{R}$, weights $\mathbf{w} \in \mathbb{R}^N$ and a map $\phi : \mathbb{R}^D \to \mathbb{R}^N$ into the *feature space* containing a finite number (N) of basis functions, e.g. for $D = 1$, polynomials: $\phi(x) = (1, x, x^2, x^3, x^4)$, i.e. $N = 5$, which can but don't need to be linear unlike the usage of the weights $\mathbf{w}$ in f.

As we model a deterministic function f, we can calculate with the Gaussian noise assumption for the n iid. samples due to the independence that

$$p(\bar{y}|\bar{\mathbf{x}}, \mathbf{w}) = \mathcal{N}\left(\bar{y}|\Phi(\bar{\mathbf{x}})^T \mathbf{w}, \sigma^2 \mathrm{I}\right) \tag{3.2}$$

where $\Phi(\bar{\mathbf{x}})$ is the $N \times n$ matrix consisting of columns $\phi\left(\mathbf{x}^{(j)}\right)$ with $\mathbf{x}^{(j)}$ being the columns of $\bar{\mathbf{x}}$, $j = 1, ..., n$, i.e. the n samples of $\mathbf{X}$.

In a Bayesian context we now set a prior over the parameters, which in this case are the weights: $p(\mathbf{w}) = \mathcal{N}(0, \Sigma_p)$. By Bayes' rule we can now compute the *posterior distribution*:

$$p(\mathbf{w}|\bar{y}, \bar{\mathbf{x}}) = \frac{p(\bar{y}|\bar{\mathbf{x}}, \mathbf{w})p(\mathbf{w})}{p(\bar{y}|\bar{\mathbf{x}})} \tag{3.3}$$

with the *normalizing constant* $p(\bar{y}|\bar{\mathbf{x}}) = \int p(\bar{y}|\bar{\mathbf{x}}, \mathbf{w})p(\mathbf{w})d\mathbf{w}$. The posterior can be determined analytically without the need to compute the previous integral by calculating the product of multivariate normal likelihood and prior and matching the distribution. We obtain the exact posterior distribution

$$p(\mathbf{w}|\bar{y}, \bar{\mathbf{x}}) = \mathcal{N}\left(\mathbf{w}|\frac{1}{\sigma^2}A^{-1}\Phi(\bar{\mathbf{x}})\bar{y}, A^{-1}\right) \tag{3.4}$$

where $A = \frac{1}{\sigma^2}\Phi(\bar{\mathbf{x}})\Phi(\bar{\mathbf{x}})^T + \Sigma_p^{-1}$.

If we now consider the $(\bar{\mathbf{x}}, \bar{y})$ as training samples and further recognize test samples $\bar{\mathbf{x}}_*$, the *posterior predictive distribution* with $\bar{f}_*$ being the prediction values for $f(\bar{\mathbf{x}}_*)$ is

$$\begin{aligned} p\left(\bar{f}_*|\bar{\mathbf{x}}_*, \bar{\mathbf{x}}, \bar{y}\right) &= \int p\left(\bar{f}_*|\bar{\mathbf{x}}_*, \mathbf{w}\right) p\left(\mathbf{w}|\bar{\mathbf{x}}, \bar{y}\right) d\mathbf{w} \\ &= \mathcal{N}\left(\bar{f}_*|\frac{1}{\sigma^2}\Phi(\bar{\mathbf{x}}_*)^T A^{-1}\Phi(\bar{\mathbf{x}})\bar{y}, \Phi(\bar{\mathbf{x}}_*)^T A^{-1}\Phi(\bar{\mathbf{x}}_*)\right) \\ &= \mathcal{N}\left(\bar{f}_*|\Phi_*^T\Sigma_p\Phi(K + \sigma^2\,\mathrm{I})^{-1}\bar{y}, \Phi_*^T\Sigma_p\Phi_* - \Phi_*^T\Sigma_p\Phi(K + \sigma^2\,\mathrm{I})^{-1}\Phi^T\Sigma_p\Phi_*\right) \end{aligned} \tag{3.5}$$

where $\Phi(\bar{\mathbf{x}}_*) = \Phi_*$, $\Phi(\bar{\mathbf{x}}) = \Phi$ and $K = \Phi^T\Sigma_p\Phi$ (see Rasmussen and Williams (2006)[pp. 9] for algebaric details).

We see that the feature maps ϕ only appear in terms of the sort $\Phi^T\Sigma_p\Phi$, $\Phi_*^T\Sigma_p\Phi$ (and $\Phi^T\Sigma_p\Phi_*$) or $\Phi_*^T\Sigma_p\Phi_*$. Any entry of such matrices can be written in the form $k(\mathbf{x}, \mathbf{x}') := \phi(\mathbf{x})\Sigma_p\phi(\mathbf{x}')$, where $\mathbf{x}, \mathbf{x}'$ are training or test values. We call $k(\cdot, \cdot)$ the *kernel function* in our regression setting. Since Σ_p is positive definite as covariance matrix of a multivariate Gaussian with density, i.e. $\mathbf{v}^T\Sigma_p\mathbf{v} > 0$ for all $\mathbf{v} \neq \mathbf{0}, \mathbf{v} \in \mathbb{R}^N$, we can define a matrix $\Sigma_p^{1/2}$ (e.g. via the singular value decomposition), such that $(\Sigma_p^{1/2})^2 = \Sigma_p$. Thus, with $\psi(\mathbf{x}) = \Sigma_p^{1/2}\phi(\mathbf{x})$, we have $k(\mathbf{x}, \mathbf{x}') = \psi(\mathbf{x}) \cdot \psi(\mathbf{x}')$, i.e. an inner product in the feature space, justifying the name *kernel* (see Steinwart and Christmann (2008)[p. 112]).

If we can directly access this kernel function and replace all the computations of inner products in the feature space by an expression of $k(\mathbf{x}, \mathbf{x}')$ in closed form, we apply the *kernel trick*. In our case this allows us to take advantage of an infinite number of basis functions. Say, for example, we choose $\Sigma_p = \sigma_p^2\,\mathrm{I}$ and basis functions $\phi_c(x) = \exp\left(-\frac{(x-c)^2}{2l^2}\right)$ ($D = 1$, $l \in \mathbb{R}^+$, $\sigma_p^2 > 0$). This leads to a kernel function

$$k_{finite}(x, x') = \sigma_p^2 \sum_{c=1}^{N} \phi_c(x)\phi_c(x') \tag{3.6}$$

After slightly modifying the scaling and two limiting procedures (one of which is $N \to \infty$, for details see Rasmussen and Williams (2006)[pp. 83], MacKay (1998)[pp. 8]) we get a kernel ($x, x' \in \mathbb{R}$)

$$k(x, x') = \sqrt{\pi} l \sigma_p^2 \exp\left(-\frac{(x - x')^2}{2(\sqrt{2}l)^2}\right) \tag{3.7}$$

which is a rescaled version of what we will define as the squared exponential kernel. In fact, by Mercer's theorem, we can identify every given *positive definite kernel function*, which is defined below, under mild conditions with a (maybe infinite) series of basis functions (see e.g. Minh et al. (2006)).

Definition 3.1 (**Positive Definite Kernel Function**).
*A symmetric function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ (i.e. we have $\forall x, x' \in \mathcal{X}\ k(x, x') = k(x', x)$) for some nonempty set $\mathcal{X}$ is called **positive definite kernel**, if $\forall n \in \mathbb{N}, (c_1, ..., c_n) \in \mathbb{R}^n$, $(x^{(1)}, ..., x^{(n)}) \in \mathcal{X}^n$*

$$\sum_{i=1}^{n} \sum_{j=1}^{n} c_i c_j k(x^{(i)}, x^{(j)}) \geq 0 \tag{3.8}$$

*Further, we call k **strictly positive definite kernel**, if for mutually distinct $x^{(1)}, ..., x^{(n)} \in \mathcal{X}$ we have equality in (3.8) only for $c_1 = ... = c_n = 0$.*

We call the $n \times n$ matrix $K := \left(k(x^{(i)}, x^{(j)})\right)_{i,j}$ for fixed $x^{(1)}, ..., x^{(n)} \in \mathcal{X}$ the **Gram matrix**. A positive definite kernel leads to a positive semi-definite Gram matrix, i.e. $\mathbf{v}^T K \mathbf{v} \geq 0$ for all $\mathbf{v} \neq \mathbf{0}, \mathbf{v} \in \mathbb{R}^N$, and a strictly positive definite kernel by the same arguments to a positive definite K.

An example for a positive definite kernel is the already mentioned **squared exponential kernel** (with length scale $\gamma \in \mathbb{R}_+$, $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^D$)

$$k_{se}(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{||\mathbf{x} - \mathbf{x}'||_2^2}{\gamma^2}\right) \tag{3.9}$$

(see e.g. Steinwart and Christmann (2008)[pp. 116]). Since for mutually distinct $\mathbf{x}^{(1)}, ..., \mathbf{x}^{(n)} \in \mathbb{R}^D$ we have $k_{se}(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) > 0$ for $i, j \in \{1, ..., n\}$ the squared exponential kernel even is strictly positive definite. By the previous considerations, we thus have a positive definite Gram matrix K_{se} from the squared exponential kernel.

3.2 The Distributed Functions View

For this section see Rasmussen and Williams (2006)[ch. 2], Kuss (2006)[ch. 3], Kanagawa et al. (2018)[pp. 7], Seeger (2004).

The essential idea of Gaussian process regression is the application of stochastic processes in regression tasks.

Definition 3.2 (**Stochastic Process**).
A (real valued) ***stochastic process*** *is a function* $f : \mathcal{X} \times \Omega \to \mathbb{R}$ *mapping a state* $x \in \mathcal{X}$ *(nonempty set) and an outcome* $\omega \in \Omega$ *to a real number, such that for every fixed* $x \in \mathcal{X} : f(x, \cdot)$ *is a measurable function, i.e. a random variable on* Ω*. For a fixed realization* $\omega \in \Omega$ *we call* $f(\cdot, \omega)$ *a sample path.*

The states x are often identified with points in time. Yet in the GPR view we identify them with points in our general input space $\mathbf{x} \in \mathbb{R}^D$. Then for the random variable $f(\mathbf{x})$ we consider the value of the **mean function**

$$m(\mathbf{x}) = \mathbb{E}[f(\mathbf{x})] \tag{3.10}$$

and for the two random variables $f(\mathbf{x}), f(\mathbf{x}')$ we consider the value of the **covariance function**

$$k(\mathbf{x}, \mathbf{x}') = \mathbb{E}[(f(\mathbf{x}) - m(\mathbf{x}))(f(\mathbf{x}') - m(\mathbf{x}'))] \tag{3.11}$$

a measure of similarity between two points, which is a positive definite kernel (see Seeger (2004))[p. 73] and can be associated with the kernel function of the regression setting of the previous section ($\mathbf{w} \sim \mathcal{N}(0, \Sigma_p)$) by computing the covariance:

$$\mathbb{E}[f(\mathbf{x})f(\mathbf{x}')] = \phi(\mathbf{x})^T \mathbb{E}[\mathbf{w}\mathbf{w}^T]\phi(\mathbf{x}') = \phi(\mathbf{x})^T \Sigma_p \phi(\mathbf{x}') \tag{3.12}$$

Together with the mean $\mathbb{E}[f(\mathbf{x})] = \phi(\mathbf{x})^T \mathbb{E}[\mathbf{w}] = 0$ we already have a characterization of a Gaussian process, which we can now define as the central object of this chapter in the (for us relevant) setting of real numbers:

Definition 3.3 (**Gaussian Process**).
A ***Gaussian process*** *is a stochastic process* f *with mean function* $m : \mathbb{R}^D \to \mathbb{R}$ *and covariance function* $k : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$ *denoted* $f \sim \mathcal{GP}(m, k)$ *such that for any finite collection represented by a matrix* $\bar{\boldsymbol{x}} = (\boldsymbol{x}^{(1)}, ..., \boldsymbol{x}^{(n)}) \in \mathbb{R}^{D \times n}$ *with* $n \in \mathbb{N}$ *the random vector with values* $\in \mathbb{R}^n$

$$f_{\bar{\boldsymbol{x}}} = (f(\boldsymbol{x}^{(1)}), ..., f(\boldsymbol{x}^{(n)}))^T \sim \mathcal{N}(m_{\bar{\boldsymbol{x}}}, K_{\bar{\boldsymbol{x}},\bar{\boldsymbol{x}}}) \tag{3.13}$$

where the multivariate normal distribution $\mathcal{N}(m_{\bar{\boldsymbol{x}}}, K_{\bar{\boldsymbol{x}},\bar{\boldsymbol{x}}})$ *is determined by mean vector* $m_{\bar{\boldsymbol{x}}} = (m(\boldsymbol{x}^{(1)}), ..., m(\boldsymbol{x}^{(n)}))^T$ *and* $n \times n$ *covariance matrix* $K_{\bar{\boldsymbol{x}},\bar{\boldsymbol{x}}} = (k_{\bar{\boldsymbol{x}}^{(i)},\bar{\boldsymbol{x}}^{(j)}})_{i,j}$.

Reversely, for a given function $m : \mathbb{R}^D \to \mathbb{R}$ and positive definite kernel function $k : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$ there exists a unique corresponding Gaussian process $f \sim \mathcal{GP}(m, k)$, i.e. a pair (m, k) exactly identifies a Gaussian process (one-to-one correspondence) (see Kanagawa et al. (2018)[p. 9]). Note, that the covariance matrix $K_{\bar{\mathbf{x}},\bar{\mathbf{x}}}$ relates to the Gram matrix of the previous section.

As already declared, we will use a Bayesian framework to utilize the definition of Gaussian processes for our purpose of regression. Therefore, we think of (3.13) as a prior belief: Given

some (*test*) data on the input side $\bar{\mathbf{x}}_* \in \mathbb{R}^{D \times n_*}$ consisting of n_* data points, we assume a prior distribution of

$$f_{\bar{\mathbf{x}}_*} \sim \mathcal{N}(m_{\bar{\mathbf{x}}_*}, K_{\bar{\mathbf{x}}_*, \bar{\mathbf{x}}_*}) \tag{3.14}$$

i.e. a sample from this multivariate normal distribution yields the values of the respective sample path of this **Gaussian process prior** at the input points $\bar{\mathbf{x}}_*$.

For a data set split in training and test values on the input side we have n input training values $\bar{\mathbf{x}} \in \mathbb{R}^{D \times n}$ and n_* input test values $\bar{\mathbf{x}}_* \in \mathbb{R}^{D \times n_*}$.

In a *noise free* framework, i.e. we assume that observed function values are not disturbed by any noise, i.e. we consider a regression setting $y = f(\mathbf{x})$ with $y \in \mathbb{R}$, $\mathbf{x} \in \mathbb{R}^D$. Thus, we have n training inputs $\mathbf{x}^{(i)} \in \mathbb{R}^D$ and a prior given data $\bar{\mathbf{x}}, \bar{\mathbf{x}}_*$ (assuming a zero-mean prior belief)

$$f_{\bar{\mathbf{x}}}, f_{\bar{\mathbf{x}}_*} | \bar{\mathbf{x}}, \bar{\mathbf{x}}_* \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K_{\bar{\mathbf{x}}, \bar{\mathbf{x}}} & K_{\bar{\mathbf{x}}, \bar{\mathbf{x}}_*} \\ K_{\bar{\mathbf{x}}_*, \bar{\mathbf{x}}} & K_{\bar{\mathbf{x}}_*, \bar{\mathbf{x}}_*} \end{bmatrix}\right) \tag{3.15}$$

In order to predict the function values at the test points, we need to access the **posterior predictive distribution** as in the previous section. Luckily, as we are working with multivariate normals, we can utilize their convenient calculus (for algebraic details see Rasmussen and Williams (2006)[p. 16], $f_{\bar{\mathbf{x}}} \in \mathbb{R}^n$):

$$f_{\bar{\mathbf{x}}_*} | \bar{\mathbf{x}}, f_{\bar{\mathbf{x}}}, \bar{\mathbf{x}}_* \sim \mathcal{N}\left(K_{\bar{\mathbf{x}}_*, \bar{\mathbf{x}}} K_{\bar{\mathbf{x}}, \bar{\mathbf{x}}}^{-1} f_{\bar{\mathbf{x}}},\ K_{\bar{\mathbf{x}}_*, \bar{\mathbf{x}}_*} - K_{\bar{\mathbf{x}}_*, \bar{\mathbf{x}}} K_{\bar{\mathbf{x}}, \bar{\mathbf{x}}}^{-1} K_{\bar{\mathbf{x}}, \bar{\mathbf{x}}_*}\right) \tag{3.16}$$

In a *noisy* framework, i.e. assuming the regression setting $y = f(\mathbf{x}) + \epsilon$, with additive iid. Gaussian noise $\epsilon \sim \mathcal{N}(0, \sigma^2)$ and $y \in \mathbb{R}$, $\mathbf{x} \in \mathbb{R}^D$, we have n training inputs $\mathbf{x}^{(i)} \in \mathbb{R}^D$, the response random vector $\mathbf{Y} = f_{\bar{\mathbf{x}}} + (\epsilon_1, ..., \epsilon_n)^T$ and a prior given data $\bar{\mathbf{x}}, \bar{\mathbf{x}}_*$ (assuming a zero-mean prior belief)

$$\begin{pmatrix} \mathbf{Y} \\ f_{\bar{\mathbf{x}}_*} \end{pmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K_{\bar{\mathbf{x}}, \bar{\mathbf{x}}} + \sigma^2 \mathrm{I} & K_{\bar{\mathbf{x}}, \bar{\mathbf{x}}_*} \\ K_{\bar{\mathbf{x}}_*, \bar{\mathbf{x}}} & K_{\bar{\mathbf{x}}_*, \bar{\mathbf{x}}_*} \end{bmatrix}\right) \tag{3.17}$$

where we have $\mathbf{Y} \sim \mathcal{N}(\mathbf{0}, K_{\bar{\mathbf{x}}, \bar{\mathbf{x}}} + \sigma^2 \mathrm{I})$ because the covariance for single outputs y_p, y_q from $\mathbf{y}$ reads

$$\mathbb{E}[(f(\mathbf{x}_p) + \epsilon)(f(\mathbf{x}_q) + \epsilon)] = \mathrm{cov}[f(\mathbf{x}_p), f(\mathbf{x}_q] + \mathrm{cov}[\epsilon, \epsilon] = k(\mathbf{x}_p, \mathbf{x}_q) + \sigma^2 \delta_{pq} \tag{3.18}$$

due to the iid. assumption on the noises and with the Kronecker delta δ_{pq}. Again, we have a closed form expression of the posterior predictive distribution

$$\begin{aligned} &f_{\bar{\mathbf{x}}_*} | \bar{\mathbf{x}}, \mathbf{y}, \bar{\mathbf{x}}_* \sim \\ &\mathcal{N}\left(K_{\bar{\mathbf{x}}_*, \bar{\mathbf{x}}} (K_{\bar{\mathbf{x}}, \bar{\mathbf{x}}} + \sigma^2 \mathrm{I})^{-1} \mathbf{y},\ K_{\bar{\mathbf{x}}_*, \bar{\mathbf{x}}_*} - K_{\bar{\mathbf{x}}_*, \bar{\mathbf{x}}} (K_{\bar{\mathbf{x}}, \bar{\mathbf{x}}} + \sigma^2 \mathrm{I})^{-1} K_{\bar{\mathbf{x}}, \bar{\mathbf{x}}_*}\right) \end{aligned} \tag{3.19}$$

which corresponds to (3.5) for the specific choice of covariance/kernel function in the prevoius section.

In this noisy setting we introduce the (logarithmic) **marginal likelihood** derived from the full GP prior by marginalization

$$\log p(\mathbf{y}|\bar{\mathbf{x}}) = -\frac{1}{2}\mathbf{y}^T(K_{\bar{\mathbf{x}},\bar{\mathbf{x}}} + \sigma^2\,\mathrm{I})^{-1}\mathbf{y} - \frac{1}{2}\log\det(K_{\bar{\mathbf{x}},\bar{\mathbf{x}}} + \sigma^2\,\mathrm{I}) - \frac{n}{2}\log(2\pi) \tag{3.20}$$

which is particularly relevant for the next section.

3.3 Learning The Gaussian Process Regression Model

For this section see Rasmussen and Williams (2006)[ch. 2,5].

In principle, given an estimate of σ^2 in the noisy case, we could apply Gaussian process regression by sampling from the appropriate distributions of the preceding section with a kernel of fixed form associated with our prior belief about the functional shape. In practice, however, we can improve on the fitting capabilities of our regression method by learning **kernel hyperparameters**, i.e. free parameters within the kernel/covariance function as for example the length scale in the squared exponential kernel (3.9).

The term *hyper*parameter stresses the reality that we are in fact not dealing with regular parameters in the sense of common regression models of type (3.1) as we saw that the correspondence of these models with GPR contains the integration over the regular parameters (i.e. the weights) in the Bayesian framework (3.5) and GP kernels might even coincide with an infinite number of basis function and thus parameters. Therefore, GP models are sometimes called *infinite dimensional* and *non-parametric*. From now on we will also include the noise σ^2 in the collection of hyperparameters $\boldsymbol{\theta} \in \boldsymbol{\Theta}$ we want to learn, which for the example of the simple squared exponential kernel with only the length scale as kernel hyperparameter $\boldsymbol{\theta}_k = \gamma$ would then be $\boldsymbol{\theta} = (\sigma^2, \boldsymbol{\theta}_k) = (\sigma^2, \gamma) \in \mathbb{R}_+ \times \mathbb{R}_+$.

From a Bayesian perspective we enter this learning or **model selection** procedure in the case of our GP setting at level 2, as the first level concerns the Bayesian update of the regular parameters or weights from our early regression setting (3.1), which we don't have to deal with directly precisely because of our Gaussian process kernel method.

Continuing with the notation of the previous section, we have at level 2 the marginal likelihood $p(\mathbf{y}|\bar{\mathbf{x}}, \boldsymbol{\theta})$ conditioned on the hyperparameters. Further, we need to define a "hyper prior" on the hyperparameters $p(\boldsymbol{\theta})$, Then Bayes' rule yields for the posterior over hyerparameters

$$p(\boldsymbol{\theta}|\mathbf{y}, \bar{\mathbf{x}}) = \frac{p(\mathbf{y}|\bar{\mathbf{x}}, \boldsymbol{\theta})p(\boldsymbol{\theta})}{\int p(\mathbf{y}|\bar{\mathbf{x}}, \boldsymbol{\theta})p(\boldsymbol{\theta})\,d\boldsymbol{\theta}} \tag{3.21}$$

If a finite set of models $\{\mathcal{M}_1, ..., \mathcal{M}_L\}$ is considered, which could for example represent the application of various kernels, we can advance to level 3 by conditioning any density

in level 2 on $\mathcal{M}_l$, $l \in 1, ..., L$ and compute the posterior over models $p(\mathcal{M}_l|\mathbf{y}, \bar{\mathbf{x}})$ by Bayes' rule as well.

In practice the integration in (3.21) can often only be done numerically. To avoid the downsides of that as well as the choice of hyper priors, the hyperparameters are often estimated by the maximum likelihood method utilizing the marginal likelihood (3.20), i.e.

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta} \in \boldsymbol{\Theta}} p_{\boldsymbol{\theta}}(\mathbf{y}|\bar{\mathbf{x}}) \tag{3.22}$$

This can either be seen as a maximum a posteriori (MAP) estimate with flat prior of the Bayesian update (3.21) in level 2, in which case we speak of type II maximum likelihood estimation or we leave the Bayesian view and justify our estimation as simple application of the maximum likelihood method. In either case we should keep in mind that the possibility of overfitting exists. The choice of maximum likelihood is further supported by the fact that the derivatives for the objective function wrt. all hyperparameters can be calculated, which is beneficial for the optimization procedure using versions of gradient descent.

3.4 Dealing With Higher Input Dimensions

For this section see Duvenaud et al. (2011), Rasmussen and Williams (2006)[ch. 5].

The question of how to construct a kernel for multiple dimensions can be answered in different ways, eg. directly with a multidimensional squared exponential kernel (3.9). In order to generalize this approach as well as regression models with D inputs of GAM(Generalized Additive Model)-form, i.e. the response is computed from a sum of functions $f_1(x_1) + ... + f_D(x_D)$, we concentrate in this thesis on the relatively new method of **additive Gaussian processes** by Duvenaud et al. (2011), which they have shown to be competitive in practice among various alternatives.

We define a one dimensional ($x_q, x_q' \in \mathbb{R}$) **base kernel** $k_q(x_q, x_q')$, which could be for example squared exponential, and with that consider an **r-th order additive kernel** ($r \in \{1, ..., D\}$, $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^D$)

$$k_{add_r}(\mathbf{x}, \mathbf{x}') = \sigma_r^2 \sum_{1 \leq s_1 < ... < s_r \leq D} \left(\prod_{t=1}^{r} k_{s_t}(x_{s_t}, x'_{s_t}) \right) \tag{3.23}$$

with additional hyperparameter $\sigma_r^2 > 0$ measuring the influence of the r-th order interaction. The full additive kernel is then merely the sum over the additive kernels of all orders up to a highest order of our choice. If we only choose to consider the 1-st order we recover a GAM. The computations don't interfere with properties of (strictly) positive definitness of a given base kernel for the new full additive kernel and we can still calculate derivatives wrt. all hyperparameters for the optimization in maximum likelihood schemes.

3.5 Advantages Of GPR For Our Setting

For this section see Duvenaud (2014)[pp. 4].

A commonly known downside of GPR is the need to deal with the inversion of matrices that scale with the number of samples (see e.g. covariance matrix in (3.20)). As long as we are in a situation of few samples however, we can truly take advantage of the benefits of GPR such as the large expressive capability geared toward highly nonlinear functions and all the leverage we gain by dealing with (multivariate) normal distributions, which in our case leads to easily computable estimates of conditional distributions as we will see in the next chapter.

4 The Gaussian Process SCM (GP-SCM)

We build on the groundwork of von Kügelgen et al. (2019) and thus indirectly Friedman and Nachman (2013).

4.1 The GP-SCM Model

In this chapter we merge results of the two previous chapters by examining the use of Gaussian process regression to model the structural equations among the variables in the SCM (2.8). The following definition is similar to one defining the same term by Karimi et al. (2020)[pp. 4]:

Definition 4.1 (**The Gaussian Process SCM (GP-SCM)**).
*A **Gaussian process SCM** $\mathfrak{S}_{\mathcal{GP}} = (\mathfrak{S}, \boldsymbol{f})$ consists of an underlying SCM $\mathfrak{S}$ with fixed, deterministic, yet unknown assignments in the included structural equations of the form*

$$\begin{aligned} S_i: \quad x_i &:= f_i^u(\boldsymbol{pa}_i) + n_i \\ X_i &:=_{as} f_i^u(\boldsymbol{PA}_i) + N_i \end{aligned} \tag{4.1}$$

with additive noise variables N_i. Further we have an independent collection $\boldsymbol{f} = (f_1, ..., f_n)$, where the Gaussian process f_i has a prior $f_i \sim \mathcal{GP}(0, k_i)$ and $N_i \sim \mathcal{N}(0, \sigma_i^2)$ representing the additive noise in the GPR setting if X_i is not a source in the associated graph $\mathcal{G}$. Else, for the sources we have $f_i \equiv 0$ and random variables N_i with some density $p(n_i)$.

By modeling the unknown functions f_i^u with f_i, i.e. the non-parametric method of GPR, we need data from the variables $\mathbf{X}$ in order to sensibly use the GP-SCM as laid out in the next section.

4.1.1 Inference Given A GP-SCM

Consider a non-source node X_i with D_i parents $\mathbf{PA}_i$ and $j = 1, ..., n$ realizations collected in $\mathbf{x}_i \in \mathbb{R}^n$ and $\mathbf{p\bar{a}}_i \in \mathbb{R}^{D_i \times n}$ and a strictly positive definite kernel k. With m given test values $\mathbf{p\bar{a}}_{i*} \in \mathbb{R}^{D_i \times m}$ we have a joint GP prior of

$$p(\mathbf{x}_i, \mathbf{x}_{i*}) = \mathcal{N}\left(\begin{bmatrix} \mathbf{x}_i \\ \mathbf{x}_{i*} \end{bmatrix} \middle| \mathbf{0}, \begin{bmatrix} K_{\mathbf{p\bar{a}}_i, \mathbf{p\bar{a}}_i} + \sigma_i^2 \, \mathrm{I} & K_{\mathbf{p\bar{a}}_i, \mathbf{p\bar{a}}_{i*}} \\ K_{\mathbf{p\bar{a}}_{i*}, \mathbf{p\bar{a}}_i} & K_{\mathbf{p\bar{a}}_{i*}, \mathbf{p\bar{a}}_{i*}} + \sigma_i^2 \, \mathrm{I} \end{bmatrix} \right) \tag{4.2}$$

for $\mathbf{x}_i$ and $\mathbf{x}_{i*}$, which differs from (3.17) in the lower right corner of the covariance matrix as we look at the test prediction of the variable X_i not only the function prediction, so by the argument of (3.18) we get (4.2). Further by (3.19), this yields the conditional density

$$\begin{aligned} p(\mathbf{x}_{i*}|\mathbf{p\bar{a}}_i, \mathbf{x}_i, \mathbf{p\bar{a}}_{i*}) = &\mathcal{N}(\mathbf{x}_{i*}|K_{\mathbf{p\bar{a}}_{i*},\mathbf{p\bar{a}}_i}(K_{\mathbf{p\bar{a}}_i,\mathbf{p\bar{a}}_i} + \sigma_i^2\,\mathrm{I})^{-1}\mathbf{x}_i, \\ &K_{\mathbf{p\bar{a}}_{i*},\mathbf{p\bar{a}}_{i*}} + \sigma_i^2\,\mathrm{I} - K_{\mathbf{p\bar{a}}_{i*},\mathbf{p\bar{a}}_i}(K_{\mathbf{p\bar{a}}_i,\mathbf{p\bar{a}}_i} + \sigma_i^2\,\mathrm{I})^{-1}K_{\mathbf{p\bar{a}}_i,\mathbf{p\bar{a}}_{i*}}) \end{aligned} \tag{4.3}$$

If we want to sample from a known structure with fixed graph, source densities and GP kernels including the variances σ_i, these conditional densities together with the densities of the sources nodes $p(n_i)$ model the Markov kernels (conditioned on data $\mathbf{p\bar{a}}_i, \mathbf{x}_i$) - i.e. for sample size 1 ($x_{i*} \in \mathbb{R}$, $\mathbf{pa}_{i*} \in \mathbb{R}^{D_i}$)

$$\hat{p}(x_{i*}|\mathbf{pa}_{i*}) = p(x_{i*}|\mathbf{p\bar{a}}_i, \mathbf{x}_i, \mathbf{pa}_{i*}) \tag{4.4}$$

- in the Markov factorization, which exists due to the underlying SCM $\mathfrak{S}$, which also dictates the sampling hierarchy as described in Section 2.3. As we have the associated graph $\mathcal{G}$ and access to the full joint distribution via (4.4), which yields an estimated full joint distribution $\hat{\mathbb{P}}_{\mathbf{X}}$, as well as the Markov property stemming from the Markov factorization we can treat $(\mathcal{G}, \hat{\mathbb{P}}_{\mathbf{X}})$ as graphical model.

Apart from that, we can also sample the learned GP's f_i or alternatively take the mean at a certain number of test points, i.e. a resolution, and interpolate the function values between these knots in order to gain deterministic estimates of the latent f_i^u. Then we have a normal SCM according to the first definition 2.8 and all results for SCM obviously hold.

4.1.2 Learning The True GP-SCM

We assume there is a true mechanism that generates the data, i.e. we assume an underlying **true SCM** $\mathfrak{S}^t$ giving rise to the true joint distribution $\mathbb{P}_{\mathbf{X}}$ with associated **true graph** $\mathcal{G}^t$. We want to find a **true GP-SCM** $\mathfrak{S}_{\mathcal{GP}} = (\mathfrak{S}^t, \mathbf{f})$ with the underlying true SCM modeled using the Gaussian processes $\mathbf{f}$ and associated true graph $\mathcal{G}^t$.

Given data from the true joint distribution $\mathbb{P}_{\mathbf{X}}$ (and possibly interventional distributions stemming from that) and considering a set of possible graphs G including the true graph $\mathcal{G}^t \in \mathsf{G}$, this task turns into estimating the true graph $\mathcal{G}^t$ and simultaneously modeling the latent functional dependencies as described by our estimate $\hat{\mathcal{G}}^t$ with GPR. We thus deal with graphical models $(\mathcal{G}, \mathbb{P}_{\mathbf{X}})$.

For the definition of a likelihood score over an entire fixed graph structure $\mathcal{G} \in \mathsf{G}$ and simultaneously learning the GPR models f_i as described in Section 3.3 we use the marginal likelihoods under the GP prior

$$\hat{p}(\mathbf{x}_i|\mathbf{p\bar{a}}_i{}^{\mathcal{G}}) = \mathcal{N}\left(\mathbf{x}_i|\mathbf{0}, K_{\mathbf{p\bar{a}}_i{}^{\mathcal{G}},\mathbf{p\bar{a}}_i{}^{\mathcal{G}}} + \sigma_i^2\,\mathrm{I}\right) \tag{4.5}$$

as further discussed in the next Section 4.2.

4.2 Computing A Likelihood Score Given Structure

The full graph likelihood derived from the Markov factorization for n iid. samples of the variables $X_i^{(j)}$ from the observational joint distribution $\mathbb{P}_{\mathbf{X}}$, $j = 1, ..., n$ reads

$$\mathcal{L}_{obs}^{\mathcal{G}} = \prod_{j=1}^{n} \prod_{i=1}^{d} p(x_i^{(j)} | \mathbf{pa}_i^{(j),\mathcal{G}}) \tag{4.6}$$

where $x_i^{(j)} \in \mathbb{R}$, $\mathbf{pa}_i^{(j),\mathcal{G}} \in \mathbb{R}^D$.

If we have data from different intervention scenarios for interventions of assigning independent random "noise" variables $\tilde{N}_{i_l}^k$ with densities $p(\tilde{n}_{i_l}^{(k)})$ to a subset of variables, where we denote $\mathcal{I}_k = \{i_l, l = 1, ..., L_k\}$ as the index set of variables intervened on in the k^{th} of K interventions in total each realized with n_k samples of the form

$$X_i^{(k,j)} \sim \mathbb{P}_{X_i}^{\text{do}\left(X_{i_l} := \tilde{N}_{i_l}^k; X_{i_l} \in \mathcal{I}_k\right)} \tag{4.7}$$

generated from the joint density of the respective interventional distribution with respective parents $\mathbf{PA}_i^{(k,j)}$. Set always $\mathcal{I}_0 = \emptyset$, i.e. we have observational data stemming from $\mathbb{P}_{\mathbf{X}}$. In total we have data

$$\mathcal{D} = ((X_1^{(k,j)}, ..., X_d^{(k,j)}); j = 1, ..., n;\ k = 0, ..., K) \sim \bigotimes_{k=0}^{K} \bigotimes_{j=1}^{n_k} \mathbb{P}_{\mathbf{X}}^{\text{do}\left(X_{i_l} := \tilde{N}_{i_l}; X_{i_l} \in \mathcal{I}_k\right)} \tag{4.8}$$

By multiplying the likelihoods - see Eaton and Murphy (2007)[p. 2] for a way to still speak of a "real" likelihood by using special intervention nodes to condense the various interventional distributions into one distribution with the same result as here, i.e. we multiply the different likelihoods - from the various interventional distributions we get a total likelihood score on the level of single samples as in (4.6)

$$\mathcal{L}^{\mathcal{G}} = \prod_{k=0}^{K} \prod_{j=1}^{n_k} \left(\prod_{i \notin \mathcal{I}_k} p(x_i^{(k,j)} | \mathbf{pa}_i^{(k,j),\mathcal{G}}) \times \prod_{i \in \mathcal{I}_k} p(\tilde{n}_i^{(k,j)}) \right) \tag{4.9}$$

If we re-arrange the product above and group not by interventions but variables on the highest level, we get

$$\mathcal{L}^{\mathcal{G}} = \left(\prod_{i=1}^{d} \prod_{k:\, i \notin \mathcal{I}_k} \prod_{j=1}^{n_k} p(x_i^{(k,j)} | \mathbf{pa}_i^{(k,j),\mathcal{G}}) \right) \times \left(\prod_{i=1}^{d} \prod_{k:\, i \in \mathcal{I}_k} \prod_{j=1}^{n_k} p(\tilde{n}_i^{(k,j)}) \right) \tag{4.10}$$

The right-hand factor as likelihood based on densities of our chosen intervention noises doesn't matter for our purpose of scoring a graph structure along with any associated parameters $\boldsymbol{\theta}$, neither of which the factor depends on.

For the left hand factor above, from now on denoted as $\mathcal{LS}^{\mathcal{G}}$, we have the product over all variables of

$$\mathcal{LS}_i^{\mathcal{G}} = \prod_{k:\, i \notin \mathcal{I}_k} \prod_{j=1}^{n_k} p(x_i^{(k,j)} | \mathbf{pa}_i^{(k,j),\mathcal{G}}) \tag{4.11}$$

For (4.11) we adopt a term of Koller and Friedman (2009)[pp. 723]: **local likelihood**, i.e. a function representing the "prediction quality" of the variable given its parents. We aim to model these local likelihoods in the case of X_i not being a source with the GP marginal likelihoods:

$$\begin{aligned} \mathcal{LS}_i^{\mathcal{G}} &= \prod_{k:\, i \notin \mathcal{I}_k} \prod_{j=1}^{n_k} \frac{p(x_i^{(k,j)}, \mathbf{pa}_i^{(k,j),\mathcal{G}})}{p(x_i^{(k,j)})} = \frac{\prod_{k:\, i \notin \mathcal{I}_k} \prod_{j=1}^{n_k} p(x_i^{(k,j)}, \mathbf{pa}_i^{(k,j),\mathcal{G}})}{\prod_{k:\, i \notin \mathcal{I}_k} \prod_{j=1}^{n_k} p(x_i^{(k,j)})} \\ &= \frac{p(\mathbf{x}_i, \mathbf{p\bar{a}}_i{}^{\mathcal{G}})}{p(\mathbf{x}_i)} = p(\mathbf{x}_i | \mathbf{p\bar{a}}_i{}^{\mathcal{G}}) \end{aligned} \tag{4.12}$$

and under the GP prior assumption we model

$$\hat{p}(\mathbf{x}_i | \mathbf{p\bar{a}}_i{}^{\mathcal{G}}) = \mathcal{N}\left(\mathbf{0}, K_{\mathbf{p\bar{a}}_i{}^{\mathcal{G}}, \mathbf{p\bar{a}}_i{}^{\mathcal{G}}} + \sigma_i^2 \, \mathrm{I}\right) \tag{4.13}$$

where we collect data out- $x_i^{(k,j)}$ and inputs $\mathbf{pa}_i^{(k,j),\mathcal{G}}$ for the correct indices (k,j) in $\mathbf{x}_i$ and $\mathbf{p\bar{a}}_i{}^{\mathcal{G}}$ respectively. Equation (4.12) is justified through the independence assumption on the samples in an almost everywhere sense with respect to the product measure of all samples (see e.g. Schmidt (2011)[p. 310]).

For modeling the likelihood of the source nodes von Kügelgen et al. (2019)[p. 5] propose simple normal or more expressive distributions. In Section 4.4 We will separate the estimation of distributions on the sources from graph and GPR estimation for the non-sources.

4.3 Learning The GP-SCM In A Bayesian Way

For this section see Heckerman et al. (1999), Peters et al. (2017)[pp. 149], Rasmussen and Williams (2006)[pp. 18], von Kügelgen et al. (2019).

We now explore a Bayesian approach to learning the GP-SCM as graphical model (similar to von Kügelgen et al. (2019) without the intervention choice scheme). Observing data $\mathcal{D}$ as in (4.8) and considering multiple possible graph structures $\mathsf{G} = \{\mathcal{G}_1, ..., \mathcal{G}_M\}$ for a given graphical model $\mathfrak{G}_m = (\mathcal{G}_m, \mathbb{P}_{\mathbf{X}})$ we define a Bayesian score with a posterior density

$$p(\mathcal{G}_m | \mathcal{D}) = \frac{p(\mathcal{D} | \mathcal{G}_m) p(\mathcal{G}_m)}{\sum_{m=1}^{M} p(\mathcal{D} | \mathcal{G}_m) p(\mathcal{G}_m)} \tag{4.14}$$

where we have some *graph priors* $p(\mathcal{G}_m)$ $(= 1/M$ for no prior knowledge) and *graph marginal likelihoods* $p(\mathcal{D}|\mathcal{G}_m)$, which due to the Markov factorization in the GP-SCM as

graphical model (4.10) compute with the application of atomic interventions (see Definition 2.12) with $\mathbf{x}_i$ and $\mathbf{p\bar{a}}_i^{\mathcal{G}_m}$ as in the previous section as

$$\begin{aligned} p(\mathcal{D}|\mathcal{G}_m) = \mathcal{L}^{\mathcal{G}_m} &= \prod_{i=1}^{d} \prod_{k:\, i \notin \mathcal{I}_k} \prod_{j=1}^{n_k} p(x_i^{(k,j)}|\mathbf{pa}_i^{(k,j),\mathcal{G}_m}) = \prod_{i=1}^{d} p(\mathbf{x}_i|\mathbf{p\bar{a}}_i^{\mathcal{G}_m}) \\ &= \prod_{i \notin \mathbf{SO}_{\mathcal{G}_m}} \int p(\mathbf{x}_i|f_{\mathbf{p\bar{a}}_i^{\mathcal{G}_m}}, \mathbf{p\bar{a}}_i^{\mathcal{G}_m})\; p(f_{\mathbf{p\bar{a}}_i^{\mathcal{G}_m}}|\mathbf{p\bar{a}}_i^{\mathcal{G}_m})\; df_{\mathbf{p\bar{a}}_i^{\mathcal{G}_m}} \times \prod_{i \in \mathbf{SO}_{\mathcal{G}_m}} p(\mathbf{x}_i) \\ &= \prod_{i \notin \mathbf{SO}_{\mathcal{G}_m}} \mathcal{N}\left(\mathbf{x}_i|\mathbf{0}, K_{\mathbf{p\bar{a}}_i^{\mathcal{G}_m}, \mathbf{p\bar{a}}_i^{\mathcal{G}_m}} + \sigma_i^2\, \mathrm{I}\right) \times \prod_{i \in \mathbf{SO}_{\mathcal{G}_m}} p(\mathbf{x}_i) \end{aligned} \tag{4.15}$$

with $f_{\mathbf{p\bar{a}}_i^{\mathcal{G}_m}}$ corresponding to the random vector of function values (3.13) at $\mathbf{p\bar{a}}_i^{\mathcal{G}_m}$ and for likelihood and GP prior

$$\begin{aligned} p(\mathbf{x}_i|f_{\mathbf{p\bar{a}}_i^{\mathcal{G}_m}}, \mathbf{p\bar{a}}_i^{\mathcal{G}_m}) &= \mathcal{N}\left(\mathbf{x}_i|f_{\mathbf{p\bar{a}}_i^{\mathcal{G}_m}}, \sigma_i^2\, \mathrm{I}\right) \\ p(f_{\mathbf{p\bar{a}}_i^{\mathcal{G}_m}}|\mathbf{p\bar{a}}_i^{\mathcal{G}_m}) &= \mathcal{N}\left(f_{\mathbf{p\bar{a}}_i^{\mathcal{G}_m}}|\mathbf{0}, K_{\mathbf{p\bar{a}}_i^{\mathcal{G}_m}, \mathbf{p\bar{a}}_i^{\mathcal{G}_m}}\right) \end{aligned} \tag{4.16}$$

(see Rasmussen and Williams (2006)[p. 19]).

For the sources the single likelihoods $p(\mathbf{x}_i)$ can be modeled in some arbitrary, suitable fashion. The parameters stemming from these likelihoods as well as the hyperparameters from the normal GP likelihoods in (4.15) are all collected in $\boldsymbol{\theta}_{\mathcal{G}_m}$ such that we have for a lower level in Bayesian model selection

$$p(\mathcal{D}|\mathcal{G}_m) = \int p(\mathcal{D}|\boldsymbol{\theta}_{\mathcal{G}_m}, \mathcal{G}_m)\; p(\boldsymbol{\theta}_{\mathcal{G}_m}|\mathcal{G}_m)\; d\boldsymbol{\theta}_{\mathcal{G}_m} \tag{4.17}$$

with some parameter prior distribution $p(\boldsymbol{\theta}_{\mathcal{G}_m}|\mathcal{G}_m)$ and parameter dependent likelihoods

$$p(\mathcal{D}|\boldsymbol{\theta}_{\mathcal{G}_m}, \mathcal{G}_m) = \prod_{i \notin \mathbf{SO}_{\mathcal{G}_m}} \mathcal{N}\left(\mathbf{x}_i|\mathbf{0}, K_{\mathbf{p\bar{a}}_i^{\mathcal{G}_m}, \mathbf{p\bar{a}}_i^{\mathcal{G}_m}, \boldsymbol{\theta}_{\mathcal{G}_m}} + \sigma^2_{i,\boldsymbol{\theta}_{\mathcal{G}_m}}\, \mathrm{I}\right) \times \prod_{i \in \mathbf{SO}_{\mathcal{G}_m}} p_{\boldsymbol{\theta}_{\mathcal{G}_m}}(\mathbf{x}_i) \tag{4.18}$$

Using the ML-II approach (Section 3.3) we avoid the integration in (4.17) with MAP assuming flat priors $p(\boldsymbol{\theta}_{\mathcal{G}_m}|\mathcal{G}_m)$.

When we apply this Bayesian score (4.14) in simulations under the small data paradigm we can observe that the graph proposed by the score tends to favor more connections (edges) than the true graph features if we choose a method for the source likelihoods that has a low fitting capacity (e.g. simple univariate normal distributions) with GPR for the non-sources being a rather flexible technique.

If we have for example the situation of two variables and look at two graphs along with the respective (Markov) factorizations of the observational joint distribution

$$\begin{aligned} \mathcal{G}^1: &\quad X \to Y \quad p(x,y) = p(x)p(y|x) \\ \mathcal{G}^2: &\quad X \perp\!\!\!\perp Y \quad p(x,y) = p(x)p(y) \end{aligned} \tag{4.19}$$

we see that if the first graph is the true graph, the respective likelihoods (4.15) differ based on modeling $p(y|x)$ vs modeling $p(y)$ - also in any interventional situation: for intervening on Y we model $p(x)$ in both cases and intervening on X we again differ in modeling $p(y|x)$ vs $p(y)$. These are different tasks which can be fulfilled with various degrees of quality: Choosing a highly adaptive method for modeling $p(y|x)$ and some restrictive technique for the sources will yield a different result than vice versa.

Furthermore, we need to recognize the dependence on prior assumptions in Bayesian modeling, that is our choices of modeling the non-sources with GPR and the sources in some other way constitute beliefs that influence the posterior outcome of the Bayesian prediction. As the definition of adequate prior beliefs seems to be difficult in this scenario and to further our understanding of how we learn form interventions, we decide go in the direction of an alternative score that utilizes the interventions in a direct manner.

4.4 An Alternative Approach To Learning The GP-SCM

We adopt the notation of the previous sections including data $\mathcal{D}$ as in (4.8) and the setting of k interventions $\mathcal{I}_k$ using independent random variables.

The question of how exactly algorithms utilize interventions in order to discover causal structure in a framework that is (at least somewhat) universally applicable is hard to answer after the study of relevant literature. Mostly, interventional data is exploited in an implicit way such as in our Bayesian score of the previous section or questions of *experimental design* are answered, which, as von Kügelgen et al. (2019)[p. 5] already remark in their paper, stay on the distribution level and tend to neglect the interplay of experiment and algorithm. With our alternative approach, however, we want to start thinking about the former question of how interventions aid the causal discovery.

First, we now turn our attention to a special relationship the true graph has with the Markov factorization or, more precisely, its Markov kernels:

Theorem 4.2 (**Only True Graph Certainly Retains Markov Kernels**)**.**
For the true graph $\mathcal{G} = \mathcal{G}^t$ we have for any intervention k with $i \notin \mathcal{I}_k$ and for any $j = 1, ..., n_k$ that the conditional density of $X_i^{(k,j)}$ given $\boldsymbol{PA}_i^{(k,j),\mathcal{G}} = \boldsymbol{pa}_i^{(k,j),\mathcal{G}}$

$$p\left(x_i^{(k,j)}|\boldsymbol{pa}_i^{(k,j),\mathcal{G}}\right) = p\left(x_i^{(0,j)}|\boldsymbol{pa}_i^{(0,j),\mathcal{G}}\right) \tag{4.20}$$

pointwise (i.e. for $x_i^{(k,j)} = x_i^{(0,j)} \in \mathbb{R}$ and $\boldsymbol{pa}_i^{(k,j),\mathcal{G}} = \boldsymbol{pa}_i^{(0,j),\mathcal{G}} \in \mathbb{R}^{D_i}$), i.e. it is pointwise equal to the conditional density of $X_i^{(0,j)}$ given $\boldsymbol{PA}_i^{(0,j),\mathcal{G}} = \boldsymbol{pa}_i^{(0,j),\mathcal{G}}$, and furthermore that $X_i^{(k,j)}$ is independent of none of its parents $\boldsymbol{PA}_i^{(k,j),\mathcal{G}}$. For any other graph $\mathcal{G} \neq \mathcal{G}^t$ this doesn't hold in general.

Proof. For the case $\mathcal{G} = \mathcal{G}^t$ we recognize that we obtain our observational samples ($k = 0$) $(X_1^{(0,j)}, ..., X_d^{(0,j)})^T \sim \mathbb{P}_{\mathbf{X}}$ for any $j = 1, ..., n_0$ with the hierarchical procedure of generating samples according to the true SCM $\mathfrak{S}^t$ (see Section 2.8, Theorem 2.9) with associated true

graph $\mathcal{G}^t$. Considering an intervention ($k > 0$) we have the analogous procedure for the respective interventional true SCM $\tilde{\mathfrak{S}}^t$, i.e. the altered true SCM for the executed intervention. Hence, we always have that in the hierarchical sample generation we can sample noises of the jointly independent $\mathbb{P}_{\mathbf{N}}$ and intervention noises $\mathbb{P}_{\tilde{\mathbf{N}}}$ earlier that need to be realized first by the given (graph) hierarchy. Therefore, we can observe a situation in the generation process with $\mathbf{AN}_i^{(k,j),\mathcal{G}^t}$ and thus $\mathbf{PA}_i^{(k,j),\mathcal{G}^t}$ realized but $X_i^{(k,j)}$ (i.e. its respective noise $N_i^{(k,j)}$) not yet realized. For $k : i \notin \mathcal{I}_k$ we now have, given $\mathbf{PA}_i^{(k,j),\mathcal{G}^t} = \mathbf{pa}_i^{(k,j),\mathcal{G}^t}$, that (4.20) holds as the respective Markov kernel did not change. Additionally, $X_i^{(k,j)}$ is never independent of any parent $\mathbf{PA}_i^{(k,j),\mathcal{G}^t}$ by structural minimality, i.e. we speak of "real" parents in the ground truth situation.

For any other graph structure $\mathcal{G} \neq \mathcal{G}^t$, we are in at least one of three situations

(A) $\mathcal{G}^t$ has an edge (v_p, v_q) that $\mathcal{G}$ doesn't have and neither its reverse
(B) $\mathcal{G}$ has an edge (v_p, v_q) that $\mathcal{G}^t$ doesn't have and neither its reverse
(C) there exists an edge (v_q, v_p) in $\mathcal{G}^t$ and (v_p, v_q) in $\mathcal{G}$

We select one of these situations and an intervention k^s such that we intervene on X_p in our usual style assigning independent noise $\tilde{N}_p^{k^s}$, i.e. $p \in \mathcal{I}_{k^s}$. True SCM and graph dictate our calculations. The statement of the theorem is then proven with the respective counterexamples in Example 4.3.

□

Example 4.3. *For a situation of two variables X_1, X_2 we have the following concrete counterexamples for graphs $\mathcal{G} \neq \mathcal{G}^t$ in the previous Theorem 4.2:*

(A) For the two graphs with respective observational joint densities (any sample j)

$$\begin{aligned} \mathcal{G}^t: &\quad X_1 \to X_2 \quad p_{\mathcal{G}^t}(x_1^{(0,j)}, x_2^{(0,j)}) = p(x_1^{(0,j)})p(x_2^{(0,j)}|x_1^{(0,j)}) \\ \mathcal{G}: &\quad X_1 \perp\!\!\!\perp X_2 \quad p_{\mathcal{G}}(x_1^{(0,j)}, x_2^{(0,j)}) = p(x_1^{(0,j)})p(x_2^{(0,j)}) \end{aligned} \tag{4.21}$$

If we intervene in the k^s-th intervention on X_1, $\mathcal{I}_{k^s} = \{1\}$, with $X_1 := \tilde{N}_1^{k^s}$ leading to the following respective interventional joint densities

$$\begin{aligned} \mathcal{G}^t: &\quad p_{\mathcal{G}^t}^{\mathrm{do}\left(X_1 := \tilde{N}_1^{k^s}\right)}(x_1^{(k^s,j)}, x_2^{(k^s,j)}) = p(\tilde{n}_1^{(k^s,j)})p(x_2^{(k^s,j)}|\tilde{n}_1^{(k^s,j)}) \\ \mathcal{G}: &\quad p_{\mathcal{G}}^{\mathrm{do}\left(X_1 := \tilde{N}_1^{k^s}\right)}(x_1^{(k^s,j)}, x_2^{(k^s,j)}) = p(\tilde{n}_1^{(k^s,j)})p(x_2^{(k^s,j)}) \end{aligned} \tag{4.22}$$

As we actually sample from the true graph $\mathcal{G}^t$ (via hierarchical sampling with the true SCM), we see that we can quite easily construct a case, where $p(x_2^{(k^s,j)}) \neq p(x_2^{(0,j)})$: if e.g. $X_1 \sim \mathcal{N}(0,3)$, $X_2 := X_1 + \mathcal{N}(0,1)$, we have choosing an intervention $X_1 := \tilde{N}_1^{k^s} \sim \mathcal{N}(2,4)$ that $p(x_2^{(k^s,j)}) = \mathcal{N}(2,5) \neq \mathcal{N}(0,4) = p(x_2^{(0,j)})$.

(B) For the two graphs with respective observational joint densities (any sample j)

$$\begin{aligned}\mathcal{G}^t: &\quad X_1 \perp\!\!\!\perp X_2 \quad p_{\mathcal{G}^t}(x_1^{(0,j)}, x_2^{(0,j)}) = p(x_1^{(0,j)})p(x_2^{(0,j)}) \\ \mathcal{G}: &\quad X_1 \to X_2 \quad p_{\mathcal{G}}(x_1^{(0,j)}, x_2^{(0,j)}) = p(x_1^{(0,j)})p(x_2^{(0,j)}|x_1^{(0,j)}) \overset{indep.}{=} p(x_1^{(0,j)})p(x_2^{(0,j)})\end{aligned} \tag{4.23}$$

If we intervene in the k^s-th intervention on X_1, $\mathcal{I}_{k^s} = \{1\}$, with $X_1 := \tilde{N}_1^{k^s}$ leading to the following respective interventional joint densities

$$\begin{aligned}\mathcal{G}^t: &\quad p_{\mathcal{G}^t}^{\text{do}\left(X_1:=\tilde{N}_1^{k^s}\right)}(x_1^{(k^s,j)}, x_2^{(k^s,j)}) = p(\tilde{n}_1^{(k^s,j)})p(x_2^{(k^s,j)}) \\ \mathcal{G}: &\quad p_{\mathcal{G}}^{\text{do}\left(X_1:=\tilde{N}_1^{k^s}\right)}(x_1^{(k^s,j)}, x_2^{(k^s,j)}) = p(\tilde{n}_1^{(k^s,j)})p(x_2^{(k^s,j)}|\tilde{n}_1^{(k^s,j)}) \overset{indep.}{=} p(\tilde{n}_1^{(k^s,j)})p(x_2^{(k^s,j)})\end{aligned} \tag{4.24}$$

In both cases we see the effects of X_i being independent of its parent in $\boldsymbol{PA}_i^{(k,j),\mathcal{G}}$, $(k = 0, k^s)$ namely X_2.

(C) For the two graphs with respective observational joint densities (any sample j)

$$\begin{aligned}\mathcal{G}^t: &\quad X_1 \leftarrow X_2 \quad p_{\mathcal{G}^t}(x_1^{(0,j)}, x_2^{(0,j)}) = p(x_1^{(0,j)}|x_2^{(0,j)})p(x_2^{(0,j)}) \\ \mathcal{G}: &\quad X_1 \to X_2 \quad p_{\mathcal{G}}(x_1^{(0,j)}, x_2^{(0,j)}) = p(x_1^{(0,j)})p(x_2^{(0,j)}|x_1^{(0,j)})\end{aligned} \tag{4.25}$$

If we intervene in the k^s-th intervention on X_1, $\mathcal{I}_{k^s} = \{1\}$, with $X_1 := \tilde{N}_1^{k^s}$ leading to the following respective interventional joint densities

$$\begin{aligned}\mathcal{G}^t: &\quad p_{\mathcal{G}^t}^{\text{do}\left(X_1:=\tilde{N}_1^{k^s}\right)}(x_1^{(k^s,j)}, x_2^{(k^s,j)}) = p(\tilde{n}_1^{(k^s,j)})p(x_2^{(k^s,j)}) \\ \mathcal{G}: &\quad p_{\mathcal{G}}^{\text{do}\left(X_1:=\tilde{N}_1^{k^s}\right)}(x_1^{(k^s,j)}, x_2^{(k^s,j)}) = p(\tilde{n}_1^{(k^s,j)})p(x_2^{(k^s,j)}|\tilde{n}_1^{(k^s,j)}) \overset{indep.}{=} p(\tilde{n}_1^{(k^s,j)})p(x_2^{(k^s,j)})\end{aligned} \tag{4.26}$$

For e.g. $X_2 = \mathcal{N}(0,3)$, $X_1 := X_2 + \mathcal{N}(0,1)$ and given $x_1^{(0,j)} = 0$ we have that $p(x_2^{(k^s,j)}|\tilde{n}_1^{(k^s,j)}) = p(x_2^{(k^s,j)}) = \mathcal{N}(0,3) \neq \mathcal{N}(0,1) = p(x_2^{(0,j)}|x_1^{(0,j)})$.

Given a graph $\mathcal{G}$ we consider all nodes if we look at sources $\mathbf{SO}_\mathcal{G}$ and dependants $\mathbf{DP}_\mathcal{G}$. For sources we have

Corollary 4.4 (**Falsifying Proposed Sources**)**.**
For $X_q \in \boldsymbol{SO}_\mathcal{G}$ we have $X_q \notin \boldsymbol{SO}_{\mathcal{G}^t}$ if there exists an intervention k^s with $q \notin \mathcal{I}_{k^s}$ such that (for any sample j) $p\left(x_q^{(k^s,j)}\right) \neq p\left(x_q^{(0,j)}\right)$ pointwise for $x_q^{(k^s,j)} = x_q^{(0,j)} \in \mathbb{R}$.

and for dependants

Corollary 4.5 (**Falsifying Proposed Dependencies**).
For $X_q \in \boldsymbol{DP}_{\mathcal{G}}$ and considering an edge $(v_p, v_q) \in E_{\mathcal{G}}$ in the graph $\mathcal{G}$ we have that $(v_p, v_q) \notin E_{\mathcal{G}^t}$, i.e. the edge is not in the true graph $\mathcal{G}^t$, if there exists an intervention k^p with $q \notin \mathcal{I}_{k^p}$ such that (for any sample j)

$$p\left(x_i^{(k^p,j)} |\, \boldsymbol{PA}_i^{(k^p,j),\mathcal{G}} = \boldsymbol{pa}_i^{(k^p,j),\mathcal{G}}\right) \neq p\left(x_i^{(0,j)} |\, \boldsymbol{PA}_i^{(0,j),\mathcal{G}} = \boldsymbol{pa}_i^{(0,j),\mathcal{G}}\right) \tag{4.27}$$

pointwise (i.e. for $x_i^{(k^p,j)} = x_i^{(0,j)} \in \mathbb{R}$ and $\boldsymbol{pa}_i^{(k^p,j),\mathcal{G}} = \boldsymbol{pa}_i^{(0,j),\mathcal{G}} \in \mathbb{R}^{D_i}$) or alternatively $X_q \perp\!\!\!\perp X_p$.

In practice, we try to utilize these corollaries by proposing (for many purposes) reasonable assumptions in the following ways:

For **sources** we implement Corollary 4.4 in an *ad-hoc* style by choosing the interventions k^s on all other variables X_p simultaneously with a large variance gap

$$\text{" } \mathbb{V}[X_q] \approx \mathbb{V}[X_p] >> \mathbb{V}[\tilde{N}_p^{k^s}] \text{ "} \tag{4.28}$$

Then it is sensible to assume that with a true, latent functional assignment $X_q := f_q^u(..., X_p, ...) + N_q$ we may observe *ad-hoc* that " $\mathbb{V}[X_q^{(k^s,j)}] << \mathbb{V}[X_q^{(0,j)}]$ " (for any sample j) as we don't expect large variation from the additive noise N_q.

For **dependencies** we implement Corollary 4.5 in an *ad-hoc* style by again assuming that we have noises N_q with small variances around the true, latent functions f_q^u compared to the variation of the actual variables X_q, i.e.

$$\text{" } \mathbb{V}[X_q] >> \mathbb{V}[X_q | \tilde{\mathbf{PA}}_q = \tilde{\mathbf{pa}}_q] >> \mathbb{V}[X_q |\, \mathbf{PA}_q = \mathbf{pa}_q] = \mathbb{V}[N_q] \text{ "} \tag{4.29}$$

with $\tilde{\mathbf{PA}}_q \subsetneq \mathbf{PA}_q$.

With these deliberations we decide to formulate a score. For this purpose in order to detect the correct dependencies (i.e. edges for a given dependant) we intervene on every variable X_i, $i = 1, ..., d$ with $\text{do}(X_i := \tilde{N}_i^k)$ individually in intervention $k \in K_1 = \{1, ..., d\}$, where $\tilde{N}_i^k \sim \mathcal{U}(-s_i^l, s_i^r)$ is uniformly distributed on an estimated "support interval" $(-s_l, s_r) \subset \mathbb{R}$ we typically observe for (most samples of) X_i. To detect the correct sources we intervene on all combinations of $d-1$ variables X_i with $(k = d + i)$-th intervention $\text{do}(..., X_{i-1} := c_{i-1}, X_{i+1} := c_{i+1}, ...)$, $k \in K_2 = \{d+1, ..., 2d\}$, where $c_i \in (-s_i^l, s_i^r) \subset \mathbb{R}$ is some constant value in a respective support interval for X_i. Apart from that we still have observational data ($k = 0$) and in total $K = 2d$ interventions.

We want to start with the basic format of the graph likelihood score $\mathcal{LS}^{\mathcal{G}}$ as product of local likelihoods $\mathcal{LS}_i^{\mathcal{G}}$ (4.10), (4.11). As in the Bayesian score (4.18) we make a distinction between dependants and sources.

To deal with **dependencies**, we use data from interventions $k \notin K_2$ to estimate the dependencies with GPR as in (4.13). If a dependency $p(\mathbf{x}_i|\mathbf{p\bar{a}}_i{}^{\mathcal{G}})$ of a wrong graph $\mathcal{G} \neq \mathcal{G}^t$ that we want to model with GPR (4.13) has deficiencies as described in Corollary 4.5, our assumption (4.29) implies that the data we fit our GP-model on likely includes disturbed data points associated with a density that is not the conditional $p(x_i|\mathbf{pa}_i^{\mathcal{G}})$ given $\mathbf{PA}_i^{(k,j),\mathcal{G}} = \mathbf{pa}_i^{(k,j),\mathcal{G}}$ as we presume (see Figure 4.1) and thus leading to a worse GP fit as well as significantly lower GP marginal likelihood. In practice this way of modeling dependant/non-source local likelihoods is the same as in the Bayesian case (4.18) and thus the fitted GP models of $\hat{\mathcal{G}}^t$ can be used as estimations of the functional dependencies.

We model the **sources** *within the score for the sole purpose of detecting the graph structure, not estimating the (marginal) density of the variable*: in order to overcome the problem of the Bayesian score as described in (4.19) of the previous section, we try to create a symmetric situation with respect to the GP marginal likelihoods of the dependants by considering a **pseudo likelihood**, which is also just a GP marginal likelihood, that is fit on "artificial" data. This data for a source X_i as proposed by graph $\mathcal{G}$ consists of

1) observational samples of X_i, i.e. $X_i^{(0,j),\mathcal{G}}$ on the input side, paired with samples of noise $N_i^{obs} \sim \mathcal{N}(0, \epsilon_b)$, where we assume that the base variance ϵ_b is of the magnitude of typical regression variances of all considered variables X_q: " $\mathbb{V}[N_i^{obs}] \approx \mathbb{V}[N_q]$ "
2) samples from a uniform distribution on the respective support interval $\mathcal{U}(-s_i^l, s_i^r)$ on the input side, paired with samples of noise $N_i^{intv} \sim \mathcal{N}(0, \epsilon_t)$, where the variance $\epsilon_t = \epsilon_b + \epsilon_a$ with ϵ_b the same as in 1) and we have additional variance ϵ_a, which acts as a penalty that is low, if the sample variance of the samples $X_i^{(k,j),\mathcal{G}}$ with $k \in K_2, i \notin \mathcal{I}_k$, i.e. the interventions, where we fix every other variable on one value and intervene repeatedly, is high compared to the sample variance of observational data of the variable X_i indicating that we are in fact dealing with a source and vice versa.

If we now fit a GP model on those data points we have in fact a situation that is, given the various assumptions, symmetric in comparison to the situation with regular dependency GP models: for interventions we get data points for wrongly assumed edges, which are disturbed, i.e. stem from distributions with larger variance and thus lead to bad fits and low GP marginal likelihoods.

In total, we formulate a score (with learned hyperparameters for the involved GP marginal likelihoods via maximum likelihood as in the Bayesian score)

$$\mathcal{S}(\mathcal{G}) = \prod_{i \notin \mathbf{SO}_{\mathcal{G}}; k \notin K_2; j} \mathcal{N}\left(\mathbf{x}_i^{(k,j)} | \mathbf{0}, K_{\mathbf{p\bar{a}}_i{}^{(k,j),\mathcal{G}}, \mathbf{p\bar{a}}_i{}^{(k,j),\mathcal{G}}} + \sigma_i^2 \, \mathrm{I}\right) \times \prod_{i \in \mathbf{SO}_{\mathcal{G}}; k \notin K_1; j} q_i(\bar{\mathbf{x}}_i^{(k,j)}) \quad (4.30)$$

where $\bar{\mathbf{x}}_i^{(k,j)}$, $i \in \mathbf{SO}_{\mathcal{G}}; k \notin K_1$ represents the data as described in 1) and 2) above and q_i the function that yields the respective GP marginal likelihood as explained above.

We estimate the true graph structure by maximizing: $\hat{G}^t = \arg\max_{\mathcal{G}} \mathcal{S}(\mathcal{G})$. In order to get actual estimates for the marginal densities of the sources we can, after we identified the correct sources, use any method to accomplish that task. For the purpose of clarity

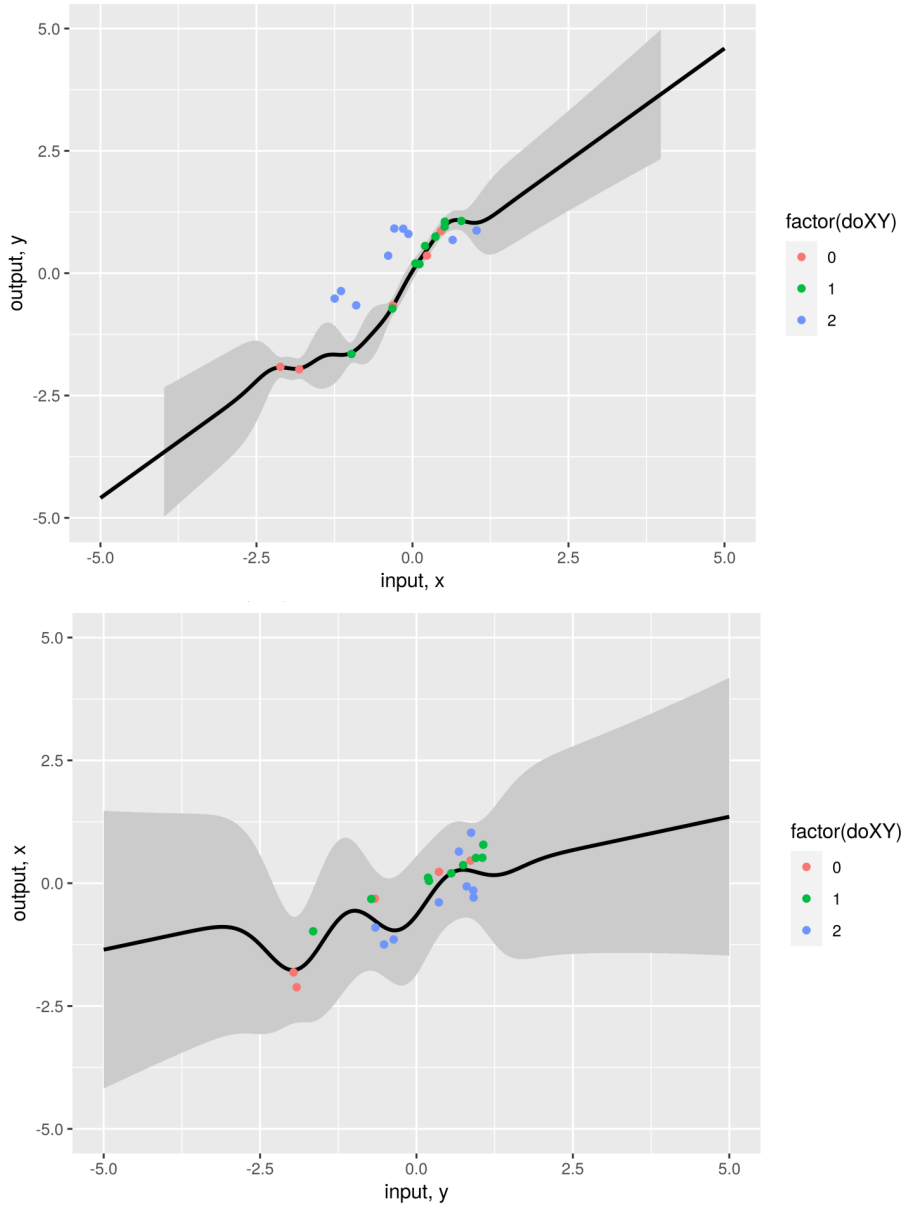

Figure 4.1: Example of two variables X, Y and a true graph $X \to Y$ with true assignments $X := N_1 \sim \mathcal{N}(0, 1)$ and $Y := \tanh(X) + N_2$; $N_2 \sim \mathcal{N}(0, 0.01)$. Orange samples stem from the observational distribution, green samples from an intervention on X and blue samples from an intervention on Y. The images show GP fits, i.e. black line: posterior mean, gray shade: area of the 95%-quantile for each marginal univariate normal distribution at respective input x or y. The upper image depicts a successful fit on observational samples and samples with intervention on input X. The lower image, shows a bad fit accompanied by a low GP marginal likelihood as we try to fit both observational samples and samples with intervention on input Y. The latter samples (blue) however are in reality associated with the marginal density of X at randomly sampled intervention values Y. See also Example 4.3 C). The plot was done with `ggplot2`, see Wickham (2016).

we stress that this proposal is merely an ad-hoc demonstration of how thinking along the lines of Theorem 4.2 can lead to causal structure identification methods.

5 Simulations

We consider a setting of 3 variables X_1, X_2, X_3 and a list of all possible models, i.e. graph structures, which is all 25 DAG's that exist for 3 variables:

```
models <- list(
  # v-structures
  graph(edges = c(1,3, 2,3), n = 3, directed = T),
  graph(edges = c(1,2, 3,2), n = 3, directed = T),
  graph(edges = c(2,1, 3,1), n = 3, directed = T),
  # reversed v-structures
  graph(edges = c(1,2, 1,3), n = 3, directed = T),
  graph(edges = c(2,1, 2,3), n = 3, directed = T),
  graph(edges = c(3,1, 3,2), n = 3, directed = T),
  # hierarchical lines
  graph(edges = c(1,2, 2,3), n = 3, directed = T),
  graph(edges = c(1,3, 3,2), n = 3, directed = T),
  graph(edges = c(2,1, 1,3), n = 3, directed = T),
  graph(edges = c(2,3, 3,1), n = 3, directed = T),
  graph(edges = c(3,1, 1,2), n = 3, directed = T),
  graph(edges = c(3,2, 2,1), n = 3, directed = T),
  # v-structures plus parent dependency
  graph(edges = c(1,2, 1,3, 2,3), n = 3, directed = T),
  graph(edges = c(1,3, 1,2, 3,2), n = 3, directed = T),
  graph(edges = c(2,1, 2,3, 1,3), n = 3, directed = T),
  graph(edges = c(2,3, 2,1, 3,1), n = 3, directed = T),
  graph(edges = c(3,1, 3,2, 1,2), n = 3, directed = T),
  graph(edges = c(3,2, 3,1, 2,1), n = 3, directed = T),
  # one independent variable
  graph(edges = c(1,2), n = 3, directed = T),
  graph(edges = c(2,1), n = 3, directed = T),
  graph(edges = c(1,3), n = 3, directed = T),
  graph(edges = c(3,1), n = 3, directed = T),
  graph(edges = c(2,3), n = 3, directed = T),
  graph(edges = c(3,2), n = 3, directed = T),
  # all independent
  graph(edges = c(), n = 3, directed = T)
)
```

For all simulations we use a squared exponential kernel. All hyperparameters of the various GP marginal likelihoods are estimated with a maximum likelihood scheme using the function `optimr` of the identically named package, which automatically selects a well performing optimization method.

5.1 A Crucial First Simulation

We start with a troublesome case for the Bayesian score based on the issue discussed in Section 4.3 for a two variable example (4.19).

Here, we define a true graph $\mathcal{G}^t$ with structural assignments

$$\begin{aligned} X_1 &:= N_1 \sim \mathcal{N}(0,1) \\ X_2 &:= N_2 \sim \mathcal{N}(1,1) \\ X_3 &:= 2\tanh(X_1) + 4\tanh(X_2) + N_3,\ N_3 \sim \mathcal{N}(0, 0.01) \end{aligned} \tag{5.1}$$

Then, the Bayesian score with single variable interventions ($|\mathcal{I}_k| = 1$) as laid out in the appendix Chapter 7 with a normal distribution assumption on the sources and ML-II scheme to estimate (hyper)parameters yields posterior probabilities for the individual graphs

```
[1] 1.439681e-04 6.475304e-31 1.224263e-28 9.194583e-28 2.583109e-22
   7.404563e-27 3.650746e-24 2.517955e-26 6.505686e-26 4.116493e-22
   2.703855e-28 4.646381e-27 1.393418e-02
[14] 2.403088e-26 9.859219e-01 1.803989e-20 7.066772e-27 3.244935e-25
   2.477551e-32 1.753007e-30 9.499852e-30 2.793626e-30 3.771955e-26
   6.784823e-31 2.559808e-34
```

(see `BAYESIAN_SCORE_3Var_first_sim.RData`) with the true graph being the first one, we see that graph structures 13 and 15 outperform the true graph with larger probabilities. These are exactly the graphs with an additional edge between X_2 and X_3 compared to the true graph $\mathcal{G}^t$. We thus are dealing with a three variable case of the issue mentioned above from Section 4.3.

For the alternative score as laid out in the appendix Chapter 7 on the other hand we get considering (5.1) the following graph probabilities (from the likelihood score through normalization)

```
[1] 1.000000e+00 3.421358e-47 1.079388e-51 2.915308e-53 1.450137e-47
   9.935375e-76 7.857591e-45 6.797782e-53 4.094551e-57 8.422600e-54
   2.886526e-77 9.744980e-75 3.368771e-29
[14] 2.007097e-56 3.064108e-29 1.139806e-52 1.151110e-71 3.920842e-77
   6.022905e-45 4.926119e-52 3.221704e-26 1.374352e-43 2.549783e-21
   2.834248e-45 2.659305e-22
```

(see `ALTERNATIVE_SCORE_3Var_first_sim.RData`) i.e. the first and true graph has the largest probability/score compared to all other possible graphs.

5.2 Further Simulations

With the same setting as before we will now present further simulations for other ground truth functions and graph structures.

5.2.1 Another v-structure

With structural assignments

$$\begin{aligned} X_1 &:= N_1 \sim \mathcal{N}(-1, 1) \\ X_2 &:= N_2 \sim \mathcal{N}(0, 1) \\ X_3 &:= 3X_1^3 + 2X_2^2 + N_3,\ N_3 \sim \mathcal{N}(0, 0.1) \end{aligned} \tag{5.2}$$

we have the following graph probabilities for the Bayesian score

```
[1]  1.502097e-04 6.303566e-109 1.488111e-105  2.835263e-15  1.321781e
   -26 1.593816e-104  6.312110e-25  6.336669e-15  5.937152e-17  1.095274
   e-24 7.131332e-105 1.923424e-106
[13]  9.793420e-01  2.835265e-15  2.050780e-02  1.490129e-21 7.131336e
   -105 2.168400e-101 6.303563e-109 1.319991e-110  4.348674e-19 1.093791
   e-108  9.681398e-29 1.408814e-108
[25] 9.668288e-113
```

and for the alternative score

```
[1] 1.000000e+00 1.373201e-45 1.747991e-37 9.334440e-23 8.878353e-42
   3.528637e-54 1.916680e-47 4.412823e-26 1.029244e-18 5.103542e-43
   1.945122e-54 2.511776e-50 1.244737e-13
[14] 1.976179e-23 3.984552e-14 6.558364e-39 1.683596e-55 1.537473e-54
   4.696852e-43 1.291076e-39 3.191373e-12 1.798024e-47 3.154397e-36
   3.839792e-46 1.386127e-40
```

We see that we have the same situation as in the previous section: the Bayesian score has trouble not proposing the additional edge between the sources and with the alternative score we pick the correct graph.

5.2.2 Reversed v-structure

With structural assignments

$$\begin{aligned} X_1 &:= N_1 \sim \mathcal{N}(0, 1) \\ X_2 &:= 3\tanh(X_1) + N_2,\ N_2 \sim \mathcal{N}(0, 0.01) \\ X_3 &:= \tanh(X_1) + N_3,\ N_3 \sim \mathcal{N}(0, 0.01) \end{aligned} \tag{5.3}$$

we have the following graph probabilities for the Bayesian score

```
[1] 1.139379e-34 1.775000e-21 4.952725e-41 9.940379e-01 2.399730e-39
   1.429283e-38 1.221974e-13 4.116050e-26 1.952106e-26 8.076875e-39
   3.451760e-13 4.246560e-39 1.503564e-03
[14] 4.458576e-03 2.952721e-29 1.529329e-35 1.548224e-15 2.706299e-35
   3.957357e-19 7.771517e-45 7.532673e-32 2.615693e-44 9.259937e-45
   1.638638e-44 2.998827e-50
```

and for the alternative score

```
[1] 6.702415e-33 5.447379e-23 9.564067e-58 9.999994e-01 1.409147e-64
   2.781593e-64 2.032970e-15 9.021116e-24 8.681748e-46 7.441984e-68
   3.411806e-43 2.790289e-58 1.498713e-10
[14] 6.460225e-07 6.126305e-57 3.319391e-60 1.692261e-46 2.457876e-60
   6.547202e-23 3.645429e-65 4.591996e-31 1.499200e-63 1.239461e-45
   1.508226e-38 8.291508e-45
```

We see that in both cases we pick the correct graph [4].

5.2.3 Hierarchical Lines

With structural assignments

$$\begin{aligned} X_1 &:= N_1 \sim \mathcal{N}(0,1) \\ X_2 &:= \tanh(X_1) + N_2,\ N_2 \sim \mathcal{N}(0,0.01) \\ X_3 &:= 4\tanh(X_2) + N_3,\ N_3 \sim \mathcal{N}(0,0.01) \end{aligned} \tag{5.4}$$

we have the following graph probabilities for the Bayesian score

```
[1] 1.072843e-21 4.611478e-31 6.691595e-43 7.471541e-26 5.310373e-11
   8.111429e-35 9.992019e-01 1.835053e-35 3.970836e-36 8.680963e-13
   3.302623e-25 4.961975e-33 7.981335e-04
[14] 6.742877e-26 4.241772e-14 1.308512e-12 2.980534e-25 1.222664e-34
   5.109814e-31 2.715669e-41 1.004317e-43 4.439354e-43 1.343118e-18
   1.254999e-40 6.868563e-49
```

and for the alternative score

```
[1] 2.710151e-19 1.802531e-27 4.601883e-67 2.851785e-26 1.277507e-37
   4.100991e-65 9.997306e-01 4.045939e-43 3.296778e-67 8.326436e-41
   3.026273e-49 1.344298e-64 2.694167e-04
[14] 5.579268e-28 9.751427e-44 1.569520e-36 1.264085e-54 1.659134e-65
   5.176544e-26 9.876014e-63 3.796145e-46 2.874743e-71 3.952621e-19
   1.506601e-42 5.055249e-44
```

We see that in both cases we pick the correct graph [7].

5.2.4 v-structure Plus Parent Dependency

With structural assignments

$$\begin{aligned} X_1 &:= N_1 \sim \mathcal{N}(0,1) \\ X_2 &:= 3\tanh(X_1) + N_2,\ N_2 \sim \mathcal{N}(0,0.01) \\ X_3 &:= 2\tanh(X_1) + \tanh(X_2) + N_3,\ N_3 \sim \mathcal{N}(0,0.01) \end{aligned} \tag{5.5}$$

we have the following graph probabilities for the Bayesian score

```
[1] 2.422612e-33 1.968859e-35 6.384735e-56 8.312929e-14 3.749511e-44
   3.176197e-47 1.088819e-18 1.469480e-37 2.862680e-39 1.584076e-43
   1.796792e-23 7.518065e-48 1.000000e+00
[14] 5.470250e-18 3.443648e-26 2.323471e-43 1.182363e-27 4.658743e-47
   2.991999e-31 1.030339e-56 2.013900e-46 4.352929e-56 2.637787e-51
   5.288970e-55 7.248451e-64
```

and for the alternative score

```
[1] 1.645474e-24 1.815180e-27 1.358299e-67 4.098539e-12 5.703392e-64
   4.219067e-71 6.438697e-19 1.964621e-38 1.020953e-60 2.684915e-66
   1.055802e-46 3.997028e-67 1.000000e+00
[14] 1.745724e-14 9.633601e-47 1.002286e-61 1.890102e-51 2.066697e-69
   4.453063e-28 3.512293e-72 5.426607e-43 1.139626e-72 1.691772e-46
   9.247625e-49 2.820748e-48
```

We see that in both cases we pick the correct graph [13].

5.2.5 One Independent Variable

With structural assignments

$$\begin{aligned} X_1 &:= N_1 \sim \mathcal{N}(0,1) \\ X_2 &:= \tanh(X_1) + N_2,\ N_2 \sim \mathcal{N}(0,0.01) \\ X_3 &:= N_3 \sim \mathcal{N}(-1,1) \end{aligned} \tag{5.6}$$

we have the following graph probabilities for the Bayesian score

```
[1] 1.330589e-20 3.608666e-07 2.408651e-19 4.723143e-02 2.228811e-16
   1.675583e-14 4.723128e-02 9.256235e-16 2.228818e-16 1.121886e-16
   8.549934e-01 3.328822e-14 4.723139e-02
[14] 1.680818e-04 2.228816e-16 1.121880e-16 3.042653e-03 1.675576e-14
   1.014045e-04 4.785204e-19 1.330590e-20 2.408662e-19 1.330586e-20
   1.987286e-18 2.856736e-23
```

and for the alternative score

```
[1] 1.018193e-27 2.710557e-03 2.462031e-26 1.569613e-12 1.414364e-33
   1.368463e-27 4.157896e-14 3.116213e-25 5.266883e-32 6.170506e-41
   1.053954e-16 2.448860e-25 5.650785e-12
[14] 8.782095e-13 1.558345e-35 1.120082e-35 3.165140e-17 2.027592e-28
   9.972894e-01 6.300662e-28 4.024197e-25 6.046635e-25 1.048755e-26
   7.459849e-16 3.478308e-18
```

We see that the Bayesian score does not find the correct graph [19] unlike the alternative score.

5.2.6 All Variables Independent

With structural assignments

$$\begin{aligned} X_1 &:= N_1 \sim \mathcal{N}(0,1) \\ X_2 &:= N_2 \sim \mathcal{N}(1,1) \\ X_3 &:= N_3 \sim \mathcal{N}(-1,1) \end{aligned} \tag{5.7}$$

we have the following graph probabilities for the Bayesian score

```
[1] 9.901551e-06 2.484159e-07 1.765437e-04 6.465786e-03 2.660833e-03
   2.782906e-02 2.671694e-05 1.335522e-03 6.439501e-01 1.541656e-02
   1.347314e-01 4.803178e-03 1.328244e-03
[14] 9.227994e-04 1.322844e-01 2.709851e-03 1.922892e-02 4.891662e-03
   1.740578e-06 1.733502e-04 4.819998e-05 1.004371e-03 1.991646e-07
   3.595202e-07 1.297535e-08
```

and for the alternative score

```
[1] 4.741912e-24 2.371493e-23 6.508741e-25 8.425139e-55 4.113894e-53
   5.093052e-53 5.380591e-53 2.580314e-53 1.498088e-50 6.428730e-53
   6.184357e-51 4.439157e-51 1.980049e-54
[14] 1.959407e-50 3.400456e-53 2.245854e-50 2.754802e-53 4.860573e-52
   4.315258e-30 2.462836e-26 1.218939e-24 6.669845e-24 2.295934e-28
   2.160237e-27 1.000000e+00
```

We see that the Bayesian score does not find the correct graph [25] unlike the alternative score.

6 Conclusion And Outlook

We have examined a Bayesian approach of causal structure learning with Gaussian process priors for the functional dependencies. In order to address issues that arise in this case we have explored an alternative idea of defining a score for causal discovery, which has been shown to solve the issue *under certain conditions* in practice.

This alternative approach may have the potential to be developed into an advanced theory of learning causal structure with Gaussian processes from interventional data using verifiable and practicable regularity assumptions as well as some kind of finite sample guarantees. Furthermore, the algorithm could be connected to a automatic GPR model construction machinery as in Duvenaud (2014) in an effort to make the approach suitable in scenarios of few a priori assumptions about the dependencies.

All in all we can conclude that in order to present a widely applicable causal structure learning algorithm in the described setting of generally few assumptions with small samples sizes a lot of further research needs to be conducted.

7 Appendix - Source Code

We start by integrating R packages `data.table` (Dowle and Srinivasan (2019)), `emulator` (Hankin (2005)), `gridExtra` (Auguie (2017)), `optimr` (Nash), `EstimationTools` (Mosquera and Hernandez (2019)), `kernlab` (Karatzoglou et al. (2004)), `igraph` (Csardi and Nepusz (2006)), `rje` (Evans (2020)), `purrr` (Henry and Wickham (2020)) and defining the ground truth circumstances:

```
library(data.table)
library(emulator)
library(gridExtra)
library(optimr)
library(EstimationTools)
library(kernlab)
library(igraph)
library(rje)
library(purrr)

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
#------------------- GROUND TRUTH SECTION --------------------
#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>


#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
#---------- Ground Truth Setup ----------
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~


# number of variables
n_var <- 3
all_var <- rep(1:n_var)
all_var_names <- c("x_1", "x_2", "x_3")

# choose ground truth causal graph
g_true <- graph(edges = c(1,3, 2,3), n = n_var, directed = T)

# plot graph
plot(g_true)

# adjaceny matrix, rows: outgoing edeges, columns: incoming edges ... of
    that node
adjm_true <- as.matrix(g_true[])

# choose ground truth SCM
scm_true <- list(X1 = function(n) rnorm(n, 0, 1),
                 X2 = function(n) rnorm(n, 1, 1),
```

```
                X3 = function(x1, x2, n) 2*tanh(x1) + 4*tanh(x2) +
                    rnorm(n, 0, 0.1))


#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
#---------- Available functions ----------
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~


#' generate n samples (x_1. ...) from the joint underlying distribution
#'
#' @param input SCM (either true observational or some interventional)
#' @param n sample size
#'
#' @return  the samples (x_1. ...) as data table
joint_sample <- function(scm, n){

  # evaluation of variables given the input SCM
  x1 = scm$X1(n)
  x2 = scm$X2(n)

  x <- list(x_1 = x1,
            x_2 = x2,
            x_3 = scm$X3(x1, x2, n))

  return(as.data.table(x))
}


#' generate n samples (default 1) from the interventional distribution P
    (.|do(X_i = x))
#'
#' @param scm_intv the SCM that's scheduled for intervention
#' @param i_do index of variable, that is being intervened on
#' @param x_do intervention value on X_i
#' @param n number of returned samples from the interventional SCM
#'
#' @return the samples (x_1. ...) as data table
Do_sample <- function(scm_intv, i_do, x_do, n = 1){

  # set the intervention variable to the intervention value
  body(scm_intv[[i_do]]) <- substitute(x_do)

  # get new samples
  smp <- joint_sample(scm = scm_intv, n = n)

  # tag the samples
  smp <- smp[,doX := i_do,]

  return(smp)
}
```

```
#' generate n(=1) sample(s) from the interventional distribution P(.|do(
    X_i = xi, X_j = xj))
#'
#' @param scm_intv the SCM that's scheduled for intervention
#' @param i_do index of 1st variable, that is being intervened on
#' @param xi_do intervention value on X_i
#' @param j_do index of 2nd variable, that is being intervened on
#' @param xj_do intervention value on X_j
#' @param n number of returned samples from the interventional SCM
#'
#' @return the samples (x_1. ...) as data table
Do2_sample <- function(scm_intv, i_do, xi_do, j_do, xj_do, n = 1){

  # set the intervention variables to the intervention values
  body(scm_intv[[i_do]]) <- substitute(xi_do)
  body(scm_intv[[j_do]]) <- substitute(xj_do)

  # get new samples
  smp <- joint_sample(scm = scm_intv, n = n)

  # tag the samples
  smp <- smp[,doX := NA,][,doXi := i_do,][,doXj := j_do,]

  return(smp)
}
```

We define functions we need for both the Bayesian and the alternative score in order to discover the causal structure.

```
#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
#-------------------------- DISCOVERY --------------------------
#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>


#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
#---------- Useful general functions ----------
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~


#' elementary symmetric polynomials
#'
#' @param z vector of inputs
#' @param o_p order of polynomial
#'
#' @return value of elementary symmetric polynomial
es_poly <- function(z, o_p){

  # input dimension
  d = length(z)
```

```
  # compute the powerset of z up to order o_p
  pS <- powerSet(z, m = o_p)

  # filter out the subsets not of that order
  S <- Filter(Negate(is.null), lapply(pS, function(i) if(length(i) == o_
     p) i))

  # return the sum of products of the respective interactions
  return(sum(unlist(lapply(S, function(i) prod(i)))))
}


#' customized base kernel: squared exponential (+ linear, turned off)
#'
#' @param v outer coefficient of exponential
#' @param w outer coefficient of exponential
#' @param a coeffincient of linear
#' @param x first input
#' @param y second input
#'
#' @return kernel value
k_base <- function(v, w, a, x, y) return(v*exp(-w*((abs(x-y))^2)/2))

# define class from kernlab package
class(k_base) <- "kernel"


#' generic kernel/covariance function with customized kernel with noise
#'
#' @param lhp list of log parameters v, w, a, k_add_o, vv
#' @param X_l data in design matrix form (columsn = features resp.
    dimensions)
#' @param X_r secondary data for K(X_l,X_r) in design matrix form
#' @param vv_incl Boolean for inclusion of variance in resulting kernel
    matrix, only sensible for X_l == X_r
#'
#' @return the kernel/covariance matrix
cov_cust <- function(lhp, X_l, X_r, vv_incl = F){

  # number of input dimensions
  D = ncol(X_l)

  # additive kernel
  ker_cust <- function(x,y){

    # base kernel values (see Duvenaud)
    z <- k_base(v = exp(lhp$ex.v), w = exp(lhp$ex.w), a = exp(lhp$lin.a)
       , x = x, y = y)

    # full additive kernel w/ resp. weights up to full order D

    # (not recursive)
    k_add <- sum(exp(lhp$k_add_w) * sapply(1:D, function(o) es_poly(z, o
       )))
```

```
    # switch to  alternative recursive, if D >= 5
    # list of (o-1)th order kernels, 1st elemnt dummy for the sum in the
        for loop
    #k_add_l <- list(1)

    # recursive formula "Newton-Girard" to efficiently compute the
       elementary symmetric polynomials
    #for(o in 1:D){
    #      k_add_l[[o + 1]] <- 1/o * sum(sapply(1:o, function(k) (-1)^(k
       -1) * k_add_l[[(o-k) + 1]] * sum(z^k)))
    #}

    #k_add <- sum(exp(lhp$k_add_w) * unlist(k_add_l)[2:(D+1)])

    return(k_add)
  }

  if(vv_incl){

    # case of X_l == X_r, number of observations:
    n <- nrow(X_l)
    return(kernelMatrix(kernel = ker_cust, x = as.matrix(X_l), y = as.
       matrix(X_r)) + diag(exp(lhp$vv), n))
  }

  else return(kernelMatrix(kernel = ker_cust, x = as.matrix(X_l), y = as
     .matrix(X_r)))
}


#' derivative of generic kernel/covariance function with customized
   kernel with noise
#'
#' @param lhp list of log parameters v, w, a, vv
#' @param X data
#'
#' @return "gradient" of the kernel/covariance matrix (order of
   hyperparameters in lhp maintained)
#'         i.e. list of derivative matrices
D_cov_cust <- function(lhp, X){

  # number of observations and dimensions
  n = nrow(X)
  D = ncol(X)


  # derivatives ...

  # ... wrt ex.v
  ker_cust_dv <- function(x, y){

    # base kernel values (see Duvenaud)
    z <- k_base(v = exp(lhp$ex.v), w = exp(lhp$ex.w), a = exp(lhp$lin.a)
```

```
      , x = x, y = y)

  # apply chain rule: df/dz * dz/dv, where z = c(z_1, z_2, ..., z_D)
      and f = k_add_o
  # dk_add_o/dz: D 1xD matrices (see Duvenaud)
  dk_add_o_diff_dz <- t(sapply(1:D, function(o) sapply(1:D, function(d
      ) es_poly(z[-d],o-1))))

  # dz/dv
  dz_diff_dv <- as.matrix(k_base(v = 1, w = exp(lhp$ex.w), a = 0, x =
      x, y = y))

  # column vector partial order derivatives w/o additive weights
  dk_add_o_diff_dv <- dk_add_o_diff_dz %*% dz_diff_dv

  # entire partial derivative
  return(sum(exp(lhp$k_add_w) * dk_add_o_diff_dv))
}

cov_dv <- kernelMatrix(kernel = ker_cust_dv, x = as.matrix(X), y = as.
    matrix(X))


# ... wrt ex.w
ker_cust_dw <- function(x, y){

  # base kernel values (see Duvenaud)
  z <- k_base(v = exp(lhp$ex.v) * (-((abs(x-y))^2)/2), w = exp(lhp$ex.
      w), a = 0, x = x, y = y)

  # apply chain rule: df/dz * dz/dv, where z = c(z_1, z_2, ..., z_D)
      and f = k_add_o
  # dk_add_o/dz: D 1xD matrices (see Duvenaud)
  dk_add_o_diff_dz <- t(sapply(1:D, function(o) sapply(1:D, function(d
      ) es_poly(z[-d],o-1))))

  # dz/dv
  dz_diff_dv <- as.matrix(k_base(v = 1, w = exp(lhp$ex.w), a = 0, x =
      x, y = y))

  # column vector partial order derivatives w/o additive weights
  dk_add_o_diff_dv <- dk_add_o_diff_dz %*% dz_diff_dv

  # entire partial derivative
  return(sum(exp(lhp$k_add_w) * dk_add_o_diff_dv))
}

cov_dw <- kernelMatrix(kernel = ker_cust_dw, x = as.matrix(X), y = as.
    matrix(X))


# ... wrt lin.a
ker_cust_da <- function(x, y){
```

```
    # base kernel values (see Duvenaud)
    z <- k_base(v = 0, w = 0, a = 1, x = x, y = y)

    # apply chain rule: df/dz * dz/dv, where z = c(z_1, z_2, ..., z_D)
       and f = k_add_o
    # dk_add_o/dz: D 1xD matrices (see Duvenaud)
    dk_add_o_diff_dz <- t(sapply(1:D, function(o) sapply(1:D, function(d
       ) es_poly(z[-d],o-1))))

    # dz/dv
    dz_diff_dv <- as.matrix(k_base(v = 1, w = exp(lhp$ex.w), a = 0, x =
       x, y = y))

    # column vector partial order derivatives w/o additive weights
    dk_add_o_diff_dv <- dk_add_o_diff_dz %*% dz_diff_dv

    # entire partial derivative
    return(sum(exp(lhp$k_add_w) * dk_add_o_diff_dv))
  }

  cov_da <- kernelMatrix(kernel = ker_cust_da, x = as.matrix(X), y = as.
     matrix(X))


  # ... wrt k_add_w[]: list of ker_cust functions and then list of
     covariance matrices
  ker_cust_dadd_w <- lapply(1:D, function(d) return(function(x, y){

    # base kernel values (see Duvenaud)
    z <- k_base(v = exp(lhp$ex.v), w = exp(lhp$ex.w), a = exp(lhp$lin.a)
       , x = x, y = y)

    # derivative of full additive kernel w/ respect to weights, not
       recursive
    dk_add <- es_poly(z, d)

    return(dk_add)
  }))

  cov_dadd_w <- lapply(1:D, function(d) kernelMatrix(kernel = ker_cust_
     dadd_w[[d]], x = as.matrix(X), y = as.matrix(X)))


  # ... wrt vv
  cov_dvv <- diag(n)

  # list w/ all cov matrices
  cov_list <- list(cov_dv, cov_dw, cov_da)
  for(d in 1:D) cov_list[[d+3]] <- cov_dadd_w[[d]]
  cov_list[[D+4]] <- cov_dvv

  return(cov_list)
}
```

```
#' source node estimation w/ ML and simple univariate Gaussians
#'
#' @param xtrain univariate samples (expected still in data.table format
    )
#'
#' @return list of estimated mean and variance
est_source <- function(xtrain){

  # simple ML scheme for Gaussian
  f_Xloglike <- maxlogL(x = as.double(unlist(xtrain)), dist = 'dnorm',
     start=c(0, 1), lower=c(-4, 0.01), upper=c(4, 4))

  # estimates for mean and variance
  return(list(X_mea = f_Xloglike$fit$par[1], X_var = f_Xloglike$fit$par
     [2]^2))
}


#' dependant node estimation: GPR w/ ML-II scheme for hyperparameter
    optimization
#'
#' @param ytrain univariate data of the dependant node
#' @param xtrain possibly multivariate data of
#' @param dep_par list of parameters for the regression situation (level
     of dep_param[[1]][[1]])
#'
#' @return list of estimates in standard form:
est_dependant <- function(ytrain, xtrain, dep_par){

  # ensure canonical form: column vectors for the various dimensions
  ytrain <- t(t(ytrain))
  xtrain <- t(t(xtrain))

  # number of observations and dimensions
  n = nrow(xtrain)
  D = ncol(xtrain)

  # get initial hyperparameter
  log_hp <- dep_par$para


  #------ functions

  #' (marginal) GP loglikelohood
  #'
  #' @param lpara log hyperparameter
  #'
  #' @return likelihood value
  minus_maloglike <- function(lpara){

    lhp <- list("ex.v" = lpara[1], "ex.w" = lpara[2], "lin.a" = lpara
       [3], "k_add_w" = lpara[4:(D+3)], "vv" = lpara[(D+4)])
    Cv <- cov_cust(lhp, X_l = xtrain, X_r = xtrain, vv_incl = T)
```

```
    return(0.5*quad.form(solve(Cv), unlist(ytrain)) + 0.5*log(det(Cv)) +
        0.5*n*log(2*pi))
  }

  #' gradients of those  minus GP loglikelihoods
  #'
  #' @param lpara log hyperparameter
  #'
  #' @return likelihood value
  D_minus_maloglike <- function(lpara){

    lhp <- list("ex.v" = lpara[1], "ex.w" = lpara[2], "lin.a" = lpara
       [3], "k_add_w" = lpara[4:(D+3)], "vv" = lpara[(D+4)])
    Cv <- cov_cust(lhp, X_l = xtrain, X_r = xtrain, vv_incl = T)
    Cv_inv <- solve(Cv)
    D_Cv <- D_cov_cust(lhp, X = xtrain)

    alph <- Cv_inv %*% as.matrix(ytrain)

    # return derivative of minus loglikelihood
    return(unlist(lapply( D_Cv, function(i) return(- 0.5 * tr(((alph %*%
        t(alph)) - Cv_inv) %*% i)))))
  }


  #------ main

  lhp_init <- list("ex.v" = log(5), "ex.w" = log(10), "lin.a" = log(10),
                   "k_add_w" = log(rep(1/D, D)), "vv" = log(1))

  o_res <- optimr(fn = minus_maloglike, gr = D_minus_maloglike, par = as
     .double(unlist(lhp_init)))


  dep_par$para = list("ex.v" = o_res$par[1], "ex.w" = o_res$par[2], "lin
     .a" = o_res$par[3],
                      "k_add_w" = o_res$par[4:(D+3)], "vv" = o_res$par[(
                         D+4)])

  dep_par$conv = o_res$convergence

  # calculate covariance matrices with optimized parameters
  dep_par$Cov <- cov_cust(dep_par$para, X_l = xtrain, X_r = xtrain, vv_
     incl = T)

  return(dep_par)
}


#' pseudo source node estimation: GPR w/ ML-II scheme for hyperparameter
    optimization
#'
#' @param data data table with observational data and fixed value
```

```
    intervention on all other d-1 variables
#' @param psou_par list of parameters for the regression situation (
    level of psou_param[[1]][[1]])
#'
#' @return list of estimates in standard form:
est_pseudo_source <- function(psou_data, psou_par){

  # select correct (regular + additional) observational sample data,
     restricted to max. 3 variable case
  if(psou_par$sou == 1){
    psou_data <- psou_data[,-c(2,3)]
    sou_data_obs_add <- data_obs_add[,1]
  }
  if(psou_par$sou == 2){
    psou_data <-psou_data[,-c(1,3)]
    sou_data_obs_add <- data_obs_add[,2]
  }
  if(psou_par$sou == 3){
    psou_data <- psou_data[,-c(1,2)]
    sou_data_obs_add <- data_obs_add[,3]
  }

  # calculate observational sample variance (corrected), also from
     additional observational data
  smp_var_obs = as.double(var(rbind(psou_data[doX == 0][,1],sou_data_obs
     _add)))

  # calculate intventional sample variance (corrected)
  smp_var_intv = as.double(var(psou_data[is.na(doX)][,1]))

  # set base noise variance equal to the noise variance around
     functional values in GPR settings (base level now fixed sd = 0.1,
     var = 0.01)
  eps_base <- 0.01

  # set additional noise according to linear function:
  #     if intervetional sample variance is high (probably source with
     fixed interventions) -> small additional noise, low penalty
  #     else -> additional variance as high as spread of entire variable
      'smp_var_obs', high penalty
  eps_add <- max(0,-smp_var_intv + smp_var_obs)

  # calculate total noise as sum of base and additional noise
  eps_total <- eps_add + eps_base



  # generate pseudo GP fitting data, first the observational data with
     just base noise
  data_pseudofit_obs <- list(x_1 = psou_data[doX == 0][,1],
                             x_2 = rnorm(n_obs, 0, sqrt(eps_base)))

  # then the for the "interventional data" randomly distributed points
     with full noise eps_total, change with different "support interval"
```

```
  data_pseudofit_intv <- list(x_1 = runif(((n_var-1)*n_intv), min = -2,
     max = 2),
                              x_2 = rnorm(((n_var-1)*n_intv), 0, sqrt(
                                 eps_total)))

  # entire data to do GP pseudo fit on
  data_pseudofit <- rbind(as.data.table(data_pseudofit_obs), as.data.
     table(data_pseudofit_intv))

  #plot(data_pseudofit)

  # fit pseudo GP
  res_psou_par <- est_dependant(ytrain = data_pseudofit$x_2, xtrain =
     data_pseudofit$x_1, dep_par = psou_par)

  res_psou_par[["ydat"]] <- data_pseudofit$x_2

  return(res_psou_par)
}
```

For the Bayesian score we consider the following learning setup and causal discovery scheme for the case of 3 considered variables:

```
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
#---------- Learning Setup ----------
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

# considered models
models <- list(
  # v-structures
  graph(edges = c(1,3, 2,3), n = 3, directed = T),
  graph(edges = c(1,2, 3,2), n = 3, directed = T),
  graph(edges = c(2,1, 3,1), n = 3, directed = T),
  # reversed v-structures
  graph(edges = c(1,2, 1,3), n = 3, directed = T),
  graph(edges = c(2,1, 2,3), n = 3, directed = T),
  graph(edges = c(3,1, 3,2), n = 3, directed = T),
  # hierarchical lines
  graph(edges = c(1,2, 2,3), n = 3, directed = T),
  graph(edges = c(1,3, 3,2), n = 3, directed = T),
  graph(edges = c(2,1, 1,3), n = 3, directed = T),
  graph(edges = c(2,3, 3,1), n = 3, directed = T),
  graph(edges = c(3,1, 1,2), n = 3, directed = T),
  graph(edges = c(3,2, 2,1), n = 3, directed = T),
  # v-structures plus parent dependency
  graph(edges = c(1,2, 1,3, 2,3), n = 3, directed = T),
  graph(edges = c(1,3, 1,2, 3,2), n = 3, directed = T),
  graph(edges = c(2,1, 2,3, 1,3), n = 3, directed = T),
  graph(edges = c(2,3, 2,1, 3,1), n = 3, directed = T),
  graph(edges = c(3,1, 3,2, 1,2), n = 3, directed = T),
  graph(edges = c(3,2, 3,1, 2,1), n = 3, directed = T),
  # one independent variable
  graph(edges = c(1,2), n = 3, directed = T),
  graph(edges = c(2,1), n = 3, directed = T),
  graph(edges = c(1,3), n = 3, directed = T),
  graph(edges = c(3,1), n = 3, directed = T),
  graph(edges = c(2,3), n = 3, directed = T),
  graph(edges = c(3,2), n = 3, directed = T),
  # all independent
  graph(edges = c(), n = 3, directed = T)
)

# number of models
n_models <- length(models)

# list of respective adjacency matrices
adjm <- lapply(models, function(g) as.matrix(g[]))

# distribution of considered graphs
prob_g <- rep(1/n_models, n_models)

# current sample size
n <- 5
```

```
# generate 5 samples
data <- joint_sample(scm = scm_true, n)

# intervention indicator: 0 = no (obs), 1 = doX_1, 2 = doX_2, ...
data <- data[,doX := 0,]




#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
#---------- Bayesian Causal Discovery ----------
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~


######## for purely observational data

#*****************************************************
# for source nodes: ML estimation
# get source nodes (no incoming edges means zero sum columns)
sources <- lapply(adjm, function(m) which(colSums(m) == 0))

# ESTIMATE all source parameter for all modeld considered (outer loop)
# and within each model for all source nodes (inner loop)
# result: list[models] of lists[source nodes] of lists[parameters of
    that source node]
source_param <- lapply(1:n_models, function(i_m) lapply(sources[[i_m]],
    function(i_s)
  list("sou" = i_s, "para" = est_source(data[doX == 0][,get(names(data)[
     i_s])])))))


#*****************************************************
# for non-source nodes or dependants
# identify the dependants for each model
dependants <- lapply(1:n_models, function(i_m) setdiff(all_var, sources
    [[i_m]]))

# identify parents for each dependant
# result: list[models] of lists[dependant nodes] of lists[dependant node
    (1) or his parents (2)]
dep_parent <- lapply(1:n_models, function(i_m)  lapply(dependants[[i_m
    ]], function(i_d)  list("dep" = i_d, "parent" = which(adjm[[i_m]][, i
    _d] == 1))))

# list to carry the GPR parameters to be estimated
# result: list[models] of lists[dependant nodes] of lists[dependant node
    (1)or the list of parameter of the additive kernel GPR (2)]

# INITIALIZE all dependent parameters for GPR
#' @param ex.v log linear weight for (exp)^2 part
#' @param ex.w log exp weight for (exp)^2 part
#' @param lin.a log linear weight lin part
#' @param k_add_w log vector of weights for the o-th order additive
```

```
    kernel respectively
#' @param vv log variance of GPR w/ normal noise
# i_d now list in style of "dep_parent[[1]][[1]]"
dep_param <- lapply(1:n_models, function(i_m)  lapply(dep_parent[[i_m]],
    function(i_d) {
                    i_d[["para"]] <- list("ex.v" = log(5), "ex.w" = log
                        (10), "lin.a" = log(10), "k_add_w" = log(rep(1/
                        length(i_d$parent), length(i_d$parent))), "vv" =
                        log(1))
                    i_d[["conv"]] <- 0
                    i_d[["Cov"]] <- matrix()
                    return(i_d)
    }))

# ESTIMATE all dependant parameters via the Gaussian process likelihoods
dep_param <- lapply(1:n_models, function(i_m)  lapply(dep_param[[i_m]],
                                                      function(i_d) est_
                                                         dependant(as.
                                                         matrix(data[doX
                                                          == 0])[,i_d$
                                                         dep], as.matrix
                                                         (data[doX ==
                                                         0])[,i_d$parent
                                                         ], i_d)))


dep_param[[24]][[1]]



# log likelihoods using Markov factorization
# ... for sources
maloglike_sources <- lapply(1:n_models, function(i_m) sum(sapply(source_
    param[[i_m]], function(i_s){
  return(sum(log(unlist(lapply(as.matrix(data[doX == 0])[,i_s$sou],
      dnorm, mean = i_s$para$X_mea, sd = sqrt(i_s$para$X_var))))))
})))

# .. for dependants (last model has no dependants)
maloglike_dependants <- lapply(1:(n_models-1), function(i_m) sum(sapply(
    dep_param[[i_m]], function(i_d){
  return(-0.5*quad.form(solve(i_d$Cov), as.matrix(as.matrix(data[doX ==
      0])[,i_d$dep])) - 0.5*log(det(i_d$Cov)) - 0.5*n*log(2*pi))
})))

# nothing for last model of independent variables
maloglike_dependants[[25]] <- 0


# ... total
totallike <- exp(unlist(maloglike_sources) + unlist(maloglike_dependants
    ))

# Bayesian nornalization constant
```

```
norm_const <- sum(prob_g*totallike)

# Bayesian update
prob_g = prob_g*totallike/norm_const



######## for observational and interventional data

# Experiments: get interventional data
for(i in 1:10){

  # Choose interventions at random uniformly in [-1, 1]
  x_intv <- runif(n_var, min = -1, max = 1)

  for(i_v in 1:n_var) data = rbind(data, Do_sample(scm_intv = scm_true,
     i_do = i_v, x_do = x_intv[i_v], n = 1))

  n = n + 1

}


# Learning

print("estimating␣source␣parameters...")
# ESTIMATE all source parameter for all models considered (outer loop)
# and within each model for all source nodes (inner loop)
# result: list[models] of lists[source nodes] of lists[parameters of
    that source node]
source_param <- lapply(1:n_models, function(i_m) lapply(sources[[i_m]],
    function(i_s)
     list("sou" = i_s, "para" = est_source(data[doX != i_s][,get(names(
        data)[i_s])])))))


print("estimating␣dependant␣parameters...")
# ESTIMATE all dependant parameters via the Gaussian process likelihoods
dep_param <- lapply(1:(n_models-1), function(i_m)  lapply(dep_param[[i_m
    ]],
     function(i_d) est_dependant(as.matrix(data[doX != i_d$dep])[,i_d$dep
        ], as.matrix(data[doX != i_d$dep])[,i_d$parent], i_d)))



# log likelihoods using Markov factorization
# ... for sources
maloglike_sources <- lapply(1:n_models, function(i_m) sum(sapply(source_
    param[[i_m]], function(i_s){
     return(sum(log(unlist(lapply(as.matrix(data[doX != i_s$sou])[,i_s$
        sou], dnorm, mean = i_s$para$X_mea, sd = sqrt(i_s$para$X_var)))))
        )
})))
```

```
# .. for dependants, fix n...
maloglike_dependants <- lapply(1:(n_models-1), function(i_m) sum(sapply(
   dep_param[[i_m]], function(i_d){
    return(-0.5*quad.form(solve(i_d$Cov), as.matrix(as.matrix(data[doX !
       = i_d$dep])[,i_d$dep])) - 0.5*log(det(i_d$Cov)) - 0.5*n*log(2*pi)
       )
})))

# nothing for last model of independent variables
maloglike_dependants[[25]] <- 0



# ... total
totallike <- exp(unlist(maloglike_sources) + unlist(maloglike_dependants
   ))

# Bayesian nornalization constant
norm_const <- sum(prob_g*totallike)

# Bayesian update
prob_g = prob_g*totallike/norm_const
```

For the alternative score we consider the following learning setup and causal discovery scheme for the case of 3 considered variables:

```
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
#---------- Learning Setup ----------
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

# considered models
models <- list(
  # v-structures
  graph(edges = c(1,3, 2,3), n = 3, directed = T),
  graph(edges = c(1,2, 3,2), n = 3, directed = T),
  graph(edges = c(2,1, 3,1), n = 3, directed = T),
  # reversed v-structures
  graph(edges = c(1,2, 1,3), n = 3, directed = T),
  graph(edges = c(2,1, 2,3), n = 3, directed = T),
  graph(edges = c(3,1, 3,2), n = 3, directed = T),
  # hierarchical lines
  graph(edges = c(1,2, 2,3), n = 3, directed = T),
  graph(edges = c(1,3, 3,2), n = 3, directed = T),
  graph(edges = c(2,1, 1,3), n = 3, directed = T),
  graph(edges = c(2,3, 3,1), n = 3, directed = T),
  graph(edges = c(3,1, 1,2), n = 3, directed = T),
  graph(edges = c(3,2, 2,1), n = 3, directed = T),
  # v-structures plus parent dependency
  graph(edges = c(1,2, 1,3, 2,3), n = 3, directed = T),
  graph(edges = c(1,3, 1,2, 3,2), n = 3, directed = T),
  graph(edges = c(2,1, 2,3, 1,3), n = 3, directed = T),
  graph(edges = c(2,3, 2,1, 3,1), n = 3, directed = T),
  graph(edges = c(3,1, 3,2, 1,2), n = 3, directed = T),
  graph(edges = c(3,2, 3,1, 2,1), n = 3, directed = T),
  # one independent variable
  graph(edges = c(1,2), n = 3, directed = T),
  graph(edges = c(2,1), n = 3, directed = T),
  graph(edges = c(1,3), n = 3, directed = T),
  graph(edges = c(3,1), n = 3, directed = T),
  graph(edges = c(2,3), n = 3, directed = T),
  graph(edges = c(3,2), n = 3, directed = T),
  # all independent
  graph(edges = c(), n = 3, directed = T)
)

# number of models
n_models <- length(models)

# list of respective adjacency matrices
adjm <- lapply(models, function(g) as.matrix(g[]))

# distribution of considered graphs
prob_g <- rep(1/n_models, n_models)

# current sample size
```

```
n_obs <- 5

# generate 5 samples
data <- joint_sample(scm = scm_true, n_obs)

# intervention indicator: 0 = no (obs), 1 = doX_1, 2 = doX_2, ...
data <- data[,doX := 0,]



# allow to draw 15 additional observational samples just to estimate the
    sample variances for pseudo nodes later on equal grounds,
# not used otherwise
data_obs_add <- joint_sample(scm = scm_true, 15)




#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
#---------- Causal Discovery ----------
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~


######## for observational and interventional data


#*****************************************************
# number of interventions
n_intv <- 10

# total number of values that GPs are fitted on
n_total <- ((n_var-1)*n_intv) + n_obs

# Experiments: get interventional data of K_1
for(i in 1:n_intv){

  # choose interventions at random uniformly in [-1, 1]
  x_intv <- runif(n_var, min = -1, max = 1)

  # get interventional data
  for(i_v in 1:n_var) data = rbind(data, Do_sample(scm_intv = scm_true,
     i_do = i_v, x_do = x_intv[i_v], n = 1))

}



# ... and get interventional data of K_2
# generate intervention data for fixed-value-interventions
data <- data[,doXi := NA,][,doXj := NA,]

# intervene on every variable w/ fixed intv value 0
for(not_v in 1:n_var){
```

```
  i_v <- setdiff(1:n_var, not_v)[1]
  j_v <- setdiff(1:n_var, not_v)[2]

  # sample size (n_var-1)*n_intv to match sample size for GP =  n_obs +
     (n_var-1)*n_intv
  smp <- Do2_sample(scm_intv = scm_true, i_do = i_v, xi_do = 0, j_do = j
     _v, xj_do = 0, n = ((n_var-1)*n_intv))

  data = rbind(data, smp)
}




#*****************************************************
# for source nodes: ML estimation
# get source nodes (no incoming edges means zero sum columns)
sources <- lapply(adjm, function(m) which(colSums(m) == 0))


#*****************************************************
# for non-source nodes or dependants
# identify the dependants for each model
dependants <- lapply(1:n_models, function(i_m) setdiff(all_var, sources
   [[i_m]]))

# identify parents for each dependant
# result: list[models] of lists[dependant nodes] of lists[dependant node
    (1) or his parents (2)]
dep_parent <- lapply(1:n_models, function(i_m)  lapply(dependants[[i_m
   ]], function(i_d)  list("dep" = i_d, "parent" = which(adjm[[i_m]][, i
   _d] == 1))))

# list to carry the GPR parameters to be estimated
# result: list[models] of lists[dependant nodes] of lists[dependant node
    (1)or the list of parameter of the additive kernel GPR (2)]



print("estimating␣dependant␣parameters...")
# INITIALIZE all dependant parameters for GPR
#' @param ex.v log linear weight for (exp)^2 part
#' @param ex.w log exp weight for (exp)^2 part
#' @param lin.a log linear weight lin part
#' @param k_add_w log vector of weights for the o-th order additive
   kernel respectively
#' @param vv log variance of GPR w/ normal noise
# i_d now list in style of "dep_parent[[1]][[1]]"
dep_param <- lapply(1:n_models, function(i_m)  lapply(dep_parent[[i_m]],
    function(i_d) {
                  i_d[["para"]] <- list("ex.v" = log(5), "ex.w" = log
                     (10), "lin.a" = log(10), "k_add_w" = log(rep(1/
                     length(i_d$parent), length(i_d$parent))), "vv" =
                     log(1))
```

```
                    i_d[["conv"]] <- 0
                    i_d[["Cov"]] <- matrix()
                    return(i_d)
    }))

# ESTIMATE all dependant parameters via the Gaussian process likelihoods
dep_param <- lapply(1:(n_models-1), function(i_m)  lapply(dep_param[[i_m
   ]],
  function(i_d) est_dependant(as.matrix(data[doX != i_d$dep & !is.na(doX
     )])[,i_d$dep], as.matrix(data[doX != i_d$dep & !is.na(doX)])[,i_d$
     parent], i_d)))

dep_param[[1]][[1]]



print("estimating␣pseudo␣source␣parameters...")
# INITIALIZE all pseudo source parameters for GPR
#' @param ex.v log linear weight for (exp)^2 part
#' @param ex.w log exp weight for (exp)^2 part
#' @param lin.a log linear weight lin part
#' @param k_add_w log vector of weights for the o-th order additive
   kernel respectively
#' @param vv log variance of GPR w/ normal noise
# i_d now list in style of "dep_parent[[1]][[1]]"
psou_param <- lapply(1:n_models, function(i_m)  lapply(sources[[i_m]],
     function(i_s) {
                    l <- list()
                    l[["sou"]] <- i_s
                    l[["para"]] <- list("ex.v" = log(5), "ex.w" = log
                       (10), "lin.a" = log(10), "k_add_w" = log(1), "vv"
                        = log(1))
                    l[["conv"]] <- 0
                    l[["Cov"]] <- matrix()
                    l[["ydat"]] <- c()
                    return(l)
     }))

# ESTIMATE all pseudo source parameter via the Gaussian process
   likelihoods
psou_param <- lapply(1:n_models, function(i_m)  lapply(psou_param[[i_m
   ]],
  function(i_s) est_pseudo_source(data[(doXi != i_s$sou & doXj != i_s$
     sou & is.na(doX)) | doX == 0], i_s)))




# log likelihoods using Markov factorization
# ... for sources
maloglike_psources <- lapply(1:n_models, function(i_m) sum(sapply(psou_
   param[[i_m]], function(i_s){
  return(-0.5*quad.form(solve(i_s$Cov), t(t(i_s$ydat))) - 0.5*log(det(i_
     s$Cov)) - 0.5*n_total*log(2*pi))
```

```
})))

# .. for dependants, fix n...
maloglike_dependants <- lapply(1:(n_models-1), function(i_m) sum(sapply(
    dep_param[[i_m]], function(i_d){
  return(-0.5*quad.form(solve(i_d$Cov), as.matrix(as.matrix(data[doX !=
      i_d$dep & !is.na(doX)])[,i_d$dep])) - 0.5*log(det(i_d$Cov)) - 0.5*n
      _total*log(2*pi))
})))

# nothing for last model of independent variables
maloglike_dependants[[25]] <- 0



# ... total
totallike <- exp(unlist(maloglike_psources) + unlist(maloglike_
    dependants))

# nornalization constant
norm_const <- sum(prob_g*totallike)

# probablities for the individual graphs
prob_g = prob_g*totallike/norm_const
```

Bibliography

Auguie, B. *gridExtra: Miscellaneous Functions for "Grid" Graphics*, 2017. URL `https://CRAN.R-project.org/package=gridExtra`. R package version 2.3.

Bongers, S., Peters, J., Schölkopf, B., and Mooij, J. M. Structural causal models: Cycles, marginalizations, exogenous reparametrizations and reductions. *arXiv preprint arXiv*, 1611, 2016.

Csardi, G. and Nepusz, T. The igraph software package for complex network research. *InterJournal*, Complex Systems:1695, 2006. URL `http://igraph.org`.

Dowle, M. and Srinivasan, A. *data.table: Extension of 'data.frame'*, 2019. URL `https://CRAN.R-project.org/package=data.table`. R package version 1.12.8.

Duvenaud, D. *Automatic model construction with Gaussian processes*. PhD thesis, University of Cambridge, 2014.

Duvenaud, D. K., Nickisch, H., and Rasmussen, C. E. Additive gaussian processes. In *Advances in neural information processing systems*, pages 226–234, 2011.

DAriano, G. M. Causality re-established. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 376(2123):20170313, 2018.

Eaton, D. and Murphy, K. Exact bayesian structure learning from uncertain interventions. In *Artificial intelligence and statistics*, pages 107–114, 2007.

Evans, R. *rje: Miscellaneous Useful Functions for Statistics*, 2020. URL `https://CRAN.R-project.org/package=rje`. R package version 1.10.16.

Friedman, N. and Nachman, I. Gaussian process networks. *arXiv preprint arXiv:1301.3857*, 2013.

Hankin, R. K. S. Introducing bacco, an r bundle for bayesian analysis of computer code output. *Journal of Statistical Software*, 14, October 2005.

Heckerman, D., Meek, C., and Cooper, G. A Bayesian approach to causal discovery. In *Computation, causation, and discovery*, pages 141–165. AAAI Press, Menlo Park, CA, 1999.

Heinze-Deml, C., Maathuis, M. H., and Meinshausen, N. Causal structure learning. *Annu. Rev. Stat. Appl.*, 5:371–394, 2018. ISSN 2326-8298. doi: 10.1146/annurev-statistics-031017-100630. URL `https://doi-org.eaccess.ub.tum.de/10.1146/annurev-statistics-031017-100630`.

Henry, L. and Wickham, H. *purrr: Functional Programming Tools*, 2020. URL `https://CRAN.R-project.org/package=purrr`. R package version 0.3.4.

Kanagawa, M., Hennig, P., Sejdinovic, D., and Sriperumbudur, B. K. Gaussian processes and kernel methods: A review on connections and equivalences. *arXiv preprint arXiv:1807.02582*, 2018.

Karatzoglou, A., Smola, A., Hornik, K., and Zeileis, A. kernlab – an S4 package for kernel methods in R. *Journal of Statistical Software*, 11(9):1–20, 2004. URL `http://www.jstatsoft.org/v11/i09/`.

Karimi, A.-H., von Kügelgen, J., Schölkopf, B., and Valera, I. Algorithmic recourse under imperfect causal knowledge: a probabilistic approach. *arXiv preprint arXiv:2006.06831*, 2020.

Koller, D. and Friedman, N. *Probabilistic graphical models*. Adaptive Computation and Machine Learning. MIT Press, Cambridge, MA, 2009. ISBN 978-0-262-01319-2. Principles and techniques.

Kuss, M. *Gaussian process models for robust regression, classification, and reinforcement learning*. PhD thesis, Technische Universität Darmstadt, 2006.

Lopez-Paz, D. From dependence to causation. *arXiv preprint arXiv:1607.03300*, 2016.

MacKay, D. J. Introduction to gaussian processes. *NATO ASI Series F Computer and Systems Sciences*, 168:133–166, 1998.

Minh, H. Q., Niyogi, P., and Yao, Y. Mercer's theorem, feature maps, and smoothing. In *Learning theory*, volume 4005 of *Lecture Notes in Comput. Sci.*, pages 154–168. Springer, Berlin, 2006. doi: 10.1007/11776420_14. URL `https://doi-org.eaccess.ub.tum.de/10.1007/11776420_14`.

Mooij, J. M., Janzing, D., Heskes, T., and Schölkopf, B. On causal discovery with cyclic additive noise models. In *Advances in neural information processing systems*, pages 639–647, 2011.

Mosquera, J. and Hernandez, F. *EstimationTools: Maximum Likelihood Estimation for Probability Functions from Data Sets*, 2019. URL `https://CRAN.R-project.org/package=EstimationTools`. R package version 1.2.1.

Nandy, P., Hauser, A., and Maathuis, M. H. High-dimensional consistency in score-based and hybrid structure learning. *Ann. Statist.*, 46(6A):3151–3183, 2018. ISSN 0090-5364. doi: 10.1214/17-AOS1654. URL `https://doi-org.eaccess.ub.tum.de/10.1214/17-AOS1654`.

Nash, J. C. *optimr: A Replacement and Extension of the 'optim' Function*. URL `https://CRAN.R-project.org/package=optimr`.

Pearl, J. *Causality*. Cambridge University Press, Cambridge, second edition, 2009. ISBN 978-0-521-89560-6; 0-521-77362-8. doi: 10.1017/CBO9780511803161. URL https://doi-org.eaccess.ub.tum.de/10.1017/CBO9780511803161. Models, reasoning, and inference.

Peters, J., Janzing, D., and Schölkopf, B. *Elements of causal inference*. Adaptive Computation and Machine Learning. MIT Press, Cambridge, MA, 2017. ISBN 978-0-262-03731-0. Foundations and learning algorithms.

Peters, J. M. *Restricted structural equation models for causal inference*. PhD thesis, ETH Zurich, 2012.

Rasmussen, C. E. and Williams, C. K. I. *Gaussian processes for machine learning*. Adaptive Computation and Machine Learning. MIT Press, Cambridge, MA, 2006. ISBN 978-0-262-18253-9.

Schmidt, K. D. *Maßund Wahrscheinlichkeit*. Springer, Heidelberg, revised edition, 2011. ISBN 978-3-642-21025-9; 978-3-642-21026-6.

Seeger, M. Gaussian processes for machine learning. *International journal of neural systems*, 14(02):69–106, 2004.

Shorack, G. R. *Probability for statisticians*. Springer Texts in Statistics. Springer, Cham, second edition, 2017. ISBN 978-3-319-52206-7; 978-3-319-52207-4. doi: 10.1007/978-3-319-52207-4. URL https://doi-org.eaccess.ub.tum.de/10.1007/978-3-319-52207-4.

Steinwart, I. and Christmann, A. *Support vector machines*. Information Science and Statistics. Springer, New York, 2008. ISBN 978-0-387-77241-7.

von Kügelgen, J., Rubenstein, P. K., Schölkopf, B., and Weller, A. Optimal experimental design via bayesian optimization: active causal structure learning for gaussian process networks. *arXiv preprint arXiv:1910.03962*, 2019.

Wickham, H. *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York, 2016. ISBN 978-3-319-24277-4. URL https://ggplot2.tidyverse.org.