Master Thesis Business Analytics

Connecting the Dots

Developing an Algorithm for Consistency Checking in Military Intelligence

---

Author
Milou Anna Barbara Meeuwisse
2536188

A thesis submitted in fulfillment of the requirements for the degree of Master of Science in Business Analytics

Supervisors
Dr. S.K. Smit (External supervisor)
Prof. dr. W.J. Fokkink (Graduation supervisor)
Prof. dr. R.D. van der Mei (Second reader)

February 2018 - August 2018

Host Organization
TNO
Modeling, Simulation and Gaming
Oude Waalsdorperweg 63
2597 AK Den Haag

University
Vrije Universiteit Amsterdam
Faculty of Science
De Boelelaan 1081a
1081 HV Amsterdam

Preface

The final stage of the Master's program Business Analytics is the Master Project Business Analytics. This six-month graduation project has to be carried out within a company. I conducted my research at host organization TNO, department Modelling, Simulation & Gaming. Thanks to my supervisor there, I was able to perform my research in the field of defence, safety & security. I want to thank Selmar Smit for his support during the past six months. Your expert-knowledge and endless ideas have been a great inspiration for my research. I would also like to thank Freek van Wermeskerken for his additional supervision.

Besides, I would like to thank my supervisor from the Vrije Universiteit, Wan Fokkink, for reviewing my thesis. Your feedback provided me to improve my research. Lastly, I would like to thank Rob van der Mei for being the second reader of this thesis.

Milou Meeuwisse, August 2018

Executive Summary

Military Intelligence is a discipline that uses different information collection and analysis approaches to provide commanders with situational awareness and understanding in order to support the decision-making process. Intelligence analysts have to deal with deceptive information. Correctly identifying misleading information is of high importance to provide reliable intelligence products. The overall objective of this research is to explore a technique to assist intelligence analysts in correctly identifying deceptive information. Therefore, a use-case is defined focusing on moving entities. Entities leave sightings while travelling. Concatenating these sightings creates movements of the entity over time and in space. Deceptive or false information might be present in the entity movement. Being able to filter the misinformation, leads to more reliable intelligence. Therefore, the goal of this research is to design an algorithm which is able to detect inconsistencies in observations of moving entities.

A framework is designed to generate and evaluate entity movements. An entity movement is defined as a chain of chronologically ordered sightings. Each possible combination of sightings is referred to as a hypothesis. Therefore, the algorithm framework consists of a hypothesis evaluator and hypothesis generator.

The hypothesis evaluator evaluates entity movements by Bayesian inference. Using Bayes' theorem the likelihood of a hypothesis can be computed. This likelihood depends on the reliability of the sightings and the characteristics of the sighting chain. The characteristics may consist of dependent variables. To model the dependency between variables, copula functions are integrated into the algorithm. Copula functions are able to model the dependence structure of dependent variables separate from the univariate marginals.

The hypothesis generator generates all possible combinations of sightings. These combinations can be presented in a search tree, respecting the chronological ordering of the sightings. To reduce computation time, backtracking is used to traverse the search tree. When a hypothesis is generated, it is passed on to the hypothesis evaluator. Depending on the likelihood returned by the evaluator, the hypothesis is either expanded by the next sighting in the tree, or pruned. In this way, only the promising hypotheses are generated and evaluated.

The designed algorithm is first validated on simulation data, after which it is applied to real-world data concerning fugitives. For both data sets the route characteristics and source reliability are modelled using historic data. The results show that the designed algorithm is able to correctly classify false and true sightings. However, the copula functions might introduce noise, resulting in a poor performance of the algorithm.

The potential of the algorithm is demonstrated for both intelligence and crime analysis. Due to the modularity of the framework, the algorithm can be applied to multiple use-cases in these two fields. Besides, the algorithm can be applied to predict future sightings. When hypothetical future sightings are added to a historic collection of sightings, the likelihood of future entity movements can be evaluated.

Table of Contents

List of Figures

List of Tables

1 Introduction

This chapter provides a general introduction in the subject of this thesis. First, the research problem is stated and the objective of this research is defined. Subsequently, the research goal and its corresponding sub-questions are specified. The chapter finishes with a brief description of the structure of this thesis.

1.1 Problem Statement

Military Intelligence is a discipline that uses different information collection and analysis approaches to provide commanders with situational awareness and understanding in order to support the decision-making process. Complete and accurate information about the mission environment is essential to a military unit in order to operate effectively. Intelligence is the result of dedicated collection and processing of information from different sources (e.g., imagery information, human information, open source information). While these sources provide large volumes of information, its relevance and credibility may be questionable. Intelligence analysts have to be aware of deceptive information and misinformation both intentional and unintentional. These types of information are misleading to intelligence analysts, causing unreliable intelligence products. In the end this may lead to incorrect decisions during an operation. Correctly identifying misleading information assists analysts to filter out irrelevant and unreliable information. This is however a challenging task, especially with the large volumes of available information analysts have to deal with nowadays.

Santos et al. (2005) encountered the same problem in intelligence analysis and wrote the paper *A Cognitive Framework for Information Gathering with Deception Detection for Intelligence Analysis*. They developed an information retrieval (IR) application which collects information based on the user's interests. In addition, they incorporated automatic document summarization enabling analysts to identify relevant documents more effectively. The IR application provides a way to reduce the problem of information overload in selecting relevant information [1]. Using this application, the analysts still face the problem of evaluating the reliability and credibility of the information they retrieve. Santos et al. consider monitoring for inconsistencies in the retrieved information to be the most useful approach in detecting mis- and disinformation.

1.2 Objective

The overall objective of this research is to explore an approach to consistency checking in intelligence to assist intelligence analysts in identifying misleading and deceptive information. A use-case is defined to scope the objective. Therefore, this research focuses on information concerning moving entities. Entity movements may contain important information about, for example, the adversary or allies and where they might move. Consider an entity moving through a mission area. The entity can be observed by multiple sources, at different places, at different moments in time, leading to observations of the entity. Concatenating these observations creates movements of the entity over time and in space. However, there might be intentionally or unintentionally misleading information included in the observations. An example of a moving entity is the BUK missile launcher which downed flight MH17. Open-source information is available about the movements of the BUK missile. This open-source information includes both correct and incorrect information about the BUK. Being able to detect the misinformation, might help to track down the origin of the missile launcher.

Figure 1 shows a fictive chain of four sightings of which the third is deceptive. Including the deceptive sighting, the entity movement contains misinformation. Being able to detect the deceptive sighting helps overcoming this problem and leads to more reliable information.

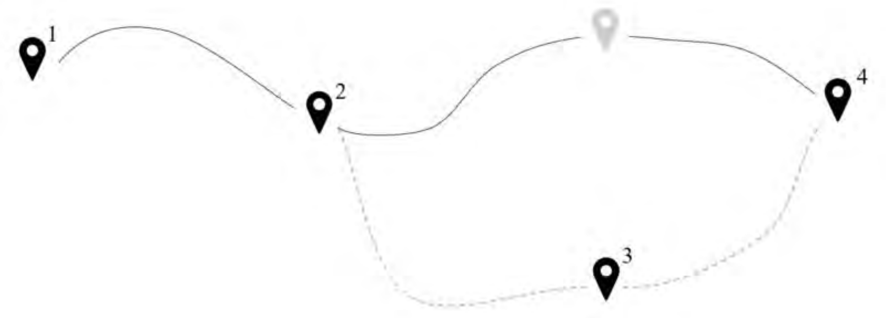


Figure 1: *Chain of four sightings of a moving entity.*

1.3 Research Goal

Regarding the main objective of this research and the use-case used for it, the main goal of this research is as follows:

Develop an algorithm which is able to detect inconsistencies in observations of moving entities.

In order to achieve this goal the research is divided into multiple sub-questions:

1. *Which techniques are suitable for evaluating entity movements?*
2. *How can characteristics belonging to a moving entity be integrated into the algorithm?*
3. *Which techniques are suitable for constructing entity movements?*
4. *Which characteristics belonging to a moving entity are required in order to apply the algorithm to a real-world example?*

NB: From this point on, entity movement(s) are referred to as route(s). Besides, the words evidence, observations and sightings are used interchangeably in this research.

1.4 Structure

This thesis is structured as follows. First, Chapter 2 provides relevant background information about military intelligence in order to be able to fully understand this research. Next, Chapter 3 discusses relevant literature regarding the research problem. In Chapter 4 the algorithm design is given. Besides, its main components are theoretically described and demonstrated. Next, the algorithm is validated on simulation data, which is described in Chapter 5. After the validation phase, the algorithm is applied to a real-world example. The real-world example and its results are described in Chapter 6. Chapter 7 provides the overall conclusion of the research. Besides, recommendations are given and suggestions for future research are presented.

2 Intelligence Background

This chapter provides background information about military intelligence. The process of intelligence is briefly described. The summary is based on the Joint Doctrine Publication 2 Intelligence of the Ministry of Defence The Netherlands [2]. In addition, a commonly used intelligence analysis technique is described. The chapter concludes with critiques on the intelligence cycle which emerged over the past decades.

2.1 Intelligence Cycle

Military operations are supported by means of intelligence. Intelligence provides commanders with situational awareness in order to support the decision-making process. The intelligence cycle is an essential part of the intelligence process and consists of four phases: *Direction*, *Collection*, *Processing* and *Dissemination*. The intelligence process is supported by the management process of the cycle, called *Information Requirements Management and Collection Management* (IRMCM). The cycle and its management process are shown in Figure 2 and briefly described below.

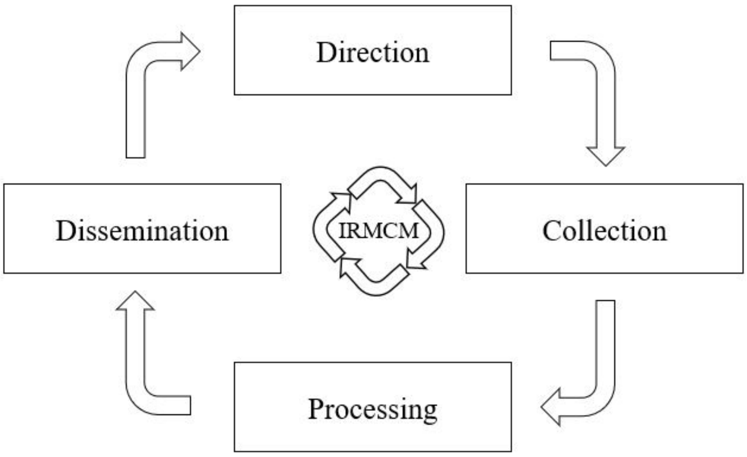


Figure 2: *The intelligence cycle.*

Direction
The intelligence cycle process is driven by the commander of the associated military operation. The objectives of an operation are defined in the so-called end state. In order to achieve this end state, the operational commander needs to have situational understanding (SU). Acquiring SU starts with having situational awareness (SA): being aware of the operational environment and who is doing what and where. Based on the SA intelligence requirements are determined to eventually meet the objectives of the operation. The intelligence requirements are contained in the Intelligence Collection Plan (ICP). The ICP consists of the Primary Intelligence Requirements (PIRs). These are a couple of broad questions representing the most important requirements. The PIRs are divided into Specific Information Requirements (SIRs). These are multiple specific sub-questions supporting the PIRs. Subsequently, the SIRs are translated to Essential Elements of Information (EEIs). EEIs are statements or questions which can actually be observed or recognised. Based on the EEIs, different collection orders can

be formulated. Each collection order is labelled by observable and recognisable indicators bounded in time and space. During the direction phase the ICP is formulated and the collection orders are issued to the collection means that are under command of the own operational commander. Besides, it is possible to send a Request for Information to other intelligence organs. The issued orders are executed in the Collection phase.

Collection

Once the ICP has been formulated and the collection orders are issued, the collection phase starts. During this phase the recipient of a collection order designs a plan to address the order. Based on the plan there is determined which source(s) are able to timely execute the order. The different sources and their information collection methods are described in *Appendix A.1.* The collected information might be pre-processed by single-source analysts. Each analyst specialises in a certain source and is able to interpret and evaluate the collected information of that source. Their processed information is called single-source intelligence. The collected information and single-source intelligence will be delivered to the processing phase.

Processing

During the processing phase the collected information is registered, evaluated, analysed, integrated and interpreted. These steps are executed sequentially as well as in parallel. First, the information is structured and compared with other collected information. After that, it is systematically registered, which results in a database. This database can be used for further analysing the information. Next, the collected information is evaluated. The reliability of the source and the credibility of the information are evaluated independently. This is because, information obtained from a generally reliable source is not necessarily true and vice versa. The credibility- and reliability values are combined to an alphanumeric value, which are explained in *Appendix A.2.* In the last three steps, the information is analysed, integrated and interpreted using various analysis techniques. During these steps, the structured and evaluated information is investigated and put in context of the operation. At the same time, existing information and knowledge is taken into account in order to detect trends and developments. Finally, the trends and developments are interpreted and conclusions are drawn regarding the current and future state of the operation.

Dissemination

The last phase in the intelligence cycle is dissemination. During this phase the intelligence needs to be distributed to the commander. There are various ways to present the intelligence products. The four most common forms are: verbal by means of briefings, written by means of reports, graphical by means of maps, and in data format. The value of intelligence decreases by the passage of time. Therefore, the timely distribution of intelligence is of high importance.

IRMCM

The coordination of all collection and processing activities in the intelligence cycle is managed by IRMCM. It monitors both the progress of these activities and the timely dissemination of the intelligence. IRMCM starts with an information requirements analysis and the investigation in which information is already available. In addition, it is assessed which information can be collected by the collection means that are under command of the own operational commander and which information must be obtained by activities of other intelligence organs. IRMCM also ensures the information is properly stored and communicates and coordinates with other intelligence staffs regarding the collection and exchange of information.

Note that the intelligence cycle is a never ending process. The collected information and intelligence are continuously evaluated and adjusted if necessary. Therefore, intelligence activities can never be considered as completed. As long as a military operation is running, the intelligence cycle is an active process.

2.2 Analysis of Competing Hypotheses

An analysis technique often used in the processing phase of the intelligence cycle is *Analysis of Competing Hypotheses*, also referred to as ACH. This technique helps intelligence analysts judging issues that require careful weighing of alternative explanations (hypotheses). ACH was introduced by Heuer [5], a former CIA staff officer. The first step of the analysis is to generate a set of alternative hypotheses representing for example, potential answers to questions of the ICP. Subsequently, the hypotheses are systematically and simultaneously evaluated based upon the collected evidence. During this evaluation there is a focus on evidence that tends to disconfirm rather than to confirm each of the hypotheses. The most probable hypothesis is usually the one with the least evidence against it, not the one with the most evidence for it [4].

Two common pitfalls in intelligence analysis are tunnel vision and confirmation bias. In general, analysts are overly influenced by a first impression of the situation and fail to explore other hypotheses. Besides, analysts tend to rely on evidence supporting their favoured hypothesis and thereby discard evidence contradicting the same hypothesis. ACH can help overcome these two common pitfalls.

In the use-case of this research, a hypothesis can be defined as an entity movement. An entity movement is equal to a chain of observations of a moving entity. Therefore, a hypothesis in this research is equal to a chain of observations. An example of two competing hypotheses for the entity movement shown in Figure 1 is shown in Figure 3.

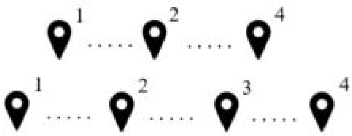


Figure 3: *Two competing hypotheses.*

2.3 Critiques Intelligence Cycle

The intelligence cycle has been discussed over the past decades. The discussion is more or less two-fold. The first point of criticism is that the intelligence cycle is not well defined. It would not be an accurate reflection of the way intelligence is produced. The second point is about the cycle being old-fashioned. Considering the development of (communication) technology and the abundance of available information, the cycle would not meet today's needs.

In 2006 Arthur Hulnick stated in his article *What's wrong with the intelligence cycle?* that the intelligence cycle is not a particularly good model. “It is really not a very good description of the ways in which the intelligence process works [6].” He argues that the intelligence cycle represents a sequential process and does not provide for iterations between steps. Especially the collection- and processing phase should be considered as operating in parallel rather than sequentially. Johnston & Johnston (2005) state that because of this weakness, understanding the challenges and responsibilities of intelligence analysis becomes much more difficult. They recommend to either redesign the traditional intelligence cycle model to depict accurately the intended goal, or to discuss explicitly its limitations whenever it is used [7].

Another reason to argue that the intelligence cycle should be redefined is that the model is outdated. In 1949 Sherman Kent published the book *Strategic Intelligence for American World Policy*. With this book Kent was the first to define the different stages of intelligence analysis. This means that the intelligence cycle has been defined more than 60 years ago. Ayden & Ozleblebici (2015) argue in their article *Is Intelligence Cycle Still Viable?* that the traditional cycle does not meet today's needs. The development of technology and communication has resulted in an enormous increase in the amount of available information. This information overload has fundamentally altered the way intelligence organizations collect, collate and analyze their data, argue Knopp et al. (2016). However, the development of tradecraft to exploit the information has lagged behind [10]. Managing and processing these quantities of data is a challenge intelligence analysts are facing today. At the time Kent defined the intelligence cycle the available information was limited compared to the information available nowadays. The traditional cycle would not be able to anticipate today's large volumes of data.

Several authors attempted to modify the intelligence cycle in order to rectify the weaknesses of the traditional cycle. Additionally, the proposed intelligence cycles aim to improve the model by increasing the effectiveness and making it more realistic. However, Wheaton (2012) argues in his article *Let's Kill the Intelligence Cycle* that “none of the proposed models seek to discard the fundamental vision of the intelligence process described by the cycle [8]” and so the discussion continues.

The mentioned critiques on the intelligence cycle have to be kept in mind when designing and developing the algorithm of this research. The algorithm must be able to respond to today's intelligence needs. Especially the big data problem should be taken into account. Handling large quantities of data should not be a problem to the algorithm. Moreover, the timely dissemination of intelligence should be taken into consideration. Despite the critiques on the cycle, the timely dissemination of intelligence will always be of high importance. This means that the algorithm must be able to handle large quantities of data on the one hand, and have a sufficient runtime on the other.

3 Relevant Work

Chapter 1 introduced the general problem of identifying misleading information in intelligence analysis. This research focuses solely on sources providing information about moving entities. Identifying inconsistencies in intelligence data describing entity movements is a specific problem within a specific domain. Literature describing both consistency checking and the military intelligence domain in a single research is scarcely available. This does not necessarily mean that this kind of research is not conducted. Instead, it might not be publicly available. Alternatively, research concerning situational awareness in military operations can be reviewed. This is relevant because the final purpose of identifying misleading and deceptive information is to contribute to better situational awareness. Approaches to provide situational awareness might be of interest for the algorithm to develop in this research. Therefore, this chapter discusses various approaches and techniques used in intelligence to contribute to better situational awareness. In this way a reasonable decision can be made about the design of the algorithm.

In 1972 Zlotnick, a former analyst at the Central Intelligence Agency, wrote the article *Bayes' Theorem for Intelligence Analysis*. In this article he argued: "The very best that intelligence can do is to make te most of the evidence without making more of the evidence than it deserves. The best recourse is often to address the probabilities [11]." The approach he suggests is Bayes' Theorem in odds-ratio form. Intelligence analysts often need to estimate the comparative merits of two competing hypotheses. Using Bayes' theorem in odds form, the estimate can be expressed in terms of the odds favouring or disfavouring the hypothesis. In this way, the analyst does not take the evidence as given. Nowadays, Bayesian analysis is a commonly used technique in creating situational awareness in military- and crime intelligence. The following two articles are examples in respectively both the domains.

Barros et al. (2014) introduce a near real-time intelligence technique in order to produce reliable and up-to-date situation awareness. This supports contemporary military operations and especially those of which time is of the essence. For these operations real-time analysis of potential threats is of vital importance. The analysis technique is a combination of Bayesian Belief Networks (BBN) and spatio-temporal modelling. The BBN is modelled by using the Hypothesis Management Framework (HMF). This framework enables the simultaneous quantitative evaluation of possible hypotheses and can be seen as the probabilistic counterpart of ACH [12]. The threats which need to be analysed can be modelled as hypotheses. Therefore the HMF is a suitable approach to quantify the probability of alternative threats based on available information. However, the threats need to be assessed in a specific time and space. These spatio-temporal factors are not included in the HMF. Therefore, the HMF is extended by spatio-temporal models in order to make the threat assessment space and time dependent.

Smit et al. (2016) also make use of Bayesian analysis in their research. They present QUIN, a support system for crime analysts able to model different crime scenarios and reason about what happened. This system helps analysts to overcome the challenge of processing large volumes of information and selecting which information to attend and which to ignore [13]. QUIN is a combination of crime-scripting and Bayesian reasoning. Domain knowledge and expert knowledge about criminal activities is captured in the crime scripts. This knowledge is used to reason about the likelihood of possible scenarios based on available evidence. Based on the ACH technique different scenarios are compared and assessed. For each defined scenario the given evidence is filled into the scenario. By using multiple BBNs conditional probabilities are calculated resulting in the likelihood of the scenario given the new element of information. Furthermore, one can obtain how the likelihood of a scenario develops by adding more evidence. When the same evidence is added to different scenarios, the development of the likelihood can be compared in order to identify which scenario has truly happened.

Both the articles of Barros et al. and Smit et al. describe a Bayesian analysis based on ACH. They start with defining possible threats or scenarios. Subsequently, these hypotheses are evaluated simultaneously by one or more BBNs. This process has a few important drawbacks. First, the analysis is totally dependent of properly defined hypotheses. This means that expert knowledge is indispensable. Second, once a threat is missing in the BBN or a scenario is not present in the knowledge base, it cannot be taken into account. This could thereby force an analyst into tunnel vision. However, these drawbacks are not relevant for our research. This is because we are not dealing with hypotheses/scenarios for which expert knowledge is required. The hypotheses we are dealing with are simple chains of sightings. Therefore, a BBN might be a suitable approach for the algorithm of this research.

Bayesian reasoning is obviously not the only intelligence technique which might be suitable for creating situational awareness. Ceolin et al. (2013) describe a research in the naval domain in which another technique is applied. In the naval domain Automated Identification System (AIS) messages are exchanged to locate and to identify ships and to avoid collisions. In this way it is possible to keep track of the positions of ships in combination with their identity. The trustworthiness of the information provided by AIS messages is questionable. Therefore, a model is designed to determine the reliability of AIS messages. One of the techniques used to obtain these results is Subjective Logic [15]. Subjective logic is a type of probabilistic logic that allows probability values to be expressed with degrees of uncertainty. Arguments in subjective logic are called opinions, represented by $\omega^{subject}_{object}$. The object can be interpreted as a proposition to which the opinion applies, such as ‘the name of this ship is X’. The subject can be interpreted as the source or the opinion holder, for example an AIS message. An opinion represents the evidence that a certain subject has up to a certain moment by means of four parameters: belief, disbelief, uncertainty and an a priori value. An important aspect of opinions is that they are equivalent to Beta or Dirichlet distribution under a specific mapping [15]. This means that an opinion can be expressed as the probability of a proposition being true. Subjective logic is useful in the maritime domain, because different subjects may have different opinions about the same object. This makes it possible to compute the ratio between agreeing and disagreeing observations, as an indicator of the trustworthiness of observed values [14]. However, in our specific research subjective logic is not applicable. The reason is that in our research hypotheses are defined as chains of observations. In terms of subjective logic, this means that the object consists of the subjects. This leads to undesirable complexity. Nevertheless, subjective logic might still be useful in the intelligence domain. Pope & Josang (2005), for example, discussed the usage of subjective logic as an elaboration of the ACH process in [16].

In addition to the discussed probabilistic approaches, artificial intelligence (AI) might also contain useful techniques for the algorithm to be designed. However, the current

AI techniques are seen as uninterpretable black-boxes, generating solutions which might be unexplainable [17]. An important requirement for techniques used in intelligence analysis is to be explainable and transparent. Otherwise, a technique is hard to trust and will not be accepted and used by analysts. Furthermore, there is not yet a strategy for applying AI in intelligence analysis [17]. For these reasons, AI methods are not investigated as potential approaches for the algorithm to design in this research.

4 Connecting the Dots

In this chapter the algorithm is designed and theoretically explained. The first section describes the design of the algorithm and the components it consists of. The two subsequent sections theoretically explain the working of the algorithm components.

4.1 Algorithm Design

Inspired by the approaches and techniques discussed in *Relevant Work*, their pitfalls and applications to our research, the algorithm can be designed. Recall that the aim of the algorithm is to detect inconsistencies in observations of moving entities. In order to detect inconsistencies in entity movements, multiple alternative movements have to be compared. This means that the algorithm should at least contain two components: a movement generator and a movement evaluator. Chapter 2 already stated that an entity movement can be defined as a chain of observations. Each possible chain of observations equals a hypothesis which can be evaluated. Therefore, the following algorithm design is able to detect inconsistencies in entity movements:

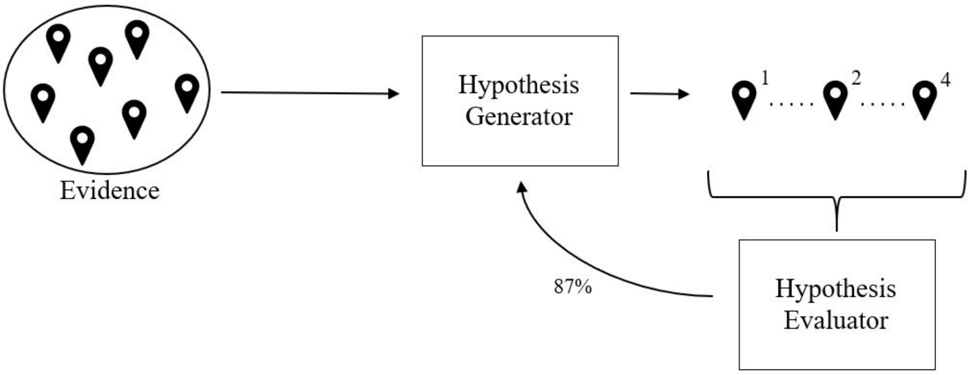


Figure 4: *Algorithm design consisting of a hypothesis generator and a hypothesis evaluator.*

The algorithm consists of two components: the hypothesis generator and the hypothesis evaluator. Besides, the input for the algorithm is the collection of sightings concerning a moving entity. This corresponds to the information collected in the *Collection*-phase of the intelligence cycle. The hypothesis generator is fed with the evidence. Subsequently, the generator generates a hypothesis, i.e. a route consisting of multiple observations. The hypothesis is evaluated by the evaluator, resulting in a likelihood corresponding to the hypothesis. Depending on the likelihood, the generator expands or adjusts the route. Using this approach, more evidence can be collected given the generated hypothesis. In this way, the design responses to the intelligence critiques mentioned in Section 2.3. Both the hypothesis generator and the hypothesis evaluator are further explained in the upcoming two sections of this chapter.

As already discussed, the analysis techniques used in both the articles of Barros et al. and Smit et al. were inspired by ACH. This is because in their research, hypotheses need to be evaluated simultaneously. Using our algorithm hypotheses are generated and evaluated one by one. Therefore, there is no reason to base the techniques used in the generator and evaluator on ACH. However, since ACH is such an important and widely used technique in intelligence analysis, we still want to incorporate this in our tracking algorithm. We therefore decided to output multiple routes with a high likelihood instead of only the route evaluated as most likely. In this way the most likely routes can further be analysed by other intelligence techniques, such as ACH. Moreover, it is impossible to exactly track down the route an entity has travelled based on its sightings. Providing an analyst with multiple highly likely possibilities is hence much more informative. Figure 5 shows the output principle of the algorithm.

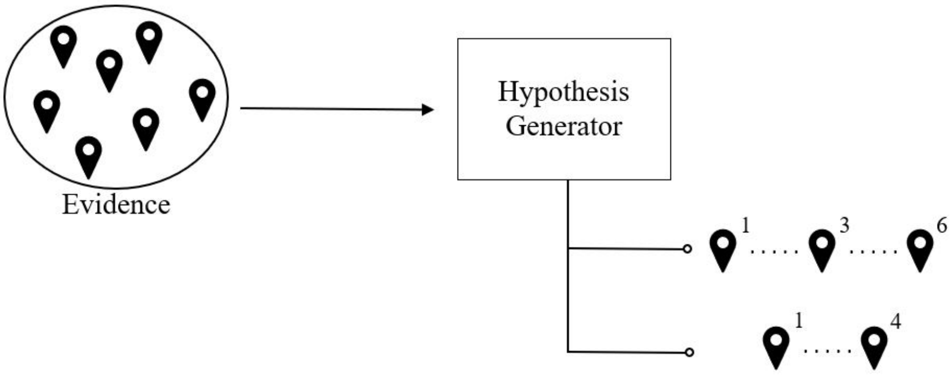


Figure 5: *Algorithm output consisting of multiple hypotheses.*

4.2 Hypothesis Evaluator

This section discusses the hypothesis evaluation component of the algorithm. The aim of this component is to provide a generated hypothesis with its corresponding likelihood value. The technique used to accomplish this is described followed by a demonstration of the evaluation technique using an example.

4.2.1 Bayesian Inference

Inspired by the papers of Barros et al. (2014) and Smit et al. (2016) a BBN might be a suitable technique for the hypothesis evaluator of the algorithm. A BBN consists of a set of nodes representing variables and a set of arcs representing conditional dependencies between the nodes. The absence of an arc indicates independence. A BBN is commonly presented as an acyclic directed graph. The extent to which the variables influence each other is given by conditional probability tables. In these tables the probability of occurrence of a variable given the state of another variable can be found. BBNs are used to calculate new probabilities when new information or evidence becomes available. The new information is applied to the network by setting a variable to a state that is consistent with the new information. The probabilities of all variables connected to the variable representing the new information are updated by Bayesian inference.

In order to use a BBN, first the variables of the network need to be defined. In case of the hypothesis evaluator, these variables are not straightforward. Since a route is defined as a chain of sightings, there is no other information about the route than the sightings it is formed by. Creating a graph consisting of sightings is impossible since there are no prior relations between them. Using a BBN as technique for the hypothesis evaluator is therefore inappropriate. However, the Bayesian inference used in BBNs is still useful.

Instead of applying inference to a network, one can iteratively apply inference.

Bayesian inference is a method to update the probability of a hypothesis as additional data or evidence is collected. The basis for Bayesian inference is derived from Bayes' Theorem. Bayes' theorem describes the probability of an event based on its association with another event. If we define H as some hypothesis and E as some newly collected evidence, Bayes' theorem can be stated as follows.

$$\mathbb{P}(H|E) = \frac{\mathbb{P}(E|H)\mathbb{P}(H)}{\mathbb{P}(E)} \tag{1}$$

Explained in words, $\mathbb{P}(H|E)$ in equation (1) asks the question: "Given some newly collected evidence, what is the probability for the hypothesis to be true?" In terms of this research an exact same question can be posed: "Given a collection of observations, how likely is the entity to really passed the positions belonging to the observations?" In order to apply Bayesian inference, two terms have to be defined within the context of this research.

Evidence
Let $E_{\{e_1,\dots,e_n\}}$ be an ordered collection of n pieces of evidence of the moving entity of interest. The ordering is in time, which means that e_1 is the first observed piece of evidence in the collection and e_n the last. For example, $E_{\{e_1,e_3,e_7\}}$ is a collection of three pieces of evidence with time ordering $e_1 \leq e_3 \leq e_7$. e_i can be interpreted as a single observation of the entity of interest by a certain source and is defined by spatio-temporal data.

Hypothesis
Let $H_{\{e_1,\dots,e_n\}}$ be the hypothesis that the entity of interest passed the positions belonging to $\{e_1, \dots e_n\}$ with time ordering $e_1 \leq \dots \leq e_n$.

Using these definitions Bayes' theorem can be put into context of this research:

$$\mathbb{P}(H_{\{e_1,\dots,e_n\}}|E_{\{e_1,\dots,e_n\}}) = \frac{\mathbb{P}(E_{\{e_1,\dots,e_n\}}|H_{\{e_1,\dots,e_n\}})\mathbb{P}(H_{\{e_1,\dots,e_n\}})}{\mathbb{P}(E_{\{e_1,\dots,e_n\}})}. \tag{2}$$

Since a hypothesis can either be true or false, the denominator in equation (2) can be rewritten to:

$$\mathbb{P}(E_{\{e_1,\dots,e_n\}}|H_{\{e_1,\dots,e_n\}})\mathbb{P}(H_{\{e_1,\dots,e_n\}}) + \mathbb{P}(E_{\{e_1,\dots,e_n\}}|\neg H_{\{e_1,\dots,e_n\}})\mathbb{P}(\neg H_{\{e_1,\dots,e_n\}}). \tag{3}$$

On the right-hand side of equation (2) three probabilities are stated.
$\mathbb{P}(E_{\{e_1,\dots,e_n\}}|H_{\{e_1,\dots,e_n\}})$ represents the likelihood that a collection of evidence will be observed given the hypothesis. This equals the true positives regarding the sources of the observations and can be obtained using historical data or expert knowledge.
$\mathbb{P}(H_{\{e_1,\dots,e_n\}})$ is the likelihood of a hypothesis to be true, without considering the reliability of evidence. How these probabilities can be computed will be discussed later.
$\mathbb{P}(E_{\{e_1,\dots,e_n\}})$ is the likelihood of a collection of evidence without any context. This value can be written in terms of $\mathbb{P}(E_{\{e_1,\dots,e_n\}}|H_{\{e_1,\dots,e_n\}})$ and $\mathbb{P}(H_{\{e_1,\dots,e_n\}})$ which are already discussed.

As discussed in Section 3.1, Zlotnick (1972) suggested to use Bayes' theorem in odds ratio form. This is useful if one needs to estimate the comparative merits of two competing hypotheses. This is not the case for the hypothesis evaluator. The evaluator needs to obtain $\mathbb{P}(H_{\{e_1,\ldots,e_n\}}|E_{\{e_1,\ldots,e_n\}})$ instead of $\frac{\mathbb{P}(H_{\{e_1,\ldots,e_n\}}|E_{\{e_1,\ldots,e_n\}})}{\mathbb{P}(\neg H_{\{e_1,\ldots,e_n\}}|E_{\{e_1,\ldots,e_n\}})}$. Therefore, the evaluator does not use Bayes' theorem in odds ratio. However, equation (2) can be rewritten to a more useful form regarding the evaluator.

Equation (4) states Bayes' theorem using (3). For readability we denote $E_{\{e_1,\ldots,e_n\}}$ by E and $H_{\{e_1,\ldots,e_n\}}$ by H.

$$\mathbb{P}(H|E) = \frac{\mathbb{P}(E|H)\mathbb{P}(H)}{\mathbb{P}(E|H)\mathbb{P}(H) + \mathbb{P}(E|\neg H)\mathbb{P}(\neg H)} \tag{4}$$

Multiplying both the numerator and the denominator in Equation (4) by $\frac{1}{\mathbb{P}(E|\neg H)\mathbb{P}(\neg H)}$ and rewriting the fraction, we end up with Equation (5).

$$\mathbb{P}(H|E) = \frac{\frac{\mathbb{P}(E|H)}{\mathbb{P}(E|\neg H)}\frac{\mathbb{P}(H)}{\mathbb{P}(\neg H)}}{1 + \frac{\mathbb{P}(E|H)}{\mathbb{P}(E|\neg H)}\frac{\mathbb{P}(H)}{\mathbb{P}(\neg H)}} \tag{5}$$

All probabilities in equation (5) can be computed. As stated before, $\mathbb{P}(E|H)$ equals the true positives regarding the sources of the observations. This means that the fraction $\frac{\mathbb{P}(E|H)}{\mathbb{P}(E|\neg H)}$ equals the true positive, false positive ratio and can be computed by historical data or expert knowledge. The remainder term in equation (5) is $\frac{\mathbb{P}(H)}{\mathbb{P}(\neg H)}$, which can be written in terms of $\mathbb{P}(H)$ by $\frac{\mathbb{P}(H)}{1-\mathbb{P}(H)}$. By computing $\mathbb{P}(H)$ the reliability of evidence cannot be taken into consideration. Therefore, all evidence belonging to the hypothesis are assumed to be accurate. This means that the collection of observations forms a route of which the likelihood can be computed. The likelihood of a route being travelled by the entity of interest depends on its characteristics. For example a route with unpaved roads might be less likely to be travelled by a certain entity than paved roads. Denote the characteristics of a route formed by $\{e_1 \ldots, e_n\}$ by $c^1_{\{e_1\ldots,e_n\}}, \ldots, c^m_{\{e_1\ldots,e_n\}}$, where $1, \ldots, m$ represent the index of the characteristics rather than an exponent. Now, the likelihood of a hypothesis to be true can be stated as:

$$\mathbb{P}(H_{\{e_1,\ldots,e_n\}}) = \mathbb{P}(c^1_{\{e_1\ldots,e_n\}} \times \cdots \times c^m_{\{e_1\ldots,e_n\}}) \tag{6}$$

Computing this probability would be straightforward when all m characteristics are assumed to be independent. However, this assumption might be invalid, since different characteristics concerning the same route are likely to highly correlate with each other. Therefore, an approach has to be determined which is able to model dependent variables. This is further discussed in Section 4.2.2. How Bayesian inference exactly can be used as hypothesis evaluation is demonstrated in Section 4.2.3.

4.2.2 Modelling Dependence

As discussed in the previous section, the different route characteristics implied in $\mathbb{P}(H_{\{e_1,\ldots,e_n\}})$ might be dependent. When this is the case, computing the likelihood of a hypothesis is not as straightforward as the multiplication of the different characteristic probabilities. Instead, the dependence between the characteristics needs to be modelled. In this section two techniques are discussed to model dependence; *Copula Functions* and *Multivariate Kernel Density Estimation.* Both techniques are theoretically described, afterwhich they are demonstrated and compared using a practical example.

4.2.2.1 Copula Functions

Dependence between random variables can be modelled by Copula functions. Copula functions are widely used to model dependence. Especially in the financial sector they have proven to be a useful methodology. Copulae were first introduced by Sklar in 1959. He stated that any multivariate cumulative distribution function can be separated into two parts: the univariate marginal distribution functions (hereafter referred to as 'margins') and the copula which describes the dependence structure between the variables. This statement is now known as Sklar's Theorem, which is formally stated below.

Sklar's Theorem

Let F be an n-dimensional distribution function with marginal distributions $F_1, \ldots, F_n$, then there exists an n-copula C such that

$$F(x_1, \ldots, x_n) = C(F_1(x_1), \ldots, F_n(x_n)). \tag{7}$$

C is uniquely defined if all $F_1(x_1), \ldots, F_n(x_n)$ are continuous.

The proof of Sklar's theorem is sketched in [18].

In order to understand how a copula function describes the dependency between random variables, Probability Integral Transformation (PIT) needs to be introduced. PIT results in the fact that any continuous random variable can be transformed to be standard uniformly distributed. The theorem is formally stated below.

Probability Integral Transformation

Let X be a random variable with continuous cumulative distribution function $F_X(x)$ and define the random variable Y as $Y = F_X(X)$. Then Y is standard uniformly distributed.

The proof for the PIT is given in equation (8).

$$\begin{aligned} F_Y(y) &= \mathbb{P}(Y \leq y) \\ &= \mathbb{P}(F_X(X) \leq y) \\ &= \mathbb{P}(X \leq F_X^{-1}(y)) \\ &= F_X(F_X^{-1}(y)) \\ &= y \end{aligned} \tag{8}$$

This equals the cumulative distribution function of a standard uniform random variable.

Using the PIT one can define that $X = F^{-1}(U)$, where U is standard uniformly distributed. Similarly, we can obtain the copula function:

$$\begin{aligned} F(x_1, \ldots, x_n) &= \mathbb{P}(X_1 \leq x_1, \ldots, X_n \leq x_n) \\ &= \mathbb{P}(F_1^{-1}(U_1) \leq x_1, \ldots, F_m^{-1}(U_n) \leq x_n) \\ &= \mathbb{P}(U_1 \leq F_1(x_1), \ldots, U_n \leq F_n(x_n)) \\ &= C(F_1(x_1), \ldots, F_n(x_n)) \end{aligned} \tag{9}$$

Note that derivation (9) is only valid in the case that all $F_1(x_1), \ldots, F_n(x_n)$ are continuous, because in this case each $F_i(x_i)$, $i = 1, \ldots, n$ has an inverse function. Besides, derivation (9) implies that an n-dimensional copula can actually be defined as a function mapping $C : [0, 1]^n \rightarrow [0, 1]$ which is a joint cumulative distribution function with uniform marginals. In this way the problem of modelling dependence between random variables is simplified to modelling dependence between uniform variables.

Copula Families

Different classes of parametric copula functions are available, also referred to as copula families. The two most popular families are Elliptical and Archimedean. The Elliptical family represents copula functions which can be derived from multivariate elliptical distributions having a symmetrical character. The two most important copulae in this family are the Gaussian and Student-t copula derived from respectively the multivariate Gaussian and the multivariate Student-t distribution. The Archimedean family consist of copula functions admitting an explicit, closed form formula. These copulae can be stated directly in contrary to the Elliptical copulae. Three examples of popular Archimedean copulae are the Clayton, Frank and Gumbel copula.

Constructing copulae can be extended to any dimension. However, as the dimension of a copula increases the construction becomes more and more difficult. This also holds for the copula families: many examples of bivariate copula families are available, whereas the multivariate generalizations are limited. The most popular bivariate copula functions in the Elliptical and Archimedean families can be found in *Appendix B.1.*

Pair-copula construction

In order to still be able to construct multivariate distributions, Joe [20] introduced in 1996 the probabilistic version of the pair-copula approach. This approach decomposes a joint multivariate cumulative distribution function into simple building blocks called pair-copulae. In 2009 Aas et al. [21] extended this approach to a multivariate density function to perform inference. Therefore, they used Sklar's theorem defined in terms of densities:

$$f(x_1, \ldots, x_n) = c(F_1(x_1), \ldots, F_n(x_n)) \cdot \prod_{i=1}^{n} f_i(x_i) \tag{10}$$

where, $c(F_1(x_1), \ldots, F_n(x_n))$ represents the density of the n-dimensional copula function $C(F_1(x_1), \ldots, F_n(x_n))$. This equation is obtained by differentiating $F(\cdot)$ and $C(\cdot)$ in equation (7) and applying the chain rule.

Another way of writing a multivariate density is in terms of conditional densities:

$$f(x_1,\dots,x_n) = f(x_1)\cdot\prod_{i=2}^{n} f(x_i|x_1,\dots x_{i-1}). \tag{11}$$

Combining equations (10) and (11) results in the pair-copula decomposition. To illustrate, the decomposition is shown for a trivariate density.

Example pair-copula construction for $n = 3$
Equation (12) states the multivariate density in products of the conditional densities.

$$f(x_1,x_2,x_3) = f_1(x_1)\cdot f_{2|1}(x_2|x_1)\cdot f_{3|1,2}(x_3|x_1,x_2) \tag{12}$$

The first term on the right hand side of the equation is the density of x_1. The second term is the density of x_2 conditioned on x_1 and can be rewritten using the definition of conditional probability for x_1 and x_2.

$$f_{2|1}(x_2|x_1) = \frac{f_{1,2}(x_1,x_2)}{f_1(x_1)} \tag{13}$$

The denominator in equation (13) can be written in terms of a bivariate copula and its marginal densities using Sklar's theorem and equation (10):

$$f_{1,2}(x_1,x_2) = c_{1,2}(F_1(x_2),F_2(x_2))\cdot f_1(x_1)\cdot f_2(x_2). \tag{14}$$

Now, equation (13) can be rewritten to:

$$\begin{aligned} f_{2|1}(x_2|x_1) &= \frac{c_{1,2}(F_1(x_1),F_2(x_2))\cdot f_1(x_1)\cdot f_2(x_2)}{f_1(x_1)} \\ &= c_{1,2}(F_1(x_1),F_2(x_2))\cdot f_2(x_2). \end{aligned} \tag{15}$$

Similarly, the third term in equation (12), $f_{3|1,2}(x_3|x_1,x_2)$, can be rewritten to:

$$\begin{aligned} f_{3|1,2}(x_3|x_1,x_2) &= \frac{f(x_1,x_2,x_3)}{f_{1,2}(x_1,x_2)} \\ &= c_{1,3|2}(F_{1|2}(x_1|x_2),F_{3|2}(x_3|x_2))\cdot f_{3|2}(x_3|x_2) \\ &= c_{1,3|2}(F_{1|2}(x_1|x_2),F_{3|2}(x_3|x_2))\cdot c_{2,3}(F_2(x_2),F_3(x_3))\cdot f_3(x_3). \end{aligned} \tag{16}$$

Now, the two conditional densities in equation (12) are written in terms of bivariate copulae and marginal densities. This results in the following pair-copula decomposition:

$$\begin{aligned} f(x_1,x_2,x_3) =& f_1(x_1)\cdot f_{2|1}(x_2|x_1)\cdot f_{3|1,2}(x_3|x_1,x_2) \\ =& f_1(x_1)\cdot f_2(x_2)\cdot f_3(x_3)\cdot c_{1,2}(F_1(x_1),F_2(x_2)) \\ &\cdot c_{1,3}(F_1(x_1),F_3(x_3))\cdot c_{2,3|1}(F_{2|1}(x_2|x_1),F_{3|1}(x_3|x_1)). \end{aligned} \tag{17}$$

Equation (17), however, is not the only pair-copula construction for three dimensions. In total there are six permutations of x_1, x_2, x_3, which eventually results in three different joint density distributions. This is due to the conditioning variables. For example, in equation (16) variable x_2 is the conditioning variable. Alternatively, one can condition on variable x_1, resulting in a different pair-copula decomposition. For three dimensions the different decompositions are relatively easy to obtain. However, the number of decompositions grows rapidly with the dimension of the joint distribution function. A dimension of seven already results in 2,580,480 different decompositions. In order to organize all possible pair-copula constructions, Bedford and Cooke [22] introduced in 2002 a graphical method called regular vines.

Regular vines
Two special cases of regular vines are called C-vines and D-vines. C-vines are suitable to model joint densities where an ordering in importance is present among the variables. D-vines are suitable when a temporal ordering is present among the variables. Figures 6 and 7 give a representation of the C-vine and D-vine decomposition respectively. The nodes in each tree T_i represent the marginals f_i and the edges represent the pair-copula functions c_i.

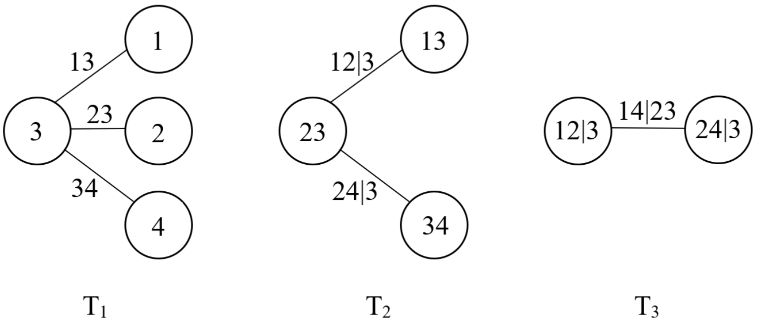


Figure 6: *Decomposition of a 4-dimensional joint density using C-vine.*

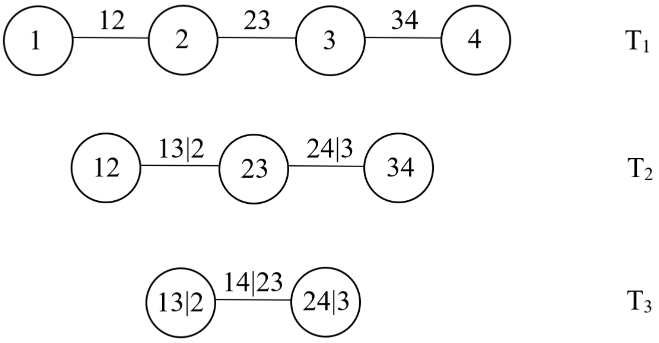


Figure 7: *Decomposition of a 4-dimensional joint density using D-vine.*

Using C-vine and D-vine one can easily obtain the number of pair-copulas involved in an n-dimensional multivariate density function. The edges in the trees represent the pair-copulae. This means that summing the edges over all trees results in the total number of pair-copulae. For both the C-vine and D-vine the first tree has $n-1$ edges, where n equals the dimension of the density. The second tree has $n-2$ edges and the third tree $n-3$. Therefore, the number of pair-copulas involved in the multivariate density equals:

$$(n-1)+(n-2)+\cdots+2+1=\frac{(n-1)n}{2}. \tag{18}$$

By equation (18) we know that the higher the dimension of a multivariate density, the more pair-copulae are involved in the decomposition. Besides, the number of conditional pair-copulae involved increases. The decompositions may be simplified by assuming conditional independence. In the example decomposition for the trivariate density, the conditional pair-copula equals $c_{2,3|1}(F_{2|1}(x_2|x_1), F_{3|1}(x_3|x_1))$. If we assume that the variables X_2 and X_3 are independent given X_1, the conditional copula will be equal to one: $c_{2,3|1}(F_{2|1}(x_2|x_1), F_{3|1}(x_3|x_1)) = 1$. Therefore, equation (17) simplifies to:

$$f(x_1,x_2,x_3)=f_1(x_1)\cdot f_2(x_2)\cdot f_3(x_3)\cdot c_{1,2}(F_1(x_1),F_2(x_2))\cdot c_{1,3}(F_1(x_1),F_3(x_3)). \tag{19}$$

Thus, assuming conditional independence reduces the number of terms involved in the density and simplifies the decomposition. This is also known as the simplified pair-copula decomposition. Whether assuming conditional independence is appropriate depends on the concerned joint distributions and its variables. However, Haff et al. (2010) showed that simplified pair-copula decomposition can always be used as an approximation. Even when the simplifying assumption is far from being fulfilled, the approximation may be in fact a good one [23].

Conclusion

Copula functions are able to model the dependence between random variables. Therefore, $\mathbb{P}(H_{\{e_1,\dots,e_n\}}) = \mathbb{P}(c^1_{\{e_1\dots,e_n\}}\times\cdots\times c^m_{\{e_1\dots,e_n\}})$ can potentially be modelled by copulae. In order to make use of this technique, the characteristics have to be continuous random variables. Besides, a sufficient amount of historical data about the characteristics has to be available. If this is the case, the joint density of the characteristics can be obtained enabling the computation of $\mathbb{P}(H_{\{e_1,\dots,e_n\}})$.

4.2.2.2 Multivariate Kernel Density Estimation

Another way of modelling the dependence between random variables is by using Kernel Density Estimation. Kernel density estimation (kde) is a form of estimating the underlying probability density function of a random variable. It works by creating a kernel function at every data point in the sample of the random variable, where a kernel is a probability density function. Adding all these kernel functions and dividing by the sample size results in the estimated density.

Formally, the procedure of kde for univariate densities can be defined as follows:

Univariate kernel density estimation

Let $X_1,\dots,X_n$ be a random sample drawn from a density $f(x)$. The density function of X can be estimated by:

$$\hat{f}(x,B)=\frac{1}{nB}\sum_{i=1}^{n}K\left(\frac{x-X_i}{B}\right) \tag{20}$$

with n equal to the sample size, B represents the bandwidth and K is the univariate kernel function.

The kernel K typically is a symmetric probability density function. It smooths out the contribution of each observed data point over a local neighborhood of that data point. Many examples of kernels are available. The most popular kernels can be found in *Appendix B.2.* Even though there is a wide range of kernels, it is not the kernel function that is crucial for the estimator. Instead, the bandwidth B has significant impact on the estimate. The bandwidth controls the size of the neighborhood around the observed data points. Determining the most appropriate value for B is known to be the hardest task in kde, since the bandwidth is a very sensitive parameter. An often used default value for the bandwidth is the following:

$$B_{default} = 1.06 \cdot \sigma_X \cdot n^{-\frac{1}{5}} \tag{21}$$

with σ_X being the standard deviation of the sample and n equals the sample size.

The procedure of kde and the sensitivity of B is illustrated in Figure 8 for a random variable X.

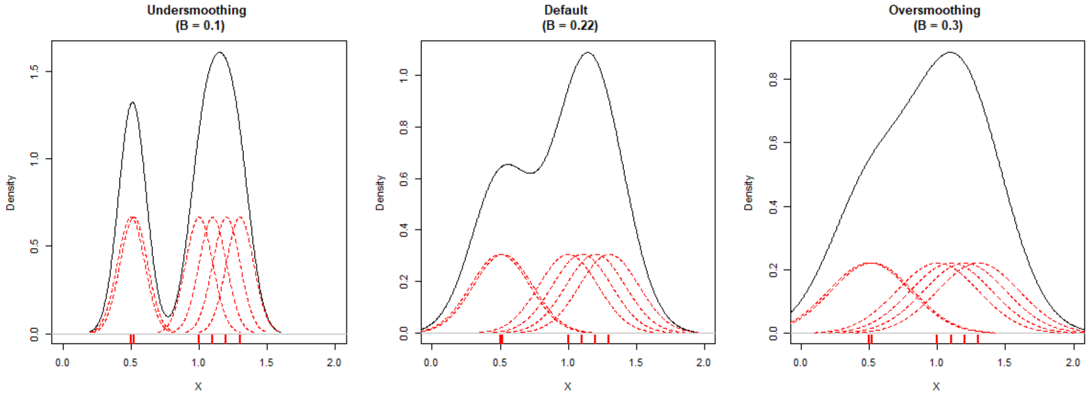


Figure 8: *Influence of bandwidth B on $\hat{f}_X(x, B)$.*

The dotted red lines are the kernel functions. In this example the kernels are chosen to be Gaussian. The vertical red lines at the bottom of the graph represent the data points at which the kernels are centered. The solid black line represents the estimated density. In the left plot the value for B is set too small leading to a spiky estimate. In contrast, the right plot is created with kernels having a bandwidth value which is set too large. This results in an oversmoothed estimate. In the middle plot the value for B is set to the default value computed by equation (21).

This univariate method of kde can be extended to obtain multivariate density functions.

Multivariate kernel density estimation
Let $\mathbf{X_1}, \ldots, \mathbf{X_n}$ be a d-variate random sample drawn from a density $f(\mathbf{x})$. The density function of $\mathbf{X}$ can be estimated by:

$$\hat{f}(\mathbf{x}, \mathbf{B}) = \frac{1}{n\mathbf{B}} \sum_{i=1}^{n} K\left(\frac{\mathbf{x} - \mathbf{X_i}}{\mathbf{B}}\right) \tag{22}$$

with $K(\cdot)$ the multivariate kernel function and $\mathbf{B}$ the $d \times d$ symmetric and positive-definite bandwidth matrix. $\mathbf{B}$ can be diagonal as well as unconstrained.

4.2.2.3 Practical Comparison

In the previous two sections, two different approaches to model dependency between random variables are discussed. In this section both the approaches are demonstrated on sample data and their performances are compared. Two random variables are defined, for which the bivariate joint density function needs to obtained. The dimension equals two, since for higher dimensions visualizing densities becomes inconvenient.

Let $\epsilon_0 = (\epsilon_{0;1}, \ldots, \epsilon_{0;1500})$, $\epsilon_1 = (\epsilon_{1;1}, \ldots, \epsilon_{1;1500})$ and $\epsilon_2 = (\epsilon_{2;1}, \ldots, \epsilon_{2;1500})$ be random samples, where $\epsilon_0, \epsilon_1, \epsilon_2 \sim Unif(0,1)$. Define the following two random variables: $X = \epsilon_0 + \epsilon_1$ and $Y = \epsilon_0 + \epsilon_2$. Assume we want to obtain the joint density $f_{X,Y}(x,y)$. Since both the random variables X and Y depend on the same random variable ϵ_0, we know that X and Y are dependent. Therefore, both a copula function and multivariate kernel density estimation are appropriate to model this dependence.

X and Y are both the sum of two independent random variables. By the central limit theorem we know that the sum of n independent random variables are approximately normal when n is large. For X and Y n equals two, which is far from large. Nevertheless, we expect X and Y to have the shape of a normal distribution. From figure 9 can be obtained that this expectation is valid. Therefore, we expect for both the approaches that the estimated density $f_{X,Y}(x,y)$ will look similar to a bivariate normal distribution.

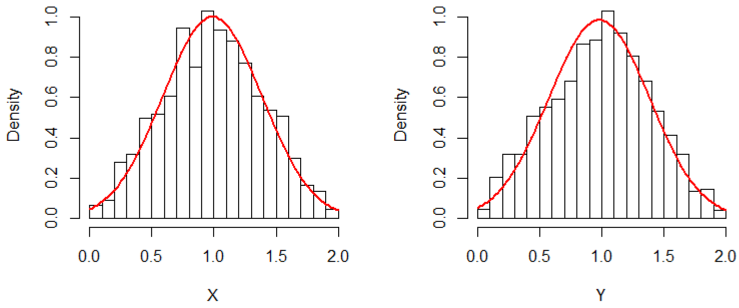


Figure 9: *Histograms for random variables X and Y respectively. The red lines represent the fitted normal distributions with $\mu_x = 0.988$, $\sigma_x = 0.399$ and $\mu_y = 0.981$, $\sigma_y = 0.405$.*

Copula function

Estimating the bivariate density $f_{X,Y}(x,y)$ using copula functions can be rewritten to:

$$\begin{aligned} f_{X,Y}(x,y) &= f_X(x) \cdot f_{Y|X}(y|x) \\ &= f_X(x) \cdot f_Y(y) \cdot c_{X,Y}(F_X(x), F_Y(y)) \end{aligned} \tag{23}$$

using the same derivation as in Section 4.2.1. This means that the marginal distributions of both X and Y need to be estimated as well as the bivariate copula function of X and Y. The marginal distributions are already estimated to be normal which can be obtained from Figure 9. Copula functions can be estimated in R using the packages *copula* [24] and *VineCopula* [25]. Figure 10 shows the estimated $c_{X,Y}(F_X(x), F_Y(y))$, which appears to have a shape similar to the bivariate normal as expected.

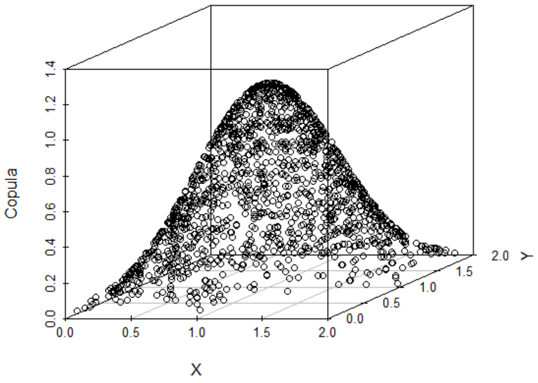


Figure 10: *Estimated bivariate copula for random variables X and Y.*

Note that Figure 10 shows the density of the copula function instead of the joint density $f_{X,Y}(x, y)$. However, since both the densities of X and Y have a shape similar to the normal distribution, the joint density will look similar to the copula density.

Kernel density estimation

Estimating the bivariate density $f_{X,Y}(x, y)$ using kernel density estimation can be rewritten to equation (22). This means that the bandwidth matrix and the kernel need to be specified. The bandwidth matrix is chosen to be diagonal with B_X and B_Y both equal to the default value (see equation (21)). The kernels are chosen to be Gaussian. Multivariate kernel density estimation can be implemented in R using the package *ks* [26]. Figure 11 shows the estimated kernel density $\hat{f}_{X,Y}(x, y)$, which appears to have a shape similar to the bivariate normal as expected.

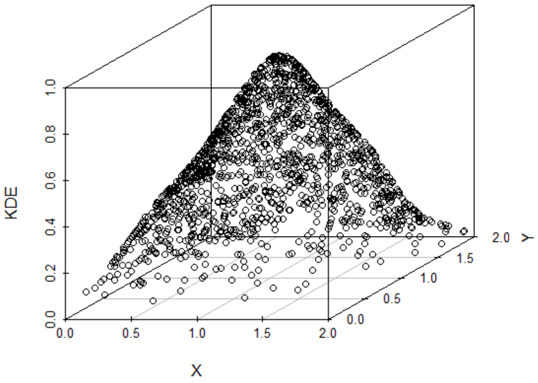


Figure 11: *Estimated kernel density for random variables X and Y.*

Conclusion

Copula functions and multivariate kernel density estimation are both able to model the dependence between random variables. Therefore, $\mathbb{P}(H_{\{e_1,\ldots,e_n\}}) = \mathbb{P}(c^1_{\{e_1\ldots,e_n\}} \times \cdots \times c^m_{\{e_1\ldots,e_n\}})$ can potentially be modelled by these techniques. In order to make use of copulae, the characteristics have to be continuous random variables. Besides, a sufficient amount of historical data about the characteristics has to be available. If these are the case, the joint density of the characteristics can be obtained enabling the computation of $\mathbb{P}(H_{\{e_1,\dots,e_n\}})$. There has to be noted that both techniques have difficulties. For multivariate kde it is the choice of the bandwidth matrix. In this example the default bandwidth matrix was used, whereas a different choice would lead to different results. For copula functions this is the pair-copula decomposition and the choice of the regular vine. These were not demonstrated in this example, since a bivariate joint density was estimated. At the same time, these difficulties are also the strengths of copula functions. A good choice of the regular vine results in a joint density which takes ordering of importance into account among the variables. Multivariate kde is not able to model such an ordering. Therefore, copula functions will be used in this research to model dependence instead of multivariate kde.

4.2.3 Demonstrative Example

Section 4.2.1 theoretically described how hypothesis evaluation can be conducted by Bayesian inference using the following formula:

$$\mathbb{P}(H|E) = \frac{\dfrac{\mathbb{P}(E|H)}{\mathbb{P}(E|\neg H)} \cdot \dfrac{\mathbb{P}(H)}{\mathbb{P}(\neg H)}}{1 + \dfrac{\mathbb{P}(E|H)}{\mathbb{P}(E|\neg H)} \cdot \dfrac{\mathbb{P}(H)}{\mathbb{P}(\neg H)}}$$

where H equals $H_{\{e_1,\dots,e_n\}}$ and E equals $E_{\{e_1,\dots,e_n\}}$.

The two components in this formula are $\dfrac{\mathbb{P}(E|H)}{\mathbb{P}(E|\neg H)}$ and $\dfrac{\mathbb{P}(H)}{\mathbb{P}(\neg H)}$. The first component represents the reliability of the source which provided the evidence. When a sufficient amount of historical data of the source is available, this probability can be computed. Otherwise, this probability can be estimated by an expert having prior knowledge about the reliability of the source. The second term represents the likelihood of a route. This likelihood can be estimated by copula functions.

Each hypothesis generated by the hypothesis generator can be evaluated by Bayesian inference. As an example, assume that the generated hypothesis consists of sightings $\{e_1, e_2, e_4\}$. This situation was also sketched in Figure 4 in Section 4.1. The hypothesis evaluator therefore needs to evaluate hypothesis $H_{\{e_1,e_2,e_4\}}$. This means that the following formula needs to be evaluated:

$$\mathbb{P}(H_{\{e_1,e_2,e_4\}}|E_{\{e_1,e_2,e_4\}}) = \frac{\dfrac{\mathbb{P}(E_{\{e_1,e_2,e_4\}}|H_{\{e_1,e_2,e_4\}})}{\mathbb{P}(E_{\{e_1,e_2,e_4\}}|\neg H_{\{e_1,e_2,e_4\}})} \cdot \dfrac{\mathbb{P}(H_{\{e_1,e_2,e_4\}})}{\mathbb{P}(\neg H_{\{e_1,e_2,e_4\}})}}{1 + \dfrac{\mathbb{P}(E_{\{e_1,e_2,e_4\}}|H_{\{e_1,e_2,e_4\}})}{\mathbb{P}(E_{\{e_1,e_2,e_4\}}|\neg H_{\{e_1,e_2,e_4\}})} \cdot \dfrac{\mathbb{P}(H_{\{e_1,e_2,e_4\}})}{\mathbb{P}(\neg H_{\{e_1,e_2,e_4\}})}} \tag{24}$$

The two components in this formula, as discussed above, can be evaluated separately.

- The term $\dfrac{\mathbb{P}(E_{\{e_1,e_2,e_4\}}|H_{\{e_1,e_2,e_4\}})}{\mathbb{P}(E_{\{e_1,e_2,e_4\}}|\neg H_{\{e_1,e_2,e_4\}})}$ represents the total reliability of the sources which provided $\{e_1, e_2, e_4\}$. Therefore, the reliability of each source needs to be evaluated, resulting in three probabilities. These can either be obtained by historical data of the sources or by expert knowledge. The multiplication of the three probabilities results in the total reliability. Note that when all three sightings are provided by the same source, all three probabilities are equal. On the other hand, when all three sightings are provided by three different sources, three different probabilities have to be obtained.

- The term $\frac{\mathbb{P}(H_{\{e_1,e_2,e_4\}})}{\mathbb{P}(\neg H_{\{e_1,e_2,e_4\}})}$ is dependent on characteristics of the route formed by $\{e_1, e_2, e_4\}$. Therefore, historical data of the different characteristics needs to be available. Using the historical data, the margins of the characteristics can be estimated. When the characteristics are dependent, copula functions need to be estimated. Subsequently, multiplying the marginals and the estimated copula functions will result in the joint probability function. Using the joint probability function, the likelihood of route $\{e_1, e_2, e_4\}$ based on its characteristics can be obtained. Note, that when the characteristics are independent, the joint probability function can simply be obtained by multiplying the marginals of the characteristics.

When the two terms are computed, they can be filled into equation (24). This results in the likelihood of an entity travelled along the positions belonging to $\{e_1, e_2, e_4\}$. Depending on the likelihood the generator either expands or adjusts the hypothesis. For example, as illustrated in Figure 4, assume that $\mathbb{P}(H_{\{e_1,e_2,e_4\}}|E_{\{e_1,e_2,e_4\}}) = 0.87$. Based on this likelihood the generator might expand the hypothesis by for example, e_5. Then, the new hypothesis $H_{\{e_1,e_2,e_4,e_5\}}$ needs to be evaluated. This can be done in the exact same way as described above. In this way, the hypothesis evaluator becomes an iterative process of Bayesian inference.

In addition, there is a possibility to expand the evaluator. Up to now, the term $\frac{\mathbb{P}(H)}{\mathbb{P}(\neg H)}$ in the evaluator is only described by the characteristics of the hypothesis. Computing this fraction can be expanded by additional modules. For example, the speed at which the entity travelled a route might indicate inconsistency. Which module suits the algorithm best depends on both the environment in which the algorithm is used and the moving entity itself. It might also be the case that an evaluator is not expanded with additional consistency check modules. Expanding the evaluator with additional modules is discussed in more detail in Section 5.2.

4.3 Hypothesis Generator

This section discusses the hypothesis generator component of the algorithm. The aim of this component is to construct entity movements from a collection of sightings. These sightings are defined by place and time. A hypothesis is formed by a chain of multiple sightings. Because of the time component of a sighting, this chain has to be ordered chronologically. Generating a chronological chain of sightings can be represented as a tree. Figure 12 shows an example of a tree for a collection consisting of six sightings. e_i corresponds to evidence i in the collection of sightings. Every edge, or combination of connected edges in the tree represents a possible hypothesis. This means that $\{e_1, e_2, e_3, e_4, e_5, e_6\}$ is a possible hypothesis as well as $\{e_3, e_4\}$ or $\{e_2, e_6\}$. The larger the collection of evidence, the higher the amount of possible hypotheses.

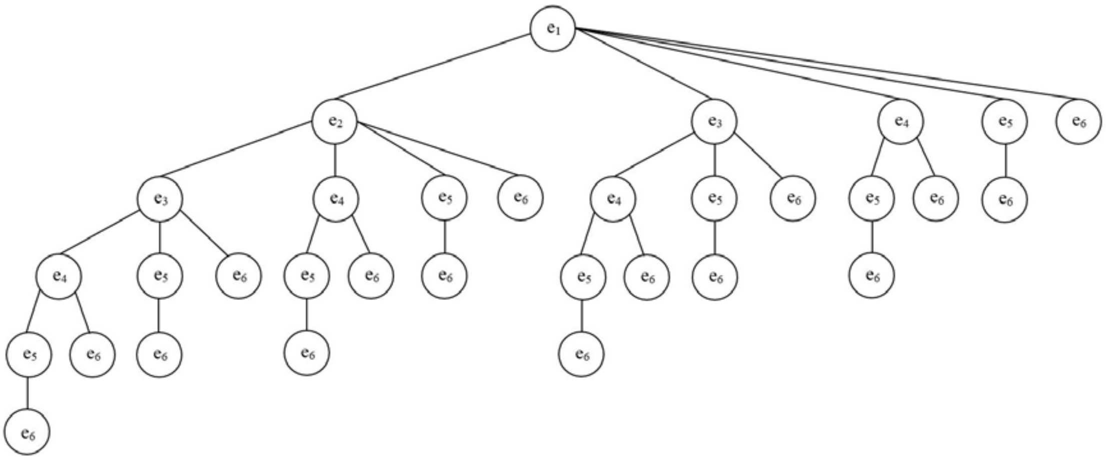


Figure 12: *Search tree for dimension of 6.*

In order to find the most likely route, all possible hypotheses need to be generated and evaluated. Therefore, the tree needs to be traversed. This can be done by a brute force manner using depth-first search. However, for large collections this approach leads to a high runtime. This is not preferable, since the algorithm should function in the intelligence cycle. From Chapter 2 it is known that the timely dissemination of intelligence is of high importance. Therefore, a technique reducing the runtime of the hypothesis generator should be investigated.

Backtracking is a refinement of the brute force approach. This technique systematically traverses the tree in depth-first order. The possible hypotheses can be represented by vectors. The vectors correspond to the chains of evidence. It starts with a vector consisting of the first sighting. At each stage the vector is extended with a new piece of evidence. Each partial vector is evaluated by the hypothesis evaluator. When the likelihood of a partial vector is smaller than a certain threshold, the algorithm backtracks by removing the last appended sighting. It then proceeds by extending the vector with an alternative sighting. When, for example, the likelihood corresponding to hypothesis $\{e_1, e_2\}$ is sufficiently low, the sub-tree rooted at e_2 can be pruned. Instead, the next generated hypothesis will be equal to $\{e_1, e_3\}$. In this way, not all possible hypotheses are necessarily generated; only the promising ones.

The value of the threshold determines how deep the tree is traversed and thus how many hypotheses are generated. When the value is chosen too low, the tree might not be traversed deep enough. On the other hand, when a threshold value is chosen too high, no sub-trees will be pruned. This may lead to a high runtime. Hence, a trade-off between the runtime and search depth has to be made in order to generate a sufficient amount of hypotheses.

5 Validation

In this chapter the designed algorithm is validated. In the previous chapter the two components of the algorithm are theoretically described. In order to investigate whether these components function in practice, the algorithm is validated in a totally controllable environment using simulated data. First, the simulator is discussed in more detail. Second, the route characteristics used to model the hypothesis evaluator are described. After that, the results of the algorithm are presented and evaluated. The aim of this chapter is to discover whether the designed algorithm functions in practice and not to obtain the best results. Therefore, the focus is more on evaluating the obtained results than improving them. The chapter ends with a concluding section.

5.1 Simulator

The simulator used for the validation of the algorithm, generates evidence in a fictive, hilly area. The terrain is a simplified representation of a real-world terrain; there are no roads, no vegetation, no facilities et cetera. This results in a totally controllable environment in which the algorithm can be validated. The simulation terrain is shown in Figure 13.

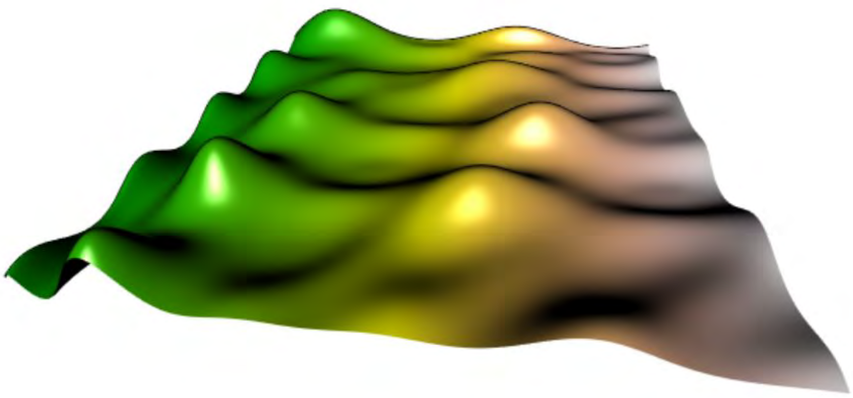


Figure 13: *Area of interest used in simulation.*

In the simulation area, nine fictive sources of information are present which might observe the entity of interest. An entity travels through the simulation area by taking routes with minimum effort. This means, that an entity travels between the hills rather than across the top of the hills. The simulator returns per source where (x- and y-coordinate) and when (time stamp) an observation took place. At most nine sightings are returned. These sightings are not guaranteed to be correct. Instead of observing the entity of interest, a source might observe another moving entity resulting in a false sighting.

The reliability of the sources in the simulation area can be summarised by a confusion matrix. The entries of the matrix should not be interpreted as counts. Instead, they correspond to conditional probabilities. The confusion matrix and its conditional probabilities are shown in Figure 14.

$$\begin{array}{ccc} & & \text{Source} \\ & & \begin{array}{cc} \text{No} & \text{Yes} \end{array} \\ \text{Actual} & \begin{array}{c} \text{No} \\ \text{Yes} \end{array} & \begin{bmatrix} \text{TN} & \text{FP} \\ \text{FN} & \text{TP} \end{bmatrix} \end{array} \qquad \begin{aligned} \text{TN} &= \mathbb{P}(\text{no source observation} \mid \text{entity not present}) \\ \text{FP} &= \mathbb{P}(\text{source observation} \mid \text{entity not present}) \\ \text{FN} &= \mathbb{P}(\text{no source observation} \mid \text{entity present}) \\ \text{TP} &= \mathbb{P}(\text{source observation} \mid \text{entity present}) \end{aligned}$$

Figure 14: *Confusion Matrix and its corresponding conditional probabilities.*

The conditional probabilities in the confusion matrix can be computed by the two parameters of the simulator, p_1 and p_2. The parameters represent probabilities. These probabilities as well as the rewritten confusion matrix are shown in Figure 15.

$$\begin{array}{ccc} & & \text{Source} \\ & & \begin{array}{cc} \text{No} & \text{Yes} \end{array} \\ \text{Actual} & \begin{array}{c} \text{No} \\ \text{Yes} \end{array} & \begin{bmatrix} 0 & p_1 \cdot p_2 \\ p_1 \cdot (1 - p_2) & 1 - p_1 \end{bmatrix} \end{array} \qquad \begin{aligned} p_1 &= \mathbb{P}(\text{incorrect source observation}) \\ p_2 &= \mathbb{P}(\text{entity not present}) \end{aligned}$$

Figure 15: *Rewritten Confusion Matrix using parameters p_1 and p_2.*

Data
The simulator returns a collection of sightings. Depending on the parameter values given to the simulator, it returns at most nine sightings, including whether a sighting is true or false. Table 1 shows an example of the simulation data. The first two columns represent the coordinate of the sightings. The third column represents the time at which the observation took place. The last column indicates whether a sighting is true or false; 1 equals a false positive sighting and 0 equals a true positive. The example data set consists of eight sightings of which two are false. The parameter values used are $p_1 = 0.3$ and $p_2 = 0.7$.

	X	**Y**	**Time**	**FP**
1	201.2849	207.8506	0.00	0
2	190.7019	216.8132	48.92	0
3	417.8686	299.3249	150.53	1
4	193.4109	198.3628	162.06	0
5	185.2104	209.7792	242.69	0
6	172.6069	201.2737	315.43	0
7	118.7394	251.5366	789.23	0
8	187.9189	187.7966	1536.16	1

Table 1: *Example simulation data of one route.*

The observations returned by the simulator can be evaluated by the algorithm designed for this research. The characteristics used to model the simulation data are explained in the upcoming section.

5.2 Hypothesis Evaluator

Recall the Bayesian inference formula obtained in Section 4.2.1:

$$\mathbb{P}(H|E) = \frac{\dfrac{\mathbb{P}(E|H)}{\mathbb{P}(E|\neg H)}\dfrac{\mathbb{P}(H)}{\mathbb{P}(\neg H)}}{1 + \dfrac{\mathbb{P}(E|H)}{\mathbb{P}(E|\neg H)}\dfrac{\mathbb{P}(H)}{\mathbb{P}(\neg H)}}$$

Using this formula the routes generated by the hypothesis generator can be evaluated.

Source reliability

The first term of the hypothesis evaluator, $\frac{\mathbb{P}(E|H)}{\mathbb{P}(E|\neg H)}$, is straightforward to compute. As discussed before, the simulator has two parameters. These parameters describe the reliability of all the sources used in the simulator. Using the confusion matrices presented in Figures 14 and 15, one can easily obtain:

$$\frac{\mathbb{P}(E|H)}{\mathbb{P}(E|\neg H)} = \frac{TP}{FP} = \frac{1 - p_1}{p_1 \cdot p_2}$$

This means that the first term of the hypothesis evaluator is totally described by the parameters of the simulator.

Route characteristics

The second term of the hypothesis evaluator, $\frac{\mathbb{P}(H)}{\mathbb{P}(\neg H)}$, is dependent of the characteristics of the simulation area. Since the surface used in the simulator is simplified, the terrain has no explicit characteristics except for the differences in elevation. The elevation can be expressed in terms of effort. The steeper the surface is, the more effort it takes to travel over that surface. The simulator defines four different effort variables, effort_1, ..., effort_4. Therefore, $\mathbb{P}(H)$ can be rewritten to:

$$\mathbb{P}(H) = \mathbb{P}(\mathit{effort}_1, \mathit{effort}_2, \mathit{effort}_3, \mathit{effort}_4) \tag{25}$$

How $\mathbb{P}(H)$ can be computed depends on the correlation between the four variables. The correlation provides information about the dependence between the variables. When all four variables appear to be independent, $\mathbb{P}(H)$ is straightforward to obtain. If two or more variables appear to be dependent, $\mathbb{P}(H)$ can be obtained using copula functions. In order to discover whether the variables are correlated, historic effort data needs to be generated. Therefore, we simulated 500 routes using $p_1 = 0$. This results in routes consisting of nine observations without false sightings. In this way ordinary behavior of the effort characteristic is modelled, making it possible to detect out-of-the-ordinary behavior. For each generated route, the values of the effort variables per route segment can be returned by the simulator. This means that a route consisting of nine sightings, has eight route segments and thus eight times four effort variables. In total the historic effort data consists of $500 \times 8 = 4000$ route segments and their corresponding values for effort_1, ..., effort_4. Table 2 shows the correlation matrix for the historic effort data.

	effort$_1$	**effort$_2$**	**effort$_3$**	**effort$_4$**
effort$_1$	1			
effort$_2$	-0.0181	1		
effort$_3$	-0.0027	-0.01441	1	
effort$_4$	0.0707	0.6372	0.7451	1

Table 2: *Correlation matrix of simulated historic effort data.*

From the correlation matrix it can be concluded that both *effort*$_2$ and *effort*$_3$ are correlated with *effort*$_4$. Besides, *effort*$_1$ is nearly uncorrelated with the other efforts. This means that $\mathbb{P}(H)$ can be obtained using a joint density consisting of two terms:

$$f(x_1, x_2, x_3, x_4) = f_1(x_1) \cdot f_{2,3,4}(x_2, x_3, x_4) \tag{26}$$

where x_i represents *effort*$_i$. The first term in Equation (26) equals the marginal of *effort*$_1$. The second term is a joint density of the dependent variables *effort*$_1$, . . . , *effort*$_4$ and can be estimated by a copula function.

Copulae

The joint density that needs to be obtained by copulae is $f_{2,3,4}(x_2, x_3, x_4)$. From the correlation matrix it is known that both *effort*$_2$ and *effort*$_3$ are correlated with *effort*$_4$. This means that an ordering in importance is present among these three variables. The regular vine suitable to model this ordering is a C-vine with *effort*$_4$ corresponding to the central variable. Note that in this situation, this is equal to a D-vine with *effort*$_4$ as central variable. Figure 16 shows the decomposition of the regular vine, where 2 represents *effort*$_2$, 3 represents *effort*$_3$ and 4 represents *effort*$_4$.

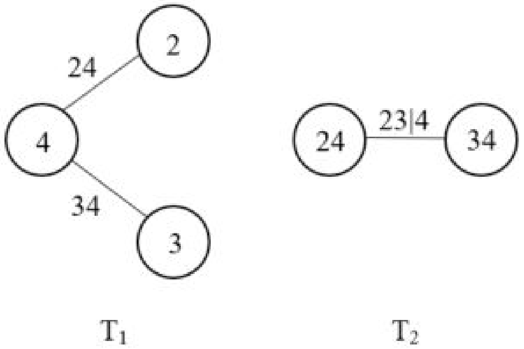


Figure 16: *Decomposition of the 3-dimensional effort density using regular vine.*

The regular vine immediately shows the pair-copula decomposition for the joint density. Equation (27) explicitly states the decomposition.

$$\begin{aligned} f(x_2, x_3, x_4) =& f_2(x_2) \cdot f_3(x_3) \cdot f_4(x_4) \cdot c_{2,4}(F_2(x_2), F_4(x_4)) \cdot c_{3,4}(F_3(x_3), F_4(x_4)) \\ & \cdot c_{2,3|4}(F_{2|4}(x_2|x_4), F_{3|4}(x_3|x_4)) \end{aligned} \tag{27}$$

Since we are dealing with a simulation area, it is difficult to exactly interpret the four effort variables. Therefore, it is unknown if the variables effort_2, effort_3 and effort_4 are conditionally independent. However, for simplicity we assume conditional independence resulting in $c_{2,3|4}(F_{2|4}(x_2|x_4), F_{3|4}(x_3|x_4)) = 1$.

Using the decomposition in Equation (27), the joint density of $\mathbb{P}(H)$ shown in Equation (26) can be rewritten to the multiplication of the four marginals and the two bivariate copulae:

$$\begin{aligned} f(x_1, x_2, x_3, x_4) =& f_1(x_1) \cdot f_2(x_2) \cdot f_3(x_3) \cdot f_4(x_4) \\ & \cdot c_{2,4}(F_2(x_2), F_4(x_4)) \cdot c_{3,4}(F_3(x_3), F_4(x_4)). \end{aligned} \tag{28}$$

The two bivariate copula functions that need to be estimated are $c_{2,4}(F_2(x_2), F_4(x_4))$ and $c_{3,4}(F_3(x_3), F_4(x_4))$. Besides, the four marginal densities need to be estimated. These estimations are performed in R using the historic effort data. Two different R-packages are compared for the estimation of the copula functions. One of the packages was also used in Section 4.2.2.3 and estimates parametric bivariate copulae. However, the parametric bivariate copula functions are limited and therefore not able to capture all sorts of dependencies. The second package estimates bivariate copulae using kernel density estimation (kde). These copulae are able to capture the non-standard dependencies. Figure 17 shows the comparison between the two different packages. The two left plots show the dependencies between effort_2 and effort_4, and effort_3 and effort_4. The middle plots show two random samples generated by the estimated kde-copulae for both dependencies. The right plots show two random samples generated by the estimated parametric copulae for both dependencies.

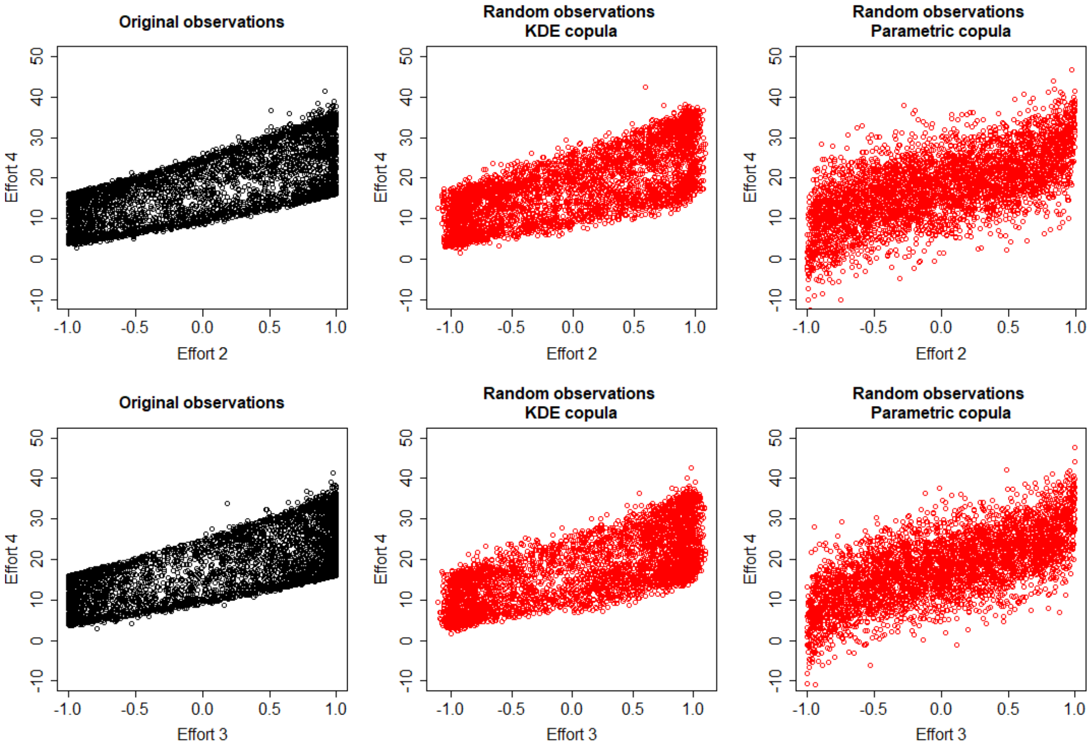


Figure 17: *Comparison parametric copula and kde-copula for effort variables.*

The random observations generated by the kde-copulae resemble more closely the original observations compared to the parameteric copulae. Especially the boundaries of the dependencies are followed better by the kde-copulae. Therefore, the R-package estimating copula functions using kernel density estimation is used to model the effort dependencies. The distributions of the variables itself are estimated parametrically. The estimated marginals are shown in Table 3.

Variable	**Distribution type**
x_1	Normal
x_2	Uniform
x_3	Uniform
x_4	Normal

Table 3: *Estimated marginal distributions for* $f(x_1, x_2, x_3, x_4)$.

Note that for the marginals the listed distributions are not the distributions that the effort data exactly follow. Instead, the historic effort data is compatible with these certain distributions.

Using the estimated distributions and copula functions, the probability density functions (pdf) of each term in Equation (28) can be obtained in R. R is not able to multiply these pdfs in order to obtain $f(x_1, x_2, x_3, x_4)$. Instead, the probability of each term in the equation can be obtained separately. Multiplying these six probabilities lead to the value corresponding to $\mathbb{P}(x_1, x_2, x_3, x_4)$. However, the effort variables are continuous variables and continuous variables have a probability density instead of a probability. This means that $\mathbb{P}(x_1, x_2, x_3, x_4) = 0$ for every given effort value $x_1, \ldots, x_4$. The pdfs can therefore not be used to obtain the probabilities of each term in Equation (28). Instead the cumulative distribution functions (cdf) can be used as an indication of each probability. The cdfs of each term can also be obtained in R. For a given value x of random variable X, a cdf returns the probability that the variable is less than or equal to x: $\mathbb{P}(X \leq x)$. Comparing this probability with the average probability of a cdf, which equals 0.5, indicates whether $\mathbb{P}(X \leq x)$ is an extreme value. When $\mathbb{P}(X \leq x) \geq 0.5$, we will use $1 - \mathbb{P}(X \leq x)$ as probability. When $\mathbb{P}(X \leq x) < 0.5$, we will use $\mathbb{P}(X \leq x)$ as probability. $\mathbb{P}(x_1, x_2, x_3, x_4)$ is thus equal to the multiplication of six estimated probabilities using cumulative distribution functions. This multiplication will lead to a small probability. To overcome this, the geometric mean is used. The geometric mean is defined as taking the nth root of a product of n numbers. Since we are dealing with six terms in order to estimate $\mathbb{P}(x_1, x_2, x_3, x_4)$, n is equal to 6.

Distance module

As discussed in Section 4.2.3 the probability of a certain route, $\mathbb{P}(H)$, can be expanded by additional modules. Effort is the only characteristic present for the simulation area. $\mathbb{P}(H)$ can therefore not be described by additional characteristics. However, the probability can be expanded by an additional module. The module chosen for the simulator is a distance module. The principle of this module is explained in Figure 18.

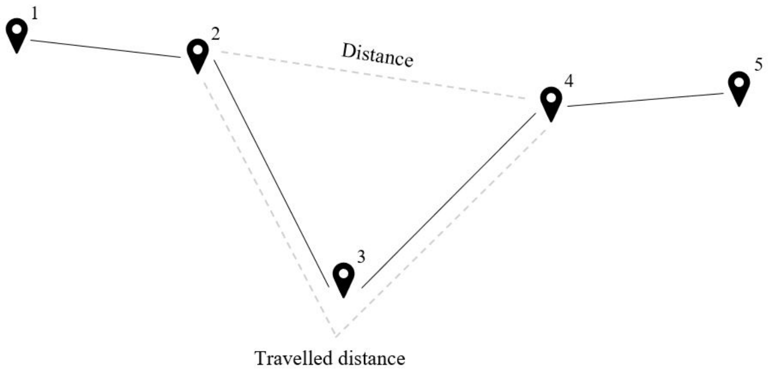


Figure 18: *Distance characteristic using short-cut distance and travelled distance.*

Figure 18 shows a generated route consisting of five sightings. The third sighting seems to be off-track in comparison with the other sightings of the route. This inconsistency might indicate a false sighting. The dotted grey lines in the route represent the travelled distance and the short-cut distance between sighting two and four as the crow flies. The closer these two distance values are, the less likely it indicates an inconsistent sighting. Note that this can only be measured for triplets of sightings.

The probability of inconsistency can be modelled using the two variables *short-cut distance* and *travelled distance*. In order to obtain whether these variables are dependent, historic data needs to be simulated. The historic distance data is obtained using the same 500 simulated routes as for the historic effort data. For each generated route, the values of the distance variables are computed per sighting triplet. Each route consists of nine sightings and thus has seven sighting triplets. In total, the historic distance data consists of $500 \times 7 = 3500$ sighting triplets and their corresponding values for *short-cut distance* and *travelled distance*. The correlation between these two variables appears to be equal to 0.9652. Therefore, a copula function needs to be estimated to obtain the joint density of the two variables. This is performed using the same procedure as for the effort copula, but using two variables instead of three. Figure 19 shows the comparison between the two different copula estimations for the distance variables.

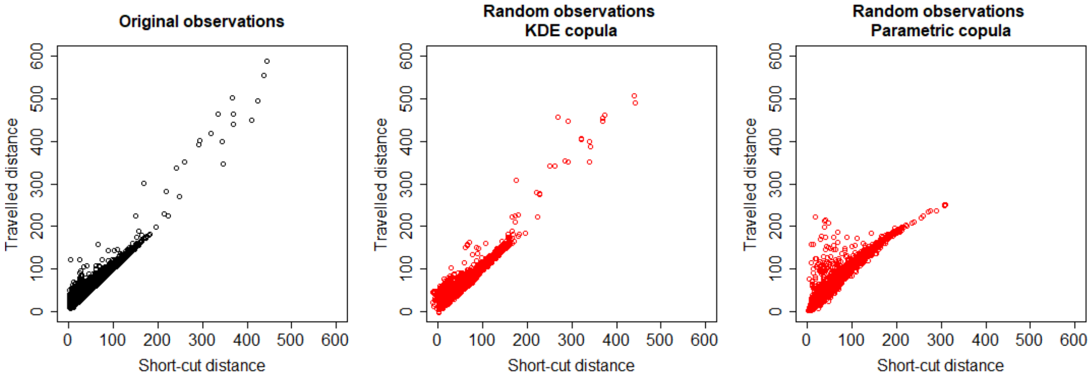


Figure 19: *Comparison parametric copula and kde-copula for distance variables.*

Again, the observations generated by the kde-copula resembles the original observations more closely. The $x = y$ boundary is not followed by the parametric copula. Instead, the boundary of the parametric copula bends towards impossible values where the short-cut distance exceeds the travelled distance. Therefore, the distance copula is also estimated by using kernel density estimation. The distributions of the distance variables are estimated parametrically and appear both to be compatible with a normal distribution. The final probabilities for the distance module are obtained in the same way as the effort probabilities.

Now the probability of a certain route is expanded by a distance module, $\mathbb{P}(H)$ is characterised by effort and distance. Equation (26) can therefore be rewritten to $\mathbb{P}(H) = \mathbb{P}_{\text{effort}} \cdot \mathbb{P}_{\text{distance}}$, with $\mathbb{P}_{\text{effort}} = \mathbb{P}(\mathit{effort}_1, \mathit{effort}_2, \mathit{effort}_3, \mathit{effort}_4)$ and $\mathbb{P}_{\text{distance}} = \mathbb{P}(\mathit{distance}, \mathit{travelled\ distance})$. Therefore, the second term of the hypothesis evaluator can be stated as:

$$\frac{\mathbb{P}(H)}{\mathbb{P}(\neg H)} = \frac{\mathbb{P}_{\text{effort}} \cdot \mathbb{P}_{\text{distance}}}{1 - (\mathbb{P}_{\text{effort}} \cdot \mathbb{P}_{\text{distance}})}. \tag{29}$$

5.3 Evaluation

The function of the hypothesis evaluator is to correctly classify false and true sightings in the simulated routes. A route is correctly returned by the algorithm when all true sightings are included in the route and all false sightings are excluded. To measure the performance of the algorithm, a confusion matrix concerning false and true sightings can be used, see Figure 20.

$$\begin{array}{ccc} & & \text{Algorithm} \\ & & \begin{array}{cc}\text{F} & \quad\text{T}\end{array} \\ \text{Actual} & \begin{array}{c}\text{F}\\ \text{T}\end{array} & \begin{bmatrix}\text{TP} & \text{FN}\\ \text{FP} & \text{TN}\end{bmatrix}\end{array}$$

Figure 20: *Confusion Matrix false (F) and true (T) sightings.*

The confusion matrix shows two possible mis-classifications for the algorithm. Classifying a false sighting as a true sighting, results in a false-positive (FP). Classifying a true sighting as a false sighting, results in a false-negative (FN). The recall of both the true and false sightings, measure the fraction of correctly classified sightings. Equations (30) and (31) state the formulas of both recalls.

$$\text{Recall}_{\text{F}} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{30}$$

$$\text{Recall}_{\text{T}} = \frac{\text{TN}}{\text{TN} + \text{FP}} \tag{31}$$

The performance of the algorithm can be measured by the two conflicting recalls. When both $\text{Recall}_\text{F} = 1$ and $\text{Recall}_\text{T} = 1$, a route is correctly returned by the algorithm. This means that both recalls are to be maximized. The threshold value t used in the hypothesis generator influences the value of the recall. When $t = 0$, all possible hypotheses in the search tree are generated. This causes the evaluator to classify all sightings in a route to be true, resulting in $\text{Recall}_\text{F} = 0$ and $\text{Recall}_\text{T} = 1$. On the other hand, when $t = 1$, no hypotheses are generated. This causes the evaluator to classify all sightings in a route to be false, resulting in $\text{Recall}_\text{F} = 1$ and $\text{Recall}_\text{T} = 0$. For all threshold values between 0 and 1, the recall values can be computed. However, it is impossible to state which of these different outcomes is the 'best', since both recalls are to maximized. The results in the next section are therefore given in a Pareto front. This front shows the optimal solutions for different threshold values.

5.4 Experimental Results

In this section the experimental results of the algorithm are shown. This section is split into two parts in order to demonstrate the impact of using copula functions in the algorithm. The section concludes with the lessons learned from the validation.

The characteristics chosen in the hypothesis evaluator are distance and effort. The aim of these characteristics is to distinguish false sightings from true sightings. The characteristics are made up of multiple variables. The dependence between the variables is modelled by copula functions. The usage of copulae are only effective when the combination of its variables play a significant role in distinguishing false and true sightings. Otherwise, just one of the involved variables could be used in isolation to classify true and false sightings. To illustrate this, Figure 21 shows two different plots of the travelled distance variable used for the distance characteristic. Recall that the travelled distance is measured over three sightings.

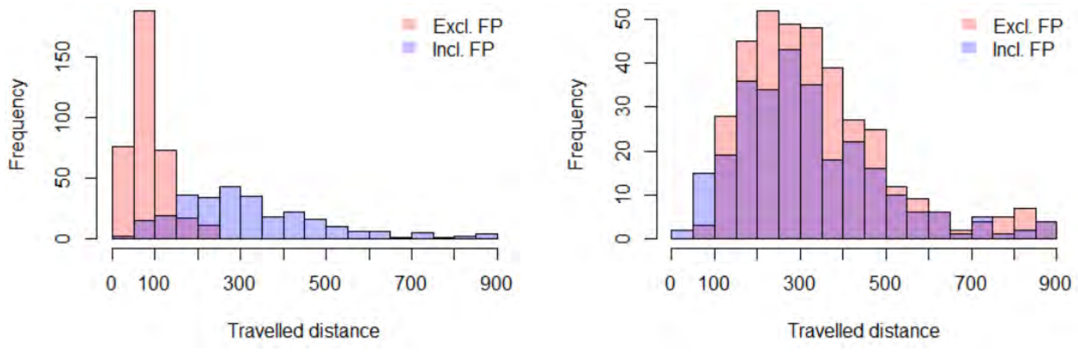


Figure 21: *Classify false and true sightings by the travelled distance variable.*

The left plot shows that the travelled distance values of sighting triplets including a false sighting are different from the values of triplets excluding a false sighting. High values indicate that one or more false sightings are present in the triplet. This means that the travelled distance variable on itself is able to distinguish false and true sightings. No complementary variable is needed. The right plot shows that the travelled distance values of sighting triplets including a false sighting are equal to the values of triplets excluding a false sighting. The value of travelled distance is not an indicator for the presence of false sightings in a triplet. This means that the travelled distance variable on itself is not able to distinguish false and true sightings. At least one complementary variable is needed, for instance the short-cut distance. This means that using a copula function for the left plot brings no added value, whereas for the right plot it does. The same conclusions can be drawn from Figure 22. This plot shows the travelled distance values versus the short-cut distance values. The black points indicate true sightings, whereas the red points indicate false sightings. The same plots and conclusions can be drawn for the effort characteristic.

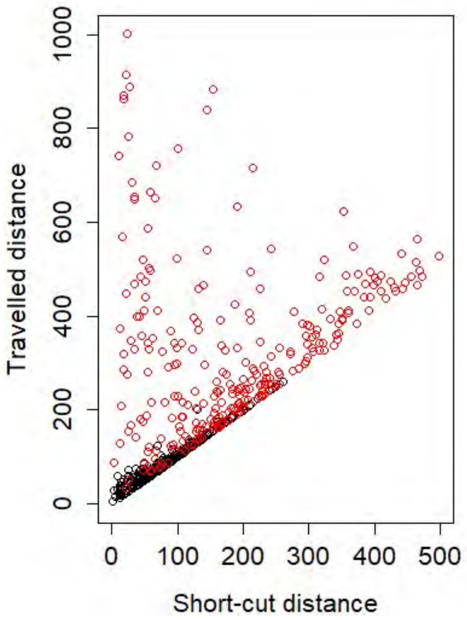


Figure 22: *Travelled distance versus short-cut distance for true (black) and false (red) sightings.*

To demonstrate the impact of using copula functions, two different simulation test sets are used in the upcoming two sections. One containing disjunct variable values, as in the left plot of Figure 21, and one containing overlapping variable values, as in the right plot of Figure 21. Subsequently, two different Pareto fronts are generated for both test sets using two variants of the algorithm. One variant uses copula functions for the characteristics and the other only uses the marginals of the characteristics. Comparing these Pareto fronts demonstrates the impact of using copula functions in the algorithm.

5.4.1 Test Set 1

In order to generate experimental results, a test set is simulated consisting of 100 collections of sightings using parameters $p_1 = 0.3$ and $p_2 = 0.7$. The number of sightings per collection range from 6 to 9. The number of false positive sightings per collection range from 0 to 5. The exact distributions of these numbers can be found in Tables 9 and 10 in *Appendix C.1.* This first test set contains disjunct variable values for the characteristics used in the hypothesis evaluator.

The algorithm is applied to each of the collections in the test set. The test set provides whether a sighting is true or false. This means that Recall_F and Recall_T can be computed for each route in the test set. This is done for 11 different values of threshold value t: $\{0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1\}$. Figure 23 shows the Pareto fronts for both the algorithms including and excluding copula functions. In these Pareto fronts the Recall_F - versus the Recall_T values are shown for both versions of the algorithm. As stated in Section 5.3, no outcome is better or worse than another outcome. The Pareto front only shows the optimal recall values for different threshold values. However, the performance of one version of the algorithm can be better or worse than the performance of the other version of the algorithm. The higher the values for both Recall_F and Recall_T, the better the performance of the algorithm.

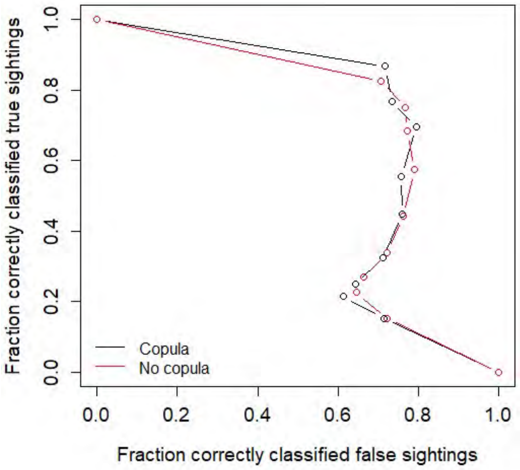


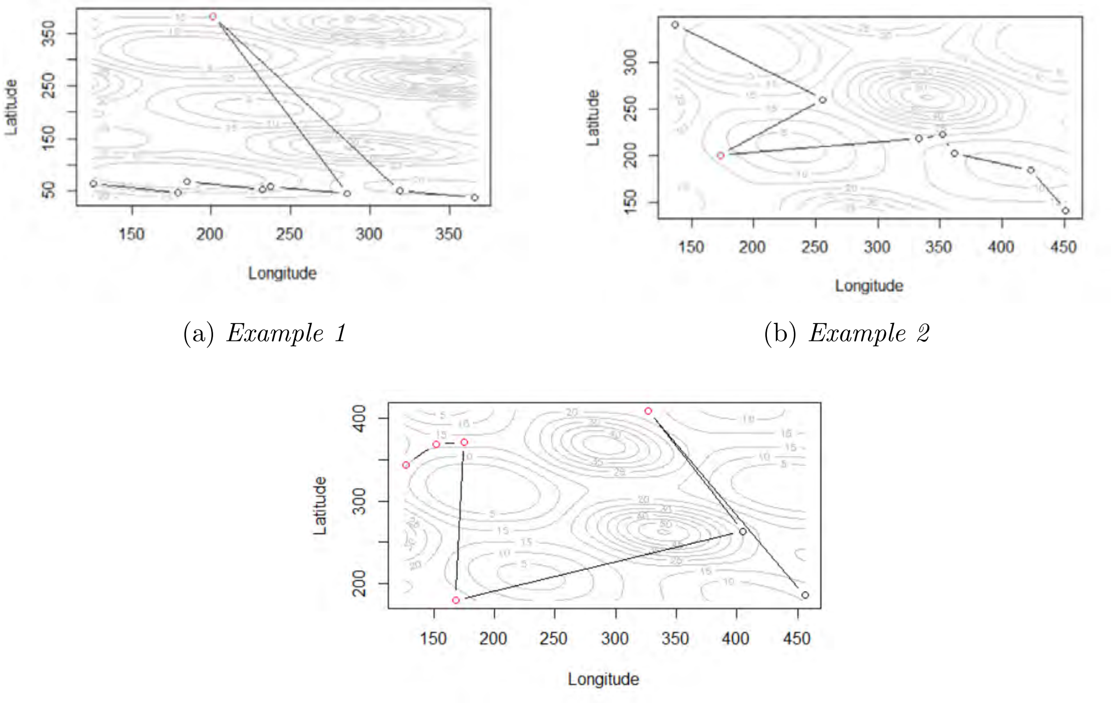
Figure 23: *Pareto fronts test set 1.*

The plots in Figure 24 on the next page, show three illustrative examples of routes from the test set. The black points in the plots represent the true sightings, whereas the red points represent the false sightings. The sightings are connected in chronological order. In the background of the plots, the contours of the simulation area are shown.

Figure 24a shows a simulation example where both Recall_F and Recall_T are equal to 1 for both algorithms. The first sighting corresponds to the point shown in the lower right corner of the plot. The plot shows that the third sighting is false and thus the correct route equals $\{1, 2, 4, 5, 6, 7, 8, 9\}$. Both the algorithms returned this route correctly. This is not surprising, since the false sighting is very inconsistent compared to the true sightings.

Figure 24b shows a simulation example where $\text{Recall}_\text{F} = 1$ and $\text{Recall}_\text{T} = \frac{6}{7}$. The first sighting corresponds to the upper left point in the plot. The plot shows that the third sighting is false and thus the correct route equals $\{1, 2, 4, 5, 6, 7, 8\}$. The route returned as most likely by both the algorithms equals $\{2, 4, 5, 6, 7, 8\}$. This means that the third sighting is correctly classified as false. However, the first sighting is incorrectly classified. The first sighting seems a bit inconsistent in comparison to the other true sightings. This might explain why this point is incorrectly classified.

Figure 24c shows a simulation example where $\text{Recall}_\text{F} = \frac{2}{5}$ and $\text{Recall}_\text{T} = 0$. The first sighting corresponds to the point shown in the lower right corner of the plot. The plot shows that only the first and third sighting are true and thus the correct route equals



(a) *Example 1*

(b) *Example 2*

(c) *Example 3*

Figure 24: *Example collections of sightings leading to different values of Recall$_F$ and Recall$_T$.*

$\{1, 3\}$. The route returned as most likely by both the algorithms equals $\{5, 6, 7\}$. The classification of the algorithm seems to be wrong. However, regarding the characteristics used in the algorithm, the classification is quite good. Sightings $\{5, 6, 7\}$ are placed in an area with low effort and the triplet is consistent in distance. This explains why these sightings are classified as correct.

5.4.2 Test Set 2

The second test set also consists of 100 collections of sightings, generated using the same parameters as the first test set; $p_1 = 0.3$ and $p_2 = 0.7$ The number of sightings per collection range from 6 to 9. The number of false positive sightings per collection range from 0 to 5. The exact distributions of these numbers can be found in Tables 11 and 12 in *Appendix C.2*. This second test set contains overlapping variable values for the characteristics used in the hypothesis evaluator.

The algorithm is applied to each of the collections in the test set. The test set provides whether a sighting is true or false. This means that Recall$_\text{F}$ and Recall$_\text{T}$ can be computed for each route in the test set. This is done for 11 different values of threshold value t: $\{0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1\}$. Figure 25 shows the Pareto fronts for both the algorithms including and excluding copula functions. The same threshold values as for the first test set are used.

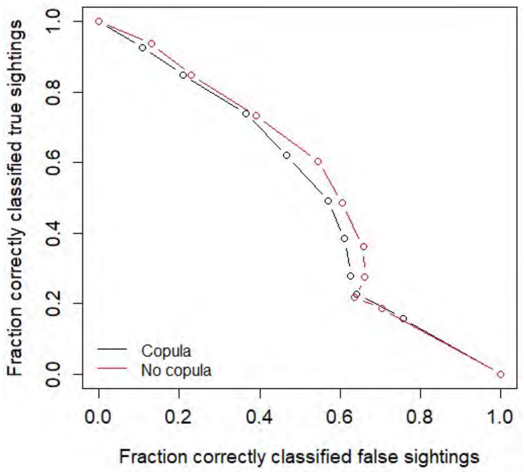


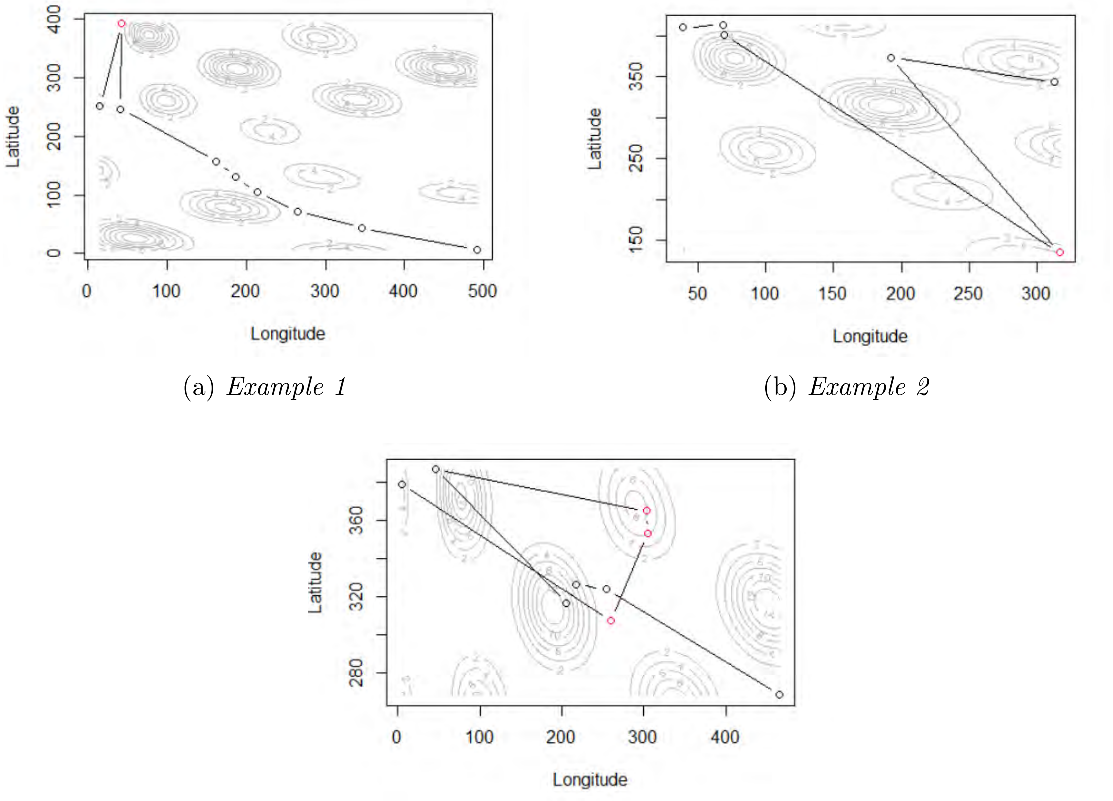
Figure 25: *Pareto fronts test set 2.*

The plots in Figure 26 on the next page, show three illustrative examples of routes from test set 2. The sightings are connected in chronological order, where the left point always represents the first sighting.

Figure 26a shows a simulation example for which the algorithm including copula functions outperforms the algorithm excluding copula functions for all values of t. Regarding the characteristics used in the algorithm, this result is not surprising for this route. The effort is low for all sightings. This means that the additional distance characteristic is needed to distinguish the false and true sightings. The travelled distance as well as the short-cut distance is comparable for all sighting triplets in the route. However, for the first three sightings the combination of the distance variables indicates the false sighting. This indicates why the algorithm including copula functions outperforms the algorithm excluding copula functions.

Figure 26b shows a simulation example for which both variations of the algorithm perform equally for all values of t. The effort of all sightings is comparable. The distance values however, differ much over the sighting triplets. The travelled distance for sightings $\{3, 4, 5\}$ is inconsistent with the rest of the triplets. This means that the inconsistency can already be defined using only the travelled distance variable. Therefore, the algorithm including and excluding copula functions perform equally.

Figure 26c shows a simulation example for which both variations of the algorithm perform poorly. The route shown has a complex structure. Regarding the characteristics used in the algorithm, it is difficult to correctly classify false and true sightings. For each value of t both algorithms returned other classifications. All classifications resulted in a small value for both Recall_F and Recall_T. However, the recall values of the algorithm excluding copulae exceed the recall values of the algorithm including copulae for all values of t.



(a) *Example 1*

(b) *Example 2*

(c) *Example 3*

Figure 26: *Example collections of sightings leading to different values of Recall$_F$ and Recall$_T$.*

5.5 Benchmark

To examine the results shown in the previous two sections, the performance of the algorithm is evaluated using a benchmark model. Since no models similar to the algorithm are known, the results cannot be benchmarked against an existing model. Therefore, a random model is used a benchmark.

The benchmark model used to evaluate the performance of the algorithm is a true-random model. This model computes the probability that a route is returned correctly by a binomial model. In other words, the true-random model computes the probability of a route returning both Recall$_\mathrm{F}$ $= 1$ and Recall$_\mathrm{T}$ $= 1$. The model is applied to the test set described in the previous section. The simulation parameters used to create the test set were $p_1 = 0.3$ and $p_2 = 0.7$. Multiplying these parameters results in $FP = 0.3 \times 0.7 = 0.21$, which indicates that approximately one out of five sightings equals a false positive. Given this value, the probability of a route returning Recall$_\mathrm{F}$ $= 1$ and Recall$_\mathrm{T}$ $= 1$ can be computed using a Binomial model. In a Binomial model, X is defined as the number of successes in n trials. The probability of X can be computed using the Binomial distribution with the following probability mass function:

$$\mathbb{P}(X = k) = \binom{n}{k} p^k (1-p)^{n-k} \tag{32}$$

with k equal to the number of successes and p the probability of a single success.

Now, define success as correctly detecting a false positive sighting and n as the number of sightings in a collection. Using Equation (32) the probability of generating a correctly returned route with a random model can be computed. Consider the example route data presented in Table 1. This route consists of eight sightings of which two are false. The random probability of generating a result 1 for this route can be computed using $k = 2$, $n = 2$ and $p = 0.3 \times 0.7 = 0.21$. This leads to the following probability:

$$\mathbb{P}(X = 2) = 0.21^2(1 - 0.21)^{8-2} = 0.0107. \tag{33}$$

Note that the binomial coefficient of Equation (32) should be omitted, since we are dealing with a chronologically ordered collection of sightings. This means that there is only one way of distributing the false positives in the collection of sightings, resulting in $\binom{n}{k} = 1$.

The true-random model computes the probabilities for all routes included in the test set. The summation of all these probabilities results in the expectation of the number of correctly generated routes by the random model. This expectation equals 3.8 for the first test set and 2.4 for the second test set. The highest number of correctly generated routes by the algorithm including copula functions equals 31 for $t = 0.1$, for the first test set. The highest number of correctly generated routes by the algorithm excluding copula functions equals 30, also for $t = 0.1$. For the second test set, the highest number of correctly generated routes by the algorithm including copula functions equals 6 for both $t = 0.2$ and $t = 0.3$. The highest number of correctly generated routes by the algorithm excluding copula functions also equals 6, for $t = 0.1$ and $t = 0.2$.

5.6 Discussion and Conclusion

The results of the first test set show that the performance of both variations of the algorithm is comparable. For each threshold value the recall values do not differ significantly. This means that using copula functions for the first test set does not add value to the performance of the algorithm. At the same time, the copula functions do not harm the results. This is in accordance with our expectations. Moreover, the example routes show that the behavior of both algorithms is explainable regarding the used characteristics. Even when the algorithms seem to perform poorly, they in fact perform well regarding the characteristics. Comparing the performance of both variations of the algorithm with the benchmark model, one can conclude that the algorithm outperforms the random model. Both variations of the algorithm perform approximately 8 times better than the random model.

The results of the second test set show that the algorithm excluding copula functions outperforms the algorithm including copula functions for most values of t. This contradicts our expectation about the impact of copula functions. Moreover, the overall performance of the algorithm is worse compared to the performance of the first test set. The example routes show two routes of which the behavior of both algorithms is perfectly explainable. However, for the third example the behavior of the algorithm is difficult to explain regarding the route characteristics. This might indicate that the characteristics used in the algorithm are not suitable for the complex routes in the test set. The algorithm excluding copula functions performs better for the complex routes. This indicates that in some way the copula functions cause noise, resulting in a worse performance. It appears that most routes in the second test set are comparable to the complex route shown in the third example. This explains the overall poor performance of the algorithms for the second test set. Comparing the performance of both variations of the algorithm with the benchmark model, one can conclude that the algorithm outperforms the random model. Both variations of the algorithm perform 2.5 times better than the random model.

The overall performance of both test sets might be increased by further investigating in additional characteristics and modules applicable to the simulation data. However, the aim of this chapter was to discover whether the designed algorithm functions in practice instead of optimizing the algorithm. Regarding the results, one can conclude that the algorithm is able to perform in practice. Nevertheless, multiple lessons learned should be kept in mind while applying the algorithm. First, it is important to choose the right characteristics for modelling the likelihood of a route. The routes in the first test set appeared to be explainable regarding the characteristics, leading to a reasonable performance of the algorithm. Those characteristics appeared to be less applicable to the complex routes in the second test, leading to a poor performance. Besides, the usage of copula functions in the algorithm influences the performance. In fact, copula functions might harm the results, as demonstrated for the second test set. Therefore, both variations of the algorithm should be applied to a collection of sightings to indicate which variation of the algorithm performs best.

6 Real-World Example

In this chapter the validated algorithm is tested on real-world data. The data originates from a television show where ordinary people go on the run and try to evade capture from a professional investigation team. While being on the run, the participants leave sightings. These sightings can be analyzed by the algorithm to check whether they are consistent with each other. In this way, the algorithm is able to assist the investigation team in tracking down the participants of the show. First, the television show is described in more detail as well as the data used in it. Second, the characteristics used to model the hypothesis evaluator are explained. After that, the results are presented and conclusions about the real-world example are drawn.

6.1 Hunted

Hunted is a real-life television show where ordinary people, referred to as *fugitives*, go on the run for 21 days. They try to evade capture from a professional investigation team; the *hunters*, who all have a police, military or intelligence background. The hunters have access to all tools the police has at its disposal nowadays. They interrogate friends and family of the fugitives, undertake home searches, monitor bank- and phone records, and make use of open-source intelligence. Furthermore, the hunters have access to all *Closed-Circuit Television* (CCTV)- and *Automatic Number Plate Recognition* (ANPR) cameras. Using all these surveillance powers, the hunters make every effort to catch the fugitives within the 21 days. At the same time, the fugitives do everything to make it the hunters as difficult as possible. They try to leave minimal evidence by avoiding surveillance cameras and minimizing the usage of their mobile phone. Moreover, the fugitives try to mislead the hunters by creating fake traces.

Data
Each fugitive is followed by a team to catch his or her journey. This team also records exactly where the fugitive went at which moment of time. Besides, the team documents whether the fugitive passed a CCTV/ANPR camera, made a call, or made a withdrawal. This results in a document consisting of a fugitive's timeline. The data used as real-world example is a conversion of the raw timelines. The converted data contain ordered activities of the fugitives defined by date-time stamps, locations and a short description of the activity. Example activities are; making a call or sending a text message, moving from one place to another using a mode of transportation, and spending the night at a certain location.

In order to check the quality of the data, an exploratory data analysis is performed on move activities. Each move activity in the data set has its own corresponding mode of transportation; car, bicycle or walk. For all three modes of transportation, the speed, distance and duration of the corresponding activities are analyzed. Figure 27 shows the histograms of speed, distance and duration per mode of transportation. The first row shows the histograms of speed, the second row of distance and the third row of duration.

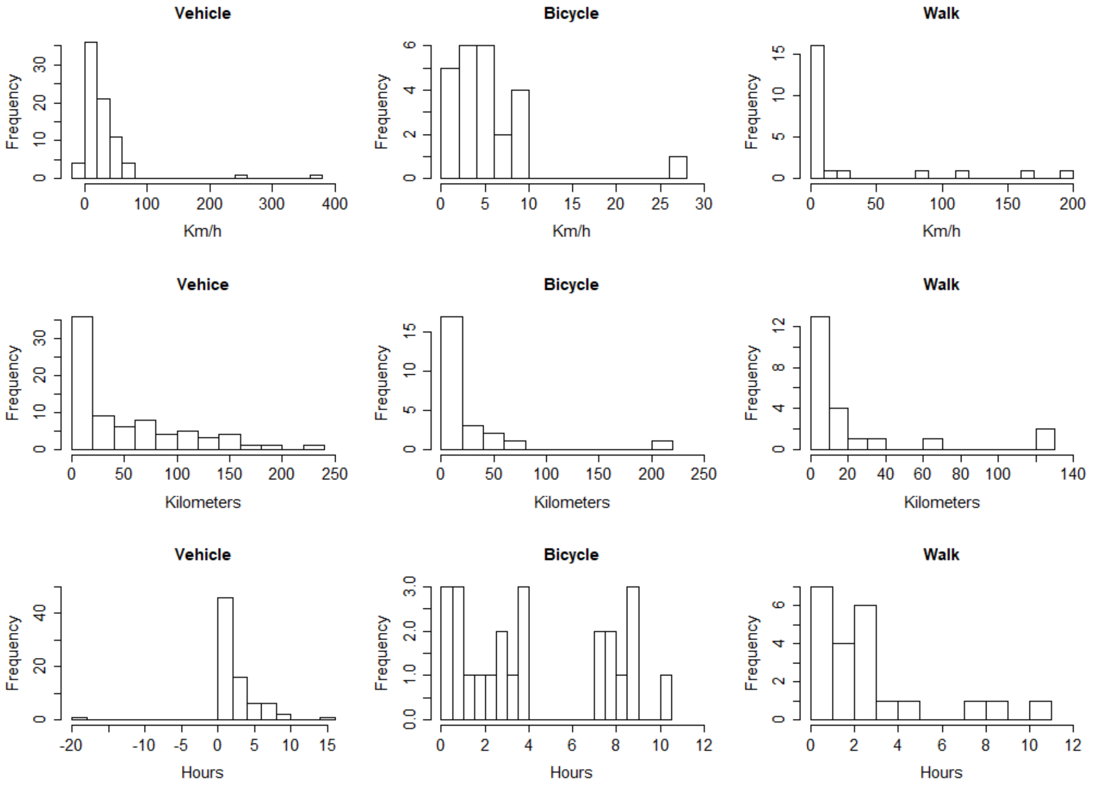


Figure 27: *Histograms of speed, distance and duration for move activities.*

Each histogram in Figure 27 shows impossible values. This indicates that the data contains faulty values. This is not surprising, since the raw timelines are created by hand. The conversion of the raw data is also done by hand. Hence, human errors in the data are inevitable. To improve the data quality, we revised the data by focusing on the impossible values indicated in the histograms of Figure 27. Afterwards, the same histograms were created using the improved data which can be found in Figure 30 in *Appendix D*. These histograms do not contain impossible values.

The revised data set contains activities of nine teams with a total of 119 days. These activities do not contain misleading or false information. Therefore, these data can be used as historic data set for creating route characteristics of a fugitive. In this way, the ordinary behavior of a fugitive is modelled, making it possible to detect out-of-the ordinary behavior. The choice of the characteristics and how they are build is discussed in the next section.

6.2 Hypothesis Evaluator

The hypothesis evaluator of the algorithm is modelled similarly to the simulator. Again the Bayesian inference formula is used. However, the effort characteristic is replaced by two other characteristics more appropriate to fugitive behavior. Besides, the source reliability is computed in a different way.

Source reliability

Five different sources are available to the hunters to track the fugitives. First, the fugitives can be observed by cameras. Two different types are possible: CCTV and ANPR. CCTV is also known as video surveillance. The cameras are placed in public areas as well as private. The surveillance footage is available to the hunters. ANPR cameras store and read vehicle number plates. They are used to create vehicle location data. The ANPR footage and location data is available to the hunters. Second, the fugitives can be tracked by tapping telephones. In this way, any communication via telephones can be monitored. The hunters have access to telephone conversations and text messages. Lastly, the fugitives can be tracked when they make a cash withdrawal. The camera footage of a cash machine is available to the hunters.

Each of the five sources have a different reliability. Therefore, each source is assigned a reliability probability. These probabilities are defined using a hunter's expert knowledge. The probabilities are used to model the first component of the hypothesis evaluator: $\frac{\mathbb{P}(E|H)}{\mathbb{P}(E|\neg H)}$.

Source	$\mathbb{P}(E\|H)/\mathbb{P}(E\|\neg H)$
Text message	0.1
Cash withdrawal	0.1
ANPR	0.2
CCTV	0.9
Phone conversation	0.9

Table 4: *The reliability probabilities assigned to the five different sources available to the hunters.*

The text message and cash withdrawal are assigned the smallest probability, because they are the least reliable. For both sources it is difficult to identify the person who sent the text message or made the cash withdrawal. It is known to a fugitive that he or she is filmed at a cash machine. Hence, the fugitives mostly disguise themselves by wearing a helmet for example. The ANPR source is also assigned a low probability. This is because recognizing a car familiar to a fugitive does not necessarily mean that the fugitive is actually in that car. The CCTV source and phone conversations are assigned the highest reliability, because identifying the fugitives is easier for these sources.

Route characteristics

The first route characteristic used for the fugitive data is speed. As described in the previous section, the historic data contain activities with its corresponding mode of transportation. However, the mode of transportation is not of interest for fugitive sightings. Two consecutive sightings can have the same mode of transportation, but that does not say anything about the mode of transportation used between these two sightings. Therefore, the speed of each activity in the historic data set is computed, irrespective of the mode of transportation. The speed of an activity is computed by dividing its distance travelled and duration. Therefore, three variables are used for the speed characteristic: *distance*, *duration* and *speed.* Using these variables in the hypothesis generator makes it possible to measure inconsistency between two consecutive sightings.

Unrealistic distance, duration and speed values between two sightings might indicate a false sighting. This can be detected by the speed variables. The values of the variables are computed for the historic data set. Table 5 shows the correlation matrix for the historic speed variables.

	Distance	**Duration**	**Speed**
Distance	1		
Duration	0.5026	1	
Speed	0.5492	-0.1331	1

Table 5: *Correlation matrix of historic speed variables.*

From the correlation matrix can be concluded that both *speed* and *duration* are correlated with *distance*. This is comparable with the three dependent effort variables of the simulator. Therefore, two bivariate copula functions need to be estimated as well as the three marginals of the variables. The copulae are estimated using kernel density estimation. The distributions of the speed variables are estimated parametrically and appear all three to be compatible with an exponential distribution.

Distance to point of interest

The second characteristic used is the distance from a sighting to the nearest point of interest (poi). The pois used are; gas stations, camp sites, hotels, bus and railway stations, highways, parking places, restaurants and large cities. The historic distributions of the distances to these pois, might contain information about the likelihood of a sighting. This characteristic consists of eight variables; the distance to each defined poi. The values of the variables are computed for the historic data set. Table 13 in *Appendix D.2* shows the correlation matrix for the historic poi variables. From the correlation matrix can be concluded that multiple variables are correlated. The variables of which the correlation value exceeds 0.5 are modelled using a copula function. In total four bivariate copula functions are estimated using kernel density estimation. The distributions of the poi variables are estimated parametrically. Table 14 in *Appendix D.2* shows an overview of the variables and its corresponding distributions.

Distance module

The additional module used for the fugitive data is the same module used for the simulator; the distance module. This module appeared to be effective for the simulation data and might also be effective for the fugitive data. The values of the two variables *short-cut distance* and *travelled distance* are computed for the historic data set. The variables appeared to have a correlation coefficient equal to 0.9571. Therefore, a bivariate copula function is estimated using kernel density estimation. The distributions of the distance variables are estimated parametrically. Both variables are compatible with the exponential distribution.

In total, two characteristics and one additional module are used to model the likelihood of a route of a fugitive. Therefore, the second term of the hypothesis evaluator can be stated as:

$$\frac{\mathbb{P}(H)}{\mathbb{P}(\neg H)} = \frac{\mathbb{P}_{\text{speed}} \cdot \mathbb{P}_{\text{poi}} \cdot \mathbb{P}_{\text{distance}}}{1 - (\mathbb{P}_{\text{effort}} \cdot \mathbb{P}_{\text{poi}} \cdot \mathbb{P}_{\text{distance}})}. \tag{34}$$

6.3 Results

The algorithm and its estimated characteristics is applied on two situations in which a fugitive intentionally created false sightings. The two situations and the result of the algorithm are described in this section. In response to the conclusions of the validation phase, both the algorithm including and excluding copula functions are applied on the two situations.

6.3.1 Situation 1

The first test set consists of two days with a total of ten sightings. On the first day, the team leaves four sightings while walking through a city and making a cash withdrawal. For the fifth sighting, the team asked an employee of a restaurant to make a phone call to a team member's wife, to mislead to hunters. At the time the employee made the phone call, the fugitives were already 35 kilometers away. This means that the fifth sighting of the test set is false. On the second day, the team leaves four sightings while looking for a ride. The last sighting of the first test set is a number plate recognition while getting a ride. To summarize, the first test set contains ten sightings, of which one is false.

Figure 28 shows the Pareto fronts for test set 1 for seven different threshold values. For $t \geq 0.6$, no hypotheses were generated. This means that both $\text{Recall}_\text{F} = 1$ and $\text{Recall}_\text{T} = 0$ for all those threshold value. Therefore, only the recall values for $t \leq 0.6$ are displayed in the figure.

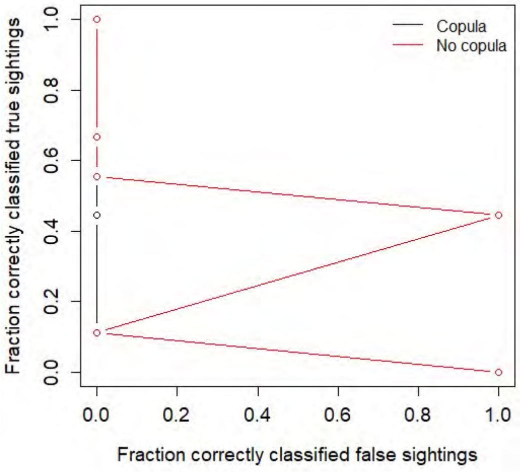


Figure 28: *Pareto fronts situation 1.*

The Pareto fronts show that both variations of the algorithm perform equally, except for $t = 0.3$. For this threshold value, the algorithm excluding copula functions outperforms the algorithm including copula functions. Besides, this is the only threshold value for which the algorithm correctly classifies the false sighting. For all other threshold values, the algorithms correctly classify some of the true sightings. Note that the correct route is never returned by the algorithms.

6.3.2 Situation 2

The second test set consists of 18 days with a total of eight sightings. This team avoided public areas as much as possible by moving and sleeping in the forest for most of the time. This resulted in only eight sightings available to the hunters in 18 days. On the first day, a friend of the team made a cash withdrawal for the team. This means that the first sighting of the test set is false. On the third day, the team was observed twice by CCTV while waiting for a ride. On the tenth and fourteenth day the team was observed by ANPR-cameras. On the sixteenth day, the team created two false sightings. Two friends of the team were asked to send a text message to a team member's girlfriend. One text message was send from a city 100 kilometers away from the team member's place. The other message was send 150 kilometers away from the team member's place. The last sighting of the test set is a phone call made by the team which was tapped by the hunters. To summarize, the second test set contains eight sightings, of which three are false.

Figure 29 shows the Pareto fronts for test set 2 for seven different threshold values. For $t \geq 0.6$, no hypotheses were generated. This means that $\text{Recall}_\text{F} = 1$ and $\text{Recall}_\text{T} = 0$ for all those threshold value. Therefore, only the recall values for $t \leq 0.6$ are displayed in the figure.

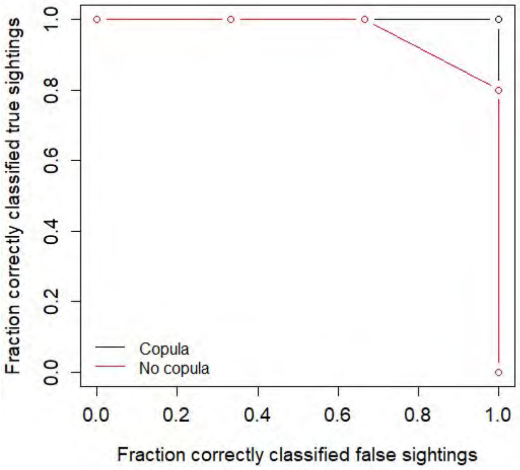


Figure 29: *Pareto fronts situation 2.*

The Pareto fronts show that the performance of both variations of the algorithm perform equally, except for $t = 0.3$. For this threshold value, the algorithm including copula functions outperforms the algorithm excluding copula functions. Besides, this is the only threshold value for which the algorithm correctly classifies all true and false sightings. It strikes that for all other threshold values both algorithms correctly classify all true sightings. Note that only five points are displayed in the Pareto front. This is caused by the fact that both $t = 0$ and $t = 0.1$ return $\text{Recall}_\text{T} = 1$ and $\text{Recall}_\text{F} = 0$.

6.4 Discussion and Conclusion

The results of both situations show that the algorithm is able to classify the hunted data. For the first situation, the algorithm excluding copula functions performs better, whereas for the second situation the algorithm including copula functions performs better. Overall, the algorithms perform better for the second situation. This may be due to multiple reasons. First, it might be caused by the fact that the false sighting of the first test set is less inconsistent compared to the false sightings of situation two. As described, the false sighting of the first situation is created 35 kilometers away from the team. For the second test set, these distances are significantly higher. Therefore, the values of the distance and speed variables might not stand out compared to the values of the true sightings of the first test set. This could imply that the algorithm is not able to detect false sightings close the actual position of a fugitive. It might also be the case that the chosen characteristics are not sufficient to classify the true and false sightings correctly. Additional modules or characteristics may improve the performance of the algorithm. However, for the second test set the characteristics appear to be adequate.

7 Overall Conclusion and Recommendations

This chapter answers the sub-questions formulated for this research, after which the overall conclusion is drawn. Besides, recommendations anf suggestions for future research are given.

7.1 Conclusion

The aim of this research was to develop an algorithm which is able to detect inconsistencies in observations of moving entities. Four sub-questions were formulated which are implicitly answered in this thesis. For convenience, the questions and their explicit, concluding answers are stated below.

1. *Which techniques are suitable for evaluating entity movements?*
   Entity movements are evaluated in the hypothesis evaluator component of the algorithm. The technique used in this research to evaluate entity movements is Bayesian inference. This technique was chosen based on a literature review. The results of both the simulation and real-world example show that Bayesian inference is a suitable technique to evaluate routes. However, the technique is highly dependent of the characteristics chosen to model the likelihood of a route. When the chosen characteristics are not suitable to evaluate entity movements, the Bayesian inference will neither work. Other techniques might as well be suitable to evaluate entity movements. This research explicitly stated that artificial intelligence techniques were not investigated, because of their uninterpretable character. Still, this does not mean that AI techniques are necessarily unsuitable. When a strategy for applying AI techniques in intelligence analysis is clear, AI techniques can be investigated to implement in the algorithm.

2. *How can characteristics belonging to a moving entity be integrated into the algorithm?*
   Characteristics belonging to a moving entity are part of the hypothesis evaluator of the algorithm. The chosen characteristics were modelled for historic data. When the historic data only contains true entity movements, the ordinary behavior of an entity can be modelled. In this way, it is possible to detect out-of-the-ordinary behavior of the moving entity. Not all characteristics can be modelled historically. For the real-world example, the source reliability is defined using expert-knowledge. In this way the characteristic is rule-based. Characteristics can therefore be implemented in both a data-driven and rule-based way. Depending on the characteristic to be modelled, one approach might be preferred over the other.

3. *Which techniques are suitable for constructing entity movements?*
   Entity movements are constructed in the hypothesis generator component of the algorithm. The technique used in this research to construct entity movements is backtracking. This technique iteratively constructs promising chains of sightings. It was chosen because of the reduction in runtime in comparison with a brute force approach. However, other techniques are available to reduce runtime. Branch-and-Bound might for example also be suitable to construct entity movements.

4. *Which characteristics belonging to a moving entity are required in order to apply the algorithm to a real-world example?*
   The characteristics to be chosen depend on the concerned moving entity. The real-world data used in this research originates from a television show where fugitives go

on the run. Therefore, characteristics suitable to detect misleading fugitive behavior were chosen. The results show that the algorithm is able to detect false sightings created by a fugitive. However, the characteristics were not completely suitable for the first situation. This shows that the suitable characteristics differ per concerned moving entity. Developing and testing multiple characteristics will lead to the most suitable configuration for the algorithm.

Regarding the results, discussion and answers to the sub-questions, we conclude that the overall goal of this research is achieved. An algorithm for detecting inconsistencies in observations of moving entities is designed, developed and tested on multiple test sets. The algorithm is designed in a modular way. Depending on the area and entity of interest, different modules can be implemented to evaluate entity movements. This is demonstrated for two different fields. The results show the potential of the algorithm in both military intelligence and crime analysis.

7.2 Recommendations

When inconsistencies have to be detected in entity movements, we recommend to use the designed framework proposed in this research. The modular nature of the framework causes the algorithm to be applicable in multiple ways for different scenarios. We recommend to take advantage of this modularity by applying the algorithm to more use-cases. Simultaneously, the potential and capabilities of the algorithm can be further investigated. A use-case one could think of is the tracking of suspected vehicles in both military intelligence and crime analysis. Another use-case for which the algorithm can be applied is alibi checking. During interrogations, information is collected about the alibis of suspects. Using the algorithm, there can be checked whether or not the information of multiple interrogations are consistent. Besides the consistency checking, the algorithm can also be adjusted to predictive tracking. Hypothetical future sightings can be added to the collection of historic sightings. In this way, the likelihood of future entity movements can be evaluated. At last, we recommend to keep in mind the lessons learned stated in the previous section, when adjusting the algorithm.

7.3 Future Research

The goal of this research is achieved, but further research is recommended to improve the performance of the algorithm. As already mentioned in the answers to the sub-questions, it might be useful to investigate other techniques for both the hypothesis evaluator and generator. Besides, the distance module used for both the simulation and real-world data can be improved. The distances used in the module are based on movements as the crow flies. In reality entities move across a road network. Further research is required to investigate how to incorporate a road network into the algorithm.

This research suggested two techniques to model dependent variables of route characteristics. Copula functions were implemented and tested in the algorithm. The algorithm was also tested without the usage of copula functions. Theoretically, the algorithm including copula functions should at least perform as good as the variant excluding copula functions. However, this is not demonstrated for all results. Further investigation is required to clarify this odd result. Furthermore, multivariate kernel density estimation can be implemented to model the dependence of variables.

The sightings used in this research were points defined by exact locations and time stamps. In reality, the location and time of a sighting may have a range. Generating hypotheses becomes more complex for this type of sightings. Instead of a fixed chronological order of sightings, the sightings might overlap. Investigating in a technique able to deal with overlapping sightings is recommended. Because then, the algorithm becomes even more widely applicable.

At TNO, the framework is now being further investigated. They are developing their own versions of the hypothesis evaluator and hypothesis generator, and apply the algorithm to new use-cases. Also some suggestions for future research mentioned in this section, are further investigated and implemented.

A Intelligence Cycle

A.1 Information Collection Methods

Acoustic Intelligence (ACINT)
Intelligence derived from the analysis of information about and from acoustic sources.

Geospatial Intelligence (GEOINT)
Intelligence derived from the analysis of geospatial and imagery information.

Human Intelligence (HUMINT)
Intelligence derived from the analysis of any type of information collected or provided by human sources.

Imagery Intelligence (IMINT)
Intelligence derived from the analysis and interpretation of imagery information.

Measurement and Signature Intelligence (MASINT)
Intelligence derived from the quantitative and qualitative analysis of scientific and technical information. This information is obtained by sensors able to identify characteristics of a target, emitting source or transmitter.

Medical Intelligence (MEDINT)
Intelligence derived from the analysis of medical, biomedical, epidemiological and environmental information.

Open Source Intelligence (OSINT)
Intelligence derived from the analysis of publicly accessible information e.g. radio, television, internet.

Signals Intelligence (SIGINT)
Intelligence derived from the analysis of information obtained in the electromagnetic spectrum. This includes Communications Intelligence (COMINT) and Electroninc Intelligence (ELINT).

A.2 Source and Information Reliability Matrix

Reliability of the source is designated by a letter between A and F signifying various degrees of confidence as indicated in Table 6.

Code	Valuation	Explanation
A	**Reliable**	**No doubt** of authenticity, trustworthiness, or competency; has a history of complete reliability.
B	**Usually reliable**	**Minor doubt** about authenticity, trustworthiness, or competency; has a history of valid information most of the time.
C	**Fairly reliable**	**Doubt** of authenticity, trustworthiness, or competency but has provided valid information in the past.
D	**Not usually reliable**	**Significant doubt** about authenticity, trustworthiness, or competency but has provided valid information in the past.
E	**Unreliable**	**Lacking** in authenticity, trustworthiness, and competency; history of invalid information.
F	**Cannot be judged**	**No basis** exists for evaluating the reliability of the source.

Table 6: *Evaluation of Source Reliability.* [3]

Credibility of information is designated by a numeral between 1 and 6 signifying various degrees of confidence as indicated in Table 7.

Code	Valuation	Explanation
1	**Confirmed**	**Confirmed** by other independent sources; **logical** in itself; **consistent** with other information on the subject.
2	**Probably true**	Not confirmed; **logical** in itself; **consistent** with other information on the subject.
3	**Possibly true**	Not confirmed; **reasonably logical** in itself; **agrees with some** other information on the subject.
4	**Doubtfully true**	Not confirmed; possible but **not logical**; **no other information** on the subject.
5	**Improbable**	Not confirmed; **not logical** in itself; **contradicted** by other information on the subject.
6	**Cannot be judged**	**No basis** exists for evaluating the validity of the information.

Table 7: *Evaluation of Information Content.* [3]

The resultant rating will be expressed in whatever combination or letter and number is appropriate. For example, information received from a “usually reliable” source which is adjusted as “probably true” will be rate as “B2”.

B Copula Families

Copula	$C_\theta(u,v)$	$\varphi_\theta(t)$	$\theta \in$
Clayton	$(max\{u^{-\theta}+v^{-\theta}-1;0\})^{\frac{-1}{\theta}}$	$\frac{1}{\theta}(t^{-\theta-1})$	$[-1,\infty)$
Frank	$-\frac{1}{\theta}\left[ln\left(1+\frac{(e^{-\theta u}-1)(e^{-\theta v}-1)}{e^{-\theta}-1}\right)\right]$	$-ln\left(\frac{e^{-\theta t}-1}{e^{-\theta}-1}\right)$	$(-\infty,\infty)$
Gumble	$\exp\{-((-ln(u))^{\theta}+(-ln(v))^{\theta})^{\frac{1}{\theta}}\}$	$(-ln(t))^{\theta})$	$[1,\infty)$

Table 8: *Three copula functions in the Archimedean family including their generator function $\varphi_\theta(t)$ and the parameter range.* [19]

C Simulation Data

C.1 Statistics Simulation Test Set 1

Table 9 shows the distribution of the number of sightings per collection in test set 1.

Nr. of sightings	Frequency
6	2
7	8
8	38
9	52

Table 9: *Distribution number of sightings per collection in test set 1.*

Table 10 shows the distribution of the number of sightings per collection in test set 1.

Nr. of false positives	Frequency
0	17
1	39
2	31
3	8
4	4
5	1

Table 10: *Distribution number of false positive sightings per collection in test set 1.*

C.2 Statistics Simulation Test Set 2

Table 11 shows the distribution of the number of sightings per collection in test set 2.

Nr. of sightings	Frequency
6	2
7	10
8	40
9	48

Table 11: *Distribution number of sightings per collection in test set 2.*

Table 12 shows the distribution of the number of sightings per collection in test set 2.

Nr. of false positives	**Frequency**
0	8
1	35
2	29
3	19
4	8
5	1

Table 12: *Distribution number of false positive sightings per collection in test set 2.*

D Real-world Data

D.1 Exploratory Data Analysis

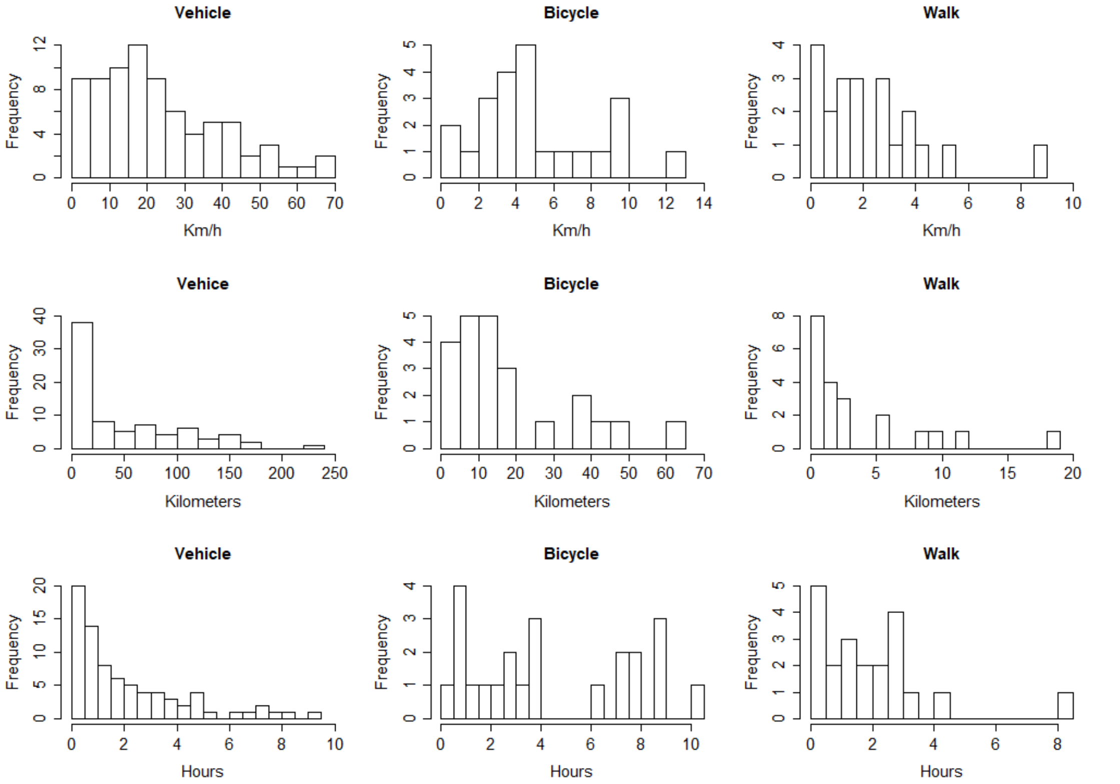


Figure 30: *Histograms of speed, distance and duration for move activities in the revised data set.*

D.2 Correlation Matrix

	Gas station	**Camp site**	**Hotel**	**Highway**	**Parking place**	**Restaurant**	**City**	**Station**
Gas station	1							
Camp site	0.0437	1						
Hotel	0.3956	0.0480	1					
Highway	0.5761	-0.0367	0.2125	1				
Parking place	0.4771	0.0546	0.3046	0.3219	1			
Restaurant	0.5519	0.1999	0.3230	0.2541	0.5817	1		
City	0.3644	-0.0969	0.2093	0.3237	0.3578	0.2824	1	
Station	0.3308	0.0861	0.3332	0.1610	0.4616	0.5440	0.1741	1

Table 13: *Correlation matrix of historic poi variables.*

Variable	**Distribution type**
Gas station	Exponential
Camp site	Gamma
Hotel	Weibull
Highway	Log Normal
Parking place	Gamma
Restaurant	Gamma
City	Weibull
Station	Log Normal

Table 14: *Estimated marginal distributions for the poi variables.*

Bibliography

[1] Santos Jr, E., Zhao, Q., Johnson, G., Nguyen, H., & Thompson, P. (2005). A cognitive framework for information gathering with deception detection for intelligence analysis. In *Proceedings of 2005 International Conference on Intelligence Analysis.*

[2] Joint Doctrine Publicatie 2 Inlichtingen; Ministerie van Defensie

[3] United States Army, Human Intelligence Collector Operations. Field Manual 2–22.3; FM 34–52. (Washington, DC: Headquarters, Department of the Army, 2006).

[4] Pherson, R. H. (2008). *Handbook of Analytic Tools & Techniques.* (pp. 13-14) Pherson Associates

[5] Heuer, R. J. (1999). *Psychology of intelligence analysis.* Center for Study of Intelligence, CIA, Washington, D.C.

[6] Hulnick, A. S. (2006). What's wrong with the Intelligence Cycle? In *Intelligence and national Security*, 21(6), (pp. 959-979).

[7] Johnston, J. M. & Johnston, R. (2005). Testing the intelligence cycle through systems modeling and simulation, In *Analytic culture in the U.S. intelligence community.* (pp. 45-57), Washington, DC: Central Intelligence Agency.

[8] Wheaton, K.J. (2012) Let's Kill the Intelligence Cycle. *Competitive Intelligence*, Vol 15, No 2, (pp. 9-24).

[9] Aydin, B., & Ozleblebici, Z. (2015). Is Intelligence Cycle Still Viable? In *International Conference on Military and Security Studies*, (pp. 95-100).

[10] Knopp, B. M., Beaghley, S., Frank, A., Orrie, R., & Watson, M. (2016). Defining the Roles, Responsibilities, and Functions for Data Science Within the Defense Intelligence Agency. *RAND National Defense Research Institute Santa Monica United States.*

[11] Zlotnick, J. (1972). Bayes' theorem for intelligence analysis. *Studies in Intelligence, 16(2)*, 43-52.

[12] Barros, A. I., van den Broek, A. C., van Dalen, J. A., Vecht, B., & Wevers, J. (2014). Producing near-real-time intelligence: predicting the world of tomorrow. *NL ARMS - Netherlands Annual Review of Military Studies, Optimal Deployment of Militairy Systems* (pp. 49-72), Asser Press, The Hague.

[13] Smit, S., van der Vecht, B., van Wermeskerken, F., & Streefkerk, J. W. (2016). QUIN: Providing Integrated Analysis Support to Crime Investigators. In *Intelligence and Security Informatics Conference (EISIC)* (pp. 120-123), IEEE.

[14] Ceolin, D., Van Hage, W. R., Schreiber, G., & Fokkink, W. (2013). Assessing trust for determining the reliability of information. In *Situation Awareness with Systems of Systems* (pp. 209-228). New York, N.Y.: Springer

[15] Josang, A. (2016). *Subjective logic.* New York, N.Y.: Springer.
doi: 10.1007/978-3-319-42337-1

[16] Pope, S., & Josang, A. (2005). Analysis of competing hypotheses using subjective logic. QUEENSLAND UNIV BRISBANE (AUSTRALIA).

[17] Weinbaum, C. & Shanahan, J.N.T (2018). Intelligence in a Data-Driven Age. In *Joint Force Quarterly* (pp. 4-9)

[18] Sklar, A. (1973). Random variables, joint distribution functions, and copulas. *Kybernetika*, 9(6), (pp. 449-460).

[19] Nelsen, R. B. (2007). An introduction to copulas. Springer Science & Business Media.

[20] Joe, H. (1996). Families of m-variate distributions with given margins and m (m-1)/2 bivariate dependence parameters. *Lecture Notes-Monograph Series*, (pp. 120-141).

[21] Aas, K., C. Czado, A. Frigessi, and H. Bakken (2009). Pair-copula constructions of multiple dependence. *Insurance: Mathematics and Economics*, 44(2), (pp. 182-198).

[22] Bedford, T., & Cooke, R. M. (2002). Vines: A new graphical model for dependent random variables. *Annals of Statistics*, (pp. 1031-1068).

[23] Haff, I. H., Aas, K., & Frigessi, A. (2010). On the simplified pair-copula construction – Simply useful or too simplistic? *Journal of Multivariate Analysis*, 101(5), (pp. 1296-1310).

[24] Marius Hofert, Ivan Kojadinovic, Martin Maechler and Jun Yan (2017). copula: Multivariate Dependence with Copulas. R package version 0.999-18 URL: https://CRAN.R-project.org/package=copula

[25] Ulf Schepsmeier, Jakob Stoeber, Eike Christian Brechmann, Benedikt Graeler, Thomas Nagler and Tobias Erhardt (2018). VineCopula: Statistical Inference of Vine Copulas. R package version 2.1.4. URL: https://CRAN.R-project.org/package=VineCopula

[26] Tarn Duong (2017). ks: Kernel Smoothing. R package version 1.10.7. URL: https://CRAN.R-project.org/package=ks