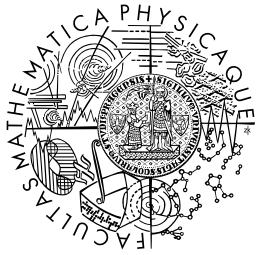


FACULTY
OF MATHEMATICS
AND PHYSICS
Charles University

**MASTER THESIS**

Bc. Ondřej Komora

# Scenario generation methods for discrete data

Department of Probability and Mathematical Statistics

Supervisor of the master thesis: Ing. Vít Procházka, Ph.D.

Study programme: Probability, Mathematical Statistics and Econometrics

Study branch: Econometrics

Prague 2024

I declare that I carried out this master thesis independently, and only with the cited sources, literature and other professional sources. It has not been used to obtain another or the same degree.

I understand that my work relates to the rights and obligations under the Act No. 121/2000 Sb., the Copyright Act, as amended, in particular the fact that the Charles University has the right to conclude a license agreement on the use of this work as a school work pursuant to Section 60 subsection 1 of the Copyright Act.

In ..... date .....  .....

Author's signature

I would like to thank the supervisor Ing. Vít Procházka, Ph.D. for valuable feedback, insights, and inspiration which helped me shape the thesis into its final form.

Title: Scenario generation methods for discrete data

Author: Bc. Ondřej Komora

Department: Department of Probability and Mathematical Statistics

Supervisor: Ing. Vít Procházka, Ph.D., Department of Probability and Mathematical Statistics

Abstract: Stochastic optimization relies heavily on scenario generation, which has a large impact on the tractability of optimization methods and the quality of obtained solutions. Despite its importance, scenario generation for discrete data is rarely studied and even when it is, it often involves a problem-oriented method. However, the development of these methods is resource-intensive, resulting in a situation where viable easy-to-use alternatives to sampling are missing. In this work, we attempt to remedy the situation by proposing a new copula-based scenario generation method for discrete data. The method is based on extending discrete random variables and subsequent use of the so-called extension copula. We demonstrate the effectiveness of this method on the stochastic knapsack problem by using several metrics like in-sample stability, out-of-sample evaluation gap, and optimality gap. The results show that our method outperforms sampling and can serve as a more challenging benchmark for problem-oriented methods.

Keywords: stochastic optimization, scenario generation, discrete data, copula

# Contents

# Introduction

Scenario generation methods play a crucial role in stochastic optimization. The choice of scenario generation method has a large impact on the tractability of optimization methods and the quality of obtained solutions. A large amount of scenarios in stochastic optimization problems may render solving these problems impossible. This is especially the case for optimization methods with exponentially increasing complexity, as an example we may take the branch-and-bound algorithm for solving mixed integer programs, or multi-stage stochastic programs. In such cases, it is necessary to condense information about the randomness, which can be represented via historical data or exact distributions, into as few scenarios as possible. This process, however, is not easy. There are several ways the scenario generation may go wrong. For example, the newly created scenarios may provide biased estimates of the "true" value of the objective function, or subsequent optimization may produce solutions that are far from optimal in terms of expectation or any other selected objective. Careful assessments of quality are therefore necessary when evaluating a scenario generation method.

The scenario generation methods can be broadly classified into two classes: *distribution-oriented* and *problem-oriented.* Distribution-oriented methods create scenarios solely by taking into consideration the underlying distribution of randomness in stochastic problems. Examples of distribution-oriented methods are sampling, moment-matching (Høyland et al. [2003]), minimizing Kantorovich-Rubinstein distance (Pflug and Pichler [2012]), or copula-based methods (Kaut [2014]).

However, neglecting the optimization problem may not always be appropriate. For example, in portfolio optimization problems, distribution-oriented methods struggle to represent rare events using scenarios. This is an undesirable behavior because rare events are the main source of risks associated with portfolio selection. A good problem-oriented method considers this and creates scenarios specifically tailored for each optimization problem. Therefore, it is expected that a well-constructed problem-oriented method will always outperform any distribution-oriented method. For an example of a successful problem-oriented method for scenario generation in portfolio selection problems, we refer to Fairbrother et al. [2018]. Another example of a problem-oriented method in the literature is presented in Prochazka and Wallace [2020] where scenario sets are obtained by minimizing the discrepancy between so-called *in-sample* and *out-of-sample* evaluations on a pool of heuristic solutions.

Despite the obvious advantages of problem-oriented methods, they require a significant investment of time, knowledge, and resources to develop. Due to these reasons, easy-to-use distribution-oriented methods like sampling are widely used when solving stochastic optimization problems, and even when the ultimate goal is to use a problem-oriented approach, sampling and other simple methods are used as benchmarks on which the approaches are compared. In this context, we believe that the development of easy-to-use distribution-oriented methods is still sensible.

Scenario generation for discrete data is usually considered a difficult problem and despite many real-world problems where discrete data naturally appears, for

example network failures (Ball et al. [1995]) and stochastic customers (Bent and Van Hentenryck [2004]), it is relatively rarely studied. Even when scenario generation for discrete data is studied, a problem-oriented approach is often used, causing problems with complicated development. Also, most distribution-oriented methods are constructed for continuous random variables. As a consequence, to the extent of our knowledge, there are no viable easy-to-use alternatives to sampling when discrete data is involved.

In this thesis, we propose a new distribution-oriented scenario generation method for discrete data. The method is based on a copula-based method introduced in Kaut [2014]. This method was proven successful for continuous data as shown by Zhang et al. [2021] on the stochastic shortest path problem in real road networks. Unfortunately, the method struggles to create reasonable scenarios when discrete data is encountered. We shall demonstrate this fact in Section 3.1. The principal idea of the newly proposed method is to *extend* discrete random variables into continuous ones, and subsequently use so-called *extension copula* in the method proposed by Kaut [2014].

The thesis is structured as follows. Chapter 1 starts with an overview of copula theory. We introduce copulas and Sklar's theorem. Then, we define so-called *discrete extensions* and provide some related theoretical properties. We finish this chapter by deriving the expression for extension copula, defined as a copula of a random vector with extended discrete margins. The summary of the method from Kaut [2014] is presented in Chapter 2. In Chapter 3 we describe in detail the adjustments we propose to the aforementioned method. We start by providing an example of why the adjustments are necessary and why the base method fails to provide reasonable scenarios in discrete cases. Then we motivate and describe these adjustments in detail. We also provide a discussion about the practical aspects of using the algorithm and about potential difficulties. The content of Chapter 4 is a case study in which we demonstrate the effectiveness of the newly proposed method using several metrics and assessments of quality, which we describe in detail. The comparison will be performed with sampling and one problem-oriented method. We conclude the thesis in the Conclusion chapter, discussing the drawbacks and advantages of the developed method, together with the potential areas of further research.

# Preliminaries

In this thesis, we work exclusively with real-valued random variables. We shall work extensively with distribution functions, which we define for a random variable $X$ as $F_X(t) = P(X \leq t), t \in \mathbb{R}$, and $F_X(-\infty) = 0, F_X(\infty) = 1$. The range of distribution function $F_X$ is defined as

$$\operatorname{Ran} F_X := \left\{ u \in [0,1] \,\middle|\, \exists t \in \overline{\mathbb{R}} : F_X(t) = u \right\}.$$

Observe that $\{0,1\} \subseteq \operatorname{Ran} F_X$ with this definition. Also, we need to define quantile functions as $F_X^{-1}(u) = \inf\{t \in \mathbb{R} : F_X(t) \geq u\}$. Notice that $F_X^{-1}(0) = -\infty$ and $F_X^{-1}(1) = \sup(\operatorname{supp} X)$. Additionally, we define joint distribution function $F$ of vector $(X_1, \ldots, X_n)$ as

$$F(t_1, \ldots, t_n) = P(X_1 \leq t_1, \ldots, X_n \leq t_n)$$

where $t_1, \ldots, t_n \in \overline{\mathbb{R}}$.

A support of a random variable $X$, denoted by $\operatorname{supp} X$, is the smallest closed set $R_X \in \mathcal{B}$ satisfying $P(X \in R_X) = 1$. We define continuous random variables as random variables which have continuous distribution functions. For discrete random variables we impose that support is countable. Furthermore, we assume that the support of a discrete variable $X$ satisfies $\operatorname{supp} X = \mathbb{N}_0$ for an infinite number of realizations, or $\operatorname{supp} X = \{0, \ldots, K\}$ for some $K \in \mathbb{N}$. Discrete random vectors have all margins discrete, mixed random vectors have at least one margin discrete and at least one continuous, and continuous random vectors have all margins continuous. Cases where the random variable is neither discrete nor continuous are not considered in this thesis.

For the purposes of this thesis, we define the median of a random variable $X$ as $\operatorname{med}(X) := F_X^{-1}\left(\frac{1}{2}\right)$ even when the $X$ is discrete, ensuring that the median is always integral. Finally, for $x \in \mathbb{R}$ we define the lower integer part as $\lfloor x \rfloor$ and the upper integer part as $\lceil x \rceil$.

# 1. Copulas and discrete extensions

## 1.1 Introduction to copulas

The principal idea of copulas is to model relationships between margins of random vectors without taking into consideration their actual distributions. Probably the simplest random vector on which we can measure dependence is a random vector with uniform margins.

**Definition 1.1** (Copula)**.** *Let $(U_1, \dots, U_n)$ be a random vector with uniform margins on interval $[0,1]$. Copula is a function $[0,1]^n \to [0,1]$ such that*

$$C(u_1, \dots, u_n) = P(U_1 \leq u_1, \dots, U_n \leq u_n).$$

*In other words, copula $C$ is distribution function of random vector $(U_1, \dots, U_n)$.*

Arguably the most important result in copula theory is Sklar's theorem first presented in Sklar [1959]. The theorem states that every random vector can be decomposed into marginal distributions and copula.

**Theorem 1.1** (Sklar's theorem)**.** *Let $F$ be a joint distribution function of random vector $X = (X_1, \dots, X_n)$. Then there exists copula $C$ such that for $t_1, \dots, t_n \in \overline{\mathbb{R}}$ it holds*

$$F(t_1, \dots, t_n) = C\left(F_{X_1}(t_1), \dots, F_{X_n}(t_n)\right). \tag{1.1}$$

*Copula $C$ is uniquely determined on $\bigtimes_{i=1}^{n} \operatorname{Ran} F_{X_i}$. Alternatively, this equation could be expressed in terms of marginal quantile functions as*

$$C(u_1, \dots, u_n) = F\left(F_{X_1}^{-1}(u_1), \dots, F_{X_n}^{-1}(u_n)\right), \tag{1.2}$$

*where $(u_1, \dots, u_n) \in \bigtimes_{i=1}^{n} \operatorname{Ran} F_{X_i}$.*

*Proof.* See for example Moore and Spruill [1975], Nelsen [2006], or Durante et al. [2013]. □

*Remark.* From now onwards, whenever we say that copula $C$ satisfies Sklar's theorem for random vector $X = (X_1, \dots, X_n)$, we mean that the copula satisfies (1.1). With reference to Sklar's theorem, all such copulas $C$ are uniquely defined on $\bigtimes_{i=1}^{n} \operatorname{Ran} F_{X_i}$.

Notice that when all margins of $X$ are continuous, vector

$$(F_{X_1}(X_1), \dots, F_{X_n}(X_n))$$

has uniformly distributed margins $[0,1]$ and copula $C$ is the distribution function of this vector. Therefore, the information about margins of $X$ is removed when computing the right-hand side of (1.1). In this context, the theorem allows us to isolate dependencies within the random vectors using copulas. Broad classes of random vectors can have the same dependence structure but differ in their

marginal distributions. Sklar's theorem allows us to model marginal distributions and dependence structures independently.

The theorem states that the copulas satisfying Sklar's theorem are uniquely determined on $\bigtimes_{i=1}^{n} \operatorname{Ran} F_{X_i}$. If the vector $X$ is continuous, the region becomes $[0,1]^n$ and there exists only one copula satisfying Sklar's theorem. On the other hand, if margins of $X$ are discrete, the copula is only uniquely defined on a grid of points. In these cases, there could be infinitely many copulas satisfying Sklar's theorem. The question that naturally arises is how to deal with these situations. One of the possible solutions is to use extensions of discrete random variables. The purpose of extensions is to make discrete random variables continuous.

## 1.2 Discrete extensions

The following definition appeared in Denuit and Lambert [2005] or in Genest and Nešlehová [2007] for discrete random vectors, but we extend it for general mixed random vectors.

**Definition 1.2.** *Assume that $X$ is a discrete random variable and $U$ is a continuous random variable on $[0,1]$ which is independent of $X$. Furthermore, assume that $U$ has a strictly increasing distribution function on the unit interval. Then we define the extension of $X$ as a random variable $X^* := X + U - 1$.*

*Let $Z = (X_1, \ldots, X_n)$ be a mixed random vector and let $U = (U_1, \ldots, U_n)$ be a random vector independent of $Z$ with independent continuous margins on $[0,1]$. Assume that marginal distribution functions of $U$ are strictly increasing on unit intervals. Then for $Z$ we define its extension as a random vector $Z^* = (Y_1, \ldots, Y_n)$ where*

$$Y_j = \begin{cases} X_j + U_j - 1 & \text{if } X_j \text{ is discrete,} \\ X_j & \text{if } X_j \text{ is continuous.} \end{cases}$$

*In other words, we replace all discrete margins of $Z$ with their extensions.*

By extending a discrete random variable, we obtain a continuous one. A similar assertion holds for mixed and discrete random vectors, which get transformed into continuous random vectors. The choice of extensions with strictly increasing distribution functions will be explained later.

Before investigating the properties of extended discrete random variables, we need to define the following.

**Definition 1.3.** *Let $X$ be a discrete random variable and $u \in [0,1]$. Then for $u \notin \operatorname{Ran} F_X$ we define*

$$\underline{F}_X^{-1}(u) = \sup\{n \in \mathbb{Z} : F_X(n) \leq u\},$$
$$\overline{F}_X^{-1}(u) = \inf\{n \in \mathbb{Z} : F_X(n) \geq u\}$$

*and otherwise $\underline{F}_X^{-1}(u) = \overline{F}_X^{-1}(u) = F_X^{-1}(u)$. We call $\underline{F}_X^{-1}(u)$ lower quantile and $\overline{F}_X^{-1}(u)$ upper quantile. Additionally, we define the lower step $u_X^-$ and the upper step $u_X^+$ as*

$$u_X^- = F_X(\underline{F}_X^{-1}(u)),$$
$$u_X^+ = F_X(\overline{F}_X^{-1}(u)).$$

*Observation.* Some observations immediately follow from Definition 1.3.

1. The lower and upper steps can coincide, i.e. $u_X^- = u_X^+$. This occurs whenever there exists $n \in \mathbb{Z}$ such that $F_X(n) = u$, or in other words $u \in \operatorname{Ran} F_X$. Then it also follows $u_X^- = u_X^+ = u$.

2. If $u \notin \operatorname{Ran} F_X$, and we observe $u_X^-, u_X^+ \in \operatorname{Ran} F_X$, according to the definition above we can obtain whenever $u_X^- > 0$

$$F_X^{-1}\left(u_X^-\right) = F_X^{-1}\left(F_X(\underline{F}_X^{-1}(u))\right) = \underline{F}_X^{-1}(u).$$

   We can derive $F_X^{-1}\left(u_X^+\right) = \overline{F}_X^{-1}(u)$ similarly.

*Remark.* In the definition above, we used the name of the random variable as a bottom-right index. We will sometimes omit this index if it is clear from the context to which random variable it is associated.

To better grasp Definition 1.3, consider a discrete random variable $X$ defined as

$$P(X = n) = \begin{cases} 0.3 & n = 0, \\ 0.15 & n = 1, \\ 0.1 & n = 2, \\ 0.4 & n = 3, \\ 0.05 & n = 4, \\ 0 & \text{otherwise.} \end{cases} \tag{1.3}$$

For $u = 0.65$ it can be easily computed $\underline{F}_X^{-1}(u) = 2, \overline{F}_X^{-1}(u) = 3, u_X^- = 0.55$ and $u_X^+ = 0.95$. These values, along with the distribution function of $X$, are plotted in Figure 1.1. The figure can be interpreted as follows. The lower quantile is the highest possible realization $n$ with a non-zero probability such that the probability $P(X \leq n)$ does not exceed $u$. Similarly, the upper quantile is the lowest possible realization $n$ with non-zero probability such that $P(X \leq n) \geq u$. A similar interpretation could be used for the lower and upper steps. The lower and upper steps are the highest and lowest numbers of set $\operatorname{Ran} F_X$, such that those steps bound $u$. More formally, the lower step $u_X^-$ is the supremum of set $\{v \in \operatorname{Ran} F_X : v \leq u\}$, and the upper step $u_X^+$ is the infimum of $\{v \in \operatorname{Ran} F_X : v \geq u\}$.

Let us now investigate the properties of extended discrete random variables. We begin by demonstrating that the distribution function of an extended discrete random variable can be viewed as an interpolation of the steps of a discrete distribution function.

**Theorem 1.2.** *Let $X$ be a discrete random variable and $X^* = X + U - 1$ its extension. Then for the distribution function of $X^*$, it holds*

$$F_{X^*}(t) = F_X(\lfloor t \rfloor) + F_U(t - \lfloor t \rfloor) \cdot [F_X(\lceil t \rceil) - F_X(\lfloor t \rfloor)], \ t \in \mathbb{R}.$$

*Proof.* The theorem follows from a straight-forward calculation

$$F_{X^*}(t) = P(X + U - 1 \leq t) = \sum_{n \in \operatorname{supp} X} P(U \leq t + 1 - n | X = n) P(X = n) = \sum_{n \in \operatorname{supp} X : n \leq \lfloor t \rfloor} P(X = n) + F_U(t - \lfloor t \rfloor) P(X = \lceil t \rceil),$$

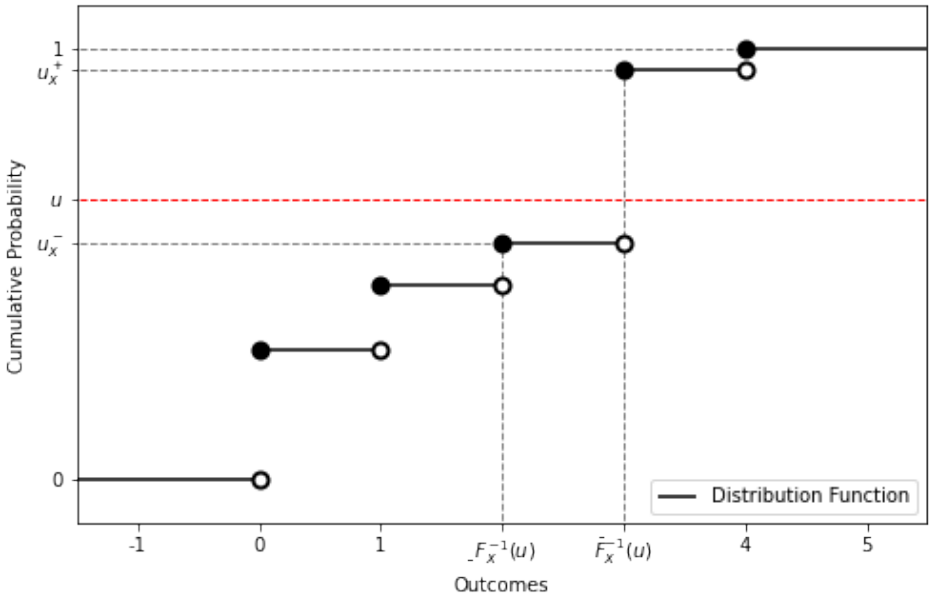



Figure 1.1: Visualization of Definition 1.3.

where we utilized independence of $U$ from $X$ and

$$P(U \leq t+1-n) = \begin{cases} 1 & n \leq \lfloor t \rfloor, \\ F_U(t - \lfloor t \rfloor) & n = \lceil t \rceil, t \notin \mathbb{Z}, \\ 0 & n > \lceil t \rceil. \end{cases}$$

Now we just need to observe

$$F_X(\lfloor t \rfloor) = \sum_{n \in \operatorname{supp} X : n \leq \lfloor t \rfloor} P(X = n)$$

and $P(X = \lceil t \rceil) = F_X(\lceil t \rceil) - F_X(\lfloor t \rfloor)$. By combining these observations we have proved the theorem. □

We defined extension in a general manner, meaning that we can use any continuous distribution on $[0, 1]$ independent of $X$, which has a strictly increasing distribution function. A natural choice might be the uniform distribution on $[0, 1]$. The uniform extension of discrete random variable defined with (1.3) is shown in Figure 1.2. The interpolation of a discrete distribution function is visible. This is a general property of all extensions. They all interpolate steps of distribution functions, although the interpolation might be non-linear, depending on the chosen distribution of the extension.

Similarly to the extended distribution function, the extended quantile function is also an interpolation of the discrete quantile function.

**Theorem 1.3.** *Consider a discrete random variable $X$ and its extension $X^* = X + U - 1$. Fix $u \in [0, 1]$, then the quantile function of $X^*$ can be computed as*

$$F_{X^*}^{-1}(u) = \begin{cases} F_X^{-1}(u) & u \in \operatorname{Ran} F_X, \\ \underline{F}_X^{-1}(u) + F_U^{-1}\left(\frac{u - u_X^-}{u_X^+ - u_X^-}\right) & u \notin \operatorname{Ran} F_X. \end{cases} \tag{1.4}$$

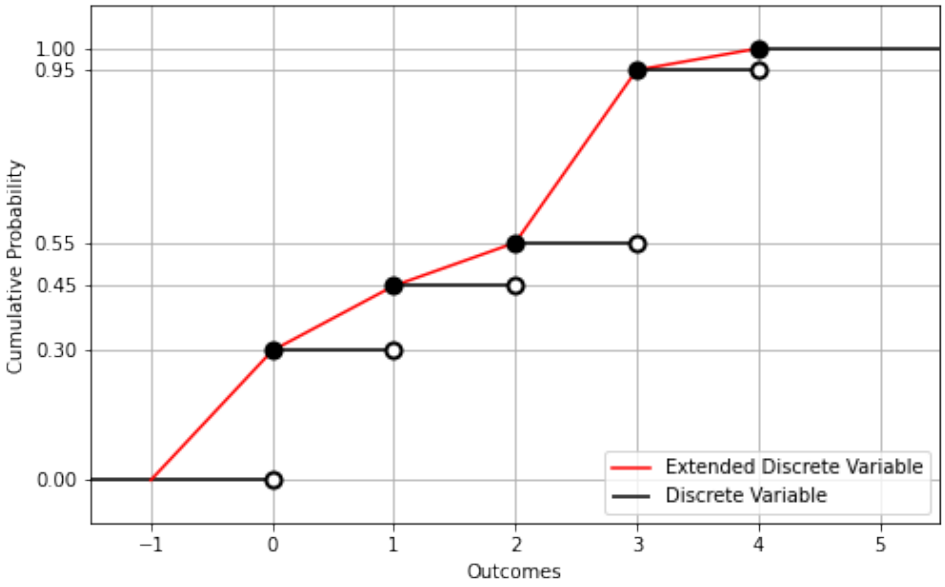



Figure 1.2: Comparison of a discrete distribution function and its extension.

*Proof.* When $u = 0$, the theorem follows trivially from the definition of a quantile function. Also, when $u = 1$, from Theorem 1.2 we have $F^{-1}_{X^*}(1) = \sup(\operatorname{supp} X) = F^{-1}_X(1)$.

Now consider $u \in (0, 1)$. According to Theorem 1.2, the $X^*$ has strictly increasing continuous distribution function on the interval $(-1, \sup(\operatorname{supp} X))$. The image of this set, when projected by $F_{X^*}$, is $(0, 1)$. Therefore it suffices to find a unique $t$ such that $F_{X^*}(t) = u$.

Case $u \in \operatorname{Ran} F_X, u \notin \{0, 1\}$, follows easily, because in such case there exists a uniquely defined $t \in \mathbb{N}_0$ such that $F_{X^*}(t) = F_X(t) = u$ and the equality of quantiles follows.

Now consider the case when $u \notin \operatorname{Ran} F_X$. We can observe according to Theorem 1.2 that $t \notin \mathbb{N}_0$ and

$$\underline{F}^{-1}_X(u) = \sup\{n \in \mathbb{Z} : F_X(n) \leq F_{X^*}(t)\} = \lfloor t \rfloor$$

and similarly $\overline{F}^{-1}_X(u) = \lceil t \rceil$. Then it also follows $u^-_X = F_X(\lfloor t \rfloor)$ and $u^+_X = F_X(\lceil t \rceil)$. With use of Theorem 1.2 we get

$$u = F_{X^*}(t) = u^-_X + F_U\left(t - \underline{F}^{-1}_X(u)\right) \cdot (u^+_X - u^-_X).$$

The theorem can be then easily derived using the fact that $F_U$ is continuous and strictly increasing, and therefore $F^{-1}_U$ exists and is uniquely defined. □

The reason why we required extension $U$ to have a strictly increasing distribution function becomes clear now. If $F_U$ was not continuous and strictly increasing, $F^{-1}_U$ would not be uniquely defined and the extended quantile function would not be calculable.

The principal idea of the algorithm we shall describe in Chapter 3 is the replacement of discrete variables with their extensions. At this point, we fore-

shadow that the algorithm works with pairs of margins and their associated copula. More specifically, if we have input random vector $Z = (X_1, \ldots, X_n)$, the algorithm works with margin pairs $(X_i, X_j), 1 \leq i < j \leq n$. We might wonder how extension affects the dependence structure and associations within these margin pairs. Take probabilistic definitions of Kendall's and Spearman's rank correlation coefficients from Nelsen [2006].

**Definition 1.4.** *Let $(X, Y)$ be a pair of random variables and define $\left(X^{(c)}, Y^{(c)}\right)$ as an independent vector from $(X, Y)$, but with the same distribution. Additionally, let $\left(X^{(c_1)}, Y^{(c_2)}\right)$ be an independent random vector from $(X, Y)$, but with the same marginal distributions that are mutually independent. Then we define Kendall's $\tau$ and Spearman's $\rho$ of $(X, Y)$ as*

$$\tau_{X,Y} = P\left[(X - X^{(c)})(Y - Y^{(c)}) > 0\right] - P\left[(X - X^{(c)})(Y - Y^{(c)}) < 0\right],$$
$$\rho_{X,Y} = 3\left\{P\left[(X - X^{(c_1)})(Y - Y^{(c_2)}) > 0\right] - P\left[(X - X^{(c_1)})(Y - Y^{(c_2)}) < 0\right]\right\}.$$

*Remark.* In the presence of a sample $(x_i, y_i), i = 1, \ldots, m$, from the distribution of vector $(X, Y)$, estimates of these coefficients attain more familiar forms

$$\hat{\tau} = \frac{\sum_{i=1}^{m} \mathbf{1}[x_i > y_i] - \mathbf{1}[x_i < y_i]}{m(m-1)/2},$$
$$\hat{\rho} = 1 - \frac{6\sum_{i=1}^{m} d_i^2}{m(m^2-1)},$$

where $d_i = r(x_i) - r(y_i)$. We denote $r(x_i)$ the index of $x_i$ in the sequence $\{x_i\}_{i=1}^{m}$ sorted in ascending order. Similiary, we define $r(y_i)$.

Interestingly, these coefficients get preserved when the pair $(X, Y)$ is extended.

**Theorem 1.4.** *Let $(X, Y)$ be a pair of discrete random variables. Then Kendall's $\tau$ and Spearman's $\rho$ coefficients are preserved when $(X, Y)$ is extended.*

*Proof.* When both $X$ and $Y$ are continuous, there is nothing to prove since the extension has the same distribution as $(X, Y)$. If $X$ and $Y$ are discrete, the proof can be found in Denuit and Lambert [2005] or Mesfioui and Tajar [2005]. The final option is if $X$ is discrete and $Y$ is continuous, or vice-versa. However, the proof from Denuit and Lambert [2005] can be easily adjusted to this case. □

This result is particularly important in terms of scenario generation. It is vital to capture dependencies in the data when generating scenarios. If we use extensions of discrete random variables, we have a guarantee of preserving dependencies and associations, at least in terms of Kendall's and Spearman's rank correlation coefficients.

The properties of extensions go even further, as they preserve a wide variety of so-called *concordance orders.* An example of concordance order is *positive quadrant dependence* (PQD). We say that $(X, Y)$ is *positive quadrant dependent* if for $s, t \in \mathbb{R}$ it holds

$$P(X \leq s)P(Y \leq t) \leq P(X \leq s, Y \leq t).$$

This property is preserved as well by extending discrete random variables. In other words, if $(X, Y)$ is PQD, its extension is also PQD. This claim was proven

for the bivariate case in Denuit and Lambert [2005], but a generalization for the mixed case is straightforward. For more examples of concordance orders that are preserved by extending a discrete random pair, we refer to Genest and Nešlehová [2007]. We believe these results can be generalized for mixed random pairs, but a formal proof would have to be provided.

Having established the basic properties of extended discrete random variables, we might focus on the properties of the copula of an extended random vector, which we call *extension copula*. The next section provides an explicit formula for its calculation.

## 1.3 Extension copula

The ultimate goal of this section is to obtain a closed expression for the extension copula. An explicit equation for extension copula is presented in Genest et al. [2014] for random vectors with all margins discrete. Unfortunately, detailed derivation is missing. In Denuit and Lambert [2005] and Nelsen [2006], we have seen an analytical description of extension copula for the bivariate case with discrete margins. However, we have not encountered an explicit formula for extension copula of mixed random vectors, let alone its derivation.

We consider our contribution in this section two-fold. Firstly, we adjust the formula for extension copula to the case of a general mixed random vector with any number of margins. Secondly, we present our derivation of extension copula, which was inspired by proof in Denuit and Lambert [2005] for the bivariate case of discrete margins. However, this proof had to be significantly updated and expanded for a general mixed random vector case. During the process, we also use some of the notation from Genest et al. [2014].

The following theorem serves as a staging point for the derivation of an explicit formula for extension copula. It is a generalization of Theorem 1.2 for extended random vectors.

**Theorem 1.5** (Distribution function of extended random vector)**.** *Consider a mixed random vector $Z = (X_1, \dots, X_k, Y_1, \dots, Y_p)$ with $X_i$ being continuous and $Y_j$ discrete margins. We choose an extension of $Z$ in the form*

$$Z^* = (X_1, \dots, X_k, Y_1 + U_1 - 1, \dots, Y_p + U_p - 1).$$

*Denote distribution functions of vectors $Z$ and $Z^*$ as $F$ and $F^*$, respectively. Let $x_1, \dots, x_k \in \overline{\mathbb{R}}$ and $y_1, \dots, y_p \in \mathbb{R}$. Then $F^*$ can be calculated using*

$$F^*(x_1, \dots, x_k, y_1, \dots, y_p) = \sum_{S \subseteq \{1,\dots,p\}} F\left(x_1, \dots, x_k, y_1^S, \dots, y_p^S\right) \prod_{i \in S} F_{U_i}(y_i - \lfloor y_i \rfloor) \prod_{j \notin S} (1 - F_{U_j}(y_j - \lfloor y_j \rfloor)), \tag{1.5}$$

*where we define $y_i^S = \lceil y_i \rceil$ if $i \in S$ and $y_i^S = \lfloor y_i \rfloor$ if $i \notin S$.*

*Proof.* Notice that whenever for any $i$ it holds $x_i = -\infty$, then

$$F^*(x_1, \dots, x_k, y_1, \dots, y_p) = F\left(x_1, \dots, x_k, y_1^S, \dots, y_p^S\right) = 0$$

and the theorem holds. Hence, we may assume $x_i \neq -\infty$. We prove this theorem by induction with respect to a number of extended discrete variables.

*Base case* $p = 1$. Let us introduce a new notation to increase readability. Write $W = (X_1, \ldots, X_k)$ and $B = \bigtimes_{i=1}^{k} B_i$, where $B_i$ is either $\mathbb{R}$ if $x_i = \infty$ or $(-\infty, x_i]$ if $x_i \in \mathbb{R}$. Utilizing law of total probability, $Y_1^* = Y_1 + U_1 - 1$, and fact that $U_1$ is independent of $W$ and $Y_1$, we derive

$$P(W \in B, Y_1^* \leq y_1) = \sum_{n \in \operatorname{supp} Y_1} P(W \in B, U_1 \leq y_1 + 1 - n | Y_1 = n) P(Y_1 = n) = \sum_{n \in \operatorname{supp} Y_1} P(W \in B | Y_1 = n) P(Y_1 = n) P(U_1 \leq y_1 + 1 - n).$$

Similarly as in the proof of Theorem 1.2, separation into individual cases yields

$$P(U_1 \leq y_1 + 1 - n) = \begin{cases} 1 & n \leq \lfloor y_1 \rfloor, \\ F_{U_1}(y_1 - \lfloor y_1 \rfloor) & n = \lceil y_1 \rceil, y_1 \notin \mathbb{Z}, \\ 0 & n > \lceil y_1 \rceil. \end{cases}$$

Using this, we can continue with

$$P(W \in B, Y_1^* \leq y_1) = P(W \in B, Y_1 \leq \lfloor y_1 \rfloor) + F_{U_1}(y_1 - \lfloor y_1 \rfloor) \cdot P(W \in B, Y_1 = \lceil y_1 \rceil). \quad (1.6)$$

Finally, if we realize that

$$P(W \in B, Y_1 = \lceil y_1 \rceil) = P(W \in B, Y_1 \leq \lceil y_1 \rceil) - P(W \in B, Y_1 \leq \lfloor y_1 \rfloor)$$

and combine it with (1.6), we get

$$P(W \in B, Y_1^* \leq y_1) = P(W \in B, Y_1 \leq \lceil y_1 \rceil) \cdot F_{U_1}(y_1 - \lfloor y_1 \rfloor) + P(W \in B, Y_1 \leq \lfloor y_1 \rfloor) \cdot (1 - F_{U_1}(y_1 - \lfloor y_1 \rfloor)). \quad (1.7)$$

Since

$$P(W \in B, Y_1 \leq t) = F(x_1, \ldots, x_k, t),$$

this is exactly (1.5). In base case $p = 1$, we are summing in (1.5) over subsets of $\{1\}$. The first summand in (1.7) is the case when $S = \{1\}$, and the second summand is the case when $S = \varnothing$.

*Induction step* $p - 1 \to p$. We define sets $B_i$ in the same way as in the proof of the base case and denote $W = (X_1, \ldots, X_k, Y_1^*, \ldots, Y_{p-1}^*)$ and $B = \left(\bigtimes_{i=1}^{k} B_i\right) \times \left(\bigtimes_{j=1}^{p-1} (-\infty, y_j]\right)$. Using the same arguments as when we derived (1.7), we obtain

$$P(W \in B, Y_p^* \leq y_p) = P(W \in B, Y_p \leq \lceil y_p \rceil) \cdot F_{U_p}(y_p - \lfloor y_p \rfloor) + P(W \in B, Y_p \leq \lfloor y_p \rfloor) \cdot (1 - F_{U_p}(y_p - \lfloor y_p \rfloor)) \quad (1.8)$$

If we combine $W$ and $Y_p$ we get a random vector with $p - 1$ extended discrete variables. Therefore we may use induction assumption and obtain

$$P(W \in B, Y_p \leq \lceil y_p \rceil) \cdot F_{U_p}(y_p - \lfloor y_p \rfloor) = \sum_{S \subseteq \{1, \ldots, p\} : p \in S} F\left(x_1, \ldots, x_k, y_1^S, \ldots, y_{p-1}^S, y_p^S\right) \cdot \prod_{i \in S} F_{U_i}(y_i - \lfloor y_i \rfloor) \prod_{j \notin S} (1 - F_{U_j}(y_j - \lfloor y_j \rfloor))$$

for the first summand in (1.8). We can obtain a similar expression for the second summand, except we would sum over subsets that do not contain $p$. We now have two sums: in one we sum over subsets of $\{1,\dots,p\}$ which contain $p$, and in other we sum over subsets of the same set which do not contain $p$. Adding these together we sum over all subsets of $\{1,\dots,p\}$ and we get (1.5) as desired. □

Now we may proceed to the derivation of an explicit formula for extension copula.

**Theorem 1.6.** *Consider a mixed random vector $Z = (X_1,\dots,X_k,Y_1,\dots,Y_p)$ with $X_i$ being continuous and $Y_j$ discrete margins. We choose an extension of $Z$ in the form*

$$Z^* = (X_1,\dots,X_k,Y_1+U_1-1,\dots,Y_p+U_p-1).$$

*Let $u_1,\dots,u_k,v_1,\dots v_p \in [0,1]$ and $v_j^-, v_j^+$ be lower and upper steps associated with $v_j$ and $Y_j$. Also let $C$ be a copula satisfying Sklar's theorem for vector $Z$. Then we may express copula $C^*$ of extended vector $Z^*$ as*

$$C^*(u_1,\dots,u_k,v_1,\dots,v_p) = \sum_{S\subseteq\{1,\dots,p\}} C\left(u_1,\dots,u_k,v_1^S,\dots,v_p^S\right)\prod_{i\in S}\lambda_i(v_i)\prod_{j\notin S}(1-\lambda_j(v_j)),$$

*where*

$$\lambda_i(v_i) = \begin{cases} \frac{v_i-v_i^-}{v_i^+-v_i^-} & v_i \notin \operatorname{Ran} F_{Y_i}, \\ 0 & v_i \in \operatorname{Ran} F_{Y_i}, \end{cases}$$

*and $v_i^S = v_i^+$ if $i \in S$ and $v_i^S = v_i^-$ otherwise. Values of $C^*$ are uniquely determined on $[0,1]^{k+p}$.*

*Proof.* Assume that $v_p \in \operatorname{Ran} F_{Y_p}$, and subsequently $v_p^S = v_p$ for any $S \subseteq \{1,\dots,p\}$. Then it holds $\lambda_p(v_p) = 0$ and $1-\lambda_p(v_p) = 1$, yielding

$$C^*(u_1,\dots,u_k,v_1,\dots,v_p) = \sum_{S\subseteq\{1,\dots,p-1\}} C\left(u_1,\dots,u_k,v_1^S,\dots,v_{p-1}^S,v_p\right)\prod_{i\in S}\lambda_i(v_i)\prod_{j\notin S}(1-\lambda_j(v_j)).$$

Inductively, we could reduce the sum in the same way for all other variables $v_j$ such that $v_j \in \operatorname{Ran} F_{Y_j}$. Therefore, it suffices to prove the theorem for $v_j \notin \operatorname{Ran} F_{Y_j}$ for all $j$.

Denote $x_i = F_{X_i}^{-1}(u_i)$ and $y_j = F_{Y_j^*}^{-1}(v_j)$. Note that $v_j \notin \operatorname{Ran} F_{Y_j}$ implies $y_j \in \mathbb{R}$. Let $F$ and $F^*$ denote distribution functions of $Z$ and $Z^*$, respectively. Since vector $Z^*$ is continuous, we can use Sklar's theorem on $Z^*$, and Theorem 1.5 (including the notation therein) to find copula $C^*$ uniquely defined on $[0,1]^{k+p}$ satisfying

$$\begin{aligned} C^*(u_1,\dots,u_k,v_1,\dots,v_p) &\overset{\text{Sklar}}{=} \\ F^*\left(F_{X_1}^{-1}(u_1),\dots,F_{X_k}^{-1}(u_k),F_{Y_1^*}^{-1}(v_1),\dots,F_{Y_p^*}^{-1}(v_p)\right) &\overset{\text{Theorem 1.5}}{=} \\ \sum_{S\subseteq\{1,\dots,p\}} F(x_1,\dots,x_k,y_1^S,\dots,y_p^S)\prod_{i\in S}F_{U_i}(y_i-\lfloor y_i\rfloor)\prod_{j\notin S}(1-F_{U_j}(y_j-\lfloor y_j\rfloor)). \end{aligned} \tag{1.9}$$

Take any $S \subseteq \{1, \ldots, p\}$. Since $u_i \in \operatorname{Ran} F_{X_i} = [0,1]$ and $v_j^+, v_j^- \in \operatorname{Ran} F_{Y_j}$, all copulas satisfying Sklar's theorem for random vector $Z$ attain same values on $(u_1, \ldots, u_k, v_1^S, \ldots, v_p^S)$. For any copula $C$ satisfying Sklar's theorem for random vector $Z$ we therefore get

$$C\left(u_1, \ldots, u_k, v_1^S, \ldots, v_p^S\right) = \\ F\left(F_{X_1}^{-1}(u_1), \ldots, F_{X_k}^{-1}(u_k), F_{Y_1}^{-1}(v_1^S), \ldots, F_{Y_p}^{-1}(v_p^S)\right).$$

Assume for now that $v_j^- > 0$ for every $j$. Consequently, we have $v_j^+ > 0$ and $v_j^S > 0$, since $v_j \notin \operatorname{Ran} F_X$. We may conclude according to Theorem 1.3 and observation bellow Definition 1.3 that

$$y_j^S = \begin{cases} \left\lceil F_{Y_j^*}^{-1}(v_j) \right\rceil = \left\lceil \underline{F}_{Y_j}^{-1}(v_j) + F_{U_j}^{-1}\left(\frac{v_j - v_j^-}{v_j^+ - v_j^-}\right) \right\rceil = \overline{F}_{Y_j}^{-1}(v_j) = F_{Y_j}^{-1}(v_j^+) & j \in S, \\ \left\lfloor F_{Y_j^*}^{-1}(v_j) \right\rfloor = \left\lfloor \underline{F}_{Y_j}^{-1}(v_j) + F_{U_j}^{-1}\left(\frac{v_j - v_j^-}{v_j^+ - v_j^-}\right) \right\rfloor = \underline{F}_{Y_j}^{-1}(v_j) = F_{Y_j}^{-1}(v_j^-) & j \notin S. \end{cases} \tag{1.10}$$

We also used observation

$$F_{U_j}^{-1}\left(\frac{v_j - v_j^-}{v_j^+ - v_j^-}\right) \in (0,1).$$

Equivalently, equation (1.10) means $y_j^S = F_{Y_j}^{-1}(v_j^S)$ whenever $v_j^S > 0$, and

$$F\left(F_{X_1}^{-1}(u_1), \ldots, F_{X_k}^{-1}(u_k), F_{Y_1}^{-1}(v_1^S), \ldots, F_{Y_p}^{-1}(v_p^S)\right) = \\ F\left(x_1, \ldots, x_k, y_1^S, \ldots, y_p^S\right).$$

However, notice that the expression holds even if $v_j^- = 0$, because then necessarily $F_{Y_j}^{-1}(v_j^-) = -\infty$ and $y_j^- = -1$. As the support of discrete random variables is contained in $\mathbb{N}_0$, the joint distribution function $F$ is zero whenever one of the arguments is negative. Consequently

$$C\left(u_1, \ldots, u_k, v_1^S, \ldots, v_p^S\right) = F\left(x_1, \ldots, x_k, y_1^S, \ldots, y_p^S\right) \tag{1.11}$$

holds for all possible values of $v_j^S$.

Looking at Theorem 1.3 and (1.10), we may also derive by realizing $y_j > \lfloor y_j \rfloor$ when $v_j \notin \operatorname{Ran} F_{Y_j}$, that

$$y_j - \lfloor y_j \rfloor = \underline{F}_{Y_j}^{-1}(v_j) + F_{U_j}^{-1}\left(\frac{v_j - v_j^-}{v_j^+ - v_j^-}\right) - \underline{F}_{Y_j}^{-1}(v_j) = F_{U_j}^{-1}\left(\frac{v_j - v_j^-}{v_j^+ - v_j^-}\right).$$

Consequently, using the properties of $F_{U_j}$ and $F_{U_j}^{-1}$ we obtain

$$F_{U_j}(y_j - \lfloor y_j \rfloor) = F_{U_j}\left(F_{U_j}^{-1}\left(\frac{v_j - v_j^-}{v_j^+ - v_j^-}\right)\right) = \frac{v_j - v_j^-}{v_j^+ - v_j^-}$$

which is equivalent to $F_{U_j}(y_j - \lfloor y_j \rfloor) = \lambda_j(v_j)$, because when $v_j \in \operatorname{Ran} F_{Y_j}$ it also holds $y_j \in \mathbb{N}_0$, and thus $F_{U_j}(y_j - \lfloor y_j \rfloor) = F_{U_j}(0) = 0$. Combining this, (1.9) and (1.11) yields the theorem. □

A key observation about Theorem 1.6 is that extension copula does not depend on distributions we use to extend discrete margins, as long as they are mutually independent. This result was mentioned, for example, in Denuit and Lambert [2005] for the bivariate case of discrete margins. We have formally proven that this result also holds for mixed random vectors of any number of margins. Also, the Theorem 1.6 can be interpreted as a multi-linear interpolation of copula values on points on which all copulas satisfying Sklar's theorem are uniquely defined, i.e. on set $[0,1]^k \times \bigtimes_{i=1}^{p} \operatorname{Ran} F_{Y_i}$.

### 1.3.1 Estimation of extension copula

Suppose we have a theoretical random vector $Z = (X_1, \ldots, X_n)$ and a sample from the distribution of this vector in the form of historical data. In the presence of historical data, exact values of copula $C$ associated with $Z$ cannot be determined exactly. We have to estimate them from the historical data. Assume that we have $m$ observations. The estimation is usually done using the empirical joint distribution function of $Z$, which we denote as $\hat{F}_m$, and empirical marginal quantile functions $\hat{F}^{-1}_{m,X_i}$. Then we can use (1.2) to compute estimate $\hat{C}_m$ of $C$, as

$$\hat{C}_m(u_1, \ldots, u_n) = \hat{F}_m\left(\hat{F}^{-1}_{m,X_1}(u_1), \ldots, \hat{F}^{-1}_{m,X_n}(u_n)\right), \tag{1.12}$$

where $u_1, \ldots, u_n \in [0,1]$. Since our goal is to compute extension copula $C^*$ of $Z^*$, we require its estimation as well. Assume the setting of Theorem 1.6, namely the separation of $Z$ into continuous and discrete margins. Denote estimation of $C^*$ by $\hat{C}^*_m$. We can use Theorem 1.6, together with the notion therein, to obtain it using

$$\begin{aligned}\hat{C}^*_m(u_1, \ldots, u_k, v_1, \ldots, v_p) = \\ \sum_{S \subseteq \{1,\ldots,p\}} \hat{C}_m\left(u_1, \ldots, u_k, \hat{v}_1^S, \ldots, \hat{v}_p^S\right) \prod_{i \in S} \hat{\lambda}_{m,i}(v_i) \prod_{j \notin S} \left(1 - \hat{\lambda}_{m,j}(v_j)\right),\end{aligned} \tag{1.13}$$

where $\hat{v}_i^S$ are the estimates of $v_i^S$ using the empirical marginal distribution functions $\hat{F}_{m,Y_i}$ and quantile functions $\hat{F}^{-1}_{m,Y_i}$, and

$$\hat{\lambda}_{m,i}(v_i) = \begin{cases} \frac{v_i - \hat{v}_i^-}{\hat{v}_i^+ - \hat{v}_i^-} & v_i \notin \operatorname{Ran} \hat{F}_{m,Y_i}, \\ 0 & v_i \in \operatorname{Ran} \hat{F}_{m,Y_i}, \end{cases}$$

which serve as estimates of $\lambda_i(v_i)$.

The properties of estimates $\hat{C}^*_m$ were studied in Genest et al. [2014] for discrete random vectors. In this article, the authors established a weak limit of $\hat{C}^*_m$ on any compact subset of

$$\bigtimes_{i=1}^{n} [0,1] \setminus \operatorname{Ran} F_{X_i},$$

although the limit is non-trivial. Notice that this set is dense on $[0,1]^n$. For more details, see Genest et al. [2014], Theorem 3.1. The reason for the existence of a weak limit only on compact subsets of this set is due to the non-existence of $i$-th partial derivative of $C^*$ on $\operatorname{Ran} F_{X_i}$ for all margins $i = 1, \ldots, n$. Despite this complication, the estimator $\hat{C}^*_m$ converges in probability to $C^*$. Formally,

$\left\|\hat{C}_m^* - C^*\right\| \xrightarrow{P} 0$ as $m \to \infty$. See the aforementioned article for proof. We believe these results could be generalized onto mixed random vectors. However, we state this as a hypothesis without providing proof.

# 2. A copula-based scenario generation method

The goal of this chapter is to provide a summary of the algorithm introduced by Kaut [2014]. We credit all results presented in this chapter to Kaut [2014], but we try to use alternative formulations and explanations. Interested readers can look into this original article, or into a well-written summary of the method in Zhang et al. [2021], Appendix A.

For the remainder of this chapter, we assume a setting where we obtained copula $C$ from random vector $X = (X_1, \ldots, X_n)$, historical data, or a combination of both, and we aim to generate $S$ scenarios. The method consists of two steps. Firstly, it constructs so-called *copula sample* which models copula $C$. This happens without reflecting the marginal distributions. In the second step, the method transforms this copula sample to account for marginal distributions.

## 2.1 Copula samples

The copulas do not take into consideration marginal distributions of random vectors. Instead, copulas focus on dependencies in terms of order. It is therefore natural to work with ranks of data. The copula samples are defined using ranks and they reflect that we want to represent copula $C$ as closely as possible using discretization into $S$ scenarios.

**Definition 2.1** (Copula sample)**.** *We define copula sample as*

$$\mathcal{C} = \{(r_1, \ldots, r_n) : 1 \leq r_i \leq S, 1 \leq i \leq n\},$$

*where each value appears exactly once in each dimension. We denote a collection of copula samples as $\Lambda$ and refer to elements of copula samples as copula assignments.*

Let us introduce a new notation. If $C$ is a copula, then by $C_r$ we mean computation of copula value from ranks

$$C_r(r_1, \ldots, r_n) := C\left(\frac{r_1}{S}, \ldots, \frac{r_n}{S}\right), \tag{2.1}$$

where $r_1, \ldots, r_n \in \{1, \ldots, S\}$. Bearing in mind this notation, we denote the distribution function of copula sample $\mathcal{C}$ as $\mathcal{C}_r$ and define it as follows

$$\mathcal{C}_r(r_1, \ldots, r_n) = \frac{1}{S} \left|\{(r'_1, \ldots, r'_n) \in \mathcal{C} : r'_i \leq r_i, 1 \leq i \leq n\}\right|.$$

The expression means that we count the number of elements in copula sample $\mathcal{C}$ which are element-wise lower than input vector $(r_1, \ldots, r_n)$, and then divide it by $S$, which is the number of elements in the copula sample.

The principal idea is to find a copula sample that is as close as possible to the copula $C$, measured by some metric. Two metrics were proposed by Kaut [2014]:

average deviance and maximum deviance. For copula sample $\mathcal{C}$, they are defined as follows

$$\begin{aligned}\text{dev}_{\text{avg}}(\mathcal{C}, C) &= \frac{1}{S^n} \sum_{r_1=1}^{n} \cdots \sum_{r_n=1}^{n} \left| \mathcal{C}_r(r_1, \ldots, r_n) - C_r(r_1, \ldots, r_n) \right|, \\ \text{dev}_{\text{max}}(\mathcal{C}, C) &= \max_{\substack{1 \le r_i \le S \\ 1 \le i \le n}} \left| \mathcal{C}_r(r_1, \ldots, r_n) - C_r(r_1, \ldots, r_n) \right|.\end{aligned}$$

The problem of finding a copula sample that is as close as possible to the target copula can be formally stated as

$$\min_{\mathcal{C} \in \Lambda} \text{dev}_{\text{avg}}(\mathcal{C}, C), \tag{2.2}$$

or its respective version for maximum deviance.

We visualize the problem (2.2) in Figure 2.1. For illustration purposes, we

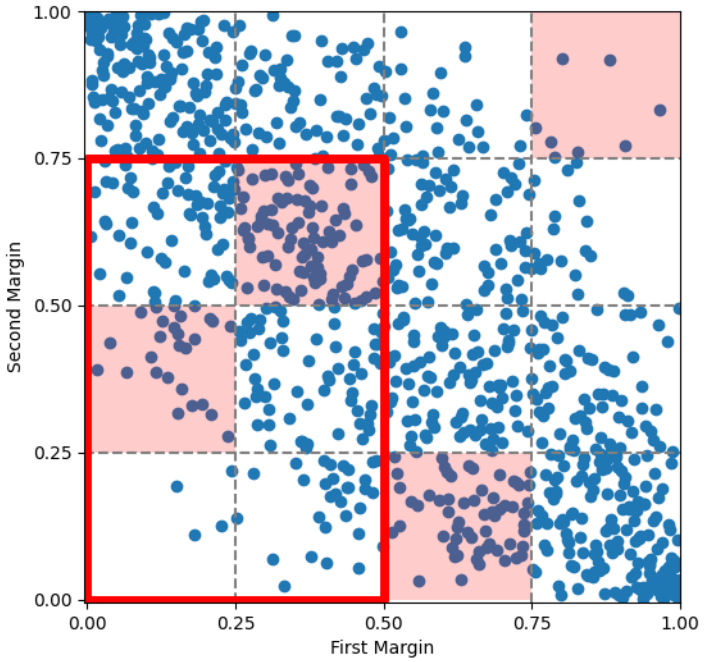



Figure 2.1: Example of copula sample laid over a sample from the two-dimensional Gaussian copula with $\rho = -0.7$. Sampled values are represented with blue points and an example of a copula sample with red squares.

assume the copula $C$ is represented through a sample from bivariate Gaussian copula with $\rho = -0.7$. The samples are plotted using blue points. The unit square is divided into a $4 \times 4$ grid where the number four represents the number of scenarios we want to generate. Each copula assignment given by a pair of ranks $(i, j), 1 \le i, j \le 4$, represents a square with rank coordinates $(i, j), (i-1, j), (i-1, j-1)$ and $(i, j-1)$, which could be easily scaled into the unit interval by diving it by the number of scenarios to generate, in our example four. Notice that the same transformation is performed in (2.1). The collection of all copula assignments and their associated squares, one in each row and column of the constructed grid

constitutes a copula sample. In the figure, an example is shown using red squares. Fix a pair of ranks $(2, 3)$, or equivalently points $\left(\frac{1}{2}, \frac{3}{4}\right)$ when scaled into the unit square. Then, the distribution function of a copula sample is the number of squares contained in the region surrounded by the red line, divided by the number of elements in the copula sample, which is four. Therefore $\mathcal{C}_r(2, 3) = \frac{1}{2}$. Similarly, the $C_r(2, 3)$ is the relative number of samples contained in the same region. If the copula $C$ was given exactly, we would compute this value using a distribution function. The difference $|\mathcal{C}_r(2, 3) - C_r(2, 3)|$ could be then easily computed. By repeating the same process for all rank grid points $\{1, 2, 3, 4\}^2$, we are able to compute the average or maximum deviation between the chosen copula sample and copula $C$. Informally, the problem (2.2) is then equivalent to finding the most optimal assignment of squares such that the deviation is minimized.

Unfortunately, it takes $\mathcal{O}(S^n)$ operations to evaluate a copula sample using both deviance metrics for a general $n$-variate case. Furthermore, the collection of copula samples $\Lambda$ can be quite large. Therefore computation complexity grows quickly and (2.2) can be solved directly only for small $S$ and $n$, meaning that we cannot solve reasonably sized problems using this approach. In the original article, it was attempted to formulate (2.2) for bivariate copula $C$ as a mixed integer linear program. However, the computational study revealed this approach is not tractable, since the computational times are rising exponentially quickly with an increasing number of scenarios. The study was performed for both average and maximum deviance.

To overcome this obstacle, Kaut [2014] proposed a heuristic algorithm that builds copula samples iteratively margin by margin. From now on, we focus only on the average deviation problem, although the algorithm could be easily modified to the maximum deviance.

## 2.2 Generation of copula samples

The core concept of the algorithm is to build copula sample $\mathcal{S}$ incrementally. Beginning with margin $2 \leq m \leq n$, we assume $\mathcal{S}$ has been built up to the $(m-1)$-th margin in the form of

$$\{(r_s^1, \ldots, r_s^{m-1}) : s \in 1, \ldots, S\}.$$

Next, we aim to assign new ranks $1, \ldots, S$ to the $m$-th margin of elements in $\mathcal{S}$ in a certain order. Considering any permutation $(i_1, \ldots, i_S)$ of ranks $(1, \ldots, S)$, we assign them to $m$-th margin of elements in $\mathcal{S}$, constituting copula sample

$$\{(r_s^1, \ldots, r_s^{m-1}, i_s) : s \in \{1, \ldots, S\}\}$$

and bivariate copula samples $\mathcal{C}^j$ of margin pairs $(j, m), 1 \leq j \leq m-1$, equal to $\{(r_s^j, i_s) : s \in \{1, \ldots, S\}\}$. Now take margin pairs $(X_j, X_m), j < m$, and their copula $C^j$. The heuristic then tries to find permutation $(i_1, \ldots, i_S)$ such that the sum

$$\sum_{j=1}^{m-1} \mathrm{dev}_{\mathrm{avg}}(\mathcal{C}^j, C^j) \tag{2.3}$$

is minimized. Then we proceed to the next margin. The process repeats until we reach the last one, yielding the final copula sample

$$\mathcal{S} = \{(r_s^1, \ldots, r_s^n) : s \in \{1, \ldots, S\}\}.$$

Unfortunately, the number of permutations is large, making it impossible to solve problem (2.3) directly. We outline a potential solution later in this chapter, along with a more detailed description of the algorithm.

By handling margins one by one, we no longer have to calculate deviation for all margins, but just for two, reducing the complexity from $\mathcal{O}(S^n)$ to $\mathcal{O}(S^2)$. Fortunately, even this can be improved upon by using a simple observation.

### 2.2.1 Heuristic for bivariate copula samples

Heuristic is based on the observation that the distribution function of copula sample $\mathcal{C}_r(i,j)$ depends only on grid points with coordinates less than $i$ and $j$, i.e. on points $(i',j')$ such that $i' \leq i, j' \leq j$. We therefore try to construct bivariate copula samples row-wise: for fixed row $j$ we try to find column $i$ such that the deviation caused is minimized. Consider a bivariate copula sample $\mathcal{C} = \{(i_{j'}, j') : j' < j\}$ which was built up until row $j-1$, and denote $\mathcal{C}^i = \mathcal{C} \cup \{(i,j)\}$. Then problem of finding unused column $i$ which minimizes deviation caused by pairing it to $j$ can be rewritten into an optimization problem

$$\min_{i \in \mathcal{I}} \delta(i,j) := \sum_{l=1}^{S} \left| \mathcal{C}_r^i(l,j) - C_r(l,j) \right|, \tag{2.4}$$

where $\mathcal{I} = \{1,\dots,S\} \setminus \{i_{j'} : j' < j\}$ denotes the set of column indices that were not used in copula sample. It can be derived that these values can be recursively computed using the formula

$$\begin{aligned}\delta(i,j) = \delta(i-1,j) &+ |\mathcal{C}_r(i-1,j-1) - C_r(i-1,j)| \\ &- \left| \mathcal{C}_r(i-1,j-1) + \frac{1}{S} - C_r(i-1,j) \right| \end{aligned} \tag{2.5}$$

with initialization

$$\delta(0,j) = \sum_{i=1}^{S} \left| \mathcal{C}_r(i,j-1) + \frac{1}{S} - C_r(i,j) \right|.$$

Detailed derivation is included in Kaut [2014]. Calculating $\delta(i,j)$ from (2.4) requires $\mathcal{O}(S)$ operations. Further, it has to be repeated for all $i = 1,\dots,S$ to decide which column minimizes deviation, totaling at $\mathcal{O}(S^2)$. Using (2.5), we can reduce it to $\mathcal{O}(S)$.

### 2.2.2 Description of the algorithm

We present the full algorithm for generating copula samples in Algorithm 1. Let us describe the algorithm step-by-step. The algorithm commences with iteration over every margin starting at $m = 2$. It is then followed by the initialization of a set of unused copula assignment indices, empty bivariate copula samples $\mathcal{C}^k$, and copulas $C^k$ of margin pairs $(k,m)$.

The first inner loop starts at Line 5. It is an iteration over rows, starting at $j = 1$. Then we compute the deviation caches for every margin pair $(k,m), k < m$. Caches are computed using the recursive rule (2.5). This is allowed by the fact that copula samples $\mathcal{C}^k$ cannot, at this point in the algorithm, contain rows with

**Algorithm 1** Heuristic for generating copula samples

**Input:** Copula $C$.
**Output:** Copula sample $\mathcal{C}$.
**Initialize:** $r_s^1 \leftarrow s,\ s \in \{1, \ldots, S\}$

1: **for** $m = 2, \ldots, n$ **do**
2: $\quad \mathcal{S} \leftarrow \{1, \ldots, S\}$
3: $\quad \mathcal{C}^k \leftarrow \emptyset,\ 1 \leq k \leq m-1$
4: $\quad$ Compute copulas $C^k$ of margin pairs $(X_k, X_m),\ 1 \leq k \leq m-1$
5: $\quad$ **for** $j = 1, \ldots, S$ **do**
6: $\quad\quad$ **for** $k = 1, \ldots, m-1$ **do**
7: $\quad\quad\quad \delta^k \leftarrow$ computeDeviationCache$(\mathcal{C}^k, C^k, j)$
8: $\quad\quad$ **end for**
9: $\quad\quad s^* \leftarrow \arg\min_{s \in \mathcal{S}} \sum_{i=1}^{m-1} \delta^i(r_s^i, j)$
10: $\quad\quad \mathcal{S} \leftarrow \mathcal{S} \setminus \{s^*\}$
11: $\quad\quad r_{s^*}^m \leftarrow j$
12: $\quad\quad \mathcal{C}^k \leftarrow \mathcal{C}^k \cup \{(r_{s^*}^k, j)\},\ k < m$
13: $\quad$ **end for**
14: **end for**
15: **return** Copula sample $\{(r_s^1, \ldots, r_s^n) : s \in \{1, \ldots, S\}\}$

an index higher or equal to $j$. The only update of copula samples happens at Line 12, from which it is clear that they cannot contain rank pairs with row index larger than, or equal to $j$.

Perhaps the most crucial part of the algorithm is happening at Line 9. From the unused copula assignments, we select the one that causes the least amount of deviance when extended on $m$-th margin by assigning it rank $j$. From another perspective, for every rank $j$ we select the copula assignment which causes the least amount of deviance. Then, at Line 10 we make the selected copula assignment unavailable in the next iteration. By doing this, we avoid the problem of systemically searching through permutations of ranks.

The iteration over margins is concluded by setting $m$-th margin of copula assignment with index $s^*$ to $j$ and updating the bivariate copula samples. After performing all iterations over margins, the final copula sample is returned.

### 2.2.3 Additional comments

Let us start by analyzing time and space complexity. Firstly, we restrict ourselves to the iteration over $m$-th margin. In each margin iteration, we iterate over $S$ ranks. Furthermore, for each rank, we compute $m-1$ deviation caches, each taking $S$ operations. Afterward, the selection of the best copula assignment takes place, which requires $\mathcal{O}(S \cdot (m-1))$ operations. Overall, the algorithm totals at $\mathcal{O}((m-1) \cdot S^2)$ operations when iterating over $m$-th margin. Because we iterate over all $n$ margins, the algorithm takes $\mathcal{O}(n^2 S^2)$ operations to finish.

In terms of space complexity, we need to store in each iteration over $m$-th margin $\mathcal{O}((m-1)S)$ for deviation caches and $\mathcal{O}((m-1)S)$ for bivariate copula samples. However, these values do not have to be stored in the next iterations over margins. Additionally, $\mathcal{O}(nS)$ is required for the output copula sample,

making the algorithm $\mathcal{O}(nS)$ in terms of space complexity. We omit from this calculation requirements to store input data, copula $C$, and copulas of all margin pairs.

The algorithm itself has a strong mathematical foundation. It is based on the important results from the copula theory like Sklar's theorem. Unfortunately, due to the computational complexity of problem (2.2), it is necessary to resort to a heuristic algorithm, at least until a new approach to model copula using $S$ scenarios is developed or until a new faster algorithm for solving (2.2) is found. By resorting to the previously described heuristic algorithm, we obtained a computationally feasible algorithm, but at the expense of losing multi-margin dependencies, since the algorithm works with pairs of margins. However, the algorithm can still capture dependencies in these margin pairs that can be non-linear.

Despite the heuristic nature of the algorithm, it was successfully applied to real-world problems involving continuous data. For example, in Zhang et al. [2021], the scenario generation for the stochastic shortest path problem on real road networks was studied. The copula-based method outperformed sampling by a significant margin using various objective functions and metrics. These results were achieved in spite of highly temporarily and spatially correlated data of travel speeds on real road networks.

## 2.3 Transformation of copula samples

Output from the first step of the algorithm is a copula sample which was found without taking into consideration marginal distributions. However, copula samples cannot be used in practice. Therefore we need to transform them to reflect marginal distributions. Assume that we have generated a copula sample $\{(r_s^1, \ldots, r_s^n) : s \in \{1, \ldots, S\}\}$ running the Algorithm 1. These ranks can be naturally scaled to the unit interval

$$\left[\frac{r_s^i - 1}{S}, \frac{r_s^i}{S}\right].$$

Applying the marginal quantile functions on these regions, we obtain

$$\left[F_{X_i}^{-1}\left(\frac{r_s^i - 1}{S}\right), F_{X_i}^{-1}\left(\frac{r_s^i}{S}\right)\right], \tag{2.6}$$

union of which over $s \in \{1, \ldots, S\}$ yields entire support of the margin $X_i$. It is sensible to transform each rank $r_s^i$ into element which is contained in (2.6). Denote such element $x_s^i$. Or in other words, it is the final value of $i$-th margin in scenario $s$ after transformation from rank $r_s^i$. Several transformations were formulated in Kaut [2014], but we list only some of them. Approaches can be split into three separate cases:

1. Distribution of $i$-th margin is known. Then we have the following reasonable options
   - $x_s^i = \operatorname{med}\left(X_i \;\middle|\; F_{X_i}^{-1}\left(\frac{r_s^i - 1}{S}\right) \leq X_i \leq F_{X_i}^{-1}\left(\frac{r_s^i}{S}\right)\right)$.
   - $x_s^i = \mathsf{E}\left[X_i \;\middle|\; F_{X_i}^{-1}\left(\frac{r_s^i - 1}{S}\right) \leq X_i \leq F_{X_i}^{-1}\left(\frac{r_s^i}{S}\right)\right]$.

2. Distribution of $i$-th margin is unknown, but we have some historical data at our disposal. Then we may take empirical quantile function $\hat{F}^{-1}_{X_i}$, or its interpolation, and proceed similarly as in the previous case.

3. We have a procedure for generating values out of marginal distributions. For example using other methods for generating scenarios (sampling, moment-matching, etc.). However, we do not discuss the details of this approach in our thesis. We refer to Kaut [2014] for more details.

From the aforementioned transformations, only conditional expectation guarantees that the resulting scenarios are not biased. However, it can be quite difficult to compute the conditional expectation. Probably due to this difficulty, in Kaut [2014], the author opted for the conditional median approach.

# 3. Extension of copula-based method for discrete data

The algorithm described in the previous chapter works well for continuous data. However, as we shall see in Section 3.1, it struggles to create reasonable scenarios when faced with discrete data. In this chapter, we propose an extension to this algorithm by utilizing discrete extensions and extension copula. We provide further intuition for the choice of extension copula in Section 3.2. The adjustments to the algorithm for copula sample generation are described in Section 3.3. In Section 3.4, new transformations of copula samples for handling discrete random variables are proposed. We claim contributions presented in this chapter as our own.

## 3.1 Necessity of extension for discrete data

In the presence of continuous data, the values of copula $C$ for continuous random vector $X = (X_1, \dots, X_n)$ are usually calculated using (1.2) if exact distributions are known, or its empirical counterpart (1.12) if only historical data is available. In either case, if the same equations are used for discrete data (i.e. without the use of any kind of extension), we illustrate in the following example that the method described in Kaut [2014] does not work properly.

Consider a simple example when we want to generate $S$ scenarios for discrete bivariate random vector $X = (X_1, X_2)$ with independent uniform margins on $\{0, 1\}$. If we compute values of copula $C$ of $X$ according to (1.2), we get

$$C(u,v) = \begin{cases} 0 & u = 0 \text{ or } v = 0, \\ \frac{1}{4} & 0 < u < \frac{1}{2},\ 0 < v < \frac{1}{2}, \\ \frac{1}{2} & 0 < u < \frac{1}{2},\ \frac{1}{2} \le v \text{ or } \frac{1}{2} \le u,\ 0 < v < \frac{1}{2}, \\ 1 & \text{otherwise.} \end{cases}$$

As described earlier, the algorithm builds copula samples row-wise. Recall that to each row we associate a column minimizing the metric defined by (2.4). For simplicity, we assume $S$ is odd. Then for $S \ge 4$ we may write

$$\begin{aligned} \delta(i,1) &= \sum_{l=1}^{S} \left| \frac{1}{S} \cdot \mathbf{1}[l \ge i] - C\left(\frac{i}{S}, \frac{1}{S}\right) \right| \\ &= \sum_{l=1}^{\frac{S-1}{2}} \left| \frac{1}{S} \cdot \mathbf{1}[l \ge i] - \frac{1}{4} \right| + \sum_{l=\frac{S+1}{2}}^{S} \left| \frac{1}{S} \cdot \mathbf{1}[l \ge i] - \frac{1}{2} \right| \\ &= \begin{cases} (i-1)\frac{1}{4} + \left(\frac{S+1}{2} - i\right)\left(\frac{1}{4} - \frac{1}{S}\right) + \frac{S+1}{2}\left(\frac{1}{2} - \frac{1}{S}\right) & i \le \frac{S-1}{2}, \\ \frac{S-1}{2}\frac{1}{4} + \left(i - \frac{S+1}{2}\right)\frac{1}{2} + (S+1-i)\left(\frac{1}{2} - \frac{1}{S}\right) & i \ge \frac{S+1}{2}. \end{cases} \end{aligned}$$

With some tedious work, it can be shown

$$\delta(i,1) = \frac{3S-7}{8} + \frac{i-1}{S}. \tag{3.1}$$

Hence, the derived deviance is monotonously increasing in $i$. Subsequently, the expression is minimized for $i = 1$ and the first row would be paired with the first column.

We proceed by induction with respect to $j$ until we reach $\frac{S-1}{2}$. The induction assumption is that the copula sample consists of pairs $(i', i'), i' \leq j-1$. Recursion (2.5) can be also performed row-wise

$$\delta(i,j) = \delta(i,j-1) + |\mathcal{C}_r(i-1,j-1) - C_r(i,j-1)| \\ - \left|\mathcal{C}_r(i-1,j-1) + \frac{1}{S} - C_r(i,j-1)\right|.$$

For more details see Kaut [2014]. Since we cannot select the same column again, we need to limit our attention only to columns with an index larger than $j-1$. Plugging into the recursive formula yields

$$\delta(i,j) = \delta(i,j-1) + \begin{cases} \left|\frac{j-1}{S} - \frac{1}{4}\right| - \left|\frac{j}{S} - \frac{1}{4}\right| & j \leq i \leq \frac{S-1}{2}, \\ \left|\frac{j-1}{S} - \frac{1}{2}\right| - \left|\frac{j}{S} - \frac{1}{2}\right| & i \geq \frac{S+1}{2}. \end{cases}$$

The most important observation is that the new contribution does not depend on column index $i$. As a consequence, $\delta(i,j) = \delta(i,1) + J$ for some $J$ which does not depend on $i$, meaning that the deviation is minimized for $i = j$ due to monotony of $\delta$.

Let us summarize what we have found. We have proved that using (1.2) to compute values of a copula, in the presence of discrete data, may lead to the addition of rank pairs $(i,i)$ up to $i = \frac{S-1}{2}$ to the copula sample. The evaluation of bounds for the transformation region (2.6) yields only set $\{0\}$ for both margins of $X$, because when $2 \leq i \leq \frac{S-1}{2}$, we have

$$F_{X_1}^{-1}\left(\frac{i-1}{S}\right) = F_{X_1}^{-1}\left(\frac{i}{S}\right) = 0,$$

and similarly for the second margin $X_2$. Notice that the same observation holds for $i = 1$ after restricting the evaluated bounds to $\mathbb{N}_0$ in which all realizations of discrete random variables lie. As a consequence, pairs $(i,i), i \leq \frac{S-1}{2}$, would be transformed into $\frac{S-1}{2}$ scenarios $(0,0)$. However, due to the algorithm constructing pairs of ranks row-wise and the algorithm continuing to row index $\frac{S+1}{2}$, we completely lost the chance to produce more than one scenario $(0,1)$. This behaviour is definitely not desired because the probability of this realization is the same as for $(0,0)$, and the discrepancy between the respective number of scenarios $(0,0)$ and $(0,1)$ increases with the rising number of scenarios $S$. Therefore it is expected that the resulting scenarios will form repeating sequences of realizations without a lot of variation. Our experiments confirmed this observation for discrete random vectors with multiple margins.

## 3.2 Intuition for choice of extension copula

Sklar's theorem states that all copulas are uniquely determined on the Cartesian product of ranges of marginal distribution functions. In the case of discrete margins, it is a grid of points. These grid points naturally form cells into which

probability associated with each grid point is evenly distributed by extending discrete variables, meaning that the density of associated extension copula with respect to the Lebesgue measure is constant on these cells.

To elaborate further, consider discrete bivariate random vector $(X, Y)$ with joint distribution function $F$. Furthermore, let $p_n = P(X \leq n), n \in \operatorname{supp} X$, and $q_m = P(Y \leq m), m \in \operatorname{supp} Y$. For convenience, also let $p_{-1} = q_{-1} = 0$. We assume $n \in \operatorname{supp} X$ and $m \in \operatorname{supp} Y$ for the remainder of this section. All copulas $C$ satisfying Sklar's theorem for $(X, Y)$ are uniquely defined on grid points $(p_n, q_m)$ and it holds

$$C(p_n, q_m) = F\left(F_X^{-1}(p_n), F_Y^{-1}(q_m)\right) = F(n, m) = P(X \leq n, Y \leq m).$$

It easily follows

$$P(X = n, Y = m) = C(p_n, q_m) - C(p_{n-1}, q_m) - C(p_n, q_{m-1}) + C(p_{n-1}, q_{m-1}).$$

Take extension $(X^*, Y^*)$ of $(X, Y)$ and its extension copula $C^*$. Since copulas $C$ and $C^*$ coincide on $\operatorname{Ran} F_X \times \operatorname{Ran} F_Y$, we get

$$\begin{aligned} P(n-1 \leq X^* \leq n, m-1 \leq Y^* \leq m) = P(X = n, Y = m) = \\ C^*(p_n, q_m) - C^*(p_{n-1}, q_m) - C^*(p_n, q_{m-1}) + C^*(p_{n-1}, q_{m-1}). \end{aligned}$$

Because the extension copula $C^*$ linearly interpolates values of copula $C$ on grid points $\{p_{n-1}, p_n\} \times \{q_{m-1}, q_m\}$, the density of $C^*$ is constant on the Cartesian product of intervals $(p_{n-1}, p_n)$ and $(q_{m-1}, q_m)$. Therefore, it is precisely this region $(p_{n-1}, p_n) \times (q_{m-1}, q_m)$ into which discrete extensions spread probability $P(X = n, Y = m)$ evenly, measured by the density of extension copula $C^*$.

Similar arguments could be made for mixed random vectors. Let $(X, Z)$ be a random vector where $Z$ is a continuous random variable and $X$ is kept unchanged. Then following the same thought process, we would arrive at

$$P(X = n, Z \leq t) = C(p_n, u) - C(p_{n-1}, u), \; u = F_Z^{-1}(t),$$

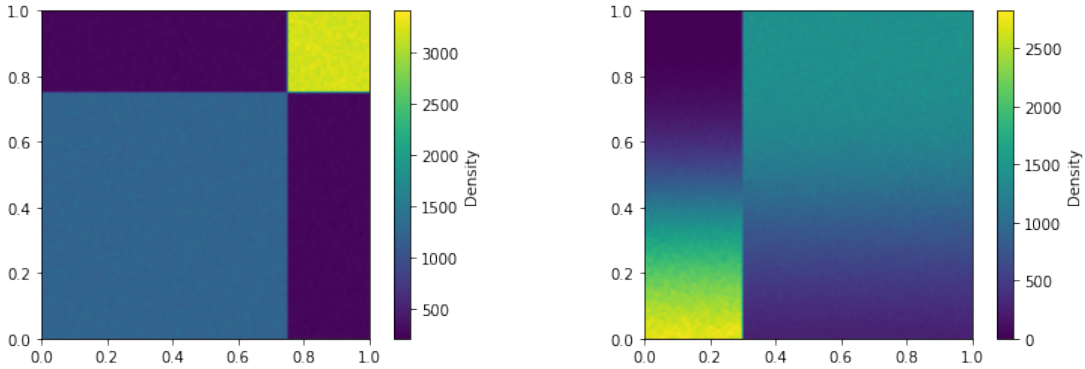
and

$$P(n-1 \leq X^* \leq n, Z \leq t) = P(X = n, Z \leq t) = C^*(p_n, u) - C^*(p_{n-1}, u).$$

Similarly, the extension copula linearly interpolates copula $C$ between points $(p_{n-1}, u)$ and $(p_n, u)$, rendering the density of $C^*$ constant along the first margin on the interval $(p_{n-1}, p_n)$. Hence, the probability $P(X = n, Z \leq t)$ is equally spread by extension $(X^*, Z)$ across first margin into $(p_{n-1}, p_n)$ in terms of density of $C^*$.

To illustrate this using graphical tools, consider two bivariate random vectors

$$P(X = i, Y = j) = \begin{cases} 0.7 & i = j = 0, \\ 0.05 & i = 0, j = 1, \\ 0.05 & i = 1, j = 0, \\ 0.2 & i = j = 1, \\ 0 & \text{otherwise.} \end{cases}$$




(a) Discrete margins.

(b) Mixed margins.

Figure 3.1: Example of extension copula densities for discrete and mixed random vectors.

and $Z \sim \text{Alt}(0.7)$ with $W|Z \sim \mathcal{N}(2Z, 1+Z)$. We compute the extension copula of those bivariate random vectors and then sample from this copula many times. The result is shown in Figure 3.1. We can see a clear divide between regions into which the probabilities are spread by extending random vectors. In the case of discrete margins, we can see that the divide is along value 0.75 for both axes since in both cases it holds $P(X \leq 0) = P(Y \leq 0) = 0.75$. These lines divide the unit square into four cells, with the lowest density being in cells corresponding to observations $(0, 1)$ and $(1, 0)$. It is worth noting that the size of the region and the associated probability does not necessarily mean high density, as shown in the left-hand side plot. In the right-hand side plot, we can see only one dividing line at 0.3 of the discrete first margin. In this plot, we can nicely see how the probability is evenly spread across discrete margins.

Observing the natural structure that discrete extensions impose on extension copulas, we can see why it is such a natural choice for extending the algorithm developed by Kaut [2014]. The splitting of unit squares into $S \times S$ grid for copula samples, as described on Figure 2.1, relates nicely to the grid-like structure that discrete extensions impose on extension copula, although the imposed grid is not necessarily equally spaced. Therefore, alternative definitions of copula samples are possible to reflect the possibly unequally spaced grids, an idea we shall pursue in our future work.

## 3.3 Generation of copula samples

The idea is relatively simple. In Line 4 of Algorithm 1, we replace copulas $C^k$ of margin pairs $(X_j, X_m), j < m$, with their extension copula $C^{k*}$. For clarity, let us rephrase Theorem 1.6 for the bivariate case.

**Theorem 3.1.** *Take $X$ and $Y$ random variables and their copula $C$ satisfying Sklar's theorem. Let $C^*$ be extension copula of random vector $(X, Y)$. Fix $u, v \in [0, 1]$, then if $u \in \text{Ran}\, F_X$ and $v \in \text{Ran}\, F_Y$ we have $C^*(u, v) = C(u, v)$. Otherwise, there are the following options*

1. *$X$ and $Y$ are continuous, then trivially $C^*(u, v) = C(u, v)$.*

2. *$X$ is continuous and $Y$ discrete. Then if $v \notin \operatorname{Ran} F_Y$ we have*

$$C^*(u,v) = \frac{v - v_Y^-}{v_Y^+ - v_Y^-} C(u, v_Y^+) + \frac{v_Y^+ - v}{v_Y^+ - v_Y^-} C(u, v_Y^-). \tag{3.2}$$

3. *$X$ is discrete and $Y$ continuous. The extension copula is similar to the previous case, but the roles of $X$ and $Y$ are switched. In other words, when $u \notin \operatorname{Ran} F_X$ then*

$$C^*(u,v) = \frac{u - u_X^-}{u_X^+ - u_X^-} C(u_X^+, v) + \frac{u_X^+ - u}{u_X^+ - u_X^-} C(u_X^-, v). \tag{3.3}$$

4. *Both $X$ and $Y$ are discrete. In this case when $u \notin \operatorname{Ran} F_X$ and $v \notin \operatorname{Ran} F_Y$, the extension copula has the most complicated, but still manageable form*

$$\begin{aligned} C^*(u,v) = &\frac{u - u_X^-}{u_X^+ - u_X^-} \frac{v - v_Y^-}{v_Y^+ - v_Y^-} C(u_X^+, v_Y^+) + \frac{u - u_X^-}{u_X^+ - u_X^-} \frac{v_Y^+ - v}{v_Y^+ - v_Y^-} C(u_X^+, v_Y^-) + \\ &\frac{u_X^+ - u}{u_X^+ - u_X^-} \frac{v - v_Y^-}{v_Y^+ - v_Y^-} C(u_X^-, v_Y^+) + \frac{u_X^+ - u}{u_X^+ - u_X^-} \frac{v_Y^+ - v}{v_Y^+ - v_Y^-} C(u_X^-, v_Y^-). \end{aligned}$$

*Cases where $u \in \operatorname{Ran} F_X$ and $v \notin \operatorname{Ran} F_Y$, or $u \notin \operatorname{Ran} F_X$ and $v \in \operatorname{Ran} F_Y$ correspond to* (3.2) *and* (3.3)*, respectively.*

*Proof.* Results are just special cases of Theorem 1.6. □

If the exact marginal distributions and values of copula $C$ at grid points $\operatorname{Ran} F_X \times \operatorname{Ran} F_Y$ are known, then the copula $C^*$ can be computed exactly using this theorem. In this case, the types of $X$ and $Y$ can be determined easily and the form of extension is chosen accordingly.

Whenever we deal with historical data, the exact calculation of $C^*$ is not possible and it is necessary to resort to estimation. The simplest approach is probably using empirical joint distribution function and empirical marginal distribution and quantile functions. We described this approach in Section 1.3.1. The types of $X$ and $Y$ could be either specified by the user or automatically inferred from the historical data. However, this may be prone to error.

Although the algorithm theoretically replaces discrete margins by their extensions, actual randomization is not necessary, meaning that the algorithm does not transform the discrete values $x$ into $x + u - 1$, where $u$ is sampled from some extension distribution. In reality, only the extended distribution and quantile functions are used for deriving the extension copula, and in the following section, we shall use them for the transformation of copula samples.

## 3.4 Discrete transformation of copula samples

Transformations mentioned in Section 2.3 work well for continuous random variables. There is no reason to discuss this case further. Discrete random variables, however, are a little bit trickier. To illustrate issues that arise with transformation, let $X$ be a discrete random variable and $X^* = X + U - 1$ its extension. Assume that we have obtained copula sample $\{(r_s^1, \ldots, r_s^n) : s \in \{1, \ldots, S\}\}$ running the extended version of the Algorithm 1 and that $X$ is $i$-th margin of input

random vector. Recall that the algorithm replaces discrete margins with their extensions. As we have seen earlier, these ranks naturally form regions

$$\left[F_{X^*}^{-1}\left(\frac{r_s^i-1}{S}\right), F_{X^*}^{-1}\left(\frac{r_s^i}{S}\right)\right]. \tag{3.4}$$

Suppose we want to investigate the probability that $X^*$ lies in this region. Extension $X^*$ is defined using underlying discrete random variable $X$. As a consequence, the probability that $X^*$ lies in the region (3.4) is influenced by the distribution of $X$. The following theorem identifies realizations of $X$ contributing to this probability. These realizations could be interpreted as sensible choices into which ranks $r_s^i$ can be transformed.

**Theorem 3.2.** *Let $L_X$ be a function defined as*

$$L_X(u) = \begin{cases} 0 & u = 0, \\ F_X^{-1}(u) + \mathbf{1}[u \in \operatorname{Ran} F_X] & u \in (0,1), \\ \sup(\operatorname{supp} X) & u = 1. \end{cases}$$

*Then only for the realizations $n \in \operatorname{supp} X$ fulfilling*

$$L_X\left(\frac{r_s^i-1}{S}\right) \le n \le F_X^{-1}\left(\frac{r_s^i}{S}\right) \tag{3.5}$$

*it holds*

$$P\left(F_{X^*}^{-1}\left(\frac{r_s^i-1}{S}\right) \le X^* \le F_{X^*}^{-1}\left(\frac{r_s^i}{S}\right)\middle| X = n\right) > 0.$$

*Proof.* Remembering $X^* = X + U - 1$ and $U$ is independent of $X$, we may write

$$\begin{aligned} P\left(F_{X^*}^{-1}\left(\frac{r_s^i-1}{S}\right) \le X^* \le F_{X^*}^{-1}\left(\frac{r_s^i}{S}\right)\middle| X = n\right) = \\ P\left(F_{X^*}^{-1}\left(\frac{r_s^i-1}{S}\right) \le X + U - 1 \le F_{X^*}^{-1}\left(\frac{r_s^i}{S}\right)\middle| X = n\right) = \\ P\left(1 - n + F_{X^*}^{-1}\left(\frac{r_s^i-1}{S}\right) \le U \le 1 - n + F_{X^*}^{-1}\left(\frac{r_s^i}{S}\right)\right). \end{aligned} \tag{3.6}$$

It suffices to decide under which conditions the derived probability for $U$ is positive. If the lower bound is higher than or equal to one, the corresponding probability is zero. Hence the $n$ must satisfy $F_{X^*}^{-1}\left(\frac{r_s^i-1}{S}\right) < n$. Because $n \in \operatorname{supp} X \subseteq \mathbb{N}_0$, this is equivalent to

$$n \ge L_X\left(\frac{r_s^i-1}{S}\right) = \begin{cases} 0 & r_s^i = 1, \\ F_X^{-1}\left(\frac{r_s^i-1}{S}\right) + 1 & \frac{r_s^i-1}{S} \in \operatorname{Ran} F_X, r_s^i \ne 1, \\ \left\lceil F_{X^*}^{-1}\left(\frac{r_s^i-1}{S}\right)\right\rceil = F_X^{-1}\left(\frac{r_s^i-1}{S}\right) & \frac{r_s^i-1}{S} \notin \operatorname{Ran} F_X, r_s^i \ne 1. \end{cases}$$

due to Theorem 1.3.

Using similar arguments, if the upper bound in (3.6) is non-positive, the probability is also zero, obtaining the necessary condition for $n$ that $1 + F_{X^*}^{-1}\left(\frac{r_s^i}{S}\right) > n$, which in turn using Theorem 1.3 yields

$$n \le \begin{cases} F_{X^*}^{-1}\left(\frac{r_s^i}{S}\right) = F_X^{-1}\left(\frac{r_s^i}{S}\right) & \frac{r_s^i}{S} \in \operatorname{Ran} F_X, \\ \left\lfloor F_{X^*}^{-1}\left(\frac{r_s^i}{S}\right) + 1\right\rfloor = F_X^{-1}\left(\frac{r_s^i}{S}\right) & \frac{r_s^i}{S} \notin \operatorname{Ran} F_X. \end{cases}$$

As a final piece of observation, notice that the size of the interval in (3.6) in which $U$ is contained is positive. This is provided by Theorem 1.3 because the extended quantile function is increasing on $(0, 1]$ and as consequence

$$F_{X^*}^{-1}\left(\frac{r_s^i - 1}{S}\right) < F_{X^*}^{-1}\left(\frac{r_s^i}{S}\right),$$

meaning that the probability is positive for every realization satisfying the derived bounds. □

The region (3.5) is always non-empty, but it can also contain multiple elements. Consequently, there are multiple sensible choices of realizations $n$ into which transform copula sample ranks. Problems occur when the number of possible realizations is large or infinite. However, the only practical limit is if we are able to iterate over every realization in a reasonable time. On a standard CPU, millions or even billions of iterations can be performed in a reasonable time. For our purposes, it suffices to define an arbitrary threshold, million for example, above which we consider a number of realizations large and below which we consider this number reasonable. We cover how to address these cases in separate paragraphs.

### 3.4.1 Case of a reasonable number of realizations

We identify two possible approaches to tackle this case. First is to transform ranks $r_s^i$ into a realization of $X$ that maximizes contribution to

$$P\left(L_X\left(\frac{r_s^i - 1}{S}\right) \leq X \leq F_X^{-1}\left(\frac{r_s^i}{S}\right)\right),$$

and the second case is similar, except we work with extension $X^*$ and we select realization of $X$ with the greatest contribution to

$$P\left(F_{X^*}^{-1}\left(\frac{r_s^i - 1}{S}\right) \leq X^* \leq F_{X^*}^{-1}\left(\frac{r_s^i}{S}\right)\right). \tag{3.7}$$

The former case is the easier one since according to Theorem 3.2 it translates to task

$$\begin{aligned} \max_{n \in \mathbb{N}_0} \quad & P(X = n) \\ \text{s.t.} \quad & L_X\left(\frac{r_s^i - 1}{S}\right) \leq n \leq F_X^{-1}\left(\frac{r_s^i}{S}\right), \end{aligned} \tag{3.8}$$

which could be easily solved by iterating over all feasible solutions. This is possible due to our assumptions.

Concerning the second case, expression (3.7) could be rewritten using the familiar law of total probability and independence of $U$ and $X$ to

$$\sum_{n \in \operatorname{supp} X} P\left(F_{X^*}^{-1}\left(\frac{r_s^i - 1}{S}\right) \leq X + U - 1 \leq F_{X^*}^{-1}\left(\frac{r_s^i}{S}\right) \middle| X = n\right) \cdot P(X = n) =$$
$$\sum_{n \in \operatorname{supp} X} P\left(1 - n + F_{X^*}^{-1}\left(\frac{r_s^i - 1}{S}\right) \leq U \leq 1 - n + F_{X^*}^{-1}\left(\frac{r_s^i}{S}\right)\right) \cdot P(X = n).$$

Summands can be easily computed using the distribution function $F_U$. Utilizing Theorem 3.2, we can transform ranks $r_s^i$ into the solution of the problem

$$\begin{aligned} \max_{n\in\mathbb{N}_0} \quad & P\left(1-n+F_{X^*}^{-1}\left(\frac{r_s^i-1}{S}\right)\leq U\leq 1-n+F_{X^*}^{-1}\left(\frac{r_s^i}{S}\right)\right)\cdot P(X=n) \\ \text{s.t.} \quad & L_X\left(\frac{r_s^i-1}{S}\right)\leq n\leq F_X^{-1}\left(\frac{r_s^i}{S}\right). \end{aligned} \tag{3.9}$$

### 3.4.2 Case of a large or infinite number of realizations

We start by pointing out that if the number of realizations is infinite, there is only one region in which there is potentially an infinite number of choices and this happens at the tail of a discrete distribution. Such discrete distributions usually have some structure that could be exploited. For example, probabilities of events in the tail could be monotonously decreasing, making the selection according to the previous section possible. If the number of realizations outside the tail is reasonable in the sense we defined earlier, we can use the approaches described in the previous section.

Unfortunately, in the case of general discrete distribution, we cannot exploit any such property. When this happens, we believe the safest approach would be to use conditional median or expectation of $X$ on region defined by (3.5). Written down in mathematical notation, it is possible to choose from

1. $\operatorname{med}\left(X \;\middle|\; L_X\left(\frac{r_s^i-1}{S}\right)\leq X\leq F_X^{-1}\left(\frac{r_s^i}{S}\right)\right)$,
2. $\mathsf{E}\left[X \;\middle|\; L_X\left(\frac{r_s^i-1}{S}\right)\leq X\leq F_X^{-1}\left(\frac{r_s^i}{S}\right)\right]$.

Conditional expectation can be non-integral and thus would have to be rounded, or left unrounded if relaxation to real numbers is not an issue. However, calculating the conditional expectation can be quite difficult. In some rare cases, the conditional expectation might be infinite. Because of this difficulty, we believe the conditional median is the better option.

### 3.4.3 Concluding remarks

Up until now, we have worked with general extension $U$. We have shown previously in Theorem 1.6 that the extension copula does not depend on extension types. However, if we decide to use approach (3.9), we need to choose the distribution of $U$. We see no reason to select distribution other than uniform on $[0,1]$. Referring to the previous example, choosing another distribution would mean that we assign greater weight to some portions of the transformation region (3.4). This contradicts the principle of extension copula which spreads the probability associated with each realization evenly.

If the number of realizations is reasonable, the size of the region (3.4) converges to zero as $S\to\infty$. As a consequence, number of realizations which satisfy condition (3.5) is reduced to one for large enough $S$. Therefore the methods (3.8) and (3.9) are almost equivalent for sufficiently large $S$.

Where we see the potential difference is when $S$ is small. Imagine, for example, that after evaluating bounds of (3.4) we arrive at an interval [1.2, 2.9]. Informally,

this would mean that we used 80% and 10% of probability associated with 2 and 3, respectively, to construct a scenario that is either 2 or 3. Yet, when using approach (3.8), realization 3 could still be selected if its probability is larger than that of 2, even if the difference in probabilities is small. The second suggested approach (3.9) takes into consideration the proportions of probabilities used. It weighs the probability of each realization with the proportion used. This leads us to think that this approach can perform slightly better for smaller $S$, although a computational study would have to be done to support this claim. Therefore, we opt for the approach (3.9) in our implementation of the algorithm.

We added discussion about a large or infinite number of realizations mainly due to completeness reasons. In reality, we expect these cases to be quite rare. There are basically two options. Either the distribution is entered exactly into the algorithm, or the historical data is used. If historical data is used, empirical distribution and quantile functions are used and thus, each realization would have to be accompanied by at least one data point which contains it. Hence, we expect the case of a large amount of realizations to be rare. Even so, the conditional median approach can be used without problem. An example of the use of exact distribution could be when we want to estimate the parameter of Poisson distribution from the historical data of a number of events, and then use a Poisson distribution with the estimated parameter for scenario generation. The proposed algorithm allows this option and described approaches for the transformation of copula samples could be used. However, it is debatable if there is a benefit to modeling such distributions with a large or infinite number of realizations as discrete and not opting for relaxation to real numbers instead.

# 4. Case study

## 4.1 Methodology

To the extent of our knowledge, there are no other easy-to-use distribution-oriented methods for scenario generation for discrete data other than sampling. Therefore we aim to compare the developed method with sampling and one problem-oriented method. We include the problem-oriented method in our analysis to see how the developed method compares to some problem-oriented approaches and if it can reach the same quality. Comparison will be performed using metrics and visual assessment described in the following section.

### 4.1.1 Quality measures of scenario generation methods

The general approach we will follow is to generate several scenario sets of the same size using different runs of the scenario generation method. Then, several metrics will be defined to assess the quality of the scenario generation method. Some methods are fundamentally random, a typical example being sampling. In such cases, different runs of scenario generation method give different scenario sets. However, some methods contain little to no randomness. The method we have described and expanded in previous chapters has this property. Each time we run these methods for a given number of scenarios, we obtain the same or very similar scenario sets, making evaluation of such sets difficult, since any defined metric will return the same or similar values for each generated scenario set. To resolve this issue, we adopted the approach to split data into separate folds of equal size, and then run the scenario generation method on each of these folds. This ensures that variability is inserted into generated scenario sets, making metric evaluation more sensible.

We define several metrics and one visual assessment to judge the quality of scenario generation methods. We cover each separately in each paragraph. But firstly let us introduce some notation.

**Notation.** Let $f$ be an objective function and by $f(x, \tau)$ define evaluation of solution $x$ using objective $f$ and scenario set $\tau$. Our general setting is then $\max_{x \in \mathcal{M}} f(x, \tau)$ for some set of feasible solutions $\mathcal{M}$. Usually, the set of feasible solutions is influenced by the choice of scenario set. However, very often violations of the constraints are transferred to the objective function using penalization. For our purposes, it suffices to assume that the set of feasible solutions is the same regardless of the chosen scenario set.

In theory, we are trying to capture the "true" distribution of randomness using some scenario set $\tau$. In practice, this distribution is rarely known and we can only approximate it, for example by using historical data. For our purposes, we assume that this "true" distribution is captured by a scenario set $\eta$, and in the context of this case study, it means exclusively all generated random data. We refer to $f(x, \eta)$ as *out-of-sample evaluation* and to $f(x, \tau)$ as *in-sample evaluation* where $\tau$ is a generated scenario set which is supposed to approximate $\eta$.

For each number of scenarios from one to $N$, we run the scenario generation method $K$-times to obtain scenario sets either by using different randomization in each run or by splitting data into $K$ folds. Collect these sets into sequence $\{\mathcal{T}_i\}_{i=1}^{N}$, i.e. each $\mathcal{T}_i$ contains $K$ scenario sets of size $i$. Some computational studies only focus on properties of scenario sets at "true" optimal solution, i.e. solution to problem $\max_{x\in\mathcal{M}} f(x,\eta)$. However, such problems are usually unsolvable. Instead, heuristic solutions might be generated to attempt to solve this problem. For this reason, we put emphasis on assessing the quality of scenario generation methods using a collection of heuristic solutions $\mathcal{X} \subseteq \mathcal{M}$. Such an approach was advocated, for example, by Prochazka and Wallace [2020].

**In-sample stability.** A reasonable scenario generation method should create scenario sets that provide similar in-sample evaluations, regardless of individual runs. There are several options to measure the in-sample stability of scenario sets. Sensible choices are variance, MSE, or RMSE. We use the maximum relative difference between the largest and lowest evaluations, averaged on a pool of solutions

$$ST_n = \frac{1}{|\mathcal{X}|}\sum_{x\in\mathcal{X}} \frac{\max_{\tau\in\mathcal{T}_n} f(x,\tau) - \min_{\tau\in\mathcal{T}_n} f(x,\tau)}{\min_{\tau\in\mathcal{T}_n} f(x,\tau)}.$$

A similar measure appeared in Zhang et al. [2021], but we use different methodology to construct sets $\mathcal{T}_n$ and we focus also on heuristic solutions.

**Out-of-sample evaluation gap.** If approximation of the "true" distribution is reasonable using generated scenario set $\tau$, it should hold $f(x,\tau) \simeq f(x,\eta)$ for a wide variety of solutions $x \in \mathcal{X}$. To measure the discrepancy between those evaluations, we use metric

$$EG_n = \frac{1}{|\mathcal{X}|}\sum_{x\in\mathcal{X}} \sqrt{\frac{1}{K}\sum_{\tau\in\mathcal{T}_n}\left(\frac{f(x,\tau) - f(x,\eta)}{f(x,\eta)}\right)^2}.$$

It is the root square error of relative differences between in-sample and out-of-sample evaluations, averaged on a pool of heuristic solutions $\mathcal{X}$. Convergence of metric to zero for $n \to \infty$ implies consistency and unbiasedness of scenario generation method. In this context, the metric incorporates bias and consistency.

**Optimality gap.** If the size of the scenario set is reasonable, it is possible to solve problems $\max_{x\in\mathcal{M}} f(x,\tau), \tau \in \mathcal{T}_n$. Denote optimal solutions to these problems $x^*_\tau$ and denote $x^*$ "true" optimal solution to $\max_{x\in\mathcal{M}} f(x,\eta)$. Assuming the scenario sets are constructed well, out-of-sample evaluations should be similar for both solutions, i.e. $f(x^*_\tau,\eta) \simeq f(x^*,\eta)$. We can measure this gap using

$$OG_n = \frac{1}{K}\sum_{\tau\in\mathcal{T}_n} \frac{f(x^*,\eta) - f(x^*_\tau,\eta)}{f(x^*_\tau,\eta)}.$$

This measure implicitly assumes that the optimal solution $x^*$ can be found. In practical applications, this is not usually the case, otherwise, scenario generation would not be necessary. However, we construct the case study such that it can be found, as this measure provides important insight into the behaviour of a scenario generation method.

**Ranking and visual assessment.** Suppose we have two solutions $x, y \in \mathcal{X}$ satisfying relationship $f(x, \eta) \geq f(y, \eta)$. Reasonable scenario set $\tau$ should preserve this ranking as many times as possible. In other words, we want $f(x, \tau) \geq f(y, \tau)$ to be satisfied for as many pairs $(x, y)$ satisfying $f(x, \eta) \geq f(y, \eta)$ as possible.

We follow the terminology used in Prochazka and Wallace [2020]. Let $\mathcal{X}$ contain $M \in \mathbb{N}$ solutions. We rank elements of $\mathcal{X}$ by their out-of-sample evaluation $f(x_{(1)}, \eta) \geq \ldots \geq f(x_{(M)}, \eta)$, assuming $\mathcal{X} = \{x_{(1)}, \ldots, x_{(M)}\}$. We plot these out-of-sample evaluations in decreasing order. Then, a scenario set $\tau$ is produced, and in-sample evaluations $f(x_{(i)}, \tau), i = 1, \ldots, M$, are plotted in the same order as for out-of-sample evaluations. We expect following to happen if $\tau$ is constructed well

1. In-sample evaluations $f(x_{(i)}, \tau)$ will form a generally decreasing sequence with few breaks of monotony.

2. If monotony is broken for pair $x_{(i)}$ and $x_{(i+1)}$, difference in their in-sample evaluations is not big. Equivalently, difference $f(x_{(i+1)}, \tau) - f(x_{(i)}, \tau)$ is small.

3. In-sample evaluations should approximate well out-of-sample evaluations. This could be written as $f(x_{(i)}, \tau) \simeq f(x_{(i)}, \eta)$.

In theory, the last point is unnecessary if the ranking is perfect. This is because the optimal solution to task $\max_{x \in \mathcal{M}} f(x, \tau)$ would also be the optimal solution to $\max_{x \in \mathcal{M}} f(x, \eta)$. Both problems would differ just by monotonic transformation and the "true" objective value could be obtained by out-of-sample evaluation. Therefore we also compute Kendall's rank correlation coefficient, described below the Definition 1.4, between in-sample and out-of-sample evaluations to assess how the constructed scenario sets preserve ranking. Kendall's $\tau$ focuses purely on the order of the two sequences without taking into consideration whether the values are close to each other or not. Ideally, this coefficient should be close to one.

For a given number of scenarios, we average this coefficient across multiple generated scenario sets using the methodology as in the previous paragraphs. In other words, for a given number of scenarios $n$, we take scenario set $\tau \in \mathcal{T}_n$ and use it to compute in-sample evaluations on a pool of solutions $\mathcal{X}$. Then we compute Kendall's rank correlation coefficients between these evaluations and their associated out-of-sample values. Finally, we average these coefficients across all possible scenario set choices $\tau \in \mathcal{T}_n$.

### 4.1.2 Problem-oriented method

We employ a problem-oriented method for scenario generation based on the framework introduced by Prochazka and Wallace [2020]. The core idea is to find a scenario set that minimizes the discrepancy between in-sample and out-of-sample evaluations on a pool of heuristic solutions. This is achieved by defining a metric that should somehow encompass various properties that are desired of scenario sets, and then finding a scenario set that minimizes it. Prochazka and Wallace [2020] proposed selection of scenario set by solving the problem

$$\min_{\tau} L(\tau; \mathcal{X}) := \sum_{x \in \mathcal{X}} \left(\alpha \cdot \mathbf{1}[f(x, \tau) > f(x, \eta)] + \beta \cdot \mathbf{1}[f(x, \tau) < f(x, \eta)]\right) \cdot \left(f(x, \tau) - f(x, \eta)\right)^2 \quad (4.1)$$

The minimized metric is essentially a weighted squared error between in-sample and out-of-sample evaluations. Weights assign different importance to in-sample evaluations which exceed or under-estimate out-of-sample evaluations. It was argued in Prochazka and Wallace [2020] that the former case is less desirable. Therefore more weight should be put on in-sample evaluations exceeding out-of-sample evaluations, manifesting in $\alpha > \beta$. Argument is that over-estimation of out-of-sample evaluation may lead to optimization algorithm selecting solution which is in reality worse than it appears. This, of course, happens at the expense of selecting a solution that performs better in terms of out-of-sample evaluation but appears to be worse due to the imprecision of in-sample evaluation.

Unfortunately, the proposed metric is usually non-linear and non-convex, rendering it more difficult to solve this problem exactly. In the aforementioned article, analysis was performed on a variant of the stochastic knapsack problem, and the problem was solved using sub-gradient descent. Sub-gradient methods work reasonably well for continuous variables, but for discrete variables, the integrality gets lost. In contrast to Prochazka and Wallace [2020], we would like to preserve the integrality of scenario variables. Therefore we solve (4.1) heuristically using simulated annealing algorithm. We shall see that solving (4.1) with simulated annealing yields sufficiently good results. We describe the simulated annealing in Appendix A.

Although the simulated annealing approach to solving (4.1) is fundamentally random, the problem itself is the same. We thus expect individual runs of the algorithm to produce similar scenario sets for a fixed scenario set size. Therefore we have decided to adopt the same strategy of separating data into folds of equal size, an approach we have discussed in the previous section for the copula-based method.

## 4.2 Stochastic knapsack

The Knapsack problem is a traditional optimization problem where we want to maximize the value of items we put into a knapsack such that the overall weight of items does not exceed its capacity. We can modify this problem to a stochastic one by making the appearance of items and prices uncertain. However, we have to decide which items to put into the knapsack before the information whether the item appeared or not is revealed. This constitutes a two-stage stochastic optimization problem. In the first stage, we decide in advance what items we try to put into the knapsack, and in the second stage the items appear or not, and the overall value of the knapsack is calculated. We assume that we have historical data of appearances and prices at our disposal to base our decisions.

The analysis will be performed on two versions of stochastic knapsack. In the first version we consider that only the appearances of items are uncertain and in the second that also prices of those items are uncertain. These versions represent problems with discrete and mixed random data, respectively.

### 4.2.1 Data generation

We generate data for item appearances as follows. Assume we want to generate $n$-dimensional random vectors with binary margins. We start by generating a

$(n+1) \times (n+1)$ positive-semidefinite matrix $C$ with values in the range $[-1, 1]$ and ones on the diagonal. Let $\text{sign}(x)$ denote sign of $x$, i.e. $\text{sign}(x) = 1$ when $x \geq 0$, and $\text{sign}(x) = -1$ otherwise. Then for a given number of samples to generate, we do the following

- Sample $(y_1, \ldots, y_{n+1})$ from the normal distribution $\mathcal{N}_{n+1}(0, C)$.
- Set $x_i = \text{sign}(y_i) \cdot \text{sign}(y_{n+1})$ for $i = 1, \ldots, n$.
- Return $(x_1, \ldots, x_n)$.

Let $f(x) = \frac{2}{\pi} \arcsin x$. We state that the resulting vectors $(x_1, \ldots, x_n)$ are samples from the distribution of binary vector $X$ with matrix of second moments equal to $f(C_n)$, where $C_n$ denotes matrix $C$ with deleted $(n+1)$-th row and column, and the function $f$ is applied element-wise. We define the matrix of second moments of $X$ as a matrix consisting of elements $\mathsf{E}\, X_i X_j$ where $i, j = 1, \ldots, n$. Furthermore, for margins of $X$ we have $\mathsf{E}\, X_i = f(c_{i,n+1}) = f(c_{n+1,i})$. We do not provide proof for this assertion, but the main idea follows the proof of Theorem 3.1 from Goemans and Williamson [1995].

The margins of the resulting vectors are either plus or minus one. We can easily obtain margins that are either zero or one by transforming minus one to zero. These binary vectors are used as the historical data for item appearance. The matrix $C$ was generated such that the samples from $\mathcal{N}_{n+1}(0, C)$ are strongly correlated. As a consequence, the item appearances are also strongly correlated. Therefore, the described procedure ensures there are strong associations between item appearances.

Prices of items were generated in two steps. The first stage is to sample $\mathcal{N}(\mu, \Sigma)$ with a priori specified means $\mu$ and covariance matrix $\Sigma$. Denote $X_i$ binary random variable of appearance of item $i$: 1 if item appears and 0 otherwise. In the second stage, we transform $C$ into

$$\tilde{C}_i = C_i + \frac{\sum_{i=1}^{n} (1 - X_i) C_i}{|\{1 \leq i \leq n : X_i = 0\}|} \cdot \mathbf{1}[X_i = 1],$$

where $n$ is the target number of items for which we generate data. The main idea is to split evenly prices of items that did not appear amongst items that appeared. This inserts dependence into the relationship between appearances and prices and also creates distributions of prices that are more complicated and multi-modal.

### 4.2.2 Problem formulation

To formulate an optimization problem, we define scenario set $\mathcal{S}$ and variables $q_i^s$ as

$$q_i^s = \begin{cases} 1 & \text{if item } i \text{ appears in scenario } s, \\ 0 & \text{otherwise,} \end{cases}$$

and $c_i^s$ which has interpretation of price of item $i$ in scenario $s$, and probabilities $p^s$ associated with each scenario. Assume that we obtain these scenarios $\mathcal{S}$ using some scenario generation method. Furthermore, we define the parameters of the stochastic knapsack problem in Table 4.1. The number of items was set to 20 and

| Notation | Meaning |
|---|---|
| $K$ | Total number of items |
| $c_i$ | Price of item $i$ |
| $w_i$ | Weight of item $i$ |
| $W$ | Capacity of knapsack |
| $Q$ | Penalty for exceeded weight |

Table 4.1: Notation for stochastic knapsack problem. Prices $c_i$ are only defined for the case of certain prices.

the capacity of the knapsack was set to half of the total weight of all the items.

We present the formulation of the stochastic knapsack problem with uncertain appearances and fixed prices using first-stage binary decisions $x_i$ (decision to try put item $i$ into knapsack) as

$$\begin{aligned} \max_{x_i,\, e_s} \quad & \sum_{s\in\mathcal{S}} p^s \left( \sum_{i=1}^{K} c_i x_i q_i^s - Q e_s \right) \\ \text{s.t.} \quad & \sum_{i=1}^{K} w_i x_i q_i^s \leq W + e_s \quad s \in \mathcal{S}, \\ & x_i \in \{0,1\} \qquad i = 1,\ldots,K, \\ & e_s \geq 0 \qquad\quad s \in \mathcal{S}. \end{aligned} \tag{4.2}$$

A similar problem appeared in Prochazka and Wallace [2020]. The objective function of this problem is just the expected value of the knapsack over constructed scenarios, penalized if the weight of the knapsack is exceeded. The crucial term in the objective is $x_i q_i^s$. It is equal to one if we try to put item $i$ into the knapsack and it becomes available under scenario $s \in \mathcal{S}$. If we do not try to put it into the knapsack or we do and the item does not appear, this term is zero, meaning that the associated value of the item is not added to the value of the knapsack. The first constraint checks the overall weight of items in the knapsack in scenario $s \in \mathcal{S}$. Whenever the maximum weight of the knapsack is exceeded in some scenario, the objective function is penalized using exceeded weight $e_s$ and penalty $Q$. Objective is maximized using binary values $x_i$ and weight excesses $e_s$.

Formulation of the problem with uncertain prices is similar to (4.2), except we replace $c_i$ by their scenario values $c_i^s$

$$\begin{aligned} \max_{x_i,\, e_s} \quad & \sum_{s\in\mathcal{S}} p^s \left( \sum_{i=1}^{K} c_i^s x_i q_i^s - Q e_s \right) \\ \text{s.t.} \quad & \sum_{i=1}^{K} w_i x_i q_i^s \leq W + e_s \quad s \in \mathcal{S}, \\ & x_i \in \{0,1\} \qquad i = 1,\ldots,K, \\ & e_s \geq 0 \qquad\quad s \in \mathcal{S}. \end{aligned}$$

**Heuristic solutions generation.** Looking at the problem formulations, we see that the decision vector $x$ is binary. This is also the case for generated data of item appearances. We therefore opt for generating a pool of item appearance

data from the same distribution and using it as a source of heuristic solutions. This pool is separate from the item appearance data we used to generate scenarios from.

### 4.2.3 Problem-oriented scenario generation

Let us now discuss details of how we approach solving problem-oriented task (4.1) for the case of stochastic knapsack. In problem formulations, $q_i^s$ and $c_i^s$ are variables representing information about item appearances and prices, respectively, and thus, the problem is solved with respect to these variables. We symbolically write metric (4.1) as $L(q^s, c^s; \mathcal{X})$, meaning that it depends on the variables $q_i^s$ and $c_i^s$.

The most important part of the simulated annealing algorithm is the perturbation of solutions. We first fix the number of scenarios $S$ and sample $S$-times without replacement from generated input data. We use this sample as the initial solution to the algorithm. Perturbation of solutions can be summarized into three steps

1. For 1% of variables $q_i^s$ perform negation $\tilde{q}_i^s \leftarrow 1 - q_i^s$.
2. For 1% of variables $q_i^s$ select another variable in the same scenario $q_j^s$ and switch them: $\tilde{q}_i^s \leftarrow q_j^s$ and $\tilde{q}_j^s \leftarrow q_i^s$.
3. In the case of uncertain prices, compute the gradient of $L$ with respect to $c_i^s$ using the updated appearances $\tilde{q}_i^s$ and perform the update

$$\tilde{c}_i^s \leftarrow c_i^s - \alpha \frac{\partial L(\tilde{q}^s, c^s; \mathcal{X})}{\partial c_i^s}.$$

   for all $i = 1, \ldots, K$ and predefined step size $\alpha$.

The new perturbed solution $\tilde{q}_i^s$ and $\tilde{c}_i^s$ is then evaluated and the algorithm proceeds as described Appendix A. We repeat this perturbation in every iteration of the simulated annealing algorithm.

### 4.2.4 Analysis results

The analysis is performed based on metrics and evaluations of the quality of scenario generation methods presented in Section 4.1.1. For each number of scenarios, we used 10 generated scenario sets. For the sampling method, they were sampled from the input data without replacement. For copula-based and problem-oriented methods we used separation into 10 separate folds. Evaluation of in-sample stability and out-of-sample evaluation gap was performed over a pool of 100 heuristic solutions. For ranking and visual assessment, we used 500 heuristic solutions. The pool of heuristic solutions used for evaluation is different from the one we used to generate problem-oriented scenarios.

Let us start analyzing the quality of the copula-based method with the assessment of in-sample stability. The evolution of in-sample stability measure with an increasing number of scenarios is presented in Figure 4.1. First of all, we see that sampling significantly lags behind copula-based and problem-oriented

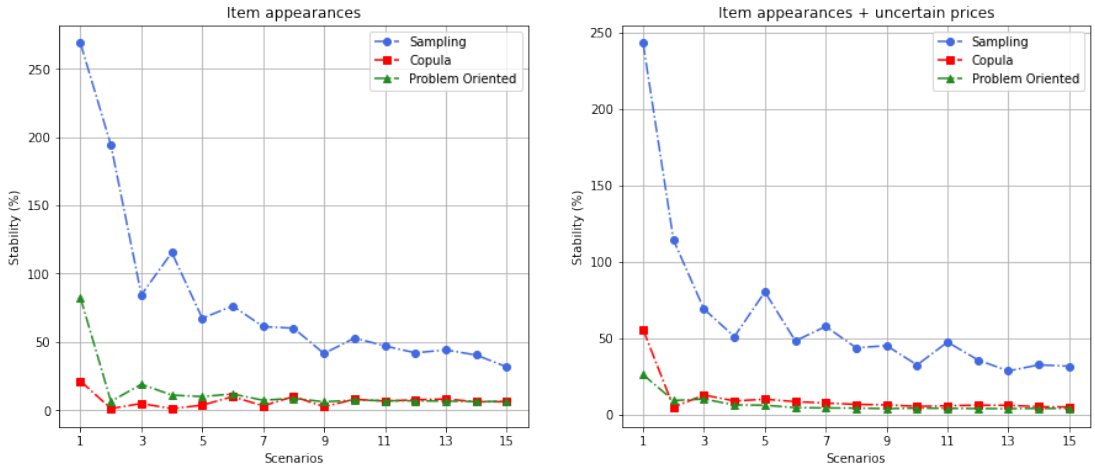



Figure 4.1: In-sample stability measure of sampling, copula-based and problem-oriented methods with an increasing number of scenarios.

methods in terms of in-sample stability. The initial (in)stability of sampling is much higher compared to other methods. Convergence of sampling's in-sample stability is slow and it takes tens of times many scenarios to achieve the same in-sample stability which copula-based and problem-oriented methods achieved in two scenarios. This holds for both versions of the stochastic knapsack problem. Copula-based and problem-oriented methods are comparable in terms of this measure. They quickly converge to the in-sample stability of under 10%. For the case of uncertain prices, the problem-oriented methods perform slightly better, but the difference is negligible.

A similar story occurs when we shift our attention to the out-of-sample evaluation gap. See Figure 4.2. The same observations apply. Sampling performs

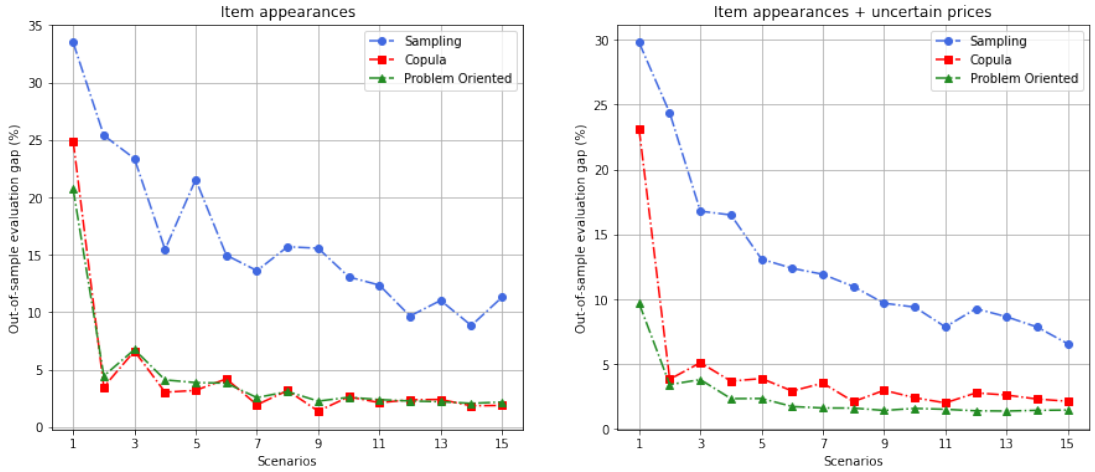



Figure 4.2: Comparison of sampling, copula-based and problem-oriented methods on out-of-sample evaluation gap.

significantly worse than both other methods. Convergence of sampling is slow, meanwhile, it takes just four scenarios to reach an out-of-sample evaluation gap under 5% for copula-based and problem-oriented methods for both variants of the stochastic knapsack problem. For the uncertain item appearances, copula-based

and problem-oriented methods perform virtually the same. If we add uncertain prices, we can see that the problem-oriented method converges faster to zero, although not significantly. Still, the copula-based method outperforms sampling in this area as well.

The third metric we mentioned in Section 4.1.1 is the optimality gap. This metric is shown in Figure 4.3. Here we see that the convergence of the optimality

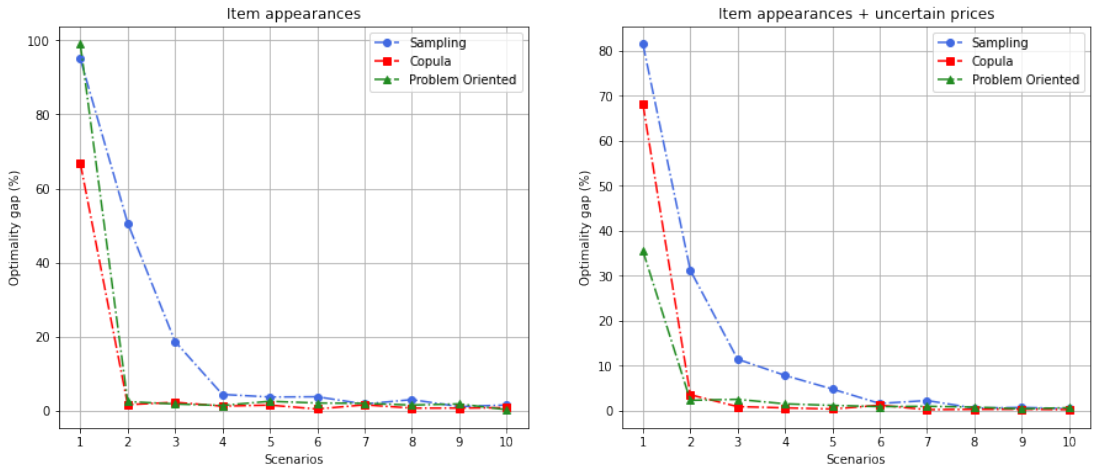



Figure 4.3: Optimality gap evolution for stochastic knapsack problem.

gap for sampling is faster than in the previous cases. Despite this, it is not enough to reach the quality of copula-based and problem-oriented methods on a small number of scenarios. The difference between one and two scenarios is striking for these methods. In essence, these results indicate that both versions of the stochastic problem can be solved using only two scenarios, while expecting that the computed solutions will be within 3% and 5% of the true optimal solution, respectively, for both variants of the stochastic knapsack problem. The same optimality gap is achieved for sampling at the sixth scenario in both variants of the problem. After the second scenario, the copula-based and problem-oriented methods perform almost the same. But for a low number of scenarios they both outperform sampling. For a large number of scenarios, the differences between all methods were mitigated.

For ranking assessment, we provide Figure 4.4 and Figure 4.5 which were created according to the methodology specified in Section 4.1.1. We can see that the in-sample evaluations for the sampling method are quite noisy. Ranking is hardly preserved and in-sample evaluations do not approximate well out-of-sample evaluations. Even when the number of scenarios is increased, the improvement is slow compared to copula-based and problem-oriented methods. On the other hand, copula-based and problem-oriented methods provide very good out-of-sample approximations, and ranking is well preserved. The fit is reasonably good even for five scenarios and it gets better with an increasing number of scenarios. We can observe an almost perfect fit for 15 scenarios. We conclude that the copula-based method outperforms sampling and is on par with the problem-oriented method.

We conclude the analysis by presenting Table 4.2 and Table 4.3 with the average Kendall rank correlation coefficients between the in-sample and out-of-sample evaluations on both versions of the stochastic knapsack problem. The coefficients were computed for all described methods and various numbers of scenarios. We

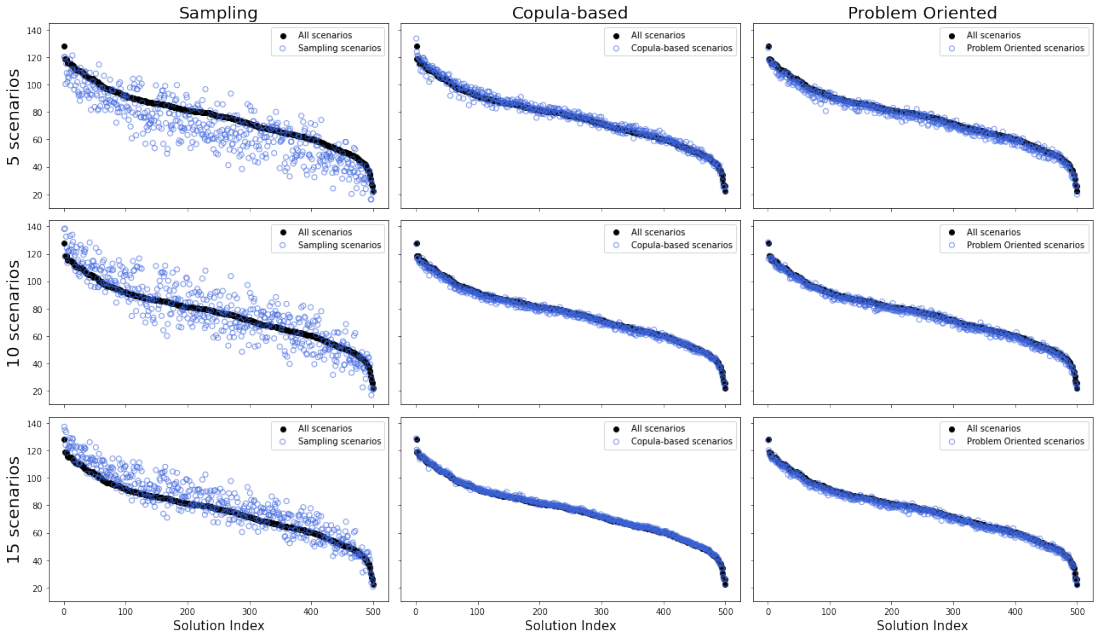



Figure 4.4: Visual assessment of ranking for the stochastic knapsack problem with uncertain appearances and fixed prices.

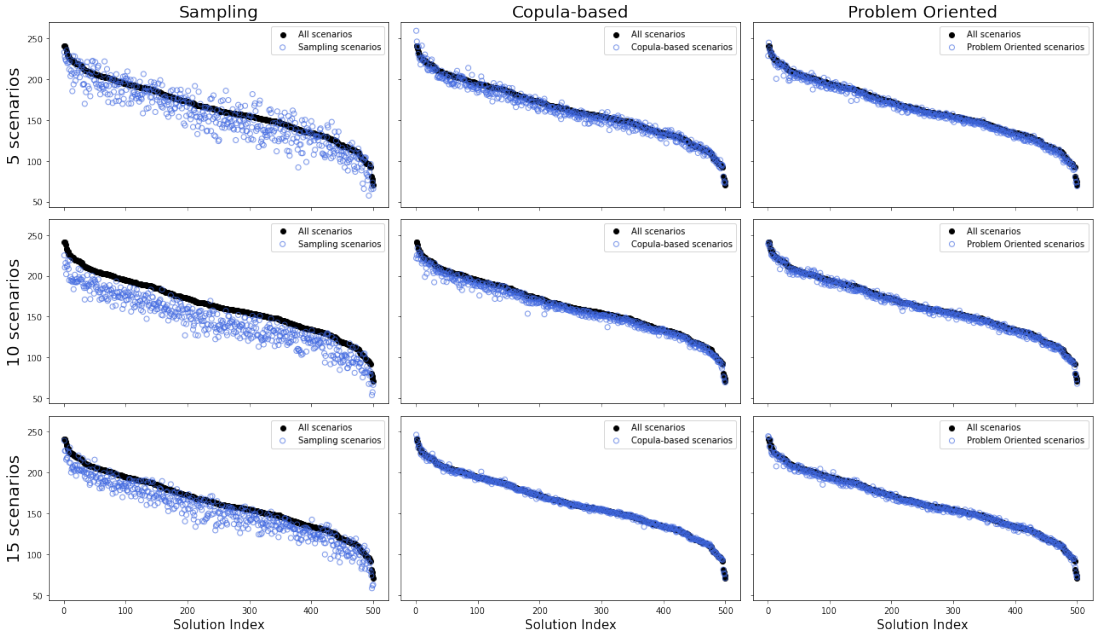



Figure 4.5: Visual assessment of ranking for the stochastic knapsack problem with uncertain appearances and prices.

| Number of Scenarios | Sampling | Copula-based | Problem-oriented |
|---|---|---|---|
| 5 scenarios | 0.729 | 0.935 | 0.931 |
| 10 scenarios | 0.750 | 0.952 | 0.953 |
| 15 scenarios | 0.835 | 0.969 | 0.959 |
| 20 scenarios | 0.898 | 0.968 | 0.961 |
| 25 scenarios | 0.901 | 0.972 | 0.962 |

Table 4.2: Average Kendall's $\tau$ between in-sample and out-of-sample evaluations on the stochastic knapsack problem with uncertain item appearances.

| Number of Scenarios | Sampling | Copula-based | Problem-oriented |
|---|---|---|---|
| 5 scenarios | 0.758 | 0.905 | 0.939 |
| 10 scenarios | 0.817 | 0.945 | 0.954 |
| 15 scenarios | 0.844 | 0.954 | 0.961 |
| 20 scenarios | 0.896 | 0.960 | 0.958 |
| 25 scenarios | 0.900 | 0.961 | 0.962 |

Table 4.3: Average Kendall's $\tau$ between in-sample and out-of-sample evaluations on the stochastic knapsack problem with uncertain item appearances and prices.

can immediately see that sampling has the lowest coefficient values. Additionally, the sampling cannot reach the same coefficient values with 25 scenarios as the copula-based and problem-oriented methods achieved with just five. In the case of the stochastic knapsack version with uncertain item appearances and fixed prices, the copula-based method slightly outperforms the problem-oriented method in terms of this measure. Concerning the second version of the stochastic knapsack problem, the problem-oriented method beats initially the copula-based method. Nonetheless, the copula-based method improved with an increasing number of scenarios until it eventually reached the quality of the problem-oriented method.

# Conclusion

In this work, we have presented a new copula-based algorithm for scenario generation for discrete random data. This algorithm is based on the method introduced by Kaut [2014] which we have covered in a separate chapter while providing alternative formulations and explanations. We have introduced the fundamentals of copulas theory and extensions of discrete random variables. We have also contributed by introducing an explicit formula for the extension copulas of mixed random vectors, which we have subsequently proven in detail. Following the description of the new algorithm, we added an analysis that showed that our method significantly outperforms sampling in terms of all the evaluations of quality we have discussed, namely in-sample stability, out-of-sample evaluation gap, optimality gap, ranking, and visual assessment. Additionally, we have shown that the method can attain similar quality as some problem-oriented approaches, although we admit more effort could have been invested into the development of the problem-oriented method described in Section 4.1.2. However, this just proves our argument that problem-oriented methods require careful handling and a significant amount of resources to develop compared to distribution-oriented methods.

Based on our analysis, the proposed algorithm appears to be promising for scenario generation whenever discrete data is involved. This method is easy to use once the algorithm is implemented, as it only requires processed data at the input, making it in our opinion particularly promising for wider use in stochastic optimization. This is even more pronounced when we take into consideration the lack of better distribution-oriented alternatives to sampling when discrete data is involved. The algorithm is, of course, what we consider to be the most important contribution in this work.

A drawback of the proposed method is the computational complexity. We mentioned that running the algorithm takes $\mathcal{O}(n^2S^2)$ operations. Therefore, the algorithm is intractable when large datasets with many variables are involved, or when the number of scenarios to generate is too large. Reducing the computational complexity is a potential path for our future research. Alternative definitions of copula samples are possible. For example, ranks no longer have to be equally spaced. We believe this approach can capture more properly the structure that extensions impose on extension copula, potentially reducing the time complexity and improving performance.

The analysis was performed on an artificially created problem. Although the introduced method outperforms sampling on this problem, real-world problems are usually more complex and more difficult to solve. Therefore we believe that comparing the newly developed method on real-world problems with sampling and other scenario generation methods would provide more insight into the properties and performance of this method. We aim to perform such analysis in the future. Also, we strive to make the implementation of the algorithm publicly available.

# A. Simulated annealing algorithm

We present the simulated annealing algorithm in Algorithm 2. The goal is to minimize objective function $f$, starting with some initial solution $S$. The initial and final temperatures, the number of iterations, and the dead-end counter have to be specified at the start of the algorithm.

**Algorithm 2** Simulated annealing algorithm

**Input:** Objective function $f$, initial solution $S$, initial temperature $T_0$, final temperature $T_1$, number of iterations $I$, dead-end counter threshold $D_0$.

**Output:** Solution $\hat{S}$ minimizing $f$

**Initialize:** $\hat{S} \leftarrow S$, $\hat{C} \leftarrow f(S)$, $\gamma \leftarrow \sqrt[I]{T_1/T_0}$, $D \leftarrow 0$

1: **for** $i = 1, \ldots, I$ **do**
2: $\quad \tilde{S} \leftarrow \text{PERTURBATE}(S)$, $\tilde{C} \leftarrow f(\tilde{S})$, $\Delta C \leftarrow \tilde{C} - \hat{C}$
3: $\quad$ **if** $\Delta C < 0$ **then**
4: $\quad\quad \hat{S} \leftarrow \tilde{S}$, $\hat{C} \leftarrow \tilde{C}$, $S \leftarrow \tilde{S}$
5: $\quad$ **else**
6: $\quad\quad T \leftarrow T_0 \gamma^i$
7: $\quad\quad$ **if** $U < e^{-\Delta C/T}$ **then**
8: $\quad\quad\quad S \leftarrow \tilde{S}$
9: $\quad\quad$ **else**
10: $\quad\quad\quad D \leftarrow D + 1$
11: $\quad\quad\quad$ **if** $D \geq D_0$ **then** $S \leftarrow \hat{S}$, $D \leftarrow 0$
12: $\quad\quad$ **end if**
13: $\quad$ **end if**
14: **end for**
15: **return** Solution $\hat{S}$

We initialize the algorithm by setting the best-known solution $\hat{S}$ to $S$, and the best-known objective $\hat{C}$ to $f(S)$. We also initialize the dead-end counter and cooling parameter $\gamma$. In each iteration of the algorithm, we use the perturbation function on the currently accepted solution $S$, obtaining the candidate solution $\tilde{S}$ and its objective value $\tilde{C}$. At Line 3, if the candidate solution is better, we update the current best-known solution and objective and also set the currently accepted solution $S$ to the best-known solution. If the candidate solution fails to outperform the currently best-known solution, we might still accept it. To decide whether we accept it, a random sample from the uniform distribution on $[0, 1]$ is performed. If the sampled value is lower than the computed acceptance probability, we accept this solution. Otherwise, we increase the dead-end counter. Finally, if the dead-end counter exceeds the default threshold $D_0$, we set the currently accepted solution to the best-known solution. This prevents the algorithm from spending too many iterations on exploring unperspective solutions.

The acceptance probability decreases geometrically, meaning that the probability of accepting worse solutions than the currently best-known is also decreasing geometrically. Rather informally, in initial iterations, we search through a wide variety of solutions by temporarily accepting and exploring worse solutions. In the later stages of the algorithm, we focus on improving the best-known solution, as the probability that a worse solution is accepted converges to zero.

# Bibliography


Michael O. Ball, Charles J. Colbourn, and J. Scott Provan. Chapter 11 network reliability. 7:673–762, 1995. ISSN 0927-0507.

Russell Bent and Pascal Van Hentenryck. Scenario-based planning for partially dynamic vehicle routing with stochastic customers. *Operations Research*, 52: 977–987, 12 2004.

Michel Denuit and Philippe Lambert. Constraints on concordance measures in bivariate discrete data. *Journal of Multivariate Analysis*, 93(1):40–57, 2005. ISSN 0047-259X.

Fabrizio Durante, Juan Fernández-Sánchez, and Carlo Sempi. A topological proof of sklar's theorem. *Applied Mathematics Letters*, 26(9):945–948, 2013. ISSN 0893-9659.

Jamie Fairbrother, Amanda Turner, and Stein W Wallace. Scenario generation for single-period portfolio selection problems with tail risk measures: coping with high dimensions and integer variables. *INFORMS Journal on Computing*, 30(3):472–491, 2018.

Christian Genest and Johanna Nešlehová. A primer on copulas for count data. *ASTIN Bulletin: The Journal of the IAA*, 37(2):475–515, 2007.

Christian Genest, Johanna G. Nešlehová, and Bruno Rémillard. On the empirical multilinear copula process for count data. *Bernoulli*, 20(3), August 2014. ISSN 1350-7265.

Michel X Goemans and David P Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM (JACM)*, 42(6):1115–1145, 1995.

Kjetil Høyland, Michal Kaut, and Stein W Wallace. A heuristic for moment-matching scenario generation. *Computational optimization and applications*, 24:169–185, 2003.

Michal Kaut. A copula-based heuristic for scenario generation. *Computational Management Science*, 11:503–516, 2014.

Mhamed Mesfioui and Abdelouahid Tajar. On the properties of some nonparametric concordance measures in the discrete case. *Nonparametric Statistics*, 17 (5):541–554, 2005.

David S. Moore and M. C. Spruill. Unified large-sample theory of general chi-squared statistics for tests of fit. *The Annals of Statistics*, 3(3):599–616, 1975. ISSN 00905364.

Roger B Nelsen. *An introduction to copulas*. Springer, 2006.

Georg Ch. Pflug and Alois Pichler. A distance for multistage stochastic optimization models. *SIAM Journal on Optimization*, 22(1):1–23, 2012.

Vit Prochazka and Stein W Wallace. Scenario tree construction driven by heuristic solutions of the optimization problem. *Computational Management Science*, 17(2):277–307, 2020.

M. Sklar. Fonctions de répartition à N dimensions et leurs marges. *Annales de l'ISUP*, VIII(3):229–231, 1959.

Dongqing Zhang, Stein W. Wallace, Zhaoxia Guo, Yucheng Dong, and Michal Kaut. On scenario construction for stochastic shortest path problems in real road networks. *Transportation Research Part E: Logistics and Transportation Review*, 152:102410, 2021. ISSN 1366-5545.