

Chapter 2

Local Causal Discovery with a Simple PC Algorithm

Abstract This chapter presents the PC-simple algorithm and illustrates how to use the algorithm in the exploration for local causal relationships around a target variable. PC-simple is a simplified version of the PC algorithm, a classic method for learning a complete casual Bayesian network. We firstly discuss how the PC algorithm establishes causal relationships by the way of detecting persistent associations, then we introduce PC-simple in detail, followed by the discussions on PC-simple. The last section of this chapter introduces the R implementation of PC-simple.

2.1 Introduction

Bayesian networks [6, 8], a type of probabilistic graphical models [4], are a major means of causal representation and inference. Given the set of variables representing a domain, a Bayesian network presents the full joint probability of the variables by a directed acyclic graph (DAG) containing nodes representing the variables and arcs indicating dependence relationships between the variables.

More formally, with a set of variables, \mathbf{V} , a Bayesian network consists of a structure, the DAG, $\mathbf{G} = (\mathbf{V}, \mathbf{E})$, and the joint probability distribution $P(\mathbf{V})$ such that the Markov condition holds, i.e. for each node $X \in \mathbf{V}$, given its parent nodes $\mathbf{P}_a(X) \subset \mathbf{V}$, X is independent of all of its non-descendant nodes according to $P(\mathbf{V})$. Hence a Bayesian network provides a graphical representation of the conditional independence relationships among all the variables in \mathbf{V} , and as a result of the Markov condition the full joint probability distribution $P(\mathbf{V})$ can be factored into $P(\mathbf{V}) = \prod_{X \in \mathbf{V}} P(X|\mathbf{P}_a(X))$.

In the exemplar Bayesian network in Fig. 2.1, conditioning on an empty set of parents, A is independent of all its non-descendants, D , E and H ; given A , variables B and C are independent of each other and they both are independent of D , E and H ; D and E both have no parents, apart from being independent of each other they both are independent of A , B , C , Z and H ; conditioning on its parents D , E , and Z ,

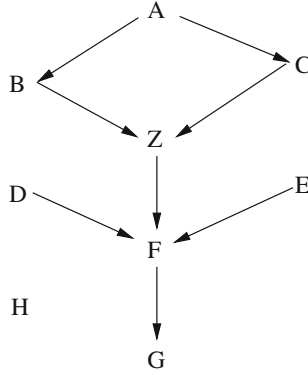


Fig. 2.1: An example Bayesian network

F is independent of A , B , C , and H ; G is independent of any other variable given its parent F ; H is isolated so it is independent of all other nodes; and finally Z is independent of A , D , E and H given its parents B and C . Furthermore, let \mathbf{V} be the set of variables in the Bayesian network, the joint probability distribution $P(\mathbf{V}) = P(A)P(B|A)P(C|A)P(D)P(E)P(F|D,E,Z)P(G|F)P(H)P(Z|B,C)$.

A causal Bayesian network is a Bayesian network when its structure is considered as a causal DAG, i.e. an edge $X \rightarrow Y$ in the DAG represents that X is a direct cause of Y . When the set of causal assumptions (discussed in Sect. 2.3.3) are made to link causal relationships and probability distributions, we can learn the causal structures from observational data.

We can discover causal relationships in a data set by learning the structure of a causal Bayesian network, but we may not know the directions of the causal relationships since the orientation of edges in a Bayesian network may not be completely determined only based on the data set (Please read Sect. 2.3.3 for detailed discussions). Therefore by learning a Bayesian network from a data set, we can conclude that two variables have a causal relationship but we may not know which variable is the cause and which variable is the effect.

The PC algorithm [5, 8] is a commonly used method for learning the structure of a causal Bayesian network. With a data set, for a pair of variables or nodes (X, Y) , the PC algorithm tests their conditional independence given the other variables, and it claims the non-existence of a causal relationship between X and Y , i.e. no edge to be drawn between X and Y , once it finds that X and Y are independent given some other variables. In other words, to determine whether there exists a persistent association between X and Y , the PC algorithm tests the association conditioning on all sub sets of all variables other than X and Y . The relationship is considered as causal only when the association exists given each of the conditioning sets.

Let us consider an exemplar binary data set containing attributes, Gender, College education, High school education, Manager, Clerk, and Salary, where the values of the variables are $\{male, female\}$ for Gender, $\{high, low\}$ for Salary, and $\{yes, no\}$ for all other variables. Note that we use the term *target variable* to represent the

outcome or effect, and *predictor variables* to represent the inputs or potential causes in this book. For example, variable Salary in this example is the target variable whose value is affected by other variables, and the remaining variables are predictor variables representing possible causes for having a high/low salary. To find out if Gender and Salary have a causal relationship, the independence between Gender and Salary is tested conditioning on each of the subsets of the other variables $\{\textit{College education}, \textit{High school education}, \textit{Manager}, \textit{Clerk}\}$, including the empty set. In the worst case, for each predictor variable, the number of conditional independence tests is 2^{m-1} where m is the number of predictor variables. Therefore, the PC algorithm only works on a data set with a small number of predictor variables.

In its algorithmic description, PC starts with a complete (undirected) graph with all the variables, \mathbf{V} , and removes the edge between a pair of nodes X and Y immediately after a subset $\mathbf{S} \subseteq \mathbf{V} \setminus \{X, Y\}$ is found such that X and Y are independent given \mathbf{S} . In order to reduce the number of conditional independence tests, the PC algorithm searches for the conditioning set for a pair of nodes in a level by level manner, i.e. searching the conditioning sets with $k + 1$ variables only when the search of all size k conditioning sets fails.

PC-simple [2] was originally developed for efficient variable selection in high dimensional linear regression models. The algorithm produces a set of variables which have strong influence on the target variable. Interestingly this algorithm turned out to be a simplified version of the PC algorithm. This possibly explains why the implementation of PC-simple by its authors (named as PC-select) is included in *pcalg* [3], the R-package of the methods for causal inference with graphical models.

PC-simple utilises the same idea as the PC algorithm to detect persistent associations, and it also conducts a level-wise search for the conditioning set for a pair of nodes. However, instead of learning a causal DAG that captures all the causal relationships among all the given variables, PC-simple can be used to discover the local relationships around a given response or target variable. That is, given the data set \mathbf{D} for a set of variables $(X_1, X_2, \dots, X_m, Z)$ where Z is the target variable, PC-simple identifies all the possible direct causes and effects of Z , i.e. all Z 's parents and children if we use the terminology for DAGs. In this case, it may be more meaningful to interpret the acronym ‘‘PC’’ in PC-simple as Parents and Children, instead of the names of the inventors (Peter Spirtes and Clark Glymour) of the PC algorithm.

With the goal of local relationship discovery, the algorithmic design of PC-simple, although follows the basic procedure of the PC algorithm, it has been simplified as presented in the next section.

2.2 The PC-simple Algorithm

Before describing the PC-simple algorithm, we firstly formally define conditional independence as follows.

Definition 2.1 (Conditional independence). Two variables X and Z are conditionally independent given a set of variables \mathbf{S} , denoted as $\text{Ind}(X, Z|\mathbf{S})$, if $P(X = x, Z =$

$z|\mathbf{S} = s) = P(X = x|\mathbf{S} = s)P(Z = z|\mathbf{S} = s)$ for all the values of X , Z , and \mathbf{S} , such that $P(\mathbf{S} = s) > 0$. The cardinality or size of \mathbf{S} , denoted as $|\mathbf{S}|$, is known as the order of the conditional independence.

Given a data set \mathbf{D} for a set of variables, the conditional independence between any two of the variables given any other variables can be tested using the data set at a specified significance level. Details of the commonly used conditional independence test methods are provided in Appendix A.

As shown in Algorithm 2.1 PC-simple takes as input the data set \mathbf{D} for predictor variables X_1, X_2, \dots, X_m and the target variable Z , and produces \mathbf{PC} , the set of parents and children of Z . The input parameter for the algorithm is the significance level used for conditional independence tests.

Initially the \mathbf{PC} set (i.e. \mathbf{PC}^0) contains all the predictor variables (Line 2 of Algorithm 2.1), and then PC-simple removes from the \mathbf{PC} set those variables that are not Z 's parents or children via conditional independence tests. The tests are done level by level of the cardinality of the conditioning sets, starting with an empty conditioning set (i.e. order zero conditional tests are done first). Each iteration of the **while** loop (Lines 3–12) generates \mathbf{PC}^k from \mathbf{PC}^{k-1} with the removal of variables that are independent of Z conditioning on $k - 1$ variables in \mathbf{PC}^{k-1} .

Specifically, in the first iteration, PC-simple initially lets \mathbf{PC}^1 equal to \mathbf{PC}^0 (Line 5 with $k = 1$), then it checks through \mathbf{PC}^0 (Lines 6 to 11 with $k = 1$), and if a variable in \mathbf{PC}^0 is independent of Z given an empty set (note that in Line 7, $|\mathbf{S}| = 0$ when $k = 1$), \mathbf{PC}^1 is updated by removing the independent variable from it (Lines 8 and 9). At the end of the first iteration, \mathbf{PC}^1 contains only the variables that are associated with Z . In the second iteration, PC-simple initially lets \mathbf{PC}^2 equal to \mathbf{PC}^1 , then it updates \mathbf{PC}^2 by removing from it the variables that are independent of Z given any other single variable in \mathbf{PC}^1 . Similarly in the third iteration, \mathbf{PC}^3 is initially equal to \mathbf{PC}^2 , then \mathbf{PC}^3 is updated by removing from it the variables that are independent of Z given any other two variables in \mathbf{PC}^2 . This process iterates until the number of variables in \mathbf{PC}^k is not more than k , and \mathbf{PC}^k is output as the final \mathbf{PC} set.

In the following, we use an example to run through the PC-simple algorithm.

Example 2.1. Supposing that we have a data set for variables $\{A, B, C, D, E, F, G, H\}$ and the target variable Z , such that the results of the conditional independence tests based on the data set can be represented by the Bayesian network structure in Fig. 2.1. Then the steps of PC-simple are given as the following.

1. $k = 0$ and $\mathbf{PC}^0 = \{A, B, C, D, E, F, G, H\}$.
2. As $|\mathbf{PC}^0| > 0$, enter the **while** loop.
3. $k = 1$ and $\mathbf{PC}^1 = \mathbf{PC}^0 = \{A, B, C, D, E, F, G, H\}$.
4. For each variable in \mathbf{PC}^1 , order zero independence test with Z is conducted given $\mathbf{S} = \emptyset$. At the end of this iteration of the **while** loop, D, E and H are removed from \mathbf{PC}^1 since they are independent of Z , and $\mathbf{PC}^1 = \{A, B, C, F, G\}$.
5. $|\mathbf{PC}^1| > k$ ($k = 1$), start the second iteration of the **while** loop.
6. $k = 2$ and $\mathbf{PC}^2 = \mathbf{PC}^1 = \{A, B, C, F, G\}$.
7. For each of the five variables in \mathbf{PC}^2 , order one independence tests are conducted conditioning on any other single variable in \mathbf{PC}^1 each time ($|\mathbf{S}| = 1$). At the end

Algorithm 2.1: The PC-simple algorithm [2]

Input: \mathbf{D} , a data set for the set of predictor variables $\mathbf{X} = \{X_1, X_2, \dots, X_m\}$ and the target variable Z ; and α , significance level for conditional independence tests.

Output: \mathbf{PC} , the subset of $\{X_1, X_2, \dots, X_m\}$ that comprises parents and children of Z

```

1: let  $k = 0$ 
2: let  $\mathbf{PC}^k = \{X_1, X_2, \dots, X_m\}$ 
3: while  $|\mathbf{PC}^k| > k$  do
4:   let  $k = k + 1$ 
5:   let  $\mathbf{PC}^k = \mathbf{PC}^{k-1}$ 
6:   for each  $X \in \mathbf{PC}^{k-1}$  do
7:     for each  $\mathbf{S} \in \mathbf{PC}^{k-1} \setminus \{X\}$  and  $|\mathbf{S}| = k - 1$  do
8:       if  $X$  and  $Z$  are independent given  $\mathbf{S}$  at significance level  $\alpha$ 
9:         let  $\mathbf{PC}^k = \mathbf{PC}^k \setminus \{X\}$ 
10:    end for
11:  end for
12: end while
13: output  $\mathbf{PC}^k$ 

```

of this iteration, only G is removed from the \mathbf{PC} set and $\mathbf{PC}^2 = \{A, B, C, F\}$ (see Fig. 2.1 which shows that G is independent of Z given its parent F , which blocks G and Z).

8. $|\mathbf{PC}^2| > k$ ($k = 2$), start the third iteration of the **while** loop.
9. $k = 3$ and $\mathbf{PC}^3 = \mathbf{PC}^2 = \{A, B, C, F\}$.
10. For each of the four variables, its independence with Z is tested given the combination of any other two variables in \mathbf{PC}^2 . Then at the end of this iteration, A is removed as Z is independent of A given its two parents B and C , and $\mathbf{PC}^3 = \{B, C, F\}$.
11. Since $|\mathbf{PC}^3| = k$ ($k = 3$), the **while** loop exits, and \mathbf{PC}^3 is output as the final \mathbf{PC} set of Z . $\mathbf{PC} = \{B, C, F\}$.

That is, the \mathbf{PC} set of Z is $\{B, C, F\}$, which is consistent with the structure in Figure 2.1.

2.3 Discussions

2.3.1 Complexity of PC-simple

Referring to Algorithm 2.1, the complexity of PC-simple comes from the time taken for conditional independence tests. In the worst case, i.e. all the variables X_1, X_2, \dots, X_m are parents or children of Z , the maximum order of conditional independence tests is $m - 1$ (starting with order zero test); at level k ($0 \leq k \leq m - 1$), all the m variables in $\{X_1, X_2, \dots, X_m\}$ need to be tested for conditional independence with Z ; and for each of the variables, there are up to $\binom{m-1}{k}$ conditioning sets to be

tested. Therefore overall, in the worst case, the number of conditional independence tests is $m \sum_{k=0}^{m-1} \binom{m-1}{k}$, which is $m2^{m-1}$. Hence in the worst case, the time complexity of PC-simple is exponential to the number of variables, which means that when the number of variables or m increases, the performance of PC-simple may degrade dramatically.

However, if most of the variables in $\{X_1, X_2, \dots, X_m\}$ are not direct causes or effects of Z (or the degree of Z is small in an underlying Bayesian network), the size of the **PC** set is reduced quickly, and the same for the number of conditioning sets in each iteration of the **while** loop. So the **while** loop will be completed after a relatively small number of iterations because $(|\mathbf{PC}^k| > k)$ is violated.

Hence when the number of direct causes and effects of a target is small, PC-simple can handle high dimensional data sets. For example, in [2] PC-simple was applied to a real world data set with over 4000 variables. Such data sets are called sparse data sets since the underlying Bayesian networks are sparse (the degree of a node is very small). Normally, when the number of causal variables is 30 or more, PC-simple will not work well on a normal desktop computer.

2.3.2 False Discoveries of PC-simple

There are two sources for the false discoveries of PC-simple (1) algorithm design; and (2) input data.

Referring to Algorithm 2.1, PC-simple updates the **PC** set of Z after order k conditional independence tests are done, by removing all the variables that have been tested to be conditionally independent of Z . This update is reasonable in terms of removing conditionally independent variables of Z as a result of order k conditional independence tests. However, as the independence tests at the next level will be conditioned on the subsets of the updated **PC** set, the removed variables will not be considered as part of the conditioning sets, which will lead to false discoveries.

For example, supposing Fig. 2.2 shows the underlying causal relationships among the nine variables, where the true **PC** set of the target variable Z is $\{B, C, F\}$. However, the discovered **PC** set by PC-simple will also include G (assuming all the conditional independence tests are correct), which is a false positive. This is caused by the removal of E based on the result of order zero conditional independence test, as E is independent of Z (given an empty set). So E will not be in any of the conditioning sets for higher order conditional independence tests, and as a result, G will not be removed since G is independent of Z given both E and F (G 's parents). G is included in the final **PC** list of Z as a false positive.

In [1], the idea of a symmetry correction is introduced to remove such false positives in local causal discovery. If G is a true parent or a child node of Z , then Z should be in the **PC** set of G . Let $\mathbf{PC}(Z)$ and $\mathbf{PC}(G)$ be the **PC** sets of Z and G respectively. In the example, PC-simple outputs $\mathbf{PC}(Z) = \{B, C, F, G\}$, and $\mathbf{PC}(G) = \{E, F\}$. Z is not in $\mathbf{PC}(G)$. Based on the symmetry property, G should not be in $\mathbf{PC}(Z)$ and hence be removed. The symmetry correction, however, introduces higher time complexity because we need to test the symmetry for each variable in $\mathbf{PC}(Z)$ in order to

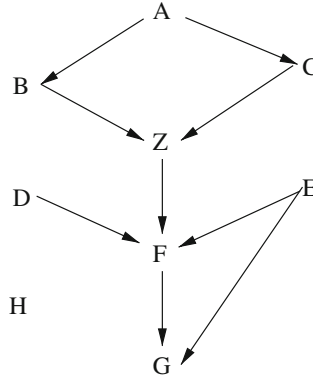


Fig. 2.2: An example scenario for false discovery of PC-simple

remove the false positives. The studies in [1] with a number of real world data sets have shown that the situations in which the false positives may occur are not often in practice.

PC-simple will produce false discoveries if a conditional independence test is incorrect at any stage. An incorrect test causes a variable to be falsely removed or included, and such a false exclusion or inclusion will result in incorrect conditioning sets for conditional independence tests at the following levels. As a consequence, an incorrect test result may cause a chain of false positives and/or false negatives. Therefore the algorithms should be used with caution if there is no sufficient number of samples or there is selection bias in the collection of samples. A number of different conditional independent tests should be used to ensure the reliability of the results.

2.3.3 The Causal Assumptions

PC-simple is based on causal Bayesian network theory, and it follows the same causal assumptions as those by causal Bayesian networks. In the context of causal Bayesian networks, to link a causal graph with a probability distribution, the two essential causal assumptions are the Causal Markov Condition and the Faithfulness Condition, as described below.

Let $\mathbf{G} = (\mathbf{V}, \mathbf{E})$ be a directed acyclic graph (DAG), where the nodes \mathbf{V} represent a set of random variables and the edges \mathbf{E} represent causal relationships between the nodes. An edge from a node X to a node Y indicates that X is a direct cause of Y , and X is known as the parent of Y . Let $P(\mathbf{V})$ be the joint probability distribution of \mathbf{V} .

Assumption 2.1 (Causal Markov Condition [8]) \mathbf{G} and $P(\mathbf{V})$ satisfy the Causal Markov Condition if and only if given the set of all its parents, a node of \mathbf{G} is independent of all its non-descendants according to $P(\mathbf{V})$.

Simply speaking, this condition states that every edge in a DAG implies a probabilistic dependence. In other words, this condition has drawn a one way link from edges to dependencies.

However, it may be possible that some of the conditional independence relationships are not reflected by the DAG, so we need the following assumption to draw the link from dependencies to edges.

Assumption 2.2 (Faithfulness Condition [8]) *G and $P(V)$ satisfy the Faithfulness Condition if and only if every conditional independence relationship in $P(V)$, $\text{Ind}(X, Z|S)$ is reflected in G in the way that S is the set of all parents of X (or Z) and Z (or X) is not a descendent or a parent of X (or Z).*

In other words, Faithfulness Condition assumes that if two variables are probabilistically dependent, there must be a corresponding edge between the two variables in the graph.

With the above two conditions, we have established the mapping between conditional dependence relationships specified by $P(V)$ and the causal relationships (edges) represented by a causal DAG. Therefore, an algorithm that discovers the correct conditional independence relationships also produces the mapped true causal relationships.

There may be more than one causal DAGs which faithfully reflect the same conditional independence relationships in a probability distribution. However, these DAGs have the same and unique underlying undirected graph (skeleton), although the directions of edges in the DAGs may be different [5]. For example, the DAGs $A \rightarrow B$ and $A \leftarrow B$ reflect the same conditional independence relationship $\neg \text{Ind}(A, B|\emptyset)$, i.e. A and B are dependent. The skeleton of both DAGs is $A - B$.

In causal Bayesian network structure learning, an algorithm is able to obtain from the data (with sufficient number of samples) an equivalence class of the causal DAGs [5]. The equivalence class can be represented as a partially directed graph, where a directed edge $X \rightarrow Y$ indicates that all causal DAGs in the class have the edge $X \rightarrow Y$ and an undirected edge $X - Y$ shows that some of the causal DAGs in the class contain $X \rightarrow Y$ while some of them contain $X \leftarrow Y$.

The set of parents and children nodes of a node in a learned causal Bayesian network are deterministic although the direction of the edge between a node in the set and the target node may be nondeterministic. Local causal discovery algorithms like PC-simple and HITON-PC are able to learn a unique parents and children set of Z in a data set with sufficient number of samples under the same causal assumptions. The cause-effect relationships around a given target variable Z are not directed, i.e. we do not know whether a variable in the learned set is the cause or effect of the target variable. Nevertheless, the undirected causal relationships are useful for causal exploration as discussed in Chap. 1.

Here we indeed have also assumed the following causal sufficiency about the data sets used by the algorithms.

Assumption 2.3 (Causal sufficiency [8]) *For every pair of variables which have their observed values in a given data set, all their common causes also have observations in the data set.*

This assumption assumes that there is no unmeasured or hidden causes. For example, if a common cause of two variables is unobserved in a data set, PC-simple may conclude that the two variables have a causal relationship while in fact they do not.

2.4 An Implementation of PC-simple

PC-simple [2] is implemented in *pcalg* [3] as the *pcSelect* function. In this section, we present some examples in *R* [7] to demonstrate the usage of the *pcSelect* function. We assume that readers are familiar with *R*. If not, the *R* introduction documentation can be found in [9].

2.4.1 Example 1: Using PC-simple in *pcalg*

In this example, we use the built-in data set, *gmB*, in the *pcalg* package for demonstrating the usage of the *pcSelect* function. The *gmB* data set includes five binary variables (columns) with 5000 samples (rows). The data is stored in *gmB*\$*x* and the known true DAG is *gmB*\$*g* as shown in Fig. 2.3. In the following, we assume that variable *V2* (i.e. node 2 in Fig. 2.3) is the target variable and other variables are predictor variables, and we apply the PC-simple algorithm to the *gmB* data set.

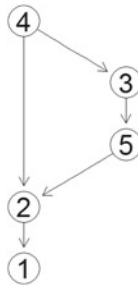


Fig. 2.3: The ground truth Bayesian network of the *gmB* data set

```

> library(pcalg)
> library(Rgraphviz) # for drawing graphs
> data(gmB)
> plot(gmB$g)
> results = pcSelect(gmB$x[,2], gmB$x[,-2], alpha=0.01)
> results

```

```

$G
      V1      V3      V4      V5
TRUE FALSE  TRUE  TRUE
$zMin
[1] 29.4191620  0.6714314  5.5774279 44.3422923

```

The result (\$G) demonstrates that variables $V1$, $V4$, $V5$ are in the parents and children set of the target variable, while variable $V3$ is not. $zMin$ is the set of z -values of the conditional independence tests corresponding to each of the predictor variables. The bigger a z -value is, the stronger the association the predictor variable and the target variable have. More details on the conditional independence test used in PC-simple can be found in Appendix A.

2.4.2 Example 2: Using the Data Sets of Figs. 2.1 and 2.2

The data set that has the same dependence relationships as the Bayesian network shown in Fig. 2.1 can be downloaded from:

<http://nugget.unisa.edu.au/Causalbook/>

The data set contains eight predictor variables and the target variable as described in Example 2.1. The data set was generated using the TETRAD software downloaded from <http://www.phil.cmu.edu/tetrad/>. Assume that the data set *Example21.csv* has been downloaded and stored in the *R* working directory. We now use the *pcSelect* function to find the parents and children set of Z .

```

> library(pcalg)
> data=read.csv("Example21.csv", header=TRUE, sep=",")
> pcSimple.Z=pcSelect(data[,9], data[, -9], alpha=0.01)
> pcSimple.Z
$G
      A      B      C      D      E      F      G      H
FALSE TRUE  TRUE FALSE FALSE  TRUE FALSE FALSE
$zMin
[1] 1.85790685 144.57474808 16.65675448 0.27538423
[5] 0.76391512 15.94735566 0.07287629 0.23537966

```

The parents and children set of Z is $\{B, C, F\}$, which is consistent with the graph in Example 2.1. We can also verify this result against the global causal structure (Fig. 2.4) learned by the PC algorithm [5, 8]. The DAGs in Figs. 2.1 and 2.4 have the same skeleton and Z has the same **PC** set in both Figures. The following codes are used to learn the causal structure from the *Example21.csv* data set using the PC algorithm.

Practical Approaches to Causal Relationship
Exploration

Li, J.; Liu, L.; Le, T.

2015, X, 80 p. 55 illus., Softcover

ISBN: 978-3-319-14432-0