Modeling Linearly and non-Linearly Dependent Simulation Input Data

Dissertation

zur

Erlangung des Doktorgrades (Dr. rer. nat.)

der

Mathematisch-Naturwissenschaftlichen Fakultät

der

Rheinischen Friedrich-Wilhelms-Universität Bonn

vorgelegt von

M.Sc. Feras Nassaj

aus

Aleppo, Syrien

Bonn, 2010

Angefertigt mit Genehmigung der
Mathematisch-Naturwissenschaftlichen Fakultät der
Rheinischen Friedrich-Wilhelms-Universität Bonn

1. Gutachter: Prof. Dr. Johann Ch. Strelen

2. Gutachter: Prof. Dr. Joachim K. Anlauf

Tag der Promotion: 21.07.2010

Erscheinungsjahr: 2010

Dedicated to the memory of my mother and father, and to my Wife Kerstin and my son Nabil

Acknowledgments

I would like to acknowledge many people for their help, support and guidance during my doctoral work. I would like first to thank my adviser, Prof. Dr. Johann Christoph Strelen, for having suggested this topic and for his guidance and his generous time and commitment throughout this work. Throughout my doctoral work he encouraged me to develop my research skills, continually stimulated my analytical thinking and greatly assisted me to enhance my scientific writing. Without this, without his inspiration as a great teacher and scientist, and without his patience and readiness to reply to my questions, this dissertation could not have been written. For everything you have done for me, Prof. Strelen, I thank you.

I thank Prof. Dr. Joachim K. Anlauf very much for his support during my work on the thesis and his time to evaluate my thesis and examine me. I also thank Prof. Dr. Helmut Baltruschat and Prof. Dr. Rolf Eckmiller very much for the time they took for my Dissertation and their readiness to evaluate my thesis and examine me. I thank Dr. H.J. Kühn. very much for his attendance and comments to my presentations and his valuable advice to consider copulas in my research. His comments and advices were always perceptive and helpful. I also thank Prof. Dr. Peter Martini for supplying me with materials related to my research.

I would like to thank my family. My sisters and brothers were a constant source of support and enthusiasm. I am grateful to my wife for her continuous encouragement and for helping me keep my life in proper perspective and balance.

Feras Nassaj
Bonn, July 26, 2010

Contents

List of Figures

List of Tables

Abstract

Input modeling software tries to fit standard probability distributions to data assuming that the data are independent. However, the input environment can generate correlated data. Ignoring the correlations might lead to serious inaccuracies in the performance measures. In the past few years, several dependence modeling packages with different properties have been developed. In our dissertation, we explain how to fit non-Gaussian autoregressive models to correlated data and compare our approach with similar dependence modeling approaches that already exist. Moreover, we extend the Yule-Walker method so as to fit non-linear models to data samples using this method.

We use in our dissertation also copulas for the purpose of fitting models to data samples. Copulas are used in finance and insurance for modeling stochastic dependency. Copulas comprehend the entire dependence structure, not only the linear correlations. In our dissertation, copulas serve the purpose to analyze measured samples of random vectors and time series, to estimate a multivariate distribution for them, and to generate random vectors with this distribution.

Chapter 1

Introduction and Motivation

Real-life systems might be too complex to be evaluated analytically. In this case, such systems must be modeled and simulated. When modeling real-life systems, one must also model the environment surrounding them. These models are called (simulation) input models. The most common way to specify simulation input models is to fit a standard distribution to the simulation input data, assuming that the data are independent. This assumption might result in input models which differ considerably from the true ones and this might affect the results of the simulations. Some complex models which take dependencies into account have been developed. We examine these models in this dissertation, modify some of them, or develop new ones, for the purpose of having more exact and/or easier to expand models.

In many already existing simulation software and packages one can fit only standard distributions as input models. Neglecting the fact that the input environment might produce dependent data might lead to serious inaccuracies in the estimated performance measures (simulation results). One system which is well understood and analytically tractable is the single server queuing system. Livny et al. (1993) has shown that when mistakenly assuming independent interarrival times and/or service times, the performance measures, e.g. mean waiting time, of the single server queuing system will be seriously in error.

Taking the case of independent interarrival and service times of a queuing system, where the utilization of the service unit is 50%, as a benchmark, Livny et al. (1993) observe that an autocorrelation in the arrival process of -0.55 reduces the mean waiting time by 32%, while an autocorrelation of 0.25 increases the mean waiting time by 80%. The impact gets even more dramatic with the strength of autocorrelation: an autocorrelation of 0.85 results in a mean waiting time that is more than 200 times than the benchmark case.

A practical example of a single server queuing system is an Internet server.

Important statistical properties of the Internet traffic are burstiness and self-similarity (Klemm et al., 2002). Consider for example increasing the average number of packets in a single burst, while spacing the bursts farther. This will keep the arrival rate of packets constant, but will increase the waiting times for the packets considerably. Therefore, there has been some researches which try to construct models which can catch not only the marginal distribution of an input process, but also the correlations which lie in the process.

Early models which can catch dependencies are those called the autoregressive (AR) and autoregressive moving average (ARMA) models (Box and Jenkins, 1976). These models imply normal marginal distributions. However, there are many physical situations in which the marginal of a process is non-normal. Later on, the transform-expand-sample (TES) processes described in Jagerman and Melamed (1992a) and Jagerman and Melamed (1992b) were developed. The TES processes allow the modeler to match the marginals distribution of the fitted process to those observed marginals as well as matching the lag-1 autocorrelations. This motivated the work of Biller and Nelson (2002) to establish what is called autoregressive to anything (ARTA) processes. ARTA approach can theoretically generate a process that is autocorrelated and has any standard marginal distribution. Our work in chapter 4 is based on the ARTA processes and similar processes and is published in the European Simulation and Modelling Conference (Nassaj and Strelen, 2005).

The above mentioned approaches can generate processes with continuous state space. The marginals of those processes can be fitted to standard distributions well. In contrast to the processes with continuous state space, there are the hybrid processes with continuous and discrete state spaces. An example is the length of packets arriving at an Internet server. Klemm et al. (2002) have measured the traffic of packets at an Internet server and noticed that the lengths of such packets can be divided into three categories. Small, medium and large packets. A small part of packet lengths lie in between these three main lengths. Figure 1.1 shows a histogram of the packet lengths.

Such histograms fit well neither standard distributions nor one of the above processes. Therefore, another type of processes must be fitted. Klemm et al. (2002) have used a Batch Markovian Arrival Process (BMAP) and fitted it to measurements (input data) which contains hybrid processes with continuous and discrete state space. We investigate the work of Klemm et al. (2002) in chapter 6 and present a model which has different characteristics from the BMAP model used by Klemm et al. (2002).

Fitting an autoregressive (AR) model to a sample requires solving equations called the Yule-Walker equations. See for example Chatfield (1996), Section 3.4.4. We investigate in chapter 5 the limitations of the AR model

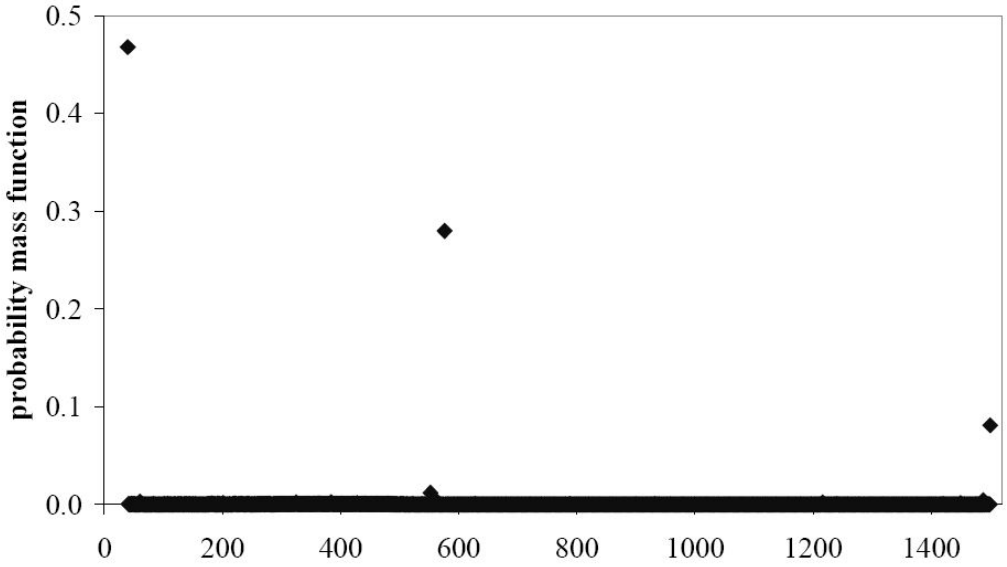

Figure 1.1: probability mass function of packet lengths arriving at an Internet server in bytes. From Klemm et al. (2002)

and extend the Yule-Walker equations to avoid some of these restrictions.

Finally we developed in chapter 7 techniques that are based on the empirical copulas. Copulas are a useful tool for modeling dependencies which has been recently often used for this purpose. However, there are many classes of copulas. We explain in chapter 7 why we use only the class of empirical copulas. We have published two papers regarding this issue, Nassaj and Strelen (2006) and Strelen and Nassaj (2007)

Before we proceed in this thesis, we would like to introduce the notation we will use, and then to highlight the main focus of our dissertation, and finally we will describe how this dissertation is organized.

1.1 Notation

We denote the random variables by capital letters such as X, Y and samples from those random variables as lowercase letters such as x, y. The marginal (cumulative) distribution function (CDF) of the random variable X is denoted as $F(x)$ or $F(x, \mathbf{p})$, where $-\infty < x < \infty$ and $\mathbf{p}$ is the set of parameters of the distribution function. Boldface type or underlined variables denote column vectors; e.g., $\mathbf{x} = \underline{x} = (x_1, x_2, ..., x_n)'$.

Throughout this thesis, Y_t, $t = 1, 2, 3, ...$, will denote independent and identically distributed (IID) random variables (perturbations), whereas Z_t,

$t = 1, 2, 3, ...$, will denote stochastic processes, in which the different random variables Z_t might exhibit dependencies among themselves. The estimated statistical quantities will be headed by a hat ($\hat{}$). For example, if μ denotes the expectation of a random variable X, $\hat{\mu}$ denotes the estimated mean found by applying the mean function over a sample generated from the random variable X.

The variance function will be denoted as Var() or σ^2, the covariance function will be denoted as Cov(), and the correlation function will be denoted as Cor() or ρ. The estimation of these functions will be headed by a hat ($\hat{}$); $\hat{\text{Var}}$, $\hat{\sigma}$, $\hat{\text{Cor}}$, $\hat{\rho}$.

1.2 Focus of the Dissertation

Fitting random variables to simulation input data is a well understood and common approach since long time, but not sufficient to model complex problems. Fitting random vectors and stochastic processes, in which the random variables might exhibit correlations or dependencies among themselves, can provide more detailed models which can be used to model more complex problems. This is much more difficult, not so popular, and it is a topic of current research. This dissertation focuses on fitting random vectors and stochastic processes to simulation input data and how to generating random vectors out of the fitted model.

The problem of fitting random variables to simulation input data can be for example solved by using the maximum likelihood estimator, which calculates the best way of fitting a mathematical model to some data. However, this approach and approaches derived from this approach, become complicated when considering fitting random vectors. Therefore, other approaches were developed to fit random vectors and stochastic processes to measurements. For example there is an approach which depends on Gaussian autoregressive processes, see 3.3. This approach shows to work well under specified conditions but to have disadvantages under other conditions.

Our focus in this dissertation is on methods useful for simulation input modeling to solve problems that are not solved by the already existing approaches. In chapter 4 we introduce an approach which for example can be used to model heavy-tailed autoregressive processes. In chapter 5 we introduce an approach which can be used to model non-linear autoregressive processes. Moreover, the world of copulas is still less explored with respect to using this technique for simulation input modeling. In chapter 7, we explore this topic with respect to simulation input modeling and show how this approach can be used to analyze and model many stochastic processes,

including heavy-tailed and non-linear ones, without to even worry about the structure of the simulation input data and which model might fit it best.

1.3 Organization

This thesis is organized as follows: The basics of statistics and stochastic processes are reviewed in chapter 2. A review on the related work that is already done with respect to fitting random vectors and stochastic processes to measurements is given in chapter 3. In Chapter 4 we introduce a new approach to model simulation input data by non-Gaussian autoregressive processes. In chapter 5 we extend a well known method for fitting autoregressive processes, the Yule-Walker method, so that it will be generalized to include non-linear dependencies. After that in Chapter 6, we compare the approaches introduced in chapters 4 and 5 with another approach used for the same purpose, the Batch Markovian arrival Processes approach. In chapter 7 we introduce the approach of empirical Copulas for the purpose of fitting simulation input data and show how this approach is quite accurate. Finally, chapter 8 outlines our contributions and gives directions to future research.

Chapter 2

Basic Stochastic

This chapter presents the basic probabilistic concepts that we need in this dissertation. So we present in section 2.1 and section 2.2 the basic concepts of the stochastic processes and simulation input modeling, respectively. In section 2.3 we make an overview of the main problem of this dissertation, the dependencies among measured data.

2.1 The Basics of Probability, Statistics and Stochastic Processes

The definitions presented in this section follow Law and Kelton (2000) unless mentioned otherwise. The set of possible outcomes of an experiment is called the *sample space*, and the outcomes are called the *sample points* or *samples*. A *random variable* is a function that assigns a real number to each point in the sample space. For example a random variable X might be the sum of two simultaneously thrown dices. By a sample point (4,3) of the sample space, the value of X is 7. The *(cumulative) distribution function* (CDF) of the random variable X is defined for each real number x as:

$$F(x) = P(X \leq x), \qquad \text{for } -\infty < x < \infty$$

where $P(X \leq x)$ is the probability associated with the event $X \leq x$. A distribution function has the following properties:

1. $0 \leq F(x) \leq 1$ for all x.
2. $F(x)$ is nondecreasing.
3. $\lim_{x \to \infty} F(x) = 1$.

4. $\lim_{x \to -\infty} F(x) = 0.$

A random variable can be discrete, continuous, or mixed. A discrete random variable takes on only a finite or countable number of values. The probability that the discrete random variable X takes on the value x_i is given by

$$p(x_i) = P(X = x_i) \quad \text{for } i = 1, 2, ...$$

and we must have

$$\Sigma_{i=1}^{\infty} p(x_i) = 1.$$

All probability statements about X can be computed from $p(x)$, which is called the *probability mass function* for the discrete random variable X.

A continuous random variable takes on an uncountably infinite number of different values. A random variable X is said to be continuous if there exists a nonnegative function $f(x)$ such that for any set of real numbers B,

$$P(X \in B) = \int_B f(x)dx,$$

and

$$\int_{-\infty}^{\infty} f(x)dx = 1.$$

$f(x)$ is called the *probability density function* for the continuous random variable X.

The *expected value* of the random variable X is denoted by $E(X)$ and for continuous random variables it is defined as

$$E(X) = \int_{-\infty}^{\infty} zf(z)dz, \quad \text{if it exists.}$$

If the random variable is discrete then its expectation is defined as

$$E(X) = \sum_{i=-\infty}^{\infty} x_i p(x_i), \quad \text{if it exists.}$$

The *variance* of a random variable X is denoted by $Var(X)$ or σ^2 and represents a measure of dispersion of the random variable about its mean. It is defined as

$$Var(X) = E(X^2) - E(X)^2, \quad \text{if it exists.}$$

The *standard deviation* of the random variable X, denoted as $\sigma(X)$, is the square root of $Var(X)$.

The correlation among two random variables X_1 and X_2 is defined as the *covariance* among X_1 and X_2 divided by the standard deviations of the two random variables:

$$\rho(X_1, X_2) = \frac{Cov(X_1, X_2)}{\sigma_{X_1}\sigma_{X_2}},$$

where $\sigma_{X_1} > 0$ and $\sigma_{X_2} > 0$ are the standard deviations of X_1 and X_2, respectively. The covariance of two random variables X_1 and X_2 is defined as

$$Cov(X_1, X_2) = E[(X_1 - E(X_1))(X_2 - E(X_2))], \quad \text{if it exists,}$$

where E is the expectation.

The random variables X and Y are *jointly continuous* if there exists a nonnegative function $f(x, y)$, called the *joint probability density function* of X and Y, such that for all sets of real numbers A and B

$$P(X \in A, Y \in B) = \int_B \int_A f(x, y)dxdy.$$

In this case, X and Y are *independent* if and only if

$$f(x, y) = f_X(x)f_Y(y) \quad \text{for all } x, y,$$

where

$$f_X(x) = \int_{-\infty}^{\infty} f(x, y)dy,$$

and

$$f_Y(y) = \int_{-\infty}^{\infty} f(x, y)dx,$$

are the (marginal) probability density functions of X and Y, respectively. The pair (X, Y) taking values in $\mathbb{R}^2$ is called *random vector*.

A *stochastic process* is a collection of random variables ordered over time, which are all defined on a common sample space. If the collection is $X_1, X_2, ...$, then we have a *discrete-time* stochastic process. If the collection is $\{X(t), t \geq 0\}$, then we have a *continuous-time* stochastic process. We will deal in this thesis with discrete-time stochastic processes.

A stochastic process is said to be strong or strictly stationary if all random variables of the process have the same distribution. In other words, the joint distribution of the random variables of a process $X_{i_1}, X_{i_2}, ..., X_{i_n}$ is the same joint distribution of the random variables $X_{i_1+j}, X_{i_2+j}, ..., X_{i_n+j}$ for all $i_1, i_2, ..., i_n$ and j. On the other hand, a weak or covariance stationary stochastic process is a process, in which the first and second moments (the mean and the variance) exist and do not change over time, and the covariance between two observations X_i and X_{i+h} depends only on the lag (also

called the separation) h and not on the actual time values i and $i + h$. In general, neither strong stationarity follows from weak stationarity, nor vice versa (Chatfield (1996)).

As we are considering only static models, and not dynamic models whose parameters might change with time or space, we consider here only fitting strongly or weakly stationary stochastic processes. Otherwise, if the stochastic process is not stationary, the parameters of the static model can not be estimated.

Many sequences $\{x_n : 0 \leq n \leq N\}$ of observations, indexed by the time at which they were taken, are suitably modeled by random processes. Such sequences are called time-series. Time-series can be modeled by an autoregressive (AR) model, moving average (MA) model, autoregressive moving average (ARMA) or other kinds of models (Chatfield (1996), section 3.4.6).

The autocorrelation between two random variables in a time-series X_t which are lag h apart is denoted as $\rho_X(h)$. The autocorrelations in a stationary time-series depend only on the lag h, not on the time when the random variables are generated.

2.2 The Basics of Simulation Input Modeling

Part of the real world is the input environment. This environment is sometimes simple and can be modeled as an independent and identically distributed (IID) random variable. In this case, the distribution of the random variable and its parameters are estimated with the help of one of several methods like the maximum likelihood estimator. However, sometimes the input environment of one system might be complex and a random variable might not represent the true model accurately. In this case, a more general stochastic process must be fitted.

Fitting an input model requires first collecting information. This information might be measurements at a real system. An example of an input model of a simulation model is the model of interarrival times of customers at a station. When measurements of such interarrival times are available, the modeler starts to search the input model which fits these measurements best. Unfortunately, there is no straight-forward method to do this.

The measurements might have more complex structure and properties than a sample from an IID random variable. There might be linear or non-linear dependencies among the measurements. The measurements might be influenced by other quantities, again linearly or non-linearly. Or they might have been generated by a mixed (continuous-discrete) process like the process of registering the interarrival times and the sizes of packets arriving at an

Internet server.

In the procedure of fitting an input model, there is need to perform one or more statistical tests, to show whether samples from the fitted model have similar statistical properties as the measurements or not. Some of these tests are the least square test, the chi-square test, the Kolmogorov-Smirnov test, and the scatter plots of the samples.

2.3 Dependencies Among Input Data

Many already existing input modeling tools assume that the input data measured at a real life system are independent. And thus they are fitted to an IID random variable. This assumption leads to models that are easy to simulate, and under suitable restrictions they are analytically tractable. However, there are many systems whose environment might produce dependent data, such as computer networks or the Internet. For example, file transfers and full motion video frames are known to be extremely bursty.

A bursty process might be modeled as a heavy-tailed process (Livny et al. (1993) and Klemm et al. (2002)). This is justified because the heavy-tailed process tends to generate "many small" random variates against "few big" ones. Considering for example the process of packet arrivals at an Internet server, the interarrival times generated from a heavy-tailed process tend to be small except a few variates, which tend to be big. The big variate might then represent the interarrival time between different bursts.

Dependencies might take other shapes. There might be for example linear dependencies among a series of random variables. Moreover, dependencies among random variables might also be non-linear. A scatter plot of linearly and non-linearly dependent processes are shown in figure 2.1. We see in this figure how the linearly dependent random variables X_1 and X_2 result in eclipsed shape when plotting them against each other in a scatter plot. The non-linear dependency among X_1 and X_2 might result in other shapes like triangle shapes. However, both processes, the linearly dependent process and the non-linearly dependent process, have the same (linear) correlations.

Another example of two processes, one with linear dependencies among its random variables and the other with non-linear dependencies, is shown in figure 2.2. The two processes have again the same linear correlations.

The most common way to model dependencies is to consider the (product-moment) correlations. In general, if the correlations among random variables are 0, then the linear dependencies among them are also 0. Most dependence modeling packages consider the correlations to generate stochastic processes. Despite its wide use, the correlations suffer several limitations. The most

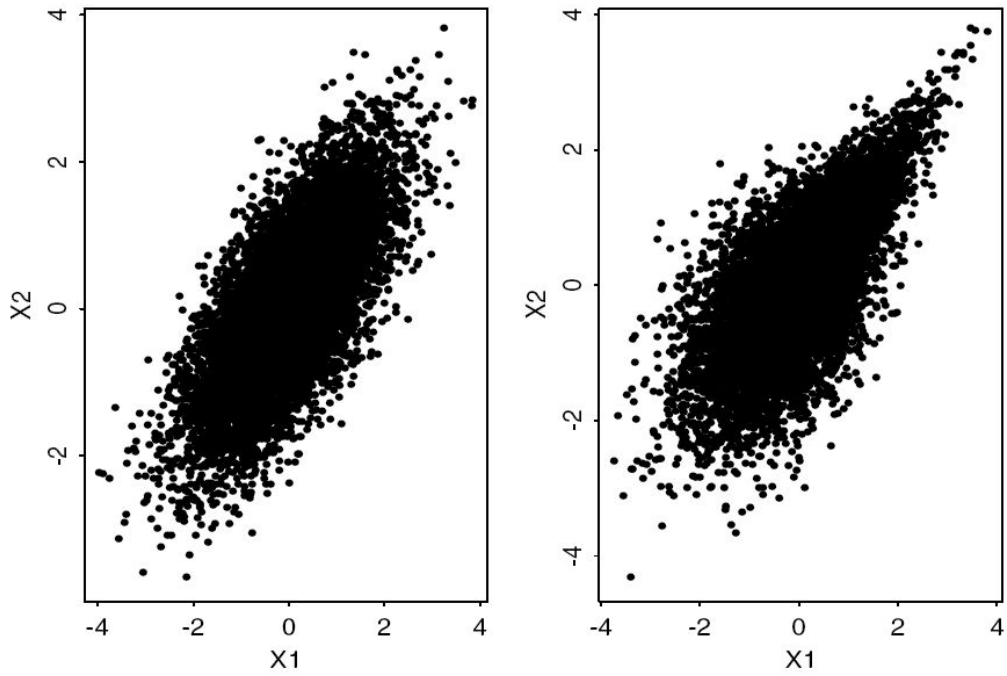

Figure 2.1: Linear (left) and non-linear (right) dependencies among X_1 and X_2. From Biller and Ghosh (2004)

.

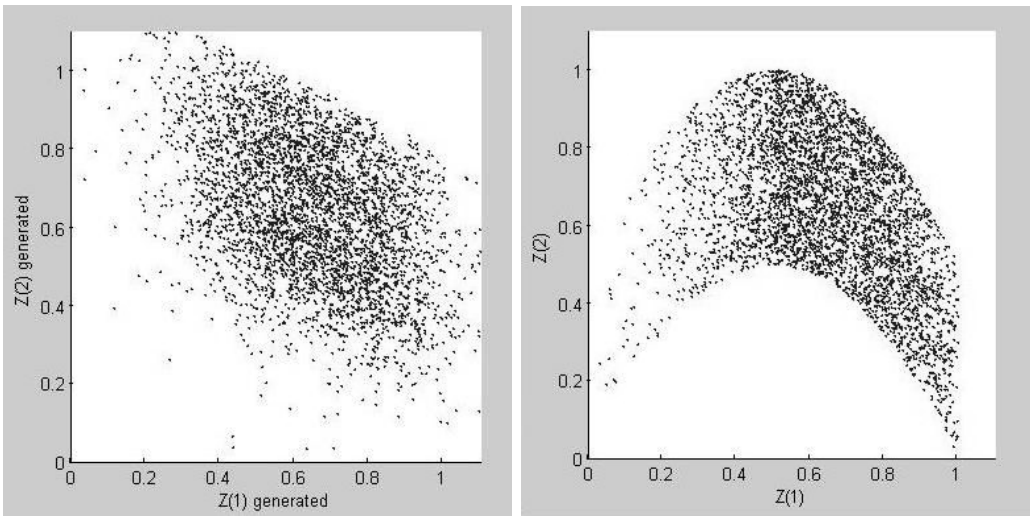

Figure 2.2: Linear (left) and non-linear (right) dependencies among Z_1 and Z_2

critical limitation is that it cannot capture non-linear dependency that may exist.

Product-moment correlations are also not invariant under transformation. They can vary with the marginal distributions of the random variables. For example, the correlations of two normally distributed random variables can take any value from the interval [-1, 1]. On the other hand, the attainable interval for the correlation between two lognormally distributed random variables, which are often generated via a transformation of normals, is [.0.09, 0.67] (Biller and Ghosh (2004)).

There exist measures of linear dependence that avoid some of these limitations like the fractile correlation. The fractile correlation is given by

$$r(X_i, X_j) = 12E[F_i(X_i)F_j(X_j)] - 3,$$

where X_i and X_j are random variables and $F_i(X_i)$ and $F_j(X_j)$ are the marginal distribution functions of the random variables. The sample analog of the fractile correation is called Spearman or rank correlation. Rank correlations can be shown to be invariant to strictly increasing transformations of the components. Thus, it provides a natural way to separate the estimation of the individual marginal distributions from that of the dependence among the variables. However, the rank correlation is less investigated than the Pearson correlation for the purpose of modeling simulation input data.

There are still other methods that can be used to specify dependencies between the random variables. For a brief review of these methods see Biller and Ghosh (2004). The methods can be in general classified into two types. Those methods which specify the complete joint distribution function of the variables, and those which specify the joint distribution function only partially, usually by specifying the marginal distribution and some dependence structure. The later method is more practical, as the first one suffers from the limitation that the amount of information they require to do a reasonable job increases enormously with dimension. In this thesis, we will consider only method which require specifying the joint distribution function only partially.

Chapter 3

Existing Approaches to Advanced Modeling of Simulation Input Data

After we have introduced the problem on which we focus in this dissertation and given a background which is necessary to proceed in this dissertation, we now look at some of the existing approaches for modeling simulation input data. We look only at those which have similarities with our approaches or which build the background for them. Section 3.1 gives an overview of the TES process. The TES process and the AR and ARMA processes described in section 3.2 build the background of the ARTA-like processes described in section 3.3. Section 3.4 describes a completely different approach for a different kind of input fitting problems, namely the approach of Batch Markovian Arrival process (BMAP). Section 3.5 review the ideas behind the copulas and the empirical copulas.

3.1 The TES Process

This Transfer-Expand-Sample (TES) approach tries to capture the marginal distribution and autocorrelation function of a given data sample (e.g. a time-series measured data). The scheme of the TES processes is an example of the general scheme *foreground/background*. Such schemes generate two autocorrelated sequences in lockstep: an auxiliary sequence Y_n (the background sequence) and the target sequence X_n (the foreground sequence) (Jagerman and Melamed (1992a)). The background sequence is usually a stationary process generated recursively by some transition function $Y_{n+1} = T_G(Y_n, Z_n)$, where Z_n is an IID Uniform(0,1) sequence. The transition function T_G is cho-

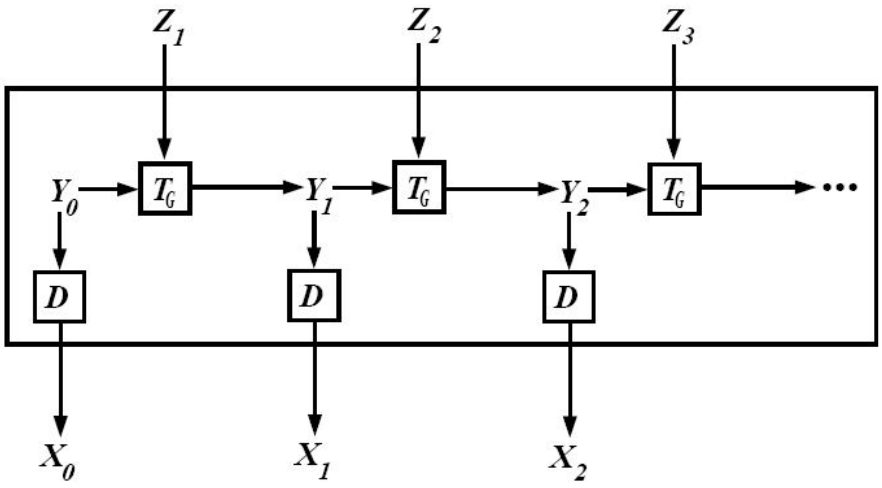

Figure 3.1: The Foreground/Background Paradigm. From Jagerman and Melamed (1992a)

sen to preserve the marginal distribution G of the initial Y_0, and to generate Y_n such that they are correlated and identically distributed. The sequence X_n is obtained from the sequence Y_n via $X_n = D(Y_n)$, where D is a mapping called distortion. Please see Fig. 3.1

A typical scheme of the TES process employs $D = F^{-1} \circ G$, where G is the marginal distribution of Y_n, F is the marginal distribution of X_n, and $\circ$ denotes functional composition. Here D is a two stage mapping: the first stage G operates on the background variate Y_n and yields a Uniform(0,1) variate, $U_n = G(Y_n)$; the second stage executes the inversion method to yield the foreground variate $X_n = F^{-1}(U_n)$ with the required marginal distribution F.

The acronym TES (*Transform-Expand-Sample*) summarizes the geometric construction of a TES process on the unit circle, where the next sample is obtained from the previous one in three steps: first, the previous sample is mapped to a point on the unit circle (*Transform*); second, the point is mapped to an interval (*Expand*); and third, a value is sampled uniformly in that interval independently of the probabilistic past (*Sample*).

There are two related classes of TES methods called TES^+ and TES^-. TES^+ realizes all values of the first autocorrelation in the range [0, 1], while TES^- cover the range [-1, 0]. The attendant TES processes U_t^+ and U_t^- are defined over a common probability space and given recursively by

$$U_t^+ = \begin{cases} U_0^+ & t = 0, \\ < U_{t-1}^+ + V_t > & t = 1, 2, ..., \end{cases}$$

$$U_t^- = \begin{cases} U_t^+ & t \text{ is even}, \\ 1 - U_t^+ & t \text{ is odd}, \end{cases}$$

where the notation $< . >$ denotes modulo-1 arithmetic, which is defined for any real x by $< x >= x - \lfloor x \rfloor$, where $\lfloor x \rfloor$=max{integer N, $n < x$}, U_0^+ is a (0, 1] uniform random variable, and V_t is a sequence of IID variates where each V_t is independent of $\{U_0^+, ..., U_{t-1}^+\}$ and $\{U_0^-, ..., U_{t-1}^-\}$.

One limitation of the TES approach is that it requires interaction with experienced users. This is due to the extreme jumps that may appear in the generated samples due to the modula-1 arithmetic, i.e., U_{t-1} can be very close to 1 while U_t is very close to 0. If this effect is wished to be alleviated, a kind of transformation called the stitching transformation, parameterized by ξ as shown in (3.1), is required. The parameter ξ is then adjusted depending on visual judgment by the user.

$$S_\xi(U_t) = \begin{cases} U_t/\xi & 0 < U_t < \xi \ , \\ (1 - U_t)((1 - \xi) & \xi < U_t < 1, \end{cases} \tag{3.1}$$

The random variables $S_\xi(U_t)$, t = 1, 2, . . ., still have (0, 1] uniform marginals, but no longer have extreme jumps.

Unfortunately, the stitching transformation changes the autocorrelation structure of $S_\xi(U_t)$, and the change is not a simple function of ξ (Biller and Ghosh (2004)). TEStool allows the user to interactively change the autocorrelation structure ξ and the distribution of V_t, and then displays the implied autocorrelation structure. The user changes the distribution until the autocorrelations of the input process match the desired autocorrelations. For this step, experience is required.

3.2 Standardized Autoregressive and Autoregressive Moving Average

A univariate linear Gaussian autoregressive (AR) model of order p can be presented as

$$Z_t^* = \alpha_1 Z_{t-1}^* + \alpha_2 Z_{t-2}^* + ... + \alpha_p Z_{t-p}^* + Y_t, \qquad t = p + 1, p + 2, ..., \infty, \tag{3.2}$$

where p is the longest lag, and the Y_t are IID normal (Gaussian) random variables with mean zero and variance ${\sigma_Y}^2$. The coefficient and Y_t can be chosen such that the Z_t^* have a standard normal distribution. The AR coefficients α_h, $h = 1, 2, ..., p$, uniquely determine the autocorrelations of the Z_t^*, $\rho_Z^*(h)$. The α_h are chosen such that the AR process is stationary. A constant c which is just a shift for the mean value of Z_t^* can be added to both sides of (3.2).

We are interested not only in standard Gaussian AR process but also in general AR processes whose marginal distribution might be more general like Weibull or Pareto, we define an AR process as:

$$Z_t = \alpha_1 Z_{t-1} + \alpha_2 Z_{t-2} + ... + \alpha_p Z_{t-p} + Y_t, \qquad t = p+1, p+2, ..., \infty, \tag{3.3}$$

where the IID random variables (perturbations) Y_t are not necessarily standard Gaussian. Therefore, the expected value of Z_t might not be zero, and the variance of Z_t might not be equal one.

As we mentioned in section 2.1, we are interested only in strong and/or weak stationary stochastic processes. Weak stationary stochastic processes have the following two properties:

1. $\mu(i) = \mu$ and $\sigma(i) = \sigma$ for all $i = 1, 2, ...n$, and
2. $\mathrm{Cov}(Z_s, Z_r)$ is a function of $(s - r)$ only, $s, r = 0, 1, ..., n$, Z_s and Z_r are random variables,

where μ is the expectation function, and σ is the standard deviation function.

The first property implies that the expectation and the standard deviation functions must be constant. The second property implies that the correlation function depends only on the difference between the lags s and r. Both properties apply also to strongly stationary stochastic processes, if they exist. The stationarity of an AR process can be satisfied if the following condition for α_h, $h = 1, 2, ..., p$, holds:

- All roots of $(1 - \sum_{h=1}^{p} \alpha_h x^h = 0)$ lie outside the unit circle in the complex plane.

Similar conditions are needed to guarantee the stationarity of multivariate AR processes (Biller and Nelson, 2003).

To calculate the AR parameters, α_h, $h = 1, 2, ..., p$, of (3.2), one can use the Yule-Walker equations. See for example Chatfield (1996), Section 3.4.4:

$$\rho_m = \textstyle\sum_{h=1}^{p} \alpha_h \rho_{m-h} + \sigma_Y^2 \delta_{m,0} \qquad p = 1, 2, ..., n \tag{3.4}$$

where $m = 0...p$, yielding $p+1$ equations. ρ_m is the autocorrelation function of the AR process Z, σ_Y is the standard deviation of the IID random variables Y_t, and $\delta_{m,j}$, $m, j = 0, 1, ...$ is the Kronecker delta function:

$$\delta_{m,j} = \begin{cases} 1 & m = j \ , \\ 0 & \text{otherwise.} \end{cases}$$

Because the last part of the equation is non-zero only if $j = 0$, the equation (3.4) is usually solved by representing it as a matrix for $m > 0$, thus getting

$$\begin{bmatrix} \rho_1 \\ \rho_2 \\ ... \\ \rho_n \end{bmatrix} = \begin{bmatrix} \rho_0 & \rho_{-1} & ... & \rho_{-n} \\ \rho_1 & \rho_0 & ... & \rho_{-n+1} \\ ... & ... & ... & ... \\ \rho_n & \rho_{n-1} & ... & \rho_{n-p} \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ ... \\ \alpha_n \end{bmatrix}$$

which is used to solve for all α. For $m = 0$ we have $\rho_0 = \sum_{h=1}^{p} \alpha_h \rho_{-h} + \sigma_Y^2$, which allows to solve for σ_Y^2. The derivation of the Yule-Walker equations in (3.4) is simple and is as follows:

For simplicity, we assume that $E[Z_t] = 0$. Multiplying both sides of (3.3) by Z_{t-m} and taking the expectation yields:

$$E[Z_t Z_{t-m}] = E\left[\sum_{i=1}^{p} \alpha_i \, Z_{t-i} Z_{t-m}\right] + E[Y_t Z_{t-m}],$$

where p is the longest lag, $t = 1, 2, ...,\ p < t$, and $m = 0...p$. Considering that $E[Z_t]$=0 yields that $E[Z_t Z_{t-m}]$ is the covariance function. The values of the IID random variables Y_t are independent of each other, and Z_{t-m} is independent of Y_t for all m greater than zero. Furthermore,

$$E[\sum_{i=1}^{p} \alpha_i \, Z_{t-i} Z_{t-m}] = \sum_{i=1}^{p} \alpha_i \, E[Z_t Z_{t-m+i}] = \sum_{i=1}^{p} \alpha_i \, Cov_{i-m}(Z_t),$$

This yields:

$$\mathrm{Cov}_m(Z_t) = \sum_{i=1}^{p} \alpha_i \, Cov_{i-m}(Z_t) + E[Y_t Z_{t-m}],$$

where p is the longest lag, $t = 1, 2, ...,\ p < t$, and $m = 0...p$

Dividing by σ_Z^2 and considering that $\mathrm{Cov}_m(Z)/\sigma_Z^2 = \rho_m$, where ρ is the autocorrelation function, and considering also that the IID random variables Y_t are independent, and Z_{t-m} is independent of Y_t for $m = 1, 2, ..., p$, and therefore $E[Y_t Z_{t-m}]$= $E[Y_t]E[Z_{t-m}]$=0, we can write

$$\rho_m = \sum_{t=1}^{p} \alpha_t \rho_{t-m} \quad m = 1, .., p, \text{ and } p \text{ is the longest lag } < t, t = p+1, .., \infty. \tag{3.5}$$

For $m = 0$

$$\rho_0 = \sum_{t=1}^{p} \alpha_t \rho_t + \sigma_Y^2. \tag{3.6}$$

Equations (3.5) and (3.6) yield the Yule-Walker equations:

$$\rho_m = \sum_{i=1}^{p} \alpha_i \rho_{m-i} + \sigma_Y^2 \delta_{m,0} \qquad .$$

If the random variables Y_t in (3.2) are defined as

$$Y_t = \epsilon_t + \sum_{i=1}^{q} \beta_i \epsilon_{t-i}, \qquad q \in \{1, 2, ..., t\},$$

where ϵ_t is a standard Gaussian process with mean =0, then the process is called an autoregressive moving average (ARMA) process. ARMA processes can be rewritten as pure AR processes, which means that both processes are equivalent. However, fitting ARMA processes might result sometimes in the need to estimate less parameters than those needed to be estimated if an AR process were fitted (Chatfield, 1996), page 47.

3.3 ARTA, NORTA, and VARTA

ARTA, NORTA, and VARTA processes have some similarities to our approach described in chapter 4, but still have different advantages and disadvantages. The abbreviations stands for autoregressive to anything, normal to anything, and vector autoregressive to anything, respectively. From now on, we will call these processes ARTA-like processes. The approaches of ARTA and VARTA try to model the dependencies in a time-series by transforming a Gaussian AR process to a non-Gaussian process. A sample of the latter processes has similar statistical properties as the time-series. The NORTA approach in turn depends on transforming Gaussian random vectors to any non-Gaussian random vectors. The later random vectors have some desired statistical properties (marginal distributions and correlation.) In the following we give a brief overview on these processes.

ARTA processes use a standard Gaussian AR process, shown in (3.2), as a base process. This base process, Z_t^*, is used to generate a series of autocorrelated uniform random variables, U_t by applying $U_t = \Phi(Z_t^*)$, where Φ is the standard normal distribution. ARTA applies then the inverse transformation method, $X_t = F_X^{-1}[U_t]$, to generate random variables having a specific distribution, F_X. The Gaussian property of the Y_t in (3.2) ensures not only that the Z_t are standard Gaussian, but also that the autocorrelation coefficients of the base process, $\rho_Z(h)$, determined by the AR coefficients α_h, $h = 1, 2, ..., p$, uniquely determine the autocorrelation coefficients of the transformed (target) process X_t, $\rho_X(h)$.

ARTA processes of Cario and Nelson (1996) are able to generate random variables having a specific distribution and autocorrelation structure, which should in turn be given explicitly. A complementary work to that is the work done by Biller and Nelson (2002). They describe how to fit ARTA processes to univariate time-series. This will enable the user to provide the ARTA tool with a time-series and to get as a result a fitted ARTA process. A sample generated from this process has similar statistical properties as the provided time-series.

Another research in this area are the NORTA processes of Cario and Nelson (1997). NORTA processes can be used to generate IID finite vectors of random variables. The random variables within the vectors can have arbitrary marginal distributions and correlation matrices. The idea behind this work is to transform a standard multivariate normal vector $\mathbf{Z} = (Z_1, Z_2, ..., Z_d)'$ into a vector $\mathbf{X} = (X_1, X_2, ..., X_d)'$, where $X_i = F_i^{-1}[\Phi(Z_i)]$. F_i, $i = 1, 2, ..., d$, may be different distribution functions. Moreover, X_i, $i = 1, 2, ..., d$, can exhibit correlations among themselves.

A generalization of ARTA and NORTA processes are the VARTA processes of Biller and Nelson (2003). VARTA can be fitted to multivariate time-series by considering the AR base process as the standard Gaussian vector AR process of order p. Similar to the case of ARTA, the autocorrelation structure of the base process, determined by the AR coefficients, specifies uniquely the target autocorrelation structure of the resulted VARTA process.

ARTA-like processes depend on a transformation of a base process into a specific process. Let us consider for example the ARTA processes. The autocorrelations in the base process of ARTA, $\rho_Z(h)$, do not match the autocorrelations of the ARTA process, $\rho_X(h)$. However, Cario and Nelson (1996) show that $\rho_X(h)$ is a continuous non-decreasing function of $\rho_Z(h)$

$$\rho_X(h) = \frac{\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} F_X^{-1}[\Phi(z_t)]F_X^{-1}[\Phi(z_{t-h})]\vartheta_{\rho_Z(h)}(z_t, z_{t-h})dz_t dz_{t-h} - \mu^2}{\sigma^2},$$

where $h = 1, 2, ...$ is the current base process autocorrelation lag, and ϑ is the bivariate normal (probability) distribution function.

NORTA and VARTA depend also on transformations similar to the above one. Current researches show that this kind of transformations has a drawback, that is, there are some random vectors, X_t, with feasible covariance matrices, Cov(X), which are transformed to non-feasible base process covariance matrices, Cov(Z). In other words, for some desired Cov(X) matrices, the transformation results in non-positive definite Cov(Z) matrices. Non-positive definite covariance matrices are invalid covariance matrices (Fishman, 1978). Cario and Nelson (1997) calls the Cov(X) matrices, that are transformed into non-positive definite Cov(Z) *defective* matrices. Please do not confuse this

definition with the definition of defective matrices in mathematics, which says, that a matrix is defective if its eigenvectors are not complete.

This drawback is discussed in Ghosh and Henderson (2001), Ghosh and Henderson (2002a), and Ghosh and Henderson (2002b) in detail for the NORTA processes. The papers provide an example of a defective covariance matrix Cov(X). They suggest a modification of the NORTA procedure such that defective matrices can be detected, and which can generate Cov(Z) matrices that are positive definite and "close" to the desired ones. VARTA processes, which are generalizations of the NORTA processes, are supposed to have the same drawback. ARTA is not yet proved to suffer from the defective matrices problem, as the defective matrix given by Ghosh and Henderson (2002a) and Ghosh and Henderson (2002b) is not a valid ARTA covariance matrix. However, Biller and Ghosh (2004) assume that ARTA can also generate defective matrices, but they do not provide detailed information.

Another drawback of the above transformation shows up when trying to fit ARTA-like processes to time-series. Let us consider ARTA for example. Fitting an ARTA process to a time-series, which have the distribution F_X with the parameters $\mathbf{p} = (p_1, p_2, ...)$ and the autocorrelation $\rho_X(h)$, requires estimating F_X along with $\mathbf{p}$ and $\rho_Z(h)$ *in parallel*. In other words, the fitting procedure assumes a distribution F_X having the parameters $\mathbf{p}$, and try to estimate $\rho_Z(h)$ using an optimization procedure. Having $\rho_Z(h)$ estimated for specified F_X and $\mathbf{p}$, the parameters $\mathbf{p}$ and maybe F_X must be estimated using an optimization procedure. The procedure iterates until "convergence". This results generally in a relatively time consuming fitting procedure.

The procedures of Biller and Nelson (2002) and Biller and Nelson (2003) fit ARTA and VARTA processes to time-series. The distributions considered in these two papers are only those from the Johnson translation system Johnson (1987). This means that the current ARTA and VARTA approaches can not generate heavy-tailed ARTA and VARTA processes. Moreover, ARTA-like processes can not fit non-linear AR models to time-series. An example of non-linear AR models is shown below.

$$Z_t = \alpha_1 {Z_{t-1}}^p + \alpha_2 {Z_{t-2}}^{p-1} + ... + \alpha_p Z_{t-p} + Y_t. \tag{3.7}$$

3.4 The Batch Markovian Arrival Processes

The Markovian Arrival Process (MAP), introduced in Neuts (1989), is a generalization of phase-type (PH) distributions, see Neuts (1995) or Riska (2002). A MAP is associated with a finite absorbing Markov chain. Once the Markov chain has entered the absorbing state and a single MAP random

variable is generated, the process restarts from the transient part again for the next random variable remembering the last transient state that reached absorption. The concept of the Markovian Arrival Process is further extended to allow for batch absorptions in the underlying Markov chain. The resulting process is known as the Batch Markovian Arrival Process (BMAP). See for example Lucantoni et al. (1994).

BMAP is an analytically tractable model of choice for aggregated traffic modeling of IP networks. The key idea of this aggregated traffic model lies in customizing the BMAP such that different lengths of IP packets are represented by rewards of the BMAP. In order to represent an aggregated traffic stream utilizing the BMAP, Klemm et al. (2002) applies the parameter estimation procedure for a BMAP with N transient states and M distinct batch sizes. The choice of N and M is crucial for an accurate capturing of the interarrival process and the reward process of the aggregated traffic, respectively.

The batch Markovian arrival process (BMAP) belongs to the class of continuous-time Markov chain (CTMC). Consider a CTMC with N+1 states, {0, 1, 2, .., N}, where the states {1, 2, .., N} are transient states, and state 0 is the absorbing state. Based on this CTMC, the BMAP can be constructed as follows: The CTMC evolves until an absorption in state 0 occurs. The chain is then instantaneously restarted in one of the transient states {1, 2, .., N}. When restarting the BMAP after absorption in a transient state j, the probability for selecting state j is allowed to depend on state i from which absorption has occurred. Thus, the distribution of the next arrival may depend on the previous history. Furthermore, there may exist multiple paths between two states i and j corresponding to different rewards, i.e., different packet lengths.

Formally, assume the BMAP is in a transient state i for an exponentially distributed time with rate λ_i. When the sojourn time has elapsed, there are (M+ N - 1) possible cases for state transitions: With probability $P(m)_{i,j}$ the BMAP enters the absorbing state 0 and an arrival of batch size m occurs. Then, the process is instantaneously restarted in state j. Note that the selection of state j ($1 < j < N$) and batch size m ($1 < m < M$) is uniquely determined by $P(m)_{i,j}$. On the other hand, with probability $P(0)_{i,j}$ the BMAP enters another transient state j, $j \neq i$, without arrivals.

Figure 3.2 shows a CTMC with the number of transient states equal to 3 (N=3), and 3 different packet lengths: L1, L2, and L3 (M=3).

Klemm et al. (2002) fits a BMAP to a trace file which consists of packet interarrival times and the corresponding packet lengths. Noticing that packet lengths of 40 bytes, 576 bytes, and 1500 bytes dominate with an overall percentage of 80% of all TCP packets, it was decided that a BMAP with

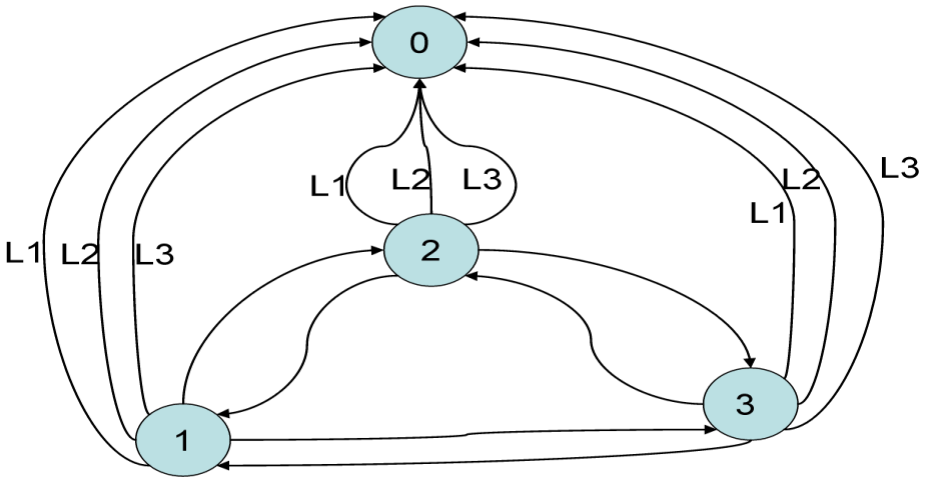

Figure 3.2: An example of a BMAP CTMC

M=3 is sufficient in order to capture the interarrival process of the considered trace.

Klemm et al. (2002) shows that the customized BMAP is advantageous over other methods like the Markov Modulated Poison Process (MMPP) and the Poisson Process. For example, the mean, variance, skewness, and kurtosis of the BMAP are in general more similar to the measured traffic than those of the MMPP or the Poisson process. The Traffic burstiness expressed in terms of the Hurst parameter is also better in the case of BMAP. Moreover, the analysis of the queuing performance shows that the BMAP model shows a similar behavior in terms of queuing performance for low traffic intensities and that the MMPP and the Poisson Process performs worse than the BMAP for all considered traffic intensities.

3.5 The Copulas and the Empirical Copulas

Copulas are functions that join or "couple" multivariate distribution functions to their one-dimensional marginal distribution functions. Formally speaking, copulas are multivariate distribution functions whose one-dimensional margins are uniform on the interval (0,1) (Nelsen, 1998). Copulas provide a way of studying scale-free measures of dependence. A foundation of many of the applications for the purpose of dependence modeling is Sklar's theorem Sklar (1959).

Theorem 3.5.1 (Sklar's Theorem) *Let H be a joint distribution function with margins F and G. Then there exists a copula C such that for all x, y in*

$\mathbb{R}$,

$$H(x, y) = C(F(x), G(y)). \tag{3.8}$$

If F and G are continuous, then C is unique; otherwise C is uniquely determined on the range of F $\times$ the range of G. Conversely, if C is a copula and F and G are distribution functions, then the function H defined by theorem 3.5.1 is a joint distribution function with margins F and G.

A proof of the above statement can be found in Nelsen (1998), section 2.3. Theorem 3.5.1 can be extended to the d-dimensional case. However, for the purpose of simplicity, we consider often the 2-dimentional copulas.

A 2-dimentional copula C is a function which has the following properties (Nelsen, 1998), page 8:

1. The range of C is the unit interval $[0, 1]$.
2. $C(u, 0) = 0 = C(0, v)$ for every $u, v \in [0, 1]$.
3. $C(u, 1) = u$ and $C(1, v) = v$ for every $u, v \in [0, 1]$.
4. C is 2-increasing. This means, for every u_1, u_2, v_1, v_2 in $[0, 1]$ such that $u_1 \leq u_2$ and $v_1 \leq v_2$,
   $C(u_1, v_1) - C(u_2, v_1) - C(u_1, v_2) + C(u_2, v_2) \geq 0$.

These properties can be extended to the d-dimensional case.

Copulas are being used for the purpose of dependence modeling between random variables, for example in financial applications. The advantage of the copula-based approach to modeling dependency is that appropriate marginal distributions for the components of a multivariate system can be selected freely, and then linked through a suitable copula (Aas, 2004).

There are several classes of copulas in use. Some of the common classes are the Marshall-Olkin copulas, the elliptical copulas, and the archimedean copulas. These copulas have been studied to a specific degree and shown to have specific properties. For example, the Marshall-Olkin copulas are asymmetric and have closed form expressions, and their tail dependence coefficients can be explicitly expressed in terms of their parameters. The Marshall-Olkin copula possesses either independence or perfect dependency, whereas the survival Marshall-Olkin copula possesses various positive upper tail dependence (Li, 2006). The class of elliptical distributions provides a rich source of multivariate distributions which share many of the tractable properties of the multivariate normal distribution and enables modeling of multivariate extremes and some other forms of dependences. Elliptical copulas are simply the copulas of elliptical distributions (Embrechts et al., 2001).

The class of archimedean copulas allow for a great variety of different dependence structures. Furthermore, in contrast to elliptical copulas, all commonly encountered archimedean copulas have closed form expressions (Embrechts et al., 2001).

One of the main issues with copulas is to choose the copula that provides the best fit for the data set at hand. According to Blum et al. (2002), giving an answer to this question is essentially as difficult as estimating the joint distribution in the first place. The choice among different copulas can be done via goodness-of-fit tests. However, while there in the one-dimensional case are a lot of well-known distribution-independent goodness-of-fit statistics available, e.g. Kolmogorov-Smirnov and Anderson-Darling, it is more difficult to build distribution-independent goodness-of-fit tests in the multidimensional framework.

The advantage of the copulas approach over the approach of fitting a multivariate distribution function is that the different marginal distributions can be of different families. However, the use of copulas was put under criticism in Mikosch (2005). Mikosch (2005) emphasizes that multivariate distributions should be used instead of the copulas approach, as the copulas approach leads to a biased view of stochastic dependence, and the statistical analysis and the goodness-of-fit of the multivariate approach are more understood.

Mikosch (2005) mentions also that the class of copulas is too big to be understood and usefull. Various copulas models (archimedean, t-, Gaussian, elliptical, extreme value) are mostly chosen because they are mathematically convenient; the explanation for their applications is obscure. Moreover, Mikosch (2005) mentions that copulas do not fit into the existing framework of stochastic processes and time series analysis; they are essentially static models and are not useful for modeling dependence through time. However, we show in this thesis that modeling dependence through time is possible with a kind of copulas called the empirical copulas.

Empirical copulas are another class of copulas. A common bivariate empirical copula is defined as follows:

Let $\{(x_i, y_i)\}_{i=1}^{n}$ denote a sample of size n from a continuous bivariate distribution. The empirical copula is the function C_n given by

$$C_n(\frac{i}{n}, \frac{j}{n}) = \frac{\text{number of pairs } (x, y) \text{ in the sample such that } x \leq x_{(i)} \text{ and } y \leq y_{(j)}}{n},$$

where $x_{(i)}$ and $y_{(j)}$, $1 \leq i, j \leq n$, denote the order statistics from the sample.

The empirical copulas can represent any class of copulas. However, the (theoretical) copulas have the disadvantages mentioned above. This encouraged us to use the empirical copulas instead of other classes of copulas for

the purpose of modeling dependencies among measurements.

The reader may note that the empirical copulas are not copulas. Their marginals are discrete. Hence they are kind of "discrete copulas". In chapter 7 we define a kind of empirical copulas with the properties mentioned on page 22 and the generalization to higher dimensionality. In other words, they are real copulas. Thus we omit the restriction to the common classes of copulas and consequently the biase mentioned.

Chapter 4

A non-Gaussian Autoregressive Modeling Approach for Simulation Input Data

Non-Gaussian autoregressive (nGAR) processes are generalization of Gaussian autoregressive processes viewed briefly in section 3.2. The marginal distribution of an nGAR process can have any known distribution and not only the (standard) Gaussian distribution. For example, if the random variables Y_t shown in (4.1) are drawn from a heavy-tailed distribution like the Pareto distribution, then the process Z_t is non-Gaussian and will also, inherently, be heavy-tailed,

$$Z_t = c + \alpha_1 Z_{t-1} + \alpha_2 Z_{t-2} + ... + \alpha_p Z_{t-p} + Y_t. \tag{4.1}$$

This process might be generalized such that the dependencies among the Z_t become non-linear. In this case, known Yule-Walker equations for solving the AR coefficients, please see section 3.2, do not apply any more, unless this method is extended as we describe in chapter 5. An example of a non-linear nGAR model is

$$Z_t = \alpha_{1,1} {Z_{t-1}}^1 + \alpha_{1,2} {Z_{t-1}}^2 + ... + \alpha_{2,1} {Z_{t-2}}^1 + \alpha_{2,2} {Z_{t-2}}^2 + ... + Y_t. \tag{4.2}$$

Combining non-Gaussian random variables Y_t and non-linear dependencies in an nGAR process results as well in a generalized nGAR process.

The nGAR models have the property that the distribution of the Z_t differs from that of the Y_t. This means that the input modeler, who does not have measurements (samples), can not generate an nGAR process which has a specific distribution and autocorrelation structure with our method. However, if a modeler has measurements, these measurements can be used

to fit an nGAR process. A sample generated from the fitted nGAR process will have similar statistical properties as the original measurements. In the world of simulation input modeling, it is common to have measurements which should be fitted to a distribution or to a stochastic process.

Before we start the explanation of the procedure of fitting nGAR models to measurements, we would like to give an overview of the genetic algorithm which is applied in our approach of fitting nGAR models to measurements.

4.1 The Genetic Algorithm for Fitting Distributions to IID sample

We use the genetic algorithm for the purpose of estimating distribution parameters by means of minimizing an objective function, namely distances between empirical distributions and fitted distributions. Estimating the parameters of a distribution can be done in different ways like the maximum likelihood or moment methods. The maximum likelihood or moment methods are applied in arena input analyzer and ExpertFit. However using the genetic algorithm for the purpose of fitting a standard distribution to IID samples is advantageous in some cases.

Strelen (2003) shows that using the genetic algorithm for fitting distributions to IID data is useful when fitting multi-mode distributions like

$$F(x) = \alpha_1 F_1(x) + \alpha_2 F_2(x), ..., +\alpha_n F_n(x),$$

where $\alpha_i > 0, i = 1, .., n$, $\alpha_1 + ... + \alpha_n = 1$ and F_1, F_2, ... might belong to the same family of distributions (e.g. Weibull) but have different parameter values, or F_1, F_2, ... might belong to different families of distributions, (e.g. F_1 is Weibull, F_2 is Gamma, and so on.) We call this method the *Genetic Algorithm Fitting* (GAF). Moreover, because we implement our algorithms in MATLAB and the algorithms of Strelen (2003) are also implemented in MATLAB, we could automate both algorithms together. This way, the fitting of an nGAR process, which includes at one point fitting a distribution to IID samples, can be accomplished in one single program and one must not interrupt the procedure at some point. On the other hand, choosing to use an available software like Arena input analyzer to fit a standard distribution to IID samples mean that at one point, the procedure of fitting an nGAR process will stop and wait for the user to intervene and use the input modeling software to fit a distribution to independent samples produced by the fitting procedure, and give the results back to the nGAR process fitting procedure.

The genetic algorithm (see for example Baeck et al. (1997) and Chipperfield et al. (1994)) is a stochastic global search method that mimics the

natural biological evolution. It differs from traditional search and optimization methods, significant differences are:

- Genetic algorithms search generations of approximations in parallel, not a single sequence
- Genetic algorithms require only the objective function, no derivatives
- Genetic algorithms use probabilistic transition rules, not deterministic ones
- Genetic algorithms work on an encoding of the parameter set rather than the parameter set itself

The genetic algorithm operates on a population of potential solutions (individuals) applying the principle of survival of the fittest to produce successively better approximations to a solution. *Individuals* are tuples of decision variable values which are encoded as strings over the binary alphabet or other alphabets. The individuals are approximations of the desired solution, and their *fitness* measures the accuracy which is fixed by an objective function. At each *generation*, a new population is created by the process of selecting individuals according to their fitness.

The genetic algorithm uses operators borrowed from the natural genetics like the selection, recombination, and mutation (Chipperfield et al., 1994). The selection operator is used to select individuals for reproduction. The recombination operator is used to exchange parts of the individuals among themselves, and the mutation operator might change the state of one bit of an individual randomly (assuming that individuals are binary encoded).

The genetic algorithm works as follows: The population of individuals is initialized randomly with uniformly distributed random numbers. The population consists of several individuals. Each individual represents a possible solution. Each individual can be assigned a fitnesses according to an objective function. Fitter individuals have higher probability to propagate to the next generation and higher probability to be selected to produce the individuals of the next generation.

In our case, the individuals of a population represent the parameters of a distribution. Each individual is a vector of the same length as the parameter vector of the distribution, and each value in this vector represent a value that the corresponding parameter in the parameter vector might take. The objective function might then depend on the principle of least squares or a similar principle and might be established as follows: Having measurements (samples) $\mathbf{y} = (y_1, y_2, ..., y_n)$, and considering a distribution $F(y, \mathbf{p})$ with

parameters $\mathbf{p}$, for each individual $\mathbf{v}$ which specifies the parameter values, the objective function is the sum of the squares of the differences between the empirical distribution function, and the distribution function at the sample points $\mathbf{y}$, where the distribution function is having the parameter values $\mathbf{v}$, divided by n. Formally, the procedure is as follows:

Having a measured sample $\mathbf{y} = (y_1, y_2, ..., y_n)$, the empirical function $F_n(y)$, $y \in \mathbb{R}$, a step function is defined by:

$$F_n(y) = \frac{(\text{number of elements in the sample} \leq y)}{n},$$

and the objective function for the genetic algorithm is

$$OF(\mathbf{v}) = \sum_{i=1}^{n} \frac{\left(F_n(y_i) - F_Y(y_i, \mathbf{v})\right)^2}{n}, \tag{4.3}$$

where $F(y_i, \mathbf{v})$ is the value of the selected distribution function with the parameters (individual) $\mathbf{v}$ at the point y_i. The objective function $OF(\mathbf{v})$ measures how accurately the selected distribution with the selected parameters (individual) fits the empirical distribution. The smaller $OF(\mathbf{v})$ is, the better the distribution with these parameters (individual) fits the empirical distribution.

4.2 The Model and the Fitting Procedure

Equations (4.1) and (4.2) are special cases of multi-variate non-Gaussian autoregressive (nGAR) models. Another special case is the bivariate nGAR model given below:

$$Z_{1,t} = \alpha_{1,1} Z_{1,t-1} + \alpha_{1,2} Z_{2,t-1} + \alpha_{1,3} Z_{1,t-2} + ... + \alpha_{1,p} Z_{2,t-p} + Y_{1,t}$$

$$Z_{2,t} = \alpha_{2,1} Z_{2,t-1} + \alpha_{2,2} Z_{1,t-1} + \alpha_{2,3} Z_{2,t-2} + ... + \alpha_{2,p} Z_{1,t-p} + Y_{2,t}$$

where the matrix of the autoregressive coefficients $\mathbf{\Sigma}_\alpha = \begin{bmatrix} \alpha_{1,1} & \alpha_{1,2} & ... & \alpha_{1,p} \\ \alpha_{2,1} & \alpha_{2,2} & ... & \alpha_{2,p} \end{bmatrix}$ are such that the stochastic process is stationary. Please see section 3.2 for the definition of stationary stochastic (nG)AR processes. $t = p + 1, ..., n$, p is the maximum lags (nGAR orders) of the sub-processes $Z_{1,t}$ and $Z_{2,t}$ respectively. For simplicity, we assume that the maximum lag of the two sub-processes is the same. $Y_{1,t}$ and $Y_{2,t}$ are IID random variables.

In general, an nGAR process might be d-variate, which means that at each point of time t, there are $Z_{1,t}$, $Z_{2,t}$, ... $Z_{d,t}$ random variables. Each random

variable $Z_{i,t}$ might depend on 0,1, or more previous random variables $Z_{i,t-j}$, where $i = 1, ...d$, $j = 1, ..., t-1$, and $t = p+1, ..., n$. Moreover, each random variable $Z_{i,t}$ might depend on 0,1, or more previous random variables $Z^k_{i,t-j}$ with an exponent k. For example, $Z_{i,t}$ might depend on $Z^2_{i,t-1}$ or $Z^k_{i,t-1}$. p is the maximum lag. A more formal and precise overview of our method and goal is described in Nassaj and Strelen (2005) and is given in the following paragraphs:

Our goal is to fit a stationary (multivariate) nGAR model, $\mathbf{Z}_t$ to a (multivariate) measured sample $\mathbf{x}_t$, $t = p+1, , ..., n$. The nGAR Model is specified by:

$$\mathbf{Z}_t = \psi(\mathbf{Z}_{t-1}, \mathbf{Z}_{t-2}, ..., \mathbf{Z}_{t-p}, \mathbf{\Sigma}_\alpha) + \mathbf{Y}_t,$$

where $\mathbf{Z}_t = (Z_{1,t}, Z_{2,t}, ..., Z_{d,t})'$ is a d-vector of random variables taken at time t. $\mathbf{Y}_t = (Y_{1,t}, Y_{2,t}, ..., Y_{d,t})'$ is a d-vector of independent and identically distributed random variables observed at time t. For each $i = 1, 2, ..., d$, the $Y_{i,t}$, $t = p+1, ..., n$, are independent and have the probability distribution $F_i(y_i)$. $\mathbf{\Sigma}_\alpha$ is a set of numerical parameters, and $\mathbf{p}$ is the a d-vector of the longest lags (orders) of the different d sub-processes of the nGAR model, and p=max(p_1, p_1, ...). $\mathbf{\Sigma}_\alpha$, $\mathbf{p}$, and ψ are such that $\mathbf{Z}_t$ is stationary for any $t = p+1, ..., , n$. The function ψ, the parameters $\mathbf{\Sigma}_\alpha$, and the vector of the longest lags $\mathbf{p}$, determine the dependency structure of the multivariate stochastic process $\mathbf{Z}_t$.

The first elements generated by the fitted nGAR model will be non-stationary because they will have different distributions and dependencies than the stationary nGAR process $\mathbf{Z}_t$. Therefore, the first elements which are generated by the nGAR model are considered as transient and will be ignored.

In the case of linear nGAR models and a single lag p for all sub-processes in the model, $\mathbf{\Sigma}_\alpha$ is a p x d matrix, $\mathbf{\Sigma}_\alpha = \begin{bmatrix} \alpha_{1,1} & \alpha_{1,2} & ... & \alpha_{1,p} \\ ... & ... & ... & ... \\ \alpha_{d,1} & \alpha_{d,2} & ... & \alpha_{d,p} \end{bmatrix}$, and ψ is simply a matrix multiplication:

$$\mathbf{Z}'_t = [\mathbf{Z}'_{t-1}, \mathbf{Z}'_{t-2}, ..., \mathbf{Z}'_{t-p}] \times \mathbf{\Sigma}_\alpha{}', \tag{4.4}$$

where $t = p+1, ..., n$, and $[\mathbf{Z}_{t-1}, \mathbf{Z}_{t-2}, ..., \mathbf{Z}_{t-p}]$ are the concatenated vectors. In the case of non-linear nGAR models, ψ is more general. An example of non-linear nGAR models is shown in (4.2).

Fitting an nGAR model to measurements corresponds to estimating the parameters $\Omega = \{\psi, \mathbf{\Sigma}_\alpha, \mathbf{p}, F(y_i), i = 1, 2, ..., d\}$. Here, the function ψ is chosen out of a finite set of given functions.

As a first step in this approach, the numerical parameters $\mathbf{\Sigma}_\alpha$ are estimated for one or several ψ functions, and one or several $\mathbf{p}$. When ψ is linear

in the $\mathbf{Z}_t$ as in (4.1), this step can be accomplished by means of the Yule-Walker or Burg method (Priestley, 1982) or by the independence method explained below. For non-linear functions ψ, the common Yule-Walker and Burg methods does not apply as the dependencies are non-linear in the random variables $\mathbf{Z}_t$ and as the common Yule-Walker methods handles only linear correlations. In this case, the independence method can be used. We also show in chapter 5 that the Yule-Walker methods can be extended so that it can catch non-linear dependencies in some special stochastic process.

In the next step of our approach, the $\hat{\mathbf{y}}_t$ are estimated for each (ψ, $\mathbf{p}$) pair and their corresponding estimated $\hat{\mathbf{\Sigma}}_\alpha$:

$$\hat{\mathbf{y}}_t = \mathbf{z}_t - \psi(\mathbf{z}_{t-1}, \mathbf{z}_{t-2}, ..., \mathbf{z}_{t-p}, \hat{\mathbf{\Sigma}}_\alpha), \tag{4.5}$$

where $t = p+1, ..., n$. Next, for each $i = 1, 2, ..., d$, a distribution is fitted to the estimated IID samples (perturbations) $y_{i,t}$. This can be accomplished with a tool like ExpertFit, Arena Input Analyzer, or with the GAF technique. At the end of step 2, estimated $\hat{\mathbf{\Sigma}}_\alpha$ and $\hat{F}(y_i)$ are available for (several) (ψ, p) pair(s). This means that different estimated nGAR model parameters $\hat{\Omega}$ might be available.

In the next step of the procedure, the best set of the estimated model parameters $\hat{\Omega}$ is chosen according to one or several statistical tests. The procedure of fitting an nGAR process in general as an algorithm is given below:

Algorithm 1. The general algorithm for fitting an nGAR model to measurements

begin

1. Select a $(\psi, \mathbf{p})$ pair that has not yet been selected
2. Estimate $(\hat{\mathbf{\Sigma}}_\alpha)$ by means of:
   a. A parametric method like the Yule-Walker or
   b. The independence method.
3. Estimate $\hat{y}_{i,t}$ for all $i = 1, 2, ...d, t = p+1, ..., n$
4. Fit all $\hat{y}_i s$ to standard distributions $\hat{F}_i(.), i = 1, 2, ...d$
5. If there remains a not yet selected parameter pair go to 1
6. Choose the $(\psi, \mathbf{p})$ pair and the corresponding $\hat{\mathbf{\Sigma}}_\alpha$ and $\hat{F}_i(.), i = 1, 2, ...d$ which gives the best goodness-of-fit test statistics

end

More details about the procedure is given in the following two subsections.

4.3 Fitting Linear nGAR models

Let us consider fitting a linear univariate nGAR model similar to that given in (4.1), where the random variables Y_t are drawn from a specific probability distribution function. As mentioned, measurements should be available. The known methods for estimating the AR order like the *Akaike* or *Schwarz information criterion* do not work here, as the provided measurements are usually not normally distributed. Instead, an order of the nGAR, $\hat{p}_{test}$, is chosen, and a parametric method like the Yule-Walker method or Burg method is used to estimate the nGAR coefficients $\hat{\alpha}_h$, $h = 1, 2, .., \hat{p}_{test}$.

Assuming that the correlation function is declining, if $\hat{p}_{test}$ is higher than the actual order, p, the estimated $\hat{\alpha}_h$ will contain "small" nGAR coefficients for lags higher than p. In the case that the sample is highly correlated and $\hat{p}_{test}$ is smaller than p, $\hat{\alpha}_{\hat{p}_{test}}$ will not be small. We mean by the term "small AR coefficient" in linearly correlated models like those given in (4.1) any value smaller than 3%. At the end, one can apply an independence test on the estimated perturbations $\hat{\mathbf{y}}_i$ to be sure that considerably high nGAR coefficients were not neglected. The whole procedure can be implemented as a sequential procedure so that it checks all lags $i = 1, 2, ..., k$ at which the value of the nGAR coefficients are not small and finally checks the independence of the $\hat{\mathbf{y}}_i$.

We noticed during our fittings that a sample which is generated from an IID distribution might have positive or negative nGAR coefficient values up to 3%. However, these values of nGAR coefficients are a result of randomness and we still can consider the variables in the sample as independent. We follow the same principle when considering real measurements. All values of nGAR Coefficients which might be result of randomness will be considered small and will not be included in the fitted model.

Having nearly independent estimated samples $\hat{\mathbf{y}}_i$, a distribution can be fitted. At this point, the parameters $\hat{\Omega}$ of the nGAR model (4.1) are estimated, a sample can be generated, and statistical goodness-of-fit tests can be applied to compare the original measurements with the generated sample.

Fitting a bivariate or multivariate nGAR model is similar to the case of fitting univariate model. However, in the case of multivariate nGAR model, the order of the different sub-processes in the model might be different. This results in the need to estimate a different orders p_i for each sub-process in the multivariate model. In subsection 4.6.2 we show an example of fitting a bivariate nGAR model assuming that the two sub-processes in the model have the same nGAR order (maximal correlation lag).

The algorithm of fitting a linearly correlated nGAR model is a simplified procedure of that shown in algorithm 1, as the function ψ is always the func-

tion of matrix multiplication when considering linearly correlated processes. The algorithm is given below:

Algorithm 2. Fitting a linearly correlated nGAR model to measurements

begin

1. Estimate $\hat{\mathbf{\Sigma}}_\alpha$ and the nGAR order $\hat{\mathbf{p}}_{test}$
2. Estimate the perturbations $\hat{\mathbf{y}}_{i,t}, i = 1, 2, ...d, t = p + 1, ...n$
3. Fit all $\hat{\mathbf{y}}_i s$ to standard distributions $\hat{F}_i(.)$,
4. Perform goodness-of-fit tests

end

4.4 The Independence Method

Another way to fit an nGAR model to measurements is to use an independence test that utilizes the Chi-square independence function. This method is used if the above described procedure is not applicable due to non-linear dependencies in the measurements or non-existing moments. Our independence method is used to accomplish step 2 in Algorithm 1 in the procedure of estimating the nGAR model parameters $\Omega = \{\psi, \mathbf{\Sigma}_\alpha, \mathbf{p}, F_i, i = 1, 2, ..., d\}$.

The chi-square independence test is used to determine whether there is dependency between a row variable and column variable in a contingency table constructed from sample data (Miket, 2006). The null hypothesis is that the variables are independent. The alternative hypothesis is that the variables are dependent.

The idea behind testing these types of claims is to compare actual counts to the counts we would expect if the null hypothesis were true (if the variables are independent). If a significant difference between the actual counts and the expected counts exists, we would take this as evidence against the null hypothesis.

The estimation of $\mathbf{\Sigma}_\alpha$ is accomplished in a recursive procedure for each hypothesized pair (ψ, $\mathbf{p}$). This requires that the function ψ is known or a set of possible functions ψ are assumed and implemented. In this recursive procedure, the perturbations $\hat{\mathbf{y}}_t$, $t = p + 1, ..., n$ are estimated using (4.5) and their independence is tested. If the perturbations are dependent, the parameters $\hat{\mathbf{\Sigma}}_\alpha$ must be adjusted. This results in an optimization procedure according to an objective function of independence.

In the following, we explain in more detail how the independence of the perturbations $\hat{\mathbf{y}}_t$ is postulated. Two vectors of realizations $\mathbf{x}$ and $\mathbf{y}$ can be

tested for independence using the chi-square independence function. The test requires in addition to the vectors a degree of freedom ν and a significance level (of rejection). If the test statistics calculated exceeds a value specified in the chi-square table under the selected ν and significance level, the vectors are said to be dependent.

The calculation of the test statistic is described for example in detail in Lehn and Wegmann (1992) and is calculated as follows: Having the two vectors of realizations $\mathbf{x}$ and $\mathbf{y}$, the corresponding sample pairs (x, y)'s from those vectors are sorted in different regions (u, v), $u = 1, 2, ..., k$, $v = 1, 2, ..., l$. The number of (x, y) pairs in each region (u, v) is then denoted as N_{uv}. $N_{u\bullet}$ denotes the number of pairs in the regions (u, v) for all $v = 1, 2, ..., l$. $N_{\bullet v}$ denotes the number of pairs in the regions (u, v) for all $u = 1, 2, ..., k$. This results in a contingency table as shown below, where N=n is the number of pairs in the sample.

Table 4.1: The contingency table

N_{11}	N_{12}	...	N_{1l}	$N_{1\bullet}$
N_{21}	N_{22}	...	N_{2l}	$N_{2\bullet}$
...	...	...	...	...
N_{k1}	N_{k2}	...	N_{kl}	$N_{k\bullet}$
$N_{\bullet 1}$	$N_{\bullet 2}$	...	$N_{\bullet l}$	N

The number of samples in the different $N_{\bullet v}$ and $N_{u\bullet}$ must guarantee to satisfy the chi-square test conditions:

1. All expected frequencies in all regions (u, v) are greater than or equal to 1 and,

2. No more than 20% of the expected frequencies are less than 5.

These two conditions are guaranteed to be satisfied by choosing the regions N_{ij} to have different widths. The Chi-square independence function (4.6) is then applied to get a positive test statistic, Q. The smaller the test statistic is, the more independent $\mathbf{x}$ and $\mathbf{y}$ are:

$$Q = n[(\sum_{u=1}^{k}\sum_{v=1}^{l}\frac{{N_{uv}}^2}{N_{u\bullet}N_{\bullet v}}) - 1]. \tag{4.6}$$

When fitting an univariate nGAR model Z_t, during the fitting procedure, the independence of all pairs of random variables $(\hat{y}_t, \hat{y}_{t-h})$, $h = 1, 2, ..., p$, where p is the maximum lag (nGAR order), must be postulated. This means

that the sample vectors $(\hat{y}_t,\ \hat{y}_{t-h})$, for all $h = 1, 2, .., p$ must be nearly independent. This requires applying (4.6) p times, one time for each h. The values of all Q_h, $h = 1, 2, ..., p$, where Q_h is the chi-square-statistics for the sample vectors $(\hat{y}_t,\ \hat{y}_{t-h})$, can then be averaged

$$\bar{Q} = \frac{\sum_{h=1}^{p} Q_h}{p}, \tag{4.7}$$

and considered as the objective function value for independence. The smaller $\bar{Q}$ is, the more independence lies among the random variables $(\hat{Y}_t,\ \hat{Y}_{t-h})$ for all $h = 1, 2, .., p$.

One can reject the hypothesis that the random variables $(\hat{Y}_t,\ \hat{Y}_{t-h})$ are independent if $\bar{Q}$ exceeds one value taken from the chi-square table. Alternatively, one can assume more difficult conditions. For example, the maximum Q_h over all h does not exceed a value. Similar to the known chi-square method mentioned before, the value chosen from the table depends on a significance level and a degree of freedom. The significance level is a personal choice and lies usually in the range [1%,5%]. The degree of freedom ν depends on the number of spans in the contingency table by the formula: $\nu = (k-1)(l-1)$.

When fitting a d-variate nGAR model $\mathbf{Z}_t$ to a d-variate sample and during the estimation of the $\mathbf{\Sigma}_\alpha$, one must postulate the independence of a d-variate $\hat{\mathbf{y}}_t$ with a target lag $\mathbf{p}$. For the purpose of testing a d-variate $\hat{\mathbf{y}}_t$ with a target lag $\mathbf{p}$ for independence, all samples $\hat{\mathbf{y}}_t$ must be independent of each other. Assuming for simplicity that all sub-processes in the nGAR model $\mathbf{Z}_t$ have the same maximum lag p, the procedure means that for all $t = p+1, ..., n$, the pairs $(\hat{y}_{i,t}, \hat{y}_{j,t-h})$ are tested for independence for each $h = 1, 2, ..., p$, and $i, j = 1, 2, ..., d$. If the maximum lag of the different sub-processes are different, one can take p as the $\max(p_i)$, $1 \leq i \leq d$. This assumption does not affect the accuracy of the procedure but results only in redundancy, which is estimating autoregressive coefficients which can be practically neglected.

Lets consider the simple example of fitting the bivariate nGAR model $\mathbf{Z_t}$:

$$Z_{1,t} = \alpha_{1,1} Z_{1,t-1} + \alpha_{1,2} Z_{1,t-2} + Y_{1,t}$$

$$Z_{2,t} = \alpha_{2,1} Z_{2,t-1} + \alpha_{2,2} Z_{2,t-2} + Y_{2,t}$$

Here we have the maximum lag (nGAR order) $p = 2$, the model is bivariate which means that $d = 2$, and there is no cross correlations among the two sub-processes Z_1 and Z_2. When fitting such a model to a sample which satisfies these conditions, during the recursive procedure of estimating

the linear autoregressive coefficients $\mathbf{\Sigma}_\alpha$=$\{\alpha_{1,1}, \alpha_{1,2}, \alpha_{2,1}, \alpha_{2,2}\}$, the modeler will notice that there is no cross dependencies between the variables in different sub-processes. In other words, all chi-square statistics which represent the cross dependencies between the random variables in different processes will be small and will most probably satisfy the chi-square test conditions to accept the null hypothesis. All other chi-square test statistics will be small enough when a good estimation of the aGAR coefficients $\hat{\mathbf{\Sigma}}_\alpha$ is reached.

In general, a multivariate nGAR model might consist of several sub-processes. Each random variable in each sub-process i might depend on the previous p_i random variables from the same sub-process, where p_i is the order of the ith sub-process. Moreover, each random variable in each sub-process might depend on random variables from another sub-process. This type of dependency is called cross-dependency. The overall number of dependencies and cross dependencies in such a d-variate process with order p for all sub-processes is equal to pd^2.

For example, in a tri-variate nGAR model with maximum lag $p = 2$, the number of dependencies in the whole nGAR process is equal to 6, 2 dependencies in each sub-process. The number of cross-dependencies among the sub-processes is equal to 12. The total number of dependencies is 18 which is equal to $2 * 3^2$. In an nGAR process with $p = 3$ and $d = 3$, the number of dependencies is 9 and the number of cross-dependencies is 18. The total number of dependencies is 27.

In a d-variate process with lag p, let $Q_{i,j,h}(\hat{\mathbf{\Sigma}}_\alpha)$ denote the chi-square test statistics for the random variables $(Y_{i,t},\ Y_{j,t-h})$ where $i, j = 1, 2, ..., d$ and $h = 1, 2, ..., p$, then, the objective function of independence might be defined as:

$$Q(\hat{\mathbf{\Sigma}}_\alpha) = \sum_{i,j=1}^{d} \sum_{h=1}^{p} Q_{i,j,h}(\hat{\mathbf{\Sigma}}_\alpha)/pd^2. \tag{4.8}$$

Different $\hat{\mathbf{\Sigma}}_\alpha$ values result in different values for $Q(\hat{\mathbf{\Sigma}}_\alpha)$. The best $\hat{\mathbf{\Sigma}}_\alpha$ is calculated as:

$$\hat{\mathbf{\Sigma}}_{\alpha,\text{best}} = \arg\min_{\hat{\mathbf{\Sigma}}_\alpha}\ Q(\hat{\mathbf{\Sigma}}_\alpha).$$

All $\hat{\mathbf{\Sigma}}_\alpha$ values must not be predefined before beginning the fitting process. Using a generic algorithm similar to that introduced in Strelen (2003) and described in 4.1, one can use the best $\hat{\mathbf{\Sigma}}_\alpha$ values to generate new $\hat{\mathbf{\Sigma}}_\alpha$ values which in turn can be used as input for Algorithm 3. These steps can be repeated till the chi-square test statistics satisfy the conditions specified by the modeler.

The algorithm for selecting the best $\hat{\mathbf{\Sigma}}_{\alpha,\text{best}}$ using the independence method, which in turn utilizes the genetic algorithm for the purpose of minimizing an objective function, is as follows:

Algorithm 3

begin

while the independence test statistics $Q(\hat{\mathbf{\Sigma}}_\alpha)$ is not smaller than a specified value **do**

1. In a predefined set of $\hat{\mathbf{\Sigma}}_\alpha$, Select a $\hat{\mathbf{\Sigma}}_\alpha$ that has not yet been selected
2. Estimate all $y_{i,t}$, $i = 1, 2, ...d,\ t = p+1, ...n$, by (4.5)
3. Estimate the chi-square test statistics for all pairs of $(y_{i,t}, y_{j,t-h})$, $i, j = 0, 1, ...d, h = 1, 2, ..., p$, by (4.6)
4. Apply (4.8) to get the independence test statistics $Q(\hat{\mathbf{\Sigma}}_\alpha)$
5. If there remains a not yet selected parameter pair go to 1
6. Choose several $\hat{\mathbf{\Sigma}}_\alpha$ which give the best independence test statistics calculated by (4.8)
7. Apply the best $\hat{\mathbf{\Sigma}}_\alpha$ as input for the genetic algorithm. The output of the genetic algorithm is a new set of $\hat{\mathbf{\Sigma}}_\alpha$

end

end

To this end, we could find the best $\hat{\mathbf{\Sigma}}_\alpha$ and $\hat{F}_i(.)$ using the independence method. This is step 2 in Algorithm 1 of section 4.2. However, if more than one ψ function or more than one p value are considered, $\hat{\mathbf{\Sigma}}_{\alpha,\text{best}}$ and the distributions $\hat{F}_i(.)$, $i = 1, 2, ..., d$ are searched for each (ψ, p) pair. $\hat{\mathbf{y}}_i$ are then estimated and tested for independence for all $i = 1, 2, ..., d$.

The (ψ, p) pairs with their corresponding $\hat{\mathbf{\Sigma}}_{\alpha,\text{best}}$ and $\hat{F}_i(.)$ are considered and therefore different nGAR parameters $\hat{\Omega}$ will be available. The parameters $\hat{\Omega}_{\text{best}}$ is then chosen with the help of goodness-of-fit tests. In other words, the original sample is tested against a sample generated from the fitted nGAR model that have the different parameters $\hat{\Omega}$. The set of parameters $\hat{\Omega}$ which delivers the best test statistics is chosen to be the "optimal" one. More than one goodness-of-fit test can also be considered. Algorithm 1 in section 4.2 summarizes the whole fitting procedure.

4.5 Goodness-of-Fit Tests

In this section, we overview one standard goodness-of-fit test (the Kolmogorov-Smirnov test) and two other tests that are functions of the residuals. The residuals are defined as:

$$\text{residual} = \text{observation-generated value}$$

4.5.1 The Mean Squared Residuals Test

The Mean Squared Residuals (MSR) test is the sum of the squares of the residuals between points generated from the fitted model and corresponding points from the original sample. The points are *made* corresponding to each other if the generated sample and the original sample are ordered. If the ordered original sample is denoted as s_o, and the ordered generated sample is denoted as s_g, the MSR is given by equation (4.9).

$$MSR = \sum_{i=1}^{n} \left(s\_o^{(i)} - s\_g^{(i)}\right)^2 / n, \tag{4.9}$$

4.5.2 The Mean Absolute Residuals Test

The MAR test is the sum of the absolute residuals between points generated from the fitted model and corresponding points in the original sample. Here ordering must also take place. If the ordered original sample is denoted as s_o, and the ordered generated sample is denoted as s_g, the MAR is given by equation (4.10).

$$MAR = \sum_{i=1}^{n} \left|(s\_o^{(i)} - s\_g^{(i)}\right| / n, \tag{4.10}$$

4.5.3 The Kolmogorov-Smirnov Test

The two-sample Kolmogorov-Smirnov goodness-of-fit hypothesis test (MathWork (2001) and Law and Kelton (2000)) is used to determine whether two samples have been drawn from the same distribution. For this purpose, for each potential value x in the first sample, the Kolmogorov-Smirnov test compares the proportion of values less than x in the first sample with the proportion of values less than x in the second sample. The first sample would be the generated sample from the fitted model and the second sample would

be the original sample. The Kolmogorov-Smirnov (KS) test function uses the maximum difference over all x_t values, $t = 1 + p, ..., n$, as its test statistics. Mathematically, this can be written as:

$$D = \max_{t=1+p,...,n} \left| \tilde{F}_1(x_t) - \tilde{F}_2(x_t) \right|$$

where $\tilde{F}_1$ and $\tilde{F}_2$ are the empirical distribution functions of the two samples and D is called the test statistics. The hypothesis regarding the distributional form is rejected if the test statistics, D, is greater than the critical value obtained from the Kolmogorov-Smirnov test table.

4.6 Examples

In this section, we give examples that show results of our fitting procedures. The examples will clarify the previously described procedures for fitting nGAR models to data obtained by measurements or generated artificially from known models. The data used in the first three examples are generated artificially from nGAR models with known parameters. We call these data as the artificial sample and the models which generated these data as the artificial models. Our goal for these the first three examples is to find out how well the fitting procedure recovers the original parameters (Ω) of the true artificial models. The data used in the fourth and fifth examples are measurements on real systems.

The fitted models are tested for accuracy by comparing statistical moments like the mean, variance, and correlations, and by using statistical goodness-of-fit tests like those overviewed in section 4.5 and by comparing scatter plots of the samples.

4.6.1 Fitting Linear Univariate nGAR Processes

We consider an nGAR model with the following parameters Ω: The function ψ is the matrix-multiplication shown in (4.4). $\mathbf{\Sigma}_\alpha$ is a 1×2 matrix $[\alpha_1, \alpha_2] = [0.4, 0.2]$. The random variables Y_t are Pareto distributed $F(y) = 1 - (b/y)^a$, $b < y < +\infty$, having b (scale parameter) $= 1.7$ and a (shape parameter) $= 3.7$. This specifies $F(y, a, b)$. The resulting model is as follows:

$$Z_t = \alpha_1 Z_{t-1} + \alpha_2 Z_{t-2} + Y_t.$$

An artificial sample with 10000 sample points from the above described nGAR model is generated. When starting to generate from the above model, the process is not stationary and the elements in the initial phase must be

ignored because they are not IID. The large sample size is considered due to the property of the Pareto distribution under the specified shape parameter a. Pareto distributions with a shape parameter $a < 2$ have infinite variance, which results in that the generated random variables are disperse along wide range. Therefore, a relatively large sample size is needed to capture enough information about the true process.

An nGAR model is fitted to the artificial sample as follows: A function ψ (linear multiplication) and an nGAR order $p = 3$ are considered. Next, $\hat{\mathbf{\Sigma}}_\alpha = [\hat{\alpha}_1, \hat{\alpha}_2, \hat{\alpha}_3]$ is estimated using the independence method. Noticing that $\hat{\alpha}_3$ is small and that the $\hat{y}_t$ are nearly independent even when we consider the order $\hat{p} = 2$, we suggest that an order $\hat{p} = 2$ is suitable.

Having $[\hat{\alpha}_1, \hat{\alpha}_2]$ and $\hat{p}$ estimated for ψ, $\hat{y}_t$ are built using

$$\hat{y}_t = z_t - \hat{\alpha}_1 z_{t-1} - \hat{\alpha}_2 z_{t-2}. \tag{4.11}$$

Next, the distribution, $F(y)$, and its parameters, $\hat{a}, \hat{b}$, that fit $\hat{y}_t$ best are estimated using the GAF techniques introduced in Strelen (2003) and reviewed shortly in section 4.4. The test statistics used to select the best fitted distribution $F(y)$ and its parameters a, b depend on the principle described in 4.5. We call the tests as MAR($F(y)$) (mean absolute residuals), MSR($F(y)$) (mean squared residuals), and KSS (Kolmogorov-Smirnov test). The three distributions that fit $\hat{y}_t$ best and their corresponding test statistics are shown in table 4.3.

Having specific sets of parameters $\hat{\Omega}$, an nGAR sample can be generated and compared statistically with the artificial sample. The MSR(process) test statistic is calculated as shown in subsection 4.5.1 whereas MAR(process) is calculated as shown in subsection 4.5.2. KSS(process) is the Kolmogorov-Smirnov test statistics calculated as shown in subsection 4.5.3. The tests MAR and MSR are applied two times. One time to compare a sample generated from the artificial random variables Y_t with a sample generated from the fitted random variable $\hat{Y}_t$. These tests are called MAR($F(y)$) or MSR($F(y)$). And one time to compare a sample generated from the artificial (original) process Z_t with a sample generated from the fitted process $\hat{Z}_t$. Those tests are called MAR(process) or MSR(process). The comparison of the samples is done by building the empirical distribution functions of the two samples to be compared and applying then the MAR and MSR functions on these empirical distributions.

Table 4.3 summarizes the results of the fitting procedure considering the three distributions $F(y)$ that deliver the best (smallest) MSR($F(y)$) values. $[\hat{\alpha}_1,\hat{\alpha}_2]$ are the fitted nGAR coefficients. $[\hat{b}, \hat{a}]$ are the best fitted parameters of the Pareto, Weibull, Lognormal, and Gamma distributions. We notice that

all of the test statistics tend to be smaller (better) in the case of choosing $F(y)$ to be the Pareto distribution. Table 4.2 shows the true nGAR parameters again for a better comparison.

Table 4.2: The parameters of the true artificial nGAR model

$[\alpha_1,\alpha_2]$	[0.4, 0.2]
$F(y)$	**Pareto**
$[b,\ a]$	[1.7, 3.7]

Table 4.3: Results summary of fitting univariate linear nGAR models to the artificial sample

$[\hat{\alpha}_1,\hat{\alpha}_2]$	[0.403, 0.193]			
$F(y)$	**Pareto**	**Weibull**	**Lognormal**	**Gamma**
$[\ \hat{b},\ \hat{a}]$	[1.69, 3.76]	[2.59, 6.99]	[1.75, 0.43]	[4.45, 1.48]
MAR ($F(y)$)	0.007	0.0618	0.037	0.04
MAR (process)	1.4	6.8	6.0	6.2
MSR ($F(y)$)	7.4565e-005	0.005	0.003	0.004
MSR (process)	35.3	325	291	294
KSS (process)	0.01	0.304	0.26	0.229
KS accepted	KS accepted	KS rejected	KS rejected	KS rejected

The scatter plots show whether a sample from the fitted nGAR models and the artificial sample have similar patterns. Figure 4.1, figure 4.2, figure 4.3, and figure 4.4 show the scatter plots (Z_t, Z_{t+1}), (Z_t, Z_{t+2}) from the artificial sample and samples from four fitted nGAR models. We notice that the sample from the fitted nGAR model with $F(y)$ of Pareto have similar patterns to the artificial sample.

Last we would like to express two more points. The first one is the computational aspects of the method described above. As the nGAR coefficients $\mathbf{\Sigma}_\alpha$ are estimated using the independence method, the computational cost for this method is relatively high. The algorithm of finding a $\hat{\mathbf{\Sigma}}_\alpha$ that makes the $\hat{Y}_t$ almost independent stops only when the value of the objective function for independence given in (4.7) lies below a critical value taken from the chi-square table, which depends as mentioned on the degree of freedom ν and a significance level. For a $\nu = 25$ and a significance level of 10% the critical value lies at 16.5, and the independence method must run on average for about 30 minutes on a Pentium 4 computer with 512 MB RAM before the objective function value reaches a value below 16.5. However, if the objective functino value does not satisfy the termination condition after

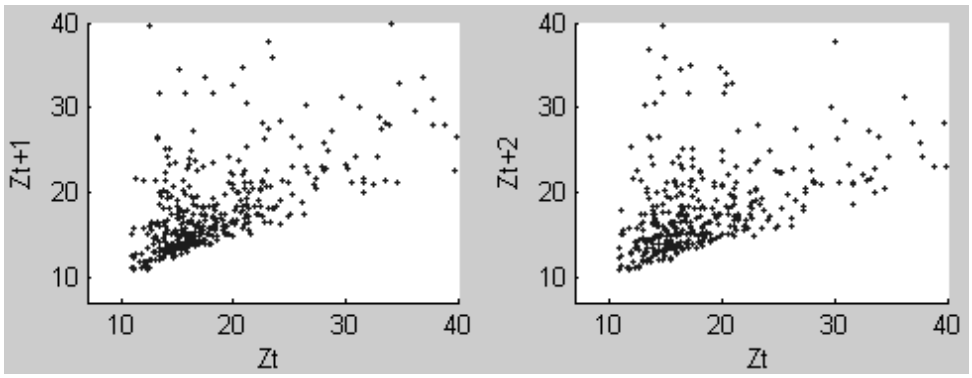

Figure 4.1: Plots from the artificial model

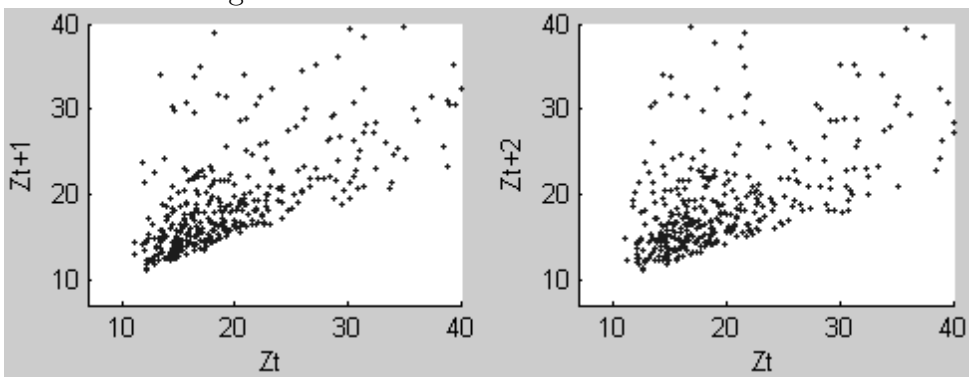

Figure 4.2: Plots from the fitted nGAR model with Pareto $F(y)$

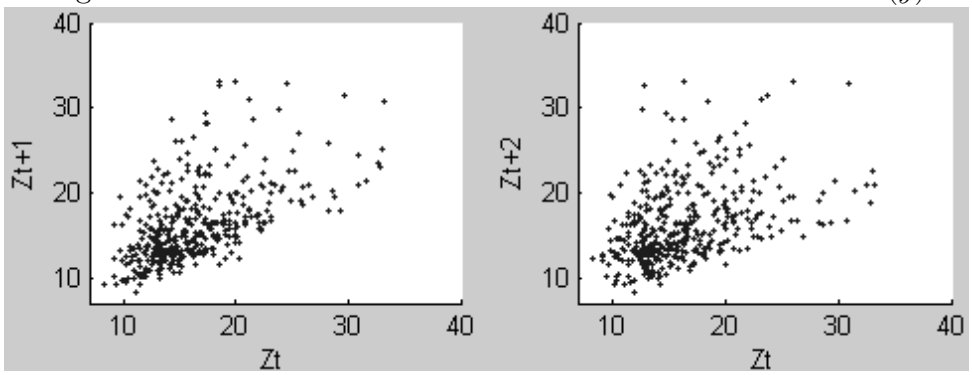

Figure 4.3: Plots from the fitted nGAR model with Lognormal $F(y)$

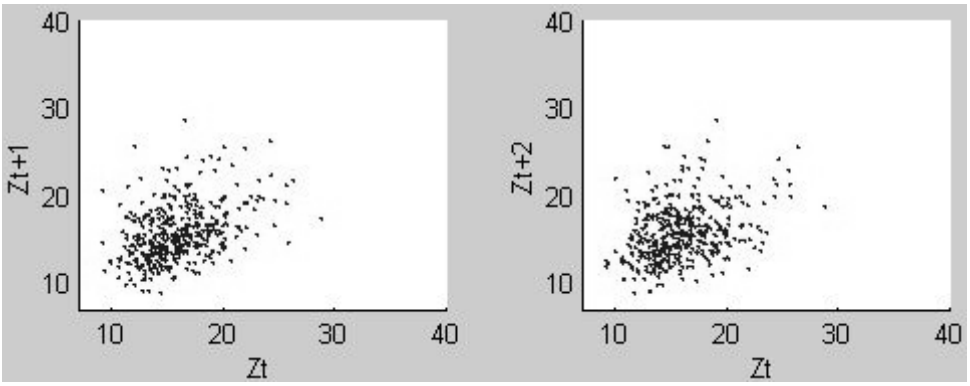

Figure 4.4: Plots from the fitted nGAR model with Gamma $F(y)$

a specified number of genetic algorithm generations, the optimization procedure can be stopped and the fitting procedure can procede with the best parameters obtained up to this point.

The second point to express is that usually the distribution of the Y_t is not known. Therefore, one in principle must try all standard distributions and choose the one which delivers the best goodness-of-fit statistics. Alternatively, one can fit an empirical distribution function. Fitting standard or empirical distribution to samples is itself not a time consuming procedure as the fitting can be accomplished very fast using for example the maximum likelihood estimator (MLE) to fit a standard distribution.

4.6.2 Fitting a Linear Bivariate nGAR Process

We generate an artificial sample of size 5000 from the following linear bivariate nGAR model:

$$A_t = \alpha_1 A_{t-1} + \alpha_2 A_{t-2} + \alpha_3 B_{t-1} + \alpha_4 B_{t-2} + Y_{At}$$
$$B_t = \beta_1 B_{t-1} + \beta_2 B_{t-2} + \beta_3 A_{t-1} + \beta_4 A_{t-2} + Y_{Bt}$$

where $\underline{\alpha}$=[0.20,0.15,-0.15,0.10], $\underline{\beta}$=[0.20,-0.10,0.15,-0.10]. The random variables Y_{A_t} are Weibull distributed with the parameters $a_1 = 2$ (shape), and $b_1 = 10$ (scale). The random variables Y_{B_t} are Weibull distributed with the parameters a_2 (shape) $= 1$ and b_2 (scale) $= 6$. Elements in the initial phase are ignored.

A bivariate nGAR model is fitted as follows: A function ψ (linear multiplication) and an nGAR order $p = 4$ are assumed. Next, $\mathbf{\Sigma}_\alpha = [\underline{\alpha}; \underline{\beta}]$ is estimated using the Yule-Walker method. Another method like the Burg method can also be used. De Hoon et al. (1996) notice that the Yule-Walker

method leads to poor parameter estimations in special cases even for a moderate sample size. De Hoon et al. (1996) shows which poor conditions of the autocovariance matrices are regarded as the cause of poor Yule-Walker estimates. Side-effects of the poor autocovariance matrix condition are an almost non-stationary, pseudo-periodic behavior of the autoregressive process and partial autocorrelation coefficients close to unity.

Having $\hat{\mathbf{\Sigma}}_\alpha$ estimated for ψ and p, the bivariate random vectors $(\hat{y}_{A_t}, \hat{y}_{B_t})'$ are estimated using (4.5). Next, the distributions and the parameters that fit $\hat{y}_{A_t}$ and $\hat{y}_{B_t}$ best are estimated. For the purpose of comparison, we fit the artificial sample directly to standard distributions neglecting the fact that the artificial sample is correlated. The best fitted distribution and parameters are shown in table 4.5.

We notice from table 4.5 that an artificial sample with a small size smaller than that used in example 1 can recover the parameters of the true model with satisfying accuracy. This is due to the fact that Y_{A_t} and Y_{B_t} are not heavy-tailed. $\underline{\hat{\alpha}}$ and $\underline{\hat{\beta}}$ are the fitted nGAR coefficients ($\hat{\mathbf{\Sigma}}_\alpha$). The fitted distributions to the $\hat{y}_{A_t}$ and $\hat{y}_{B_t}$ are Weibull. $[\hat{a}_1, \hat{b}_1]$ and $[\hat{a}_2, \hat{b}_2]$ are the fitted parameters. $MSR1$ and $MSR2$ are the mean squared residuals between the bivariate artificial sample and a sample generated from the fitted bivariate processes, while (KSS1, KSS2) are the Kolmogorov-Smirnov test statistics.

The test statistics MSR and KSS are smaller (better) when we fit an nGAR model and greater (worse) when we fit a standard distribution. This shows that the fitted standard distribution fit the *marginal distribution* of the artificial sample badly. Moreover, the standard distributions can generate only independent data and the correlations of the artificial sample can not be modeled. Theoretically, the nGAR coefficients $\underline{\hat{\alpha}}$ and $\underline{\hat{\beta}}$ are zero. Practically, they are a bit higher or smaller than zero.

Table 4.4: The parameters of the true artificial nGAR model

	nGAR model
$\underline{\alpha}$	[0.20, 0.15, -0.15, 0.10]
$\underline{\beta}$	[0.20, -0.10, 0.15, -0.10]
$F(y_A, a_1, b_1)$	Weibull (2.0, 10.0)
$F(y_B, a_2, b_2)$	Weibull (1.0, 6.0)

Figure 4.5, figure 4.6, and figure 4.7 show the plots (A_t, A_{t+1}) and (A_t, A_{t+2}) from the artificial sample, the fitted nGAR models, and a sample from a fitted independent distribution, respectively. We notice that the plots of the fitted nGAR model is more similar to the artificial sample than those from

Table 4.5: Results of fitting bivariate nGAR model and standard distributions to bivariate artificial sample

	nGAR Model	**Standard Distr.**
$\underline{\hat{\alpha}}$	[0.20, 0.149, -0.158, 0.098]	[0, 0, 0, 0]
$\underline{\hat{\beta}}$	[0.189, -0.89, 0.139, -0.85]	[0, 0, 0, 0]
$F(y_A, \hat{a}_1, \hat{b}_1)$	Weibull (2.02, 10.1)	$Gamma^*$ (6.5, 2)
$F(y_B, \hat{a}_2, \hat{b}_2)$	Weibull (.99, 6.2)	$LogN^*$ (2.06, 0.62)
(MSR1, MSR2)	(0.02, 0.07)	(0.13, 0.19)
(KSS1, KSS2)	(0.015, 0.016)	(0.1, 0.2)
KS1 rejected? KS2 rejected?	no, no	no, no

the independent distribution.

Figure 4.8, figure 4.9, and figure 4.10 show the plots (B_t, A_{t+1}) and (B_t, A_{t+2}) from the artificial sample, the fitted nGAR models, and a sample from a fitted independent distribution, respectively. We notice that the plots of the fitted nGAR model is more similar to the artificial sample than those from the independent distribution.

4.6.3 Fitting a non-Linear nGAR model

An artificial sample of size 10000 is generated from the following non-linear nGAR model:

$$Z_t = \alpha_1 {Z_{t-1}}^2 + \alpha_2 Z_{t-2} + Y_t \tag{4.12}$$

where $\underline{\alpha}$ =[0.034, 0.2]. Y_t are Weibull distributed IID random variables with the parameters $[p_1,\ p_2] = [2.7,\ 4]$. The linear correlations of the artificial sample, $\rho_Z(1)$, $\rho_Z(2)$, $\rho_Z(3)$, $\rho_Z(4)$, $\rho_Z(5)$, and $\rho_Z(6)$, have the values 0.57, 0.47, 0.32, 0.25, 0.17, and 0.10, respectively.

We consider fitting linear and non-linear nGAR models to the artificial sample. The assumed functions ψ are as follows:

$$\psi_1(Z_{t+1}, Z_{t+2}, Z_{t+3}) = \alpha_1 Z_{t+1} + \alpha_2 Z_{t+2} + \alpha_3 Z_{t+3},$$

and

$$\psi_2(Z_{t+1}, Z_{t+2}) = \alpha_1 {Z_{t+1}}^2 + \alpha_2 Z_{t+2}.$$

In both cases of ψ, the parameters $\mathbf{\Sigma}_\alpha$ are first estimated. Next, the (nearly) independent $\hat{Y}_t$ are estimated and the distribution of the $\hat{Y}_t$, $F(y)$, is determined. Table 4.7 summarizes some results of the fitting procedure.

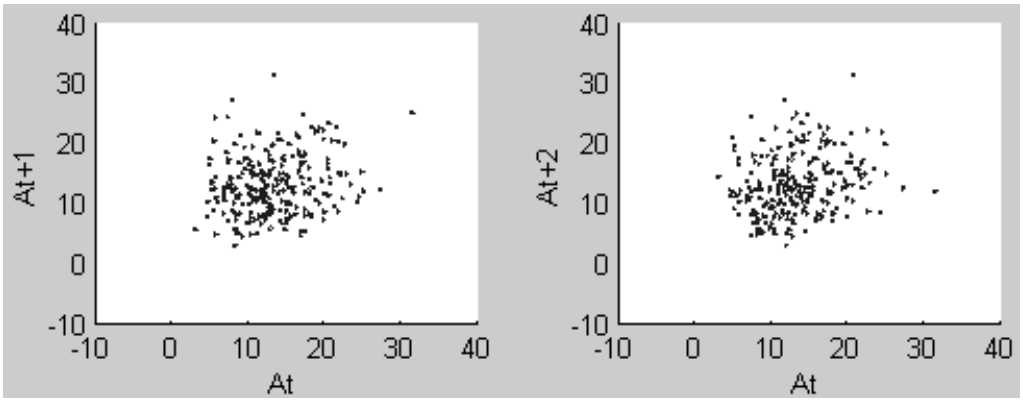

Figure 4.5: Plots from the artificial model. Correlations in sub-process A

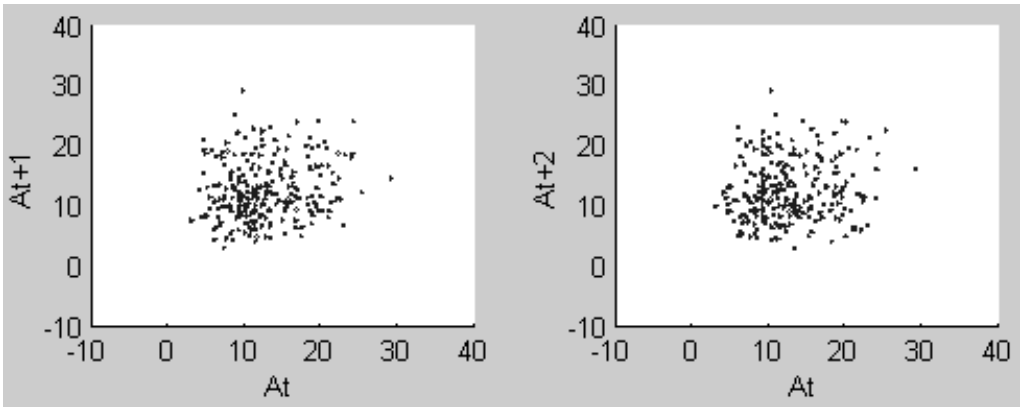

Figure 4.6: Plots from the fitted nGAR model. Correlations in sub-process A

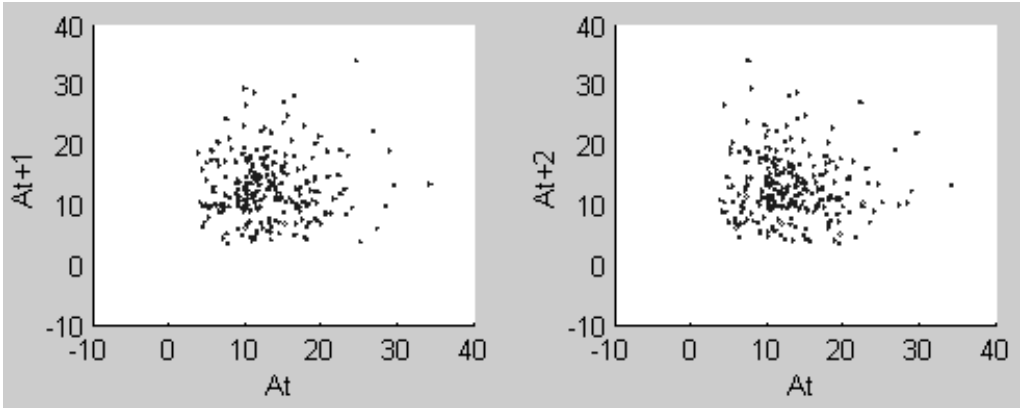

Figure 4.7: Plots from the fitted standard distribution to sub-process A without correlations

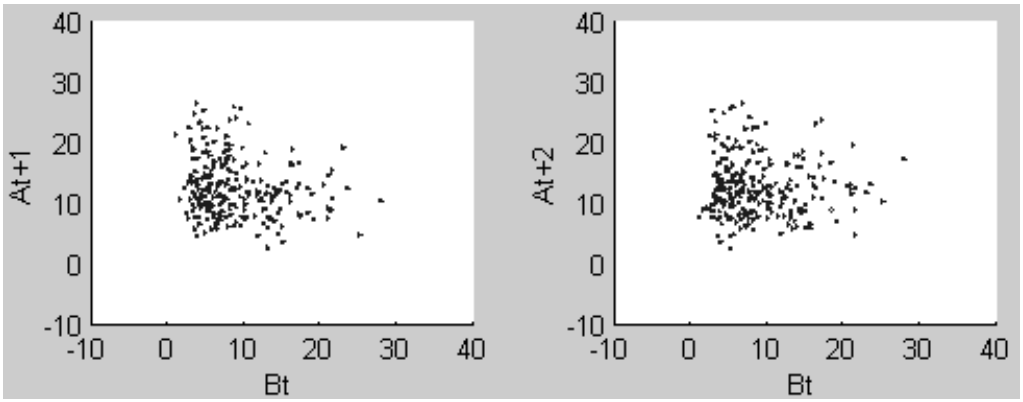

Figure 4.8: Plots from the artificial model. Cross-correlations of sub-process A and sub-process B

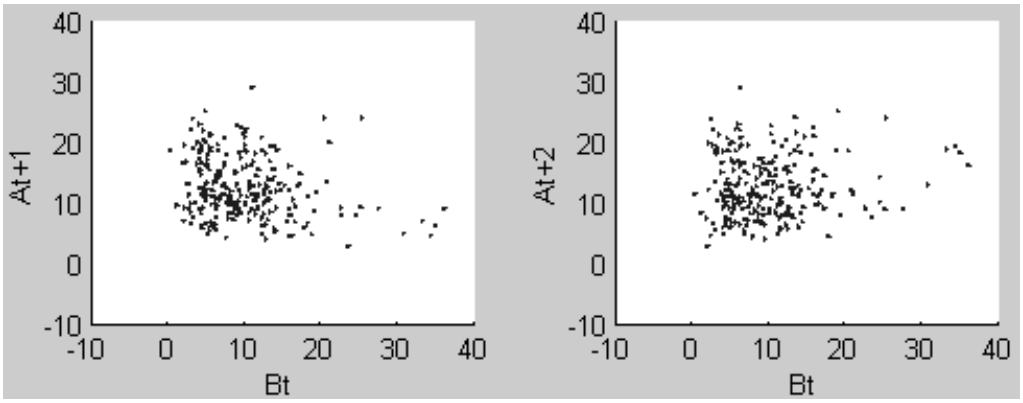

Figure 4.9: Plots from from the fitted nGAR model. Cross-correlations of sub-process A and sub-process B

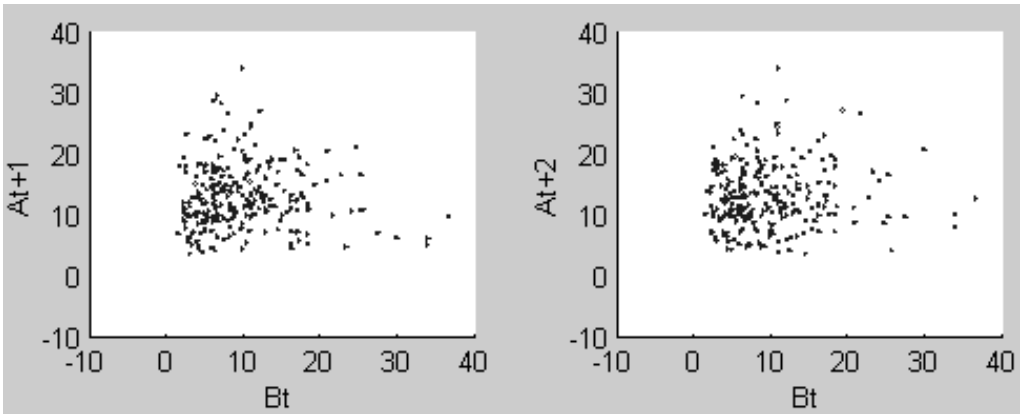

Figure 4.10: Plots from the fitted standard distribution. Cross-correlations of sub-process A and sub-process B

Table 4.6: The parameters of the true artificial model

	artificial model
$\underline{\alpha}$	[0.034, 0.2]
$F(y, a_1, b_1)$	Weibull (2.7, 4.0)
mean	6.165
variance	2.989
$\rho_Z(1)$, $\rho_Z(2)$, $\rho_Z(3)$	[0.57, 0.47, 0.32]

The entries of the table are similar to those described in the previous examples. We notice that the correlations of the linear and non-linear nGAR models are similar to those of the artificial sample. The test statistics MSR and KSS of the non-linear nGAR model are better than the linear one. Hence, one would select the non-linear nGAR model as the model which fits better. We also notice that the plots of the fitted non-linear nGAR process and the plots of the artificial sample are alike. This is not the case considering plots of the linear nGAR model. The plots are shown in figure 4.11, figure 4.12, and figure 4.13.

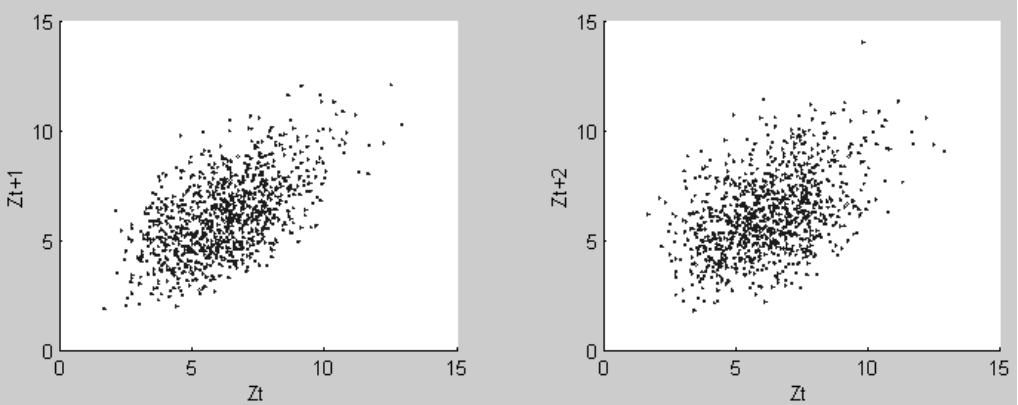

Figure 4.11: Plots from the artificial model.

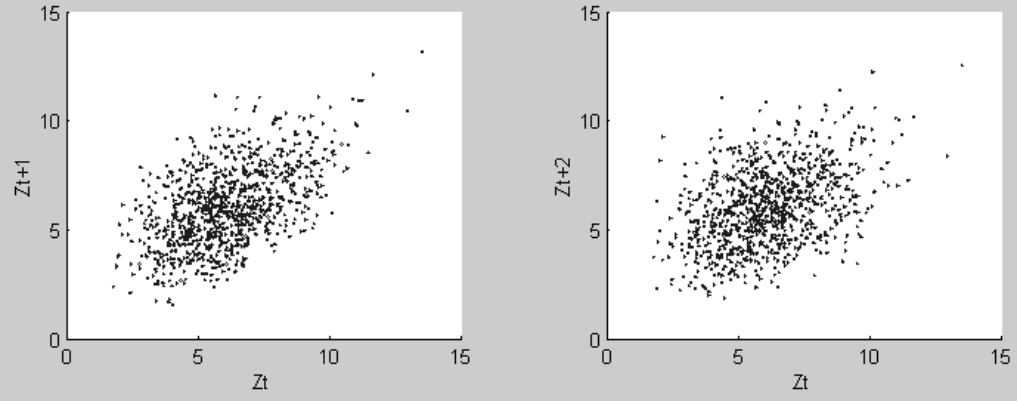

Figure 4.12: Plots from from the fitted non-linear nGAR model.

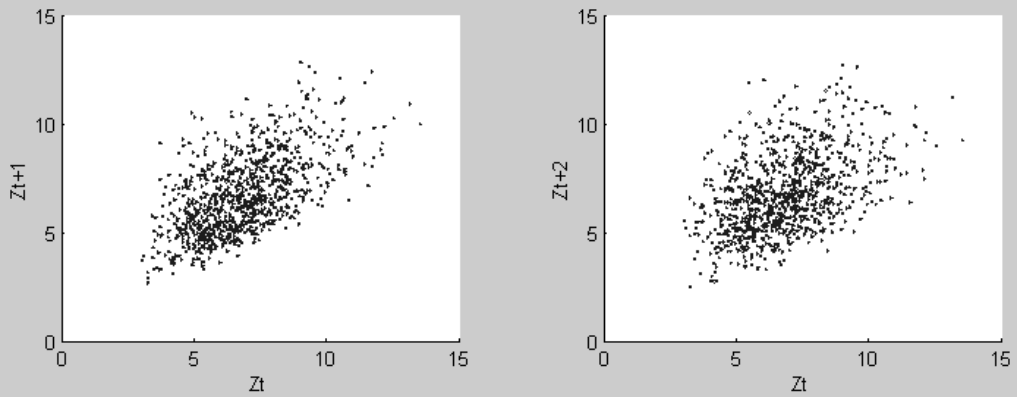

Figure 4.13: Plots from the fitted linear nGAR model.

Table 4.7: Results summary of fitting linear and non-linear nGAR models to artificial sample

	Linear nGAR	**Non-linear nGAR**
$[\underline{\hat{\alpha}}]$	[0.42 , 0.20, 0.01]	[0.033, 0.202]
$\hat{F}(y, p_1, p_2)$	Weibull(1.8 , 2.7)	Weibull(2.8, 3.9)
mean	6.327	6.222
variance	3.1857	2.7296
$\rho_Z(1)$, $\rho_Z(2)$, $\rho_Z(3)$	[0.59, 0.49, 0.31]	[0.55, 0.45, 0.28]
MSR	0.0365	0.0089
KSS	0.0486	0.0261
KSS rejected?	yes	no

4.6.4 Fitting Models to Real Measurements, the Old Faithful Geyser

In this example, we fit standard distributions and linear nGAR models to 108 observations taken from West and Ogden (1998). The observations are the duration (in minutes) for eruptions of the Old Faithful Geyser in Yellowstone National Park. The observations are correlated with correlation coefficients $\underline{\rho}$ of [-0.60, 0.57, -0.46]. Lag $p = 3$

For the fitting procedure, a linear function ψ and an nGAR order $p = 5$ are considered. The nGAR coefficients $\underline{\alpha}$ are estimated using the Burg method. Noticing that the nGAR coefficients of lag 4 and higher are small, we suggest that an order $p = 3$ is suitable. The $\hat{y}_t$ are fitted using Arena Input Analyzer to the common known distributions and the best ones which fit are the triangular and the Beta distributions. For the purpose of comparison, the observed sample are also fitted directly to the standard distributions, neglecting the fact that the observations are correlated.

Fitting the observations to a standard distribution or to an nGAR model several times gives always the same results. For example the best fitting standard distribution is always the triangular distribution and the parameters are always (1.34, 5, 4.1). However, when we generate a sample from this distribution to compare it with the original sample, we get every time (slightly) different results for the comparison. This is due to that we have a small number of observations. A small sample size of a distribution might not capture its statistical properties accurately.

In this case, the statistical goodness-of-fit tests might deliver (slightly) different results after each fit. To ensure that the test statistics lie in a

specific interval, we run 500 fittings and build 99% confidence intervals for the different statistics seperately. In table 4.9, which shows a summary of the statistical results of the fittings, only the lower bounds of the confidence interval are shown.

Table 4.8: Some moments of the observed sample (the old faithful geyser)

$[\underline{\hat{\rho}}]$	[-0.60, 0.57, -0.46]
mean	3.45
variance	1.0904
minimum	1.6700
maximum	4.9300

Table 4.9: Results summary of fitting a standard distribution and linear nGAR models to the observed sample

	Standard distr.	**Linear nGAR 1**	**Linear nGAR 2**
$[\ \underline{\hat{\rho}}]$	[-.02, -.02, -.01]	[-0.59 , 0.52, -0.44]	[-0.59 , 0.52, -0.45]
$F(y, \hat{p})$	Tri^* (1.34, 5, 4.1)	Tri (0, 4, 2.2)	Beta (4, 2.62, 2.67)
$\hat{\text{mean}}$	3.47	3.51	3.43
$\hat{\text{variance}}$	.61	1.13	1.08
$\hat{\text{minimum}}$	1.598	0.89	1.019
$\hat{\text{maximum}}$	4.85	6.50	6.23
MAR (process)	0.29	0.33	0.31
MSR (process)	0.14	0.17	0.15
KSS (process)	0.2	0.18	0.2
KS rejected?	(76%)	(42%)	(55%)

The models shown in table 4.9 are the ones which best fit the measurements. We see however that the different statistical tests give different answers for the question, which model is better. The standard distribution Tri^* for example delivers better estimation for the mean, the MAR and MSR statistics, and as a lower and upper bounded distribution, sampling from a triangular distribution delivers minimum and maximum random variates which estimate precisely the real minimum and maximum. However, the triangular distribution Tri^* completely fails with respect to modeling the variance and correlations. The KS test is also rejected in most of the cases (76%).

The two nGAR models deliver moderate goodness-of-fit test results. On one hand, the first nGAR model which consider the triangular distribution as the one to fit the IID random variables Y_t gives the best KS statistic and very good estimation of the correlations. On the other hand, the second nGAR model with the beta distribution estimates the mean, variance, and correlations quite accurately, and it delivers relatively moderate MAR, MSR and KSS statistics.

We also generate an nGAR process using some kind of empirical distribution for the independent random variables Y_i. We see in table 4.10 that using an empirical distribution gives better test statistics than the other models.

Table 4.10: Results summary of fitting a standard distribution and linear nGAR models to the observed sample with empirical distribution for the Y_i

	Empirical distribution
$[\ \underline{\hat{\rho}}]$	[-0.56, 0.52, -0.41]
$F(y, \hat{p})$	empirical
$\hat{\text{mean}}$	3.46
$\hat{\text{variance}}$	1.0374
$\hat{\text{minimum}}$	1.0482
$\hat{\text{maximum}}$	5.40
MAR (process)	0.31
MSR (process)	0.14
KSS (process)	0.17
KS rejected?	(43%)

Figures 4.14, 4.15, 4.16, 4.17, and 4.18 show the plots (Z_t, Z_{t+1}) and (Z_t, Z_{t+2}) from the observed sample, samples from the fitted triangular random variable, and samples from three fitted nGAR models respectively. We notice that the correlations of the fitted nGAR models are more similar to the correlations of the observed sample than those from the IID triangular random variable. However, the statistical tests might not be satisfactory for many modelers. The Kolmogorov-Smirnov test is rejected in 43% of the times. Moreover, the scatter plots in Figure 4.14 shows regions which do not contain any observations. The fitted processes do not produce such regions. Such models can best be fitted to models utilizing an empirical copula to capture the dependencies, instead of autoregressive models with linear or non-linear dependencies among its random variables. Models utilizing empirical copulas are introduced in chapter 7.

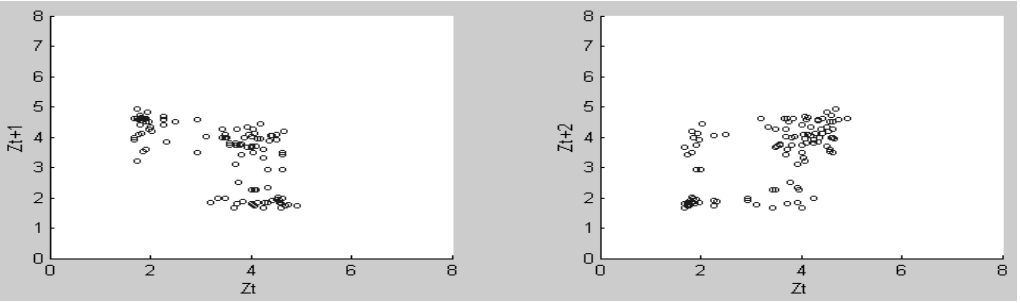

Figure 4.14: Plots from the observed sample.

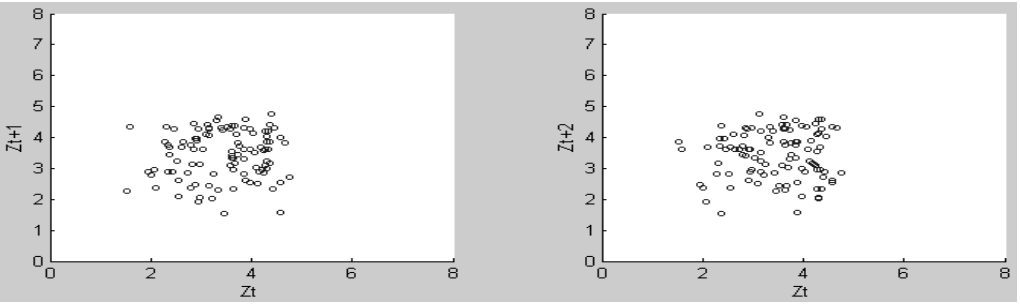

Figure 4.15: Plots from from the fitted IID random variable (triangular) without correlations

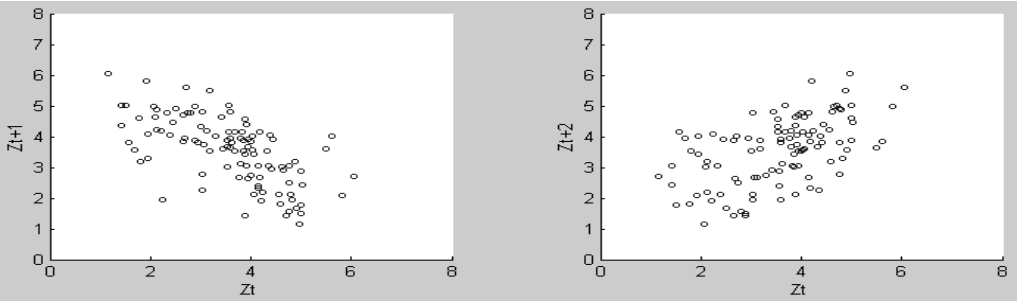

Figure 4.16: Plots from the fitted linear nGAR model (triangular Y_i).

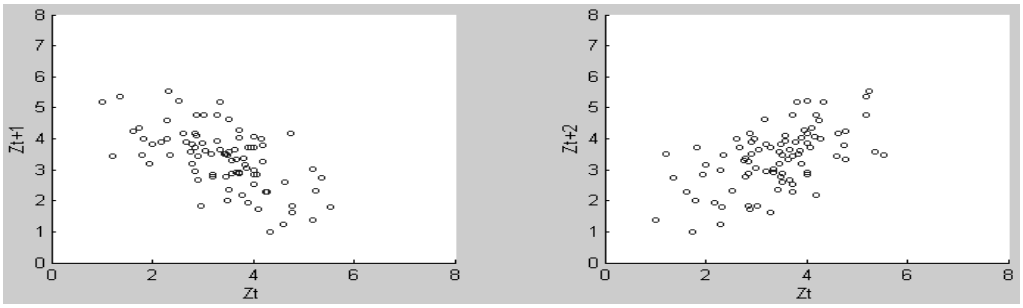

Figure 4.17: Plots from the fitted linear nGAR model (Beta Y_i).

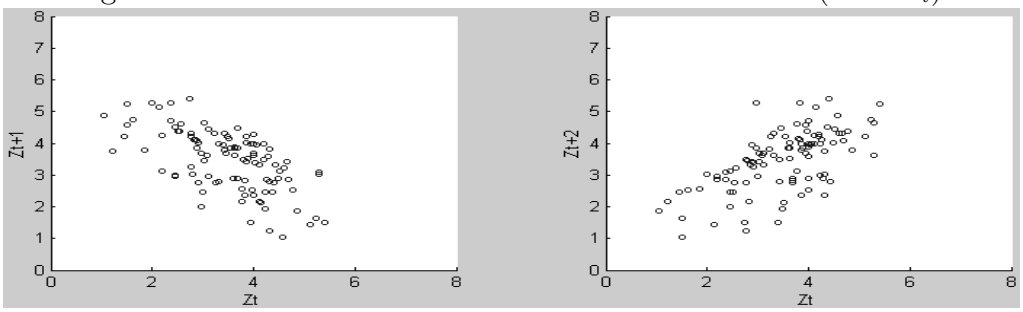

Figure 4.18: Plots from the fitted linear nGAR model (empirical Y_i).

4.6.5 Fitting Models to Real Measurements, Packet arrivals at Internet Server

In this example, we fit standard distributions and a linear nGAR model to 9000 measurements taken from of Aveiro (2006). The measurements describe the interarrival times of packets arriving at an Internet server. The measurements are correlated. The mean (in milliseconds), variance, and correlation coefficients are given in table 4.11.

Table 4.11: The moments of the observed sample (Interarrival time of packets at an Internet Server)

$[\underline{\hat{\rho}}]$	[-0.156, 0.08]
mean	0.0011
variance	1.9e-06
minimum	1.0e-006
maximum	0.0185

For the fitting procedure, a linear function ψ and an nGAR order $p = 3$ are considered. The nGAR coefficients $\underline{\alpha}$ are estimated using the Burg method.

Next, the $\hat{y}_t$ are estimated. Noticing that α_3 is small, we suggest that an order $p = 2$ is suitable. The independence of the $\hat{y}_t$ can also be verified using the independence method. For the purpose of comparison, the real measurements are fitted directly to the standard distributions, neglecting the fact that the measurements are correlated. The statistical results of the two fittings are summarized in table 4.12.

Table 4.12: Results summary of fitting a standard distribution and linear nGAR models to observed sample

	Standard distr.	**Linear nGAR**
$[\ \underline{\hat{\rho}}]$	[0.02, -.014]	[-0.16 , 0.075]
$F(y, \hat{p})$	$Beta^*$ (0.389, 7.36767, 10)	Beta (2.16, 17.7, 20)
$\hat{\text{mean}}$	1.0164	2.0297
$\hat{\text{minimum}}$	6.6888e-014	-0.8322
$\hat{\text{maximum}}$	9.8875	12.3214
$\hat{\text{variance}}$	2.3562	1.7943
MAR (process)	0.3225	0.9013
MSR (process)	0.2534	0.9146
KSS (process)	0.2879	0.4360
KS rejected?	yes	yes

$\underline{\hat{\alpha}}$ are the fitted nGAR coefficients. $F(y, \hat{p})$ are the fitted distributions and parameters. The best fitted distribution to the real measurements is $Beta^*(.282, 2.61, 10)$, whereas the best fitted distribution to the estimated independent data $\hat{y}_t$ is Beta (2.16, 17.7, 20). The MAR, MSR and the KS test statistics of the fitted standard distribution are better than those of the nGAR model. Although the (correlations of) the nGAR model fits the real measurements better than the standard distribution, we see that the other statistics like the mean, variance, ans minimum differs very much from those of the real measurements. This indicates that the nGAR approach fails in this case to generate a process with the required statistics.

The reason behind the failure is the structure of the real sample. If we look at the scatter plots of this sample in figure 4.19 we see that many points lie at or near the two axes and a few far from them. An nGAR/AR models tends to have a plot which looks like that in Fig. 4.20.

Another problem with the nGAR approach in this case is that it generates values of interarrival times which are less that 0. These values are unacceptable values for this physical model. To avoid this problem and to

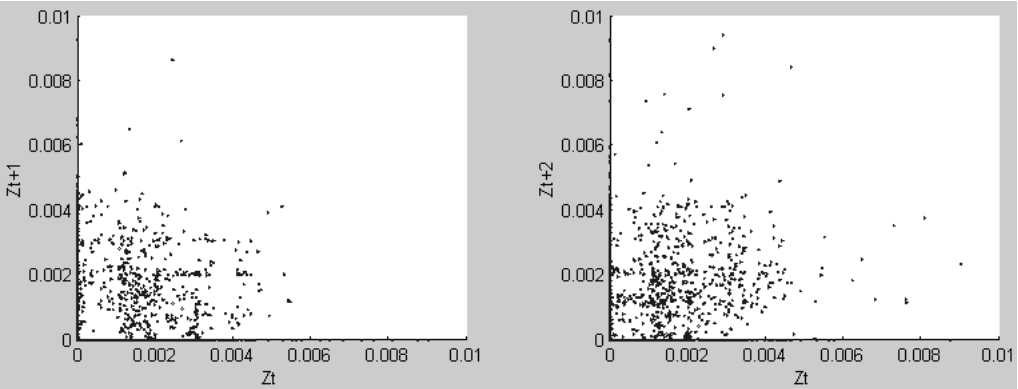

Figure 4.19: Plots from the observed sample.

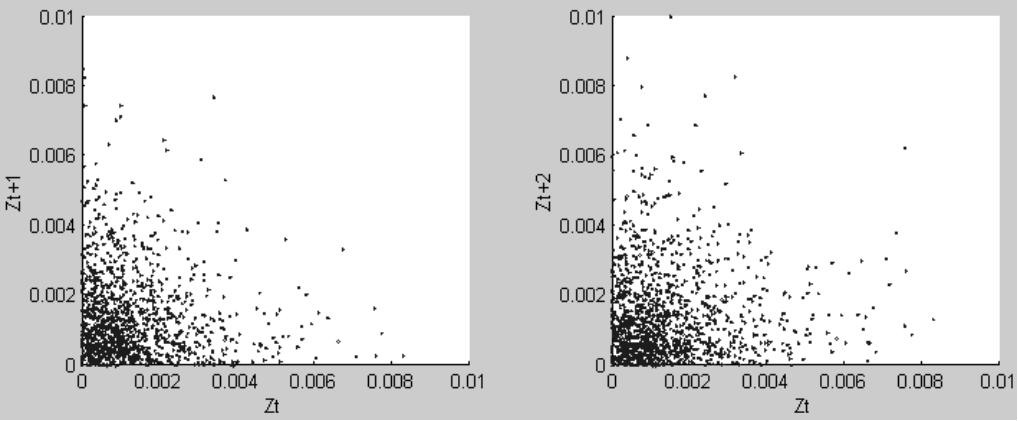

Figure 4.20: Plots from the generated nGAR model.

try to enhance the fitting procedure we took the logarithm of the interarrival times before we perform the fitting and at the end took the exponential function of the resulted model. In other words, we first perform the step $Z_i\_Logged = log(Z_i)$, where log is the logarithm function and Z_i are the measured interarrival times, then we fit Z_i_Logged to an nGAR model. After that we generate a sample from the fitted nGAR model, say Z_i_Fitted, and perform the step $Z_i\_Fitted = exp(Z_i\_Fitted)$. At the end we perform some goodness-of-fit tests to see how well the fitted sample Z_i_Fitted fit the original sample Z_i. The results of the tests are shown in table 4.13

Table 4.13: Results summary of fitting a standard distribution and linear nGAR models to the observed sample

	Linear nGAR
$[\ \underline{\hat{\rho}}]$	[-0.061, 0.0079]
$\hat{\text{mean}}$	1.78
$\hat{\text{minimum}}$	5.7020e-005
$\hat{\text{maximum}}$	209
$\hat{\text{variance}}$	36
MAR	1.09
MSR	28.43
KSS	0.24
KS rejected?	yes

We see that the results of the goodness of fit tests and other statistics are unacceptable. The mean and variance differ from the true ones considerably, and the test statistics are bad. This ensures us that the nGAR model is not a suitable one for this sample and another model must be considered. An example of another model which can be used in such cases is the empirical copulas approach described in chapter 7.

Chapter 5

The Extended Yule-Walker Method

The Yule-Walker equations were developed to estimate the autoregressive coefficients of linear autoregressive models as described in section 3.2. We extend this approach to be able to estimate the autoregressive coefficients of non-linear nGAR models like that shown in (5.1).

Assume that the following d-variate non-linear nGAR model must be fitted to a data sample:

$$Z_{i,t} = \sum_{i,j,k} \alpha_{i,j,k} Z_{i,t-j}^{k} + Y_{i,t}, \tag{5.1}$$

where $i = 1, 2, ..., d$, $j = 1, 2, ..., p$, $k = k_1, k_2, ..., k_K \in \mathbb{R}$, $t = p{+}1, p{+}2, ..., \infty$, and p is the maximum autoregressive lag. We have seen in chapter 4 that the non-Gaussian autoregressive approach utilizing the independence method (see section 4.4) can be used to estimate the autoregressive coefficients $\underline{\alpha}$ and also the distribution of the IID random variables $\mathbf{Y_t}$. However, the nGAR approach utilizing the independence method is in general time-consuming. An analytical method, if it exists, would be faster and more accurate, and therefore preferable.

We have extended the Yule-Walker method so as to be able to analytically estimate the autoregressive coefficients for weakly stationary non-linear nGAR processes. We describe in the following sections how we derived equations which can be solved to estimate the nGAR coefficients for several specific non-linear nGAR models.

5.1 Non-Linear Univariate nGAR Process

We derive formulas for the purpose of estimating the autoregressive coefficients $\underline{\alpha}$ of the non-linear univariate nGAR process shown in (5.2):

$$Z_t = \sum_{j,k} \alpha_{j,k} {Z_{t-j}}^k + Y_t, \tag{5.2}$$

where $j = 1, 2, ..., p$, $t = p+1, p+2, ..., \infty$, p is the maximum autoregressive lag and $k = k_1, k_2, ..., k_K \in \mathbb{R}$. Assuming that this process is weakly stationary, that some higher moments exist and are stationary, which we specify later, and that μ_Z is the mean of the random variable Z_t, we can write after subtracting the mean μ_Z from both sides of (5.2) and multiplying with $(Z_{t-m} - \mu_Z)$, $m = 1, 2, ...p$:

$$(Z_t - \mu_Z)(Z_{t-m} - \mu_Z) =$$
$$\sum_{j,k} \alpha_{j,k} {Z_{t-j}}^k Z_{t-m} - \mu_Z \sum_{j,k} \alpha_{j,k} {Z_{t-j}}^k - \mu_Z(Z_{t-m} - \mu_Z) + Y_t(Z_{t-m} - \mu_Z).$$

Taking the expectation results in:

$$\mathrm{Cov}(Z_t, Z_{t-m}) =$$
$$\sum_{j,k} \alpha_{j,k} E[{Z_{t-j}}^k Z_{t-m}] - \mu_Z \sum_{j,k} \alpha_{j,k} E[{Z_{t-j}}^k] - \mu_Z E[Z_{t-m} - \mu_Z] + E[Y_t(Z_{t-m} - \mu_Z)].$$

However, the term $E[Y_t(Z_{t-m} - \mu_Z)]$ is zero, because Y_t and Z_{t-m} are independent for $m = 1, 2, ...p$ and therefore $E[(Y_t)(Z_{t-m} - \mu_Z)] = E[Y_t]E[Z_{t-m} - \mu_Z]$. But $E[Z_{t-m} - \mu_Z]$ is zero as $E[Z_{t-m}] = E[Z_t] = \mu_Z$. Moreover, the term $\mu_Z E[Z_{t-m} - \mu_Z]$ is also zero as $E[Z_{t-m} - \mu_Z]$ is zero. Thus, the equation above can be rewritten as:

$$\mathrm{Cov}(Z_t, Z_{t-m}) = \sum_{j,k} \alpha_{j,k} E[{Z_{t-j}}^k Z_{t-m}] - \mu_Z \sum_{j,k} \alpha_{j,k} E[{Z_{t-j}}^k]. \tag{5.3}$$

This means that (5.3) results in p linear equations for the unknown autoregressive coefficients $\alpha_{j,k}$. Some of the $\alpha_{j,k}$ may be zero, depending on the model. Hence the number of unknowns $\alpha_{j,k}$ is less or equal to pK. When this number is greater than p, we need additional equations. They can be obtained as follows: multiply (5.2) with $Z_{t-m}^{k'}$, where $0 < m \leq p$, and $k' > 1$, and take the expectations. The linear equations can be solved if the moments of the process Z_t, $E[Z_t]$, $E[{Z_t}^k]$, ..., and the covariances exist and are known. Those can be estimated from the available sample and the covariances.

We show now in detail how the above procedure can be applied on the following specific nGAR model to derive the extended Yule-Walker equations:

$$Z_t = \alpha_1 {Z_{t-1}}^2 + \alpha_2 Z_{t-2} + Y_t. \tag{5.4}$$

For m=1 (see equation 5.3) this yields

$$\mathrm{Cov}(Z_t, Z_{t-1}) = \alpha_1 E[Z^3] + \alpha_2 E[Z_{t-2}Z_{t-1}] - \alpha_1 \mu_Z E[Z^2] - \alpha_2 {\mu_Z}^2.$$

or

$$\mathrm{Cov}(Z_t, Z_{t-1}) = \alpha_1 (E[Z^3] - \mu_Z E[Z^2]) + \alpha_2 (E[Z_{t-2}Z_{t-1}] - {\mu_Z}^2). \quad (5.5)$$

Considering equation (5.4) again, for $m = 2$:

$$\mathrm{Cov}(Z_t, Z_{t-2}) = \alpha_1 (E[{Z_{t-1}}^2 Z_{t-2}] - \mu_Z E[Z^2]) + \alpha_2 (E[Z^2] - \mu_Z^2). \quad (5.6)$$

The system of two equations (5.5) and (5.6) with two variables of autoregressive coefficients α_1 and α_2 can now be solved as all other terms in equations (5.5) and (5.6) are constants that can be estimated from the available sample directly.

Similar steps can be followed to estimate the coefficients of other nGAR processes like

$$Z_t = \alpha_1 {Z_{t-1}}^2 + Y_t,$$

to get the equation

$$\alpha_1 = \mathrm{Cov}(Z_t, Z_{t-1}) / (E[Z^3] - E[Z^2]\mu_Z).$$

5.2 Non-Linear Multivariate nGAR Process

We derive formulas for the purpose of estimating the autoregressive coefficients $\underline{\alpha}$ of the non-linear multivariate nGAR process shown in (5.7):

$$Z_{i,t} = \sum_{i,j,k} \alpha_{i,j,k} {Z_{i,t-j}}^k + Y_{i,t}, \quad (5.7)$$

where $i = 1, 2, ..., d$, d is the number of different random variables Z_t at each point of time t, $j = 1, 2, ..., p$, $t = p+1, p+2, ..., \infty$, p is the lag, and $k = k_1, k_2, ..k_K \in \mathbb{R}$. Assuming that this multivariate process is weakly stationary, that all needed moments exist and are stationary, and that μ_{Z_i} is the mean of the random variable $Z_{i,t}$ over all t, we can write after subtracting the mean μ_{Z_i} from both sides of (5.7) and multiplying with $(Z_{i,t-m} - \mu_{Z_i})$ for all $i = 1, 2, ..., d$ and $m = 1, 2, ...p$:

$$(Z_{i,t} - \mu_{Z_i})(Z_{i,t-m} - \mu_{Z_i}) =$$
$$\sum_{i,j,k} \alpha_{i,j,k} {Z_{i,t-j}}^k Z_{i,t-m} - \mu_{Z_i} \sum_{i,j,k} \alpha_{i,j,k} {Z_{i,t-j}}^k - \mu_{Z_i}(Z_{i,t-m} - \mu_{Z_i}) + Y_{i,t}(Z_{i,t-m} - \mu_{Z_i}).$$

Taking the expectation results in:

$$\mathrm{Cov}(Z_{i,t}, Z_{i,t-m}) = \sum_{i,j,k} \alpha_{i,j,k} E[{Z_{i,t-j}}^k Z_{i,t-m}] - \mu_{Z_i} \sum_{i,j,k} \alpha_{i,j,k} E[{Z_{i,t-j}}^k] - \mu_Z E[Z_{i,t-m} - \mu_{Z_i}] + E[(Y_{i,t})(Z_{i,t-m} - \mu_{Z_i})].$$

However, the term $E[(Y_{i,t})(Z_{i,t-m} - \mu_{Z_i})]$ is zero, because $Y_{i,t}$ and $Z_{i,t-m}$ are independent and therefore $E[(Y_{i,t})(Z_{i,t-m} - \mu_{Z_i})] = E[Y_{i,t}]E[Z_{i,t-m} - \mu_{Z_i}]$. But $E[Z_{i,t-m} - \mu_{Z_i}]$ is zero as $E[Z_{i,t-m}] = E[Z_{i,t}] = \mu_{Z_i}$. Moreover, the term $\mu_{Z_i} E[Z_{i,t-m} - \mu_{Z_i}]$ is also zero as $E[Z_{i,t-m} - \mu_{Z_i}]$ is zero. Thus, the equation above can be rewritten as:

$$\mathrm{Cov}(Z_{i,t}, Z_{i,t-m}) = \sum_{i,j,k} \alpha_{i,j,k} E[{Z_{i,t-j}}^k Z_{i,t-m}] - \mu_{Z_i} \sum_{i,j,k} \alpha_{i,j,k} E[{Z_{i,t-j}}^k]. \quad (5.8)$$

Equation (5.8) is a system of pd linear equations for the unknown autoregressive coefficients $\alpha_{i,j,k}$. Here, the number of unknown $\alpha_{i,j,k}$ is less or equal to pdK. Additional equations can be obtained by multiplying (5.7) with $Z_{i,t-m}^{k'}$ The linear equations can be solved if the moments of the process $Z_{i,t}$, $E[Z_{i,t}]$, $E[{Z_{i,t}}^k]$, ..., and the covariances, are known. Those can be estimated from the available sample. Similarly, cross-dependencies can be considered.

We show now this. The above procedure can be applied on the following specific nGAR model to derive the extended Yule-Walker equations:

$$Z_{1,t} = \alpha_{1,1} {Z_{2,t-1}}^2 + Y_{1,t}, \quad t = 1, 2, ... \quad (5.9)$$

$$Z_{2,t} = \alpha_{2,1} {Z_{1,t-1}}^2 + Y_{2,t}, \quad t = 1, 2, ... \quad (5.10)$$

Assuming that the random variables in equations (5.9) and (5.10) are weakly stationary, that all needed moments exist and are stationary, and that μ_{Z_1} symbolizes the mean of the first random variable, $Z_{1,t}$, and that μ_{Z_2} symbolizes the mean of the second random variable, $Z_{2,t}$, one can write

$$Z_{1,t} - \mu_{Z_1} = \alpha_{1,1} {Z_{2,t-1}}^2 - \mu_{Z_1} + Y_{1,t}, \quad t = 1, 2, ... \quad (5.11)$$

$$Z_{2,t} - \mu_{Z_2} = \alpha_{2,1} {Z_{1,t-1}}^2 - \mu_{Z_2} + Y_{1,t}, \quad t = 1, 2, ... \quad (5.12)$$

Multiplying (5.11) with $(Z_{2,t-1} - \mu_{Z_2})$ results in

$$(Z_{1,t} - \mu_{Z_1})(Z_{2,t-1} - \mu_{Z_2}) = \alpha_{1,1} {Z_{2,t-1}}^3 - \mu_{Z_1}(Z_{2,t-1} - \mu_{Z_2}) - Y_{1,t}(Z_{2,t-1} - \mu_{Z_2}) - \alpha_{1,1}\mu_{Z_2} Z_{2,t-1}^2.$$

Talking the expectation

$$\mathrm{Cov}(Z_{1,t}, Z_{2,t-1}) = \alpha_{1,1} E[{Z_{2,t-1}}^3] - \mu_{Z_1}(E[Z_{2,t-1}] - \mu_{Z_2}) - E[Y_{1,t}(Z_{2,t-1} - \mu_{Z_2})] - \alpha_{1,1}\mu_{Z_2} E[Z_{2,t-1}^2],$$

where Cov is the cross covariance function between the two different random variables $Z_{1,t}$ and $Z_{2,t-1}$.

Considering the property of the expectation operator for weak stationary processes, the property of the expectation operator for independent random variables, and rearranging the above equation, results in

$$\mathrm{Cov}(Z_{1,t}, Z_{2,t-1}) = \alpha_{1,1} E[{Z_{2,t-1}}^3] - \alpha_{1,1} \mu_{Z_2} E[{Z_{2,t}}^2].$$

and

$$\alpha_{1,1} = \mathrm{Cov}(Z_{1,t}, Z_{2,t-1}) / (E[{Z_{2,t-1}}^3] - \mu_{Z_2} E[{Z_{2,t}}^2]).$$

Multiplying (5.12) with $(Z_{1,t-1} - \mu_{Z_1})$ and performing similar operations to the above ones we get

$$\alpha_{2,1} = \mathrm{Cov}(Z_{2,t}, Z_{1,t-1}) / (E[{Z_{1,t-1}}^3] - \mu_{Z_1} E[{Z_1}^2]).$$

By solving these two equations with estimated moments we get the needed values of the autoregressive coefficients $\alpha_{1,1}$ and $\alpha_{2,1}$.

5.3 Examples

5.3.1 Example 1: Non-Linear Univariate nGAR Process

The extended Yule-Walker method is used to estimate the autoregressive coefficients of a non-linear process in the random variable which is generated artificially. The artificial process is the following:

$$Z_t = \alpha_1 {Z_{t-1}}^2 + \alpha_2 Z_{t-2} + Y_t + shift, \quad t = 1, 2, ... \tag{5.13}$$

where $\alpha_1 = 0.03$, $\alpha_2 = 0.1$ and Y_t are Weibull distributed IID random variables with the parameters $p_1 = 6$ (shape parameter) and $p_2 = 2$ (scale parameter). $shift$ is a constant equal to 1.5.

The estimated nGAR coefficients fluctuate because of the stochastic behavior of the artificial process. To ensure that the estimated results are valid, we generate 100 samples from (5.13), each with length of 500, and build confidence intervals for the estimated nGAR coefficients α_1 and α_2 with confidence level of 95%. The confidence intervals of the estimated nGAR coefficients are shown in table 5.1.

We use the process with the estimated parameters to generate a sample and we compare this sample with the artificial sample in table 5.2.

Table 5.1: The estimated nGAR coefficients of the non-linear nGAR process using the extended Yule-Walker method

	Confidence Interval	**Confidence Level**
$[\ \hat{\underline{\alpha}}_1]$	[0.0288, 0.0305]	95%
$[\ \hat{\underline{\alpha}}_2]$	[0.0937, 0.1060]	95%

The statistical moments of the artificial sample and sample generated from the fitted process are very close. The statistical goodness-of-fit tests deliver also good results. We also plot samples of the original and estimated process and we can see from the plots how the correlations, resulting from the non-linear ones, are similar in both samples.

Table 5.2: Results summary of the original and generated sample

	Artificial process	**Fitted process**
$[\ \hat{\underline{\rho}}]$	[0.48, 0.25, 0.1328, 0.0693]	[0.5, 0.26, 0.15, 0.076]
$F(y, \hat{p_1}, \hat{p_2})$	Weibull (2.7, 4)	Weibull (2.66, 3.99)
$\hat{\mu}$	4.8267	4.8201
$\hat{\text{minimum}}$	0.8670	0.5109
$\hat{\text{maximum}}$	10.4322	12.4811
$\hat{\text{Var}}$	2.6994	2.7455
MAR (process)	-	0.0446
MSR (process)	-	0.0055
KSS (process)	-	0.0212
KS rejected?	-	no

5.3.2 Example 2: Another Non-Linear Univariate nGAR Process

The extended Yule-Walker method is used to estimate the autoregressive coefficients of a non-linear process in the random variable which is generated artificially. The artificial process is the following:

$$Z_t = \alpha {Z_{t-1}}^2 + Y_t, \quad t = 1, 2, ... \tag{5.14}$$

where $\alpha_1 = .05$, and Y_t are Weibull distributed IID random variables with the parameters $p_1 = 2.7$ (shape paramter) and $p_2 = 4$ (scale paramter).

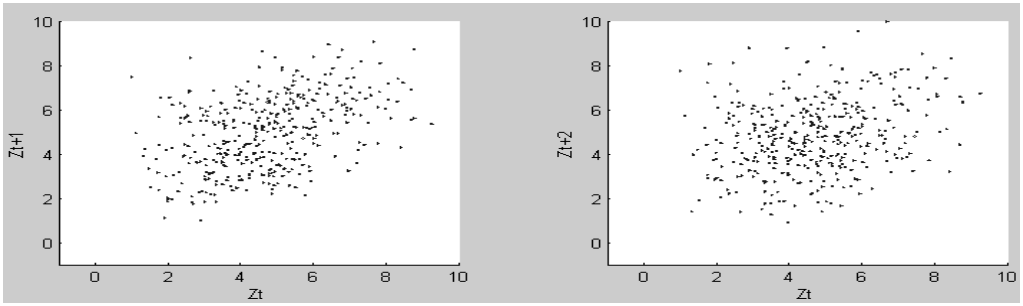

Figure 5.1: Plots from the artificial non-linear process

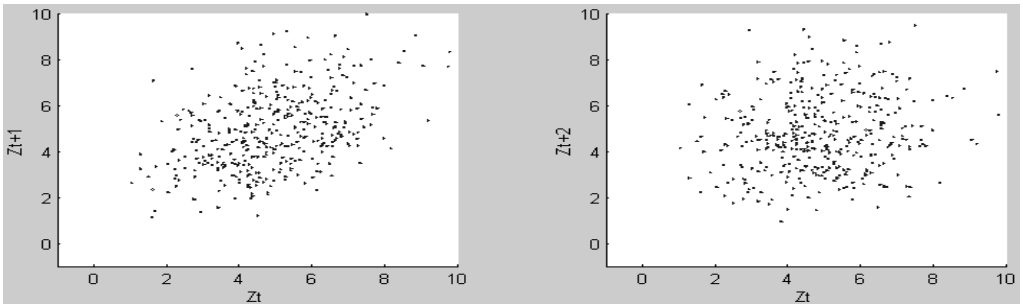

Figure 5.2: Plots from the derived non-linear nGAR process

The estimated nGAR coefficients fluctuate because of the stochastic behavior of the artificial process. To ensure that the estimated results are valid, we generate 100 samples from (5.14), each with length of 500, and build confidence intervals for the estimated nGAR coefficient $\hat{\alpha}$ with confidence level of 95%. The confidence intervals of the estimated nGAR coefficients are shown in table 5.3.

Table 5.3: The estimated nGAR coefficient of the non-linear nGAR process using the extended Yule-Walker method

	Confidence Interval	**Confidence Level**
$[\ \underline{\hat{\alpha}}]$	[0.0497, 0.0512]	95%

5.3.3 Example 3: Multivariate Non-Linear nGAR Process

The extended Yule-Walker method is used to estimate the autoregressive coefficients of a multivariate non-linear process which is generated artificially. The artificial process is the following:

$$A(t) = \alpha B_{t-1}^2 + Y_{1,t}, \quad t = 1, 2, ... \tag{5.15}$$

$$B(t) = \beta A_{t-1}^2 + Y_{2,t}, \quad t = 1, 2, ... \tag{5.16}$$

where $\alpha = 0.04$, $\beta = 0.03$, $Y_{1,t}$ are gamma distributed IID random variables with the parameters $p_1 = 5$ (shape parameter), $p_2 = 0.7$ (scale parameter), and $Y_{2,t}$ are exponentially distributed IID random variables with the parameter $\mu = 2$ (scale parameter).

We generate 100 samples from (5.15) and (5.16), each with length of 500, and build confidence intervals for the estimated nGAR coefficients $\hat{\alpha}$ and $\hat{\beta}$ with confidence level of 95%. The confidence intervals of the estimated nGAR coefficients are shown in table 5.4.

Table 5.4: The estimated nGAR coefficients of the bi-variate non-linear nGAR process using the extended Yule-Walker method

	Confidence Interval	**Confidence Level**
$[\ \underline{\hat{\alpha}}]$	[0.0393, 0.0403]	95%
$[\ \underline{\hat{\beta}}]$	[0.0290, 0.0305]	95%

We use the above fitted process to generate a bi-variate sample and we compare this sample with the artificial one in table 5.5.

The statistical moments of the artificial sample and the sample generated from the fitted process are very close. The statistical goodness-of-fit tests deliver also good results. We also plot the samples and we can see from the plots how the correlations are similar in both of them.

Table 5.5: Results summary of the artificial and fitted processes

	Artificial process	**Fitted process**
$[\hat{\underline{\rho}}(A_t, B_{t-1}), \hat{\underline{\rho}}(B_t, A_{t-1})]$	[0.27, 0.44]	[0.25, 0.38]
$F_Y(\hat{p})$	[gamma(2.7, 4), exp(2)]	[gamma(2.72, 3.89), exp(2.02)]
$\hat{\text{mean}}$	[3.89, 2.55]	[3.8308, 2.4844]
$\hat{\text{minimum}}$	[0.57, 0.06]	[0.5674 , 0.0735]
$\hat{\text{maximum}}$	[19.08, 19.87]	[13.9406, 15.3687]
$\hat{\text{variance}}$	[3.21, 4.42]	[2.8690, 3.8569]
MAR	-	[0.0628, 0.0745]
LSS	-	[0.0278, 0.0384]
KSS	-	[0.0228, 0.0165]
KS rejected?	-	[no, no]

$\hat{\underline{\rho}}$ is the cross correlations among the two sub processes A and B.

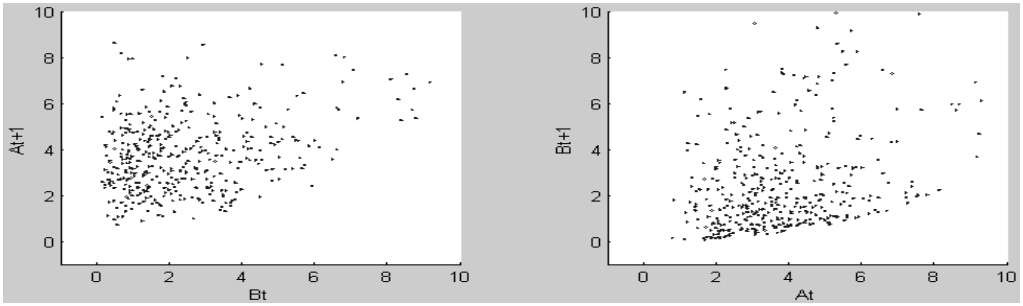

Figure 5.3: Sample from the bi-variate original process

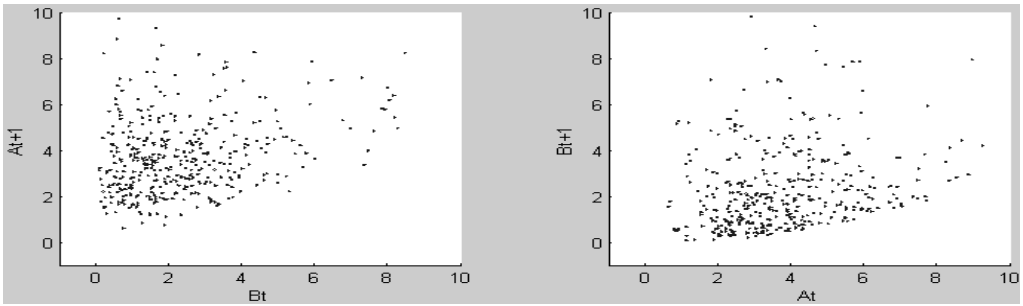

Figure 5.4: Sample from the fitted bi-variate non-linear nGAR process.

Chapter 6

The Probabilistic Transition Matrix Versus the Batch Markovian arrival Process

The Batch Markovian Arrival Process (BMAP) approach, overviewed in section 3.4 has several advantages when applied to model the Internet traffic, which are described thoroughly in Klemm et al. (2002) and Klemm et al. (2003), therefore, we will concentrate here on the drawbacks of using this approach.

As described in Klemm et al. (2002) and Klemm et al. (2003), the BMAP can generate only a prespecified number of packet lengths. Therefore, one must observe heuristically how many packet lengths is best. In other words, for the example described in Klemm et al. (2002) and Klemm et al. (2003), the result that the software delivers will be worse when one uses 2 or 4 packet lengths instead of 3 packet lengths used actually. Taking into consideration that one BMAP fitting procedure takes relatively long time (for this special case 1 hour), one sees that trying several possibilities of packets lengths is relatively time consuming.

The tool which utilizes the BMAP to model IP traffic is called by Klemm et al. (2002) IP2BMAP. We call the approach or the tool as the BMAP approach or the BMAP tool for simplicity. To have more information about this tool, we analyze the results by running it and compare the sample generated by the tool with the real one. For this purpose, we consider a 3 state Discrete Time Markov Chain (DTMC), each state represents a different packet length. Like the BMAP, we divide the packet lengths into small (S), medium (M), and big (B). Please see figure 6.1. A packet is small if it is smaller than 94 Byte, and medium if it is between 94 and 575 Byte, and big if it is bigger than 575 Byte.

The transition probabilities between the different states can be estimated using a simple counting procedure running on the real data. If one counts how many small (S) packets come after medium (M) packets and how many big packets come after big packets and so on, one can get absolute and conditional probabilities that the transitions take. The conditional transition matrix which results from this counting procedure is shown in figure 6.2. For example, T-S→S has the value 0.7185. This means that provided the current packet is small, the probability that the next packet is also small is 0.7185.

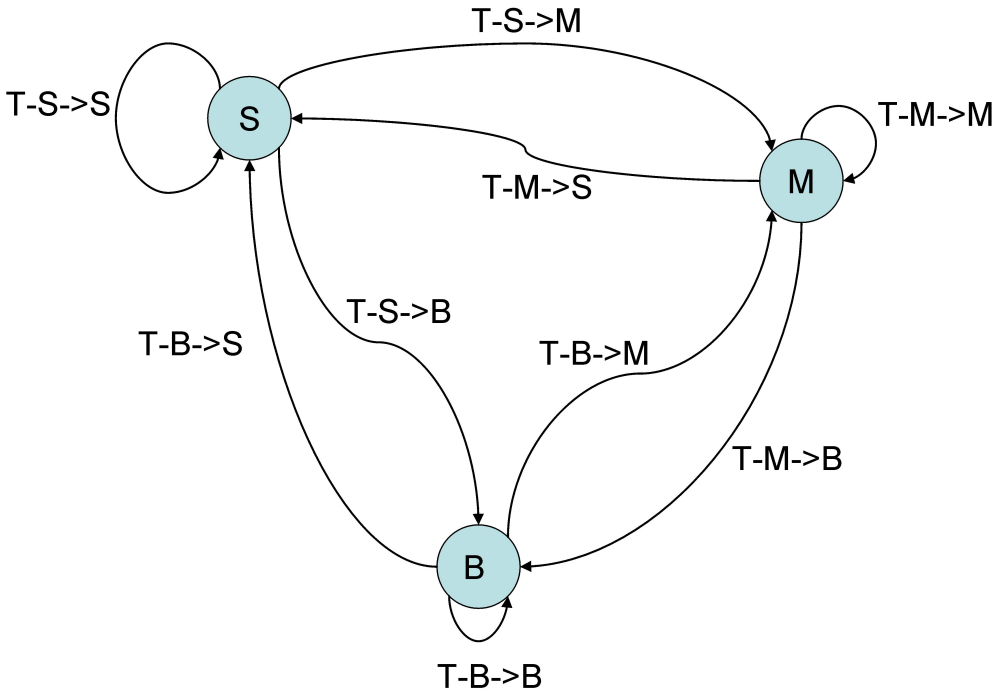

Figure 6.1: A DTMC representing the packet lengths of a measured trace at an Internet server

T-S→S	T-S→M	T-S→B	=	0.7185	0.2625	0.0189
T-M→S	T-M→M	T-M→B		0.3823	0.6103	0.0074
T-B→S	T-B→M	T-B→B		0.4712	0.1976	0.3312

Figure 6.2: The probabilities of the transitions in figure 6.1 generated by the real measurements (T-real)

To analyze the data generated by the BMAP tool, we calculate the transition probabilities between the different states of the BMAP model to get the transition matrix shown in figure 6.3.

T-S→S	T-S→M	T-S→B		0.7160	0.2667	0.0173
T-M→S	T-M→M	T-M→B	=	0.3835	0.5908	0.0257
T-B→S	T-B→M	T-B→B		0.5082	0.4706	0.0212

Figure 6.3: The probabilities of the transitions in figure 6.1 generated by the BMAP tool (T-BMAP)

The transition matrix of the real data (T-real) and the transition matrix generated from the BMAP model (T-BMAP) are shown in figure 6.2 and figure 6.3, respectively. Please notice the big difference in the transition probability T-B→B between the real data and the data generated by the BMAP tool. The true transition probability T-B→B=0.33, while the BMAP transition probability T-B→B=0.02.

The difference between the transition probability of the BMAP process and the real process means that the number of "big" packets coming after each other differs between the BMAP process and the real one. This might affect the performance measurements. The processing time of a set of message with length M at one period and then processing a set of messages with length B at the next period might be grater than the time if the packets with different lengths are mixed.

We have built a simple model depending on the DTMC in figure 6.1. We call this model the probability model. The model starts at one state and transits to the next one depending on the transition probabilities in T-real. The resulting transition matrix of packet lengths (T-pro) is similar to T-real. However, one must also take the interarrival times into consideration. Therefore, we store at each transition not only the packet length but also the corresponding interarrival time. This way, one gets different traces of interarrival times, each corresponds to a specific transition. Each trace can then be fitted to a standard or empirical distribution which in turn can be used when generating this process again.

Below we compare the linear correlations between the packet lengths, and the linear correlations between the interarrival times for the real data, for the data generated by the BMAP tool, and for the data generated by the probability model. We also compare the linear cross correlations between the packet lengths and the interarrival times. The packet lengths are noted as L_i, and the interarrival times are noted as Z_i.

Apart from the inaccuratly estimated T-B→M by the BMAP tool, we see in the tables that the BMAP tool can estimate the (cross) correlations for the different lags accurately. The probability model can not catch the second correlation lag without being modified.

Table 6.1: The correlations between the different packets lengths (L_i)

	$\rho(L_i, L_{i-1})$	$\rho(L_i, L_{i-2})$	$\rho\ (L_i, L_{i-3})$
Real data	0.3049	0.0657	0.0174
Data from the BMAP model	0.2396	0.0682	0.0191
Data from the probability model	0.3089	0.0937	0.0294

Table 6.2: The correlations between the interarrival times of packets (Z_i)

	$\rho(Z_i, Z_{i-1})$	$\rho\ (Z_i, Z_{i-2})$	$\rho\ (Z_i, Z_{i-3})$
Real data	0.0413	0.0080	0.0178
Data from the BMAP model	0.0732	0.0138	0.0002
Data from the probability model	0.0090	-0.0004	-0.0007

Table 6.3: The correlations between the interarrival times of packets (Z_i) and the packet lengths (L_i)

	$\rho\ (Z_i, L_i)$	$\rho\ (Z_i, L_{i-1})$	$\rho\ (Z_i, L_{i-2})$
Real data	-0.1592	-0.0445	-0.0148
Data from the BMAP model	-0.1478	-0.0338	-0.0069
Data from the probability model	-0.1507	-0.0380	-0.0106

Table 6.4: The correlations between the packet lengths (L_i) and the interarrival times of packets (Z_i)

	$\rho\ (L_i, Z_i)$	$\rho\ (L_i, Z_{i-1})$	$\rho\ (L_i, Z_{i-2})$
Real data	-0.1592	-0.1062	0.0428
Data from the BMAP model	-0.1478	-0.1669	-0.0389
Data from the probability model	-0.1507	-0.0497	-0.0174

The inaccuracy in the estimated transition probability T-B→M of the BMAP tool does not disappear even when changing the number of packet lengths (4 instead of 3) and the number of states in the BMAP. To try to understand this problem, we run the BMAP tool for another dataset taken from of Aveiro (2006). The results of the analysis is interesting. The problem with the probability transitions vanishes. The real probability transitions and that generated by the BMAP tool are given in tables 6.5 and 6.6 respectively.

Table 6.5: The conditional transition probabilities of the real data

T-S→S=0.5576	T-S→M1=0.0717	T-S→M2=0.0515	T-S→B=0.3193
T-M1→S=0.4786	T-M1→M1=0.1029	T-M1→M2=0.0586	T-M1→B=0.3600
T-M2→S=0.6328	T-M2→M1=0.0631	T-M2→M2=0.0478	T-M2→B=0.2564
T-B→S=0.6179	T-B→M1=0.0605	T-B→M2=0.0429	T-B→B=0.2786

Table 6.6: The conditional transition probabilities of the data generated by the BMAP tool

T-S→S=0.5698	T-S→M1=0.0766	T-S→M2=0.0490	T-S→B=0.3047
T-M1→S=0.5525	T-M1→M1=0.0831	T-M1→M2=0.0470	T-M1→B=0.3175
T-M2→S=0.6085	T-M2→M1=0.0747	T-M2→M2=0.0454	T-M2→B=0.2714
T-B→S=0.5750	T-B→M1=0.0788	T-B→M2=0.0520	T-B→B=0.2942

To be able to use the BMAP tool for the other datasets we had to change some parameters in the tool, which made the run time for BMAP tool to do the analysis and generate the needed BMAP model much longer. Moreover, the run times needed to analyse the first dataset is longer than mentioned in the documentations of the BMAP tool.

We believe that the BMAP tool tends to generate transition probabilities from all nodes to one node which are of the same order (Similar). In 6.2 we see how the transition probabilities of the real data are extremely different from each other (T-B→B is 44 times bigger than T-M→B).

The inaccuracy of some of the estimated transition probabilities of the BMAP tool might make this tool less useful for some datasets. For example, those datasets which have extreme different transition probabilities from the different nodes to one node. In addition to this drawback that the BMAP software suffers from, we could not get the BMAP software to converge for the specific example provided with the tool. We could use only a non-converged result.

Chapter 7

Empirical Copulas

The pitfalls of the theoretical classes of copulas described in Mikosch (2005) and which we present briefly in section 3.5 encouraged us to use the class of the empirical copulas. Our work is published in Nassaj and Strelen (2006) and in Strelen and Nassaj (2007). The empirical copulas can capture any type of dependencies: Linear and non-linear dependencies, tail dependencies, and others. Copulas encompass the entire dependence structure of multivariate distributions, and not only the correlations. In this chapter, we describe the procedure of fitting a multivariate model which utilizes a kind of empirical copula to dependent random vectors or time series and how to generate such random vectors or time series out of the fitted model.

We introduce two new approaches for the purpose of modeling and generating dependent random vectors or time series. The two approaches use two kinds of empirical copulas. We call the first kind of empirical copulas as the *approximate empirical copula*. The second kind is called the *piecewise multi-linear empirical copula*.

Similar to the procedure to fit a model which utilizes a theoretical class of copulas, the procedure of fitting a model which utilizes an empirical copula follows two steps: The marginal distributions can be modeled as empirical or fitted standard distributions, and the empirical copula is estimated separately as a frequency distribution. For this, one does not need any knowledge about the type of dependencies lying among the data at hand. The fitted marginal distributions together with the fitted empirical copula comprise a fitted multivariate distribution $\mathcal{A}$, from which random vectors can be generated.

The Empirical Copula and the Approximate Empirical Copula

We overview first the definition of some kind of bivariate empirical copula from the literature. A higher dimensional empirical copula can be defined similar to the bivariate one in a straight-forward manner.

Let $\{(x_k, y_k)\}_{k=1}^{n}$ denote a sample of size n from a continuous bivariate distribution. The empirical copula is given by the frequency function C_n,

$$C_n(\frac{i}{n}, \frac{j}{n}) = \frac{\text{number of pairs } (x, y) \text{ in the sample such that } x \leq x_{(i)} \text{ and } y \leq y_{(j)}}{n},$$

where $x_{(i)}$ and $y_{(j)}$, $i, j = 1, 2, ..., n$, denote the order statistics from the sample (Nelsen, 1998), for example $x_{(1)}$ is the smallest x_k and $y_{(n)}$ is the biggest y_k.

The definition above indicates that the bivariate empirical copula is a mapping function from a bi-dimensional space $\mathcal{T}$ to a uni-dimensional space of the real values. This mapping requires a search procedure to assign to each point of interest in the $\mathcal{T}$ space a real value in the range [0,1]. The bivariate empirical copula can be stored in a data structure M. So if the sample size is n, the matrix is of size n^2, see figure 7.1

In figure 7.1 we see the points $(\frac{i}{n}, \frac{j}{n})$, $i, j = 1, 2, ..., n$, and $n = 10$, as a grid of points. The value of the empirical copula at some grid points is for example: $C_{10}(\frac{1}{10}, \frac{1}{10}) = 0$. $C_{10}(\frac{1}{10}, \frac{6}{10})$=$C_{10}(\frac{2}{10}, \frac{4}{10}) = 0.1$, $C_{10}(\frac{2}{10}, \frac{6}{10}) = 0.2$, $C_{10}(\frac{3}{10}, \frac{1}{10})$ $= 0.1$ and $C_{10}(\frac{5}{10}, \frac{5}{10}) = 0.2$. The total number of grid points is $10^2 = 100$.

According to the definition of the empirical copulas, if the sample size is n, and the sample is D-dimensional, one needs a data structure of size n^D, say M. If n and/or D are not small, M will be very or too big to be handled by a personal computer. This problem can be solved by defining an *approximate empirical copula*; instead of defining a data structure M of size n^D for the empirical copula, one might combine adjacent points and thus define a data structure of size k^D for an approximate empirical copula, where $k < n$.

Alternatively, one can store only the non-zero elements in the matrix. An approach which stores only the none zero elements and combines adjacent points is followed in section 7.2.

In an approximate empirical copula, each axis [0,1] of the D-spaces is divided into a specified number K_j of sub-intervals. For simplicity, we consider K_j=K for all $j = 1, 2, ...D$, unless otherwise mentioned. This approximate empirical copula results in that the final fitted distribution is also only an approximation. However, we show that this approximation is very good. More details is in section 7.1. From now on, we call the approximate empirical copula as the *approximate copula*.

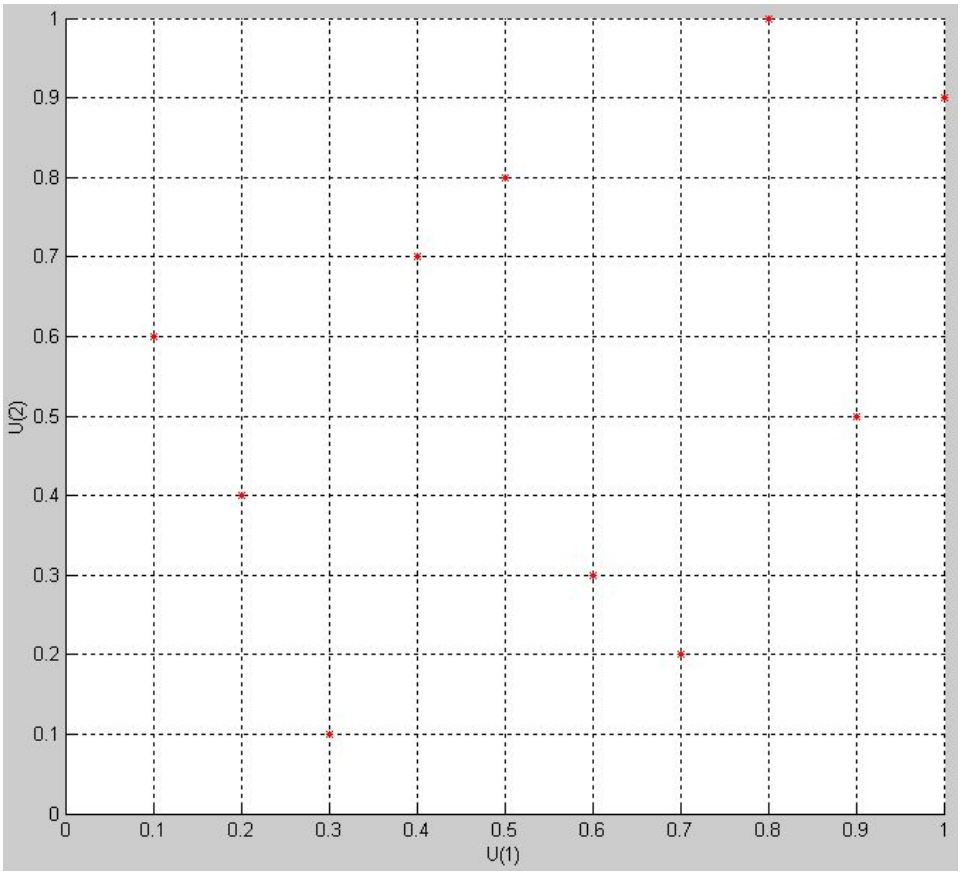

Figure 7.1: Illustration of a bivariate empirical copula

An illustration of a bivariate approximate copula can be given with the help of figure 7.2.

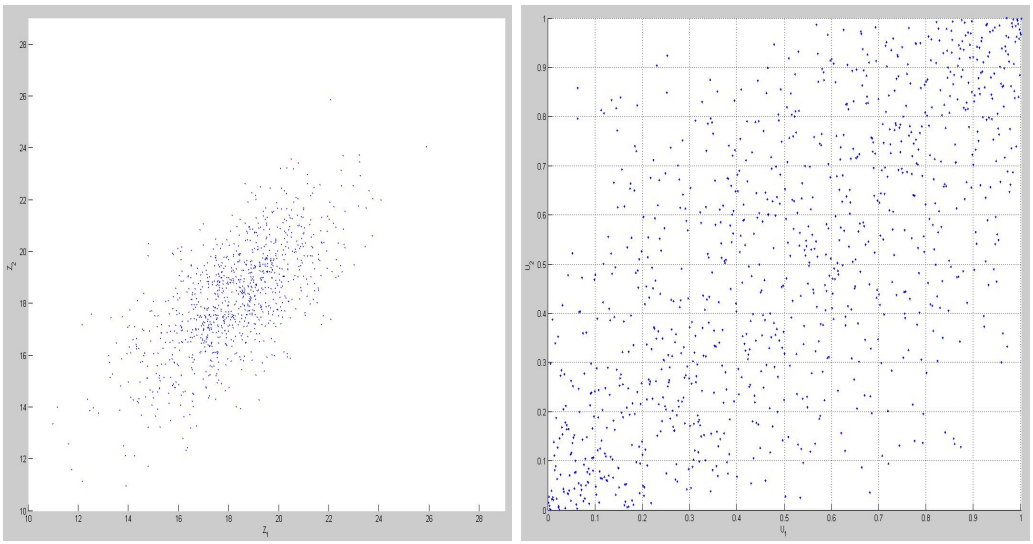

Figure 7.2: Left: Sample in the space of real numbers $\mathbb{R}^2$. Right: Sample in the $[0,1]^2$ space, $U_d = F_d(z_d)$, $d = 1, 2$.

In figure 7.2 we see a sample of a bivariate stochastic process $\mathbf{Z}$, where $\mathbf{Z}$ consists of two components, Z_1 and Z_2. The random variables Z_1 and Z_2 are linearly dependent. In figure 7.2 we see also the sample after being transformed into the space $[0,1]^2$. The components U_1 and U_2 of the transformed process $\mathbf{U}$ still exhibit a positive linear dependency as the original stochastic process does.

We have divided each of the [0,1] spaces, K, to 10 sub-spaces. This results in 100 different squares/grid points (points where the approximate copula is to be evaluated). Different squares might contain different number of points. In other words, there are different densities of points in the different squares.

Considering the grid points $(\frac{10}{100}, \frac{10}{100})$, $(\frac{10}{100}, \frac{20}{100})$, ... ,$(\frac{100}{100}, \frac{100}{100})$. One can count the number of data points that lie left and below each of these grid points. The resulting frequency function defines the approximate copula. The approximate copula is similar to the empirical copula. However, it is not defined over all points $(\frac{i}{n}, \frac{j}{n})$, but only over some points. The approximate copula is

$$C_{n,k}(\frac{l}{k}, \frac{m}{k}) = \frac{\text{number of pairs } (x,y) \text{ in the sample such that } F_1(x) \leq \frac{l}{k} \text{ and } F_2(y) \leq \frac{m}{k}}{n}, \tag{7.1}$$

where $l,\ m\ = 0, 1, 2, ..., k$. $F_1(x)$ is some marginal distribution function of the considered sample, e.g. empirical. $F_2(y)$ is the same but in the other

dimension.

This approximate empirical copula is not a true copula because the marginal distributions of the approximate copula are not uniform on [0,1], in general. In section 7.2, we define a new kind of empirical copulas and prove that it satisfies the conditions of a true copula.

Instead of calculating the approximate empirical copula, which is a discrete distribution function, we can calculate the approximate empirical copula density. The value of the approximate density function for one subspace depends on the number of sample points lying in the corresponding subspace. The exact value of the approximate copula density in those subspaces is the number of sample points lying in the subspace divided by (n * volume), where n is the total number of the sample points and volume is the area of each subspace. Each volume (area of a subspace) in figure 7.2 equals $0.1 * 0.1 = 0.01$.

The approximate empirical copula density is given by

$$c_n(l,m) = \frac{\text{number of points in subspace}(l,m)}{n * volume} \tag{7.2}$$

where $l, m = 1, ..., k$. Actually, this density is not used. Instead, we use in the approximate technique the distribution function (7.1) together with the marginal distributions. This and the complete procedure of fitting multivariate distributions which utilizes approximate empirical copulas is described in the next section. In section 7.2 we set some conditions on the empirical copulas so that they become real copulas, and explain the theory and the practice, how they can be used for modeling and generating dependent samples.

7.1 Fitting an Approximate Multivariate Distribution which Utilizes an Approximate Empirical Copula

In this section, we illustrate how an approximate multivariate distribution, which consists of an approximate copula and fitted marginal distributions, can be fitted to measured multivariate dependent data. The procedure consists in general of the following steps:

1. The marginal distributions, $F_d(Z_d),\ d = 1, ..., D$, where D is the dimension of the measured data sample, are built from the observed sample points, $\mathbf{z}$. $F_d(Z_d)$ can be empirical distribution functions of some kind, or fitted standard distributions like Exponential, Weibull, etc. We

noticed during our experiments that empirical distribution functions deliver often better results than the standard distribution functions in term of better results of statistical goodness-of-fit test. This happens when continuous or discrete standard distributions only badly fit the data.

2. Each vector of the multi-dimensional observed sample $z_i,\ i = 1, ..., D$, where D is the dimension of the sample, is transformed into points u_i of the unit D-space $[0, 1]^D$ by means of a marginal distribution function. For example, the uni-dimensional vector $\mathbf{z} = (3, 4, 7, 5)$ in $\mathbb{R}$ is mapped to $\mathbf{u} = [\frac{1}{4}, \frac{2}{4}, 1, \frac{3}{4}]$ in [0,1] according to the position of each point of $\mathbf{z}$ in the ordered vector $(3, 4, 5, 7)$.

   This step can be exemplified for a bi-dimensional sample as follows: Consider the bi-dimensional sample $\mathbf{z}$, where $z_1 = (0.10, 0.11, 0.20, 0.09)$ and $z_2 = (3, 4, 7, 5)$. First, the vectors z_1 and z_2 are mapped separately to points in $[0, 1]$. The vector z_1 is mapped to the vector $u_1 = [\frac{2}{4}, \frac{3}{4}, 1, \frac{1}{4}]$ and the vector z_2 is mapped to the vector $u_2 = [\frac{1}{4}, \frac{2}{4}, 1, \frac{3}{4}]$. This results in the bi-dimensional sample $\mathbf{u} = [(\frac{2}{4}, \frac{1}{4}), (\frac{3}{4}, \frac{2}{4}), (1, 1), (\frac{1}{4}, \frac{3}{4})]$.

3. The approximate copula density is estimated by a simple counting procedure. The copula density of the previous example is shown in figure 7.3. A sample of the bi-dimensional sample is also plotted for clarifying.

4. From the approximate copula density, the conditional probability $P\{U_2 \leq u_2 | U_1 = u_1\}$ is calculated. For the previous example, the conditional probabilities are as shown in figure 7.4.

The empirical copula density shown in figure 7.3 is actually a true copula as the marginal distributions of it are all uniform.

In the following, we give the procedure of fitting an approximate multivariate distribution which utilizes the approximate copula more formally.

1. Approximations $F_d(z),\ d = 1, ..., D$, of the unknown marginal distribution functions are built from the given sample. This can be empirical distribution functions of some kind, or fitted standard distributions like exponential, Weibull etc. For this, the sequences $z_{d,1}, z_{d,2}, ..., z_{d,n},\ d = 1, ..., D$, are ordered: $z_{d,(1)}, z_{d,(2)}, ...$ where $i < j$ implies $z_{d,(i)} \leq z_{d,(j)}$. For $z_{d,(i)}$ where $z_{d,(i-1)} < z_{d,(i)} = z_{d,(i+1)} = ... = z_{d,(i+m-1)} < z_{d,(i+m)}$, $F_d(z_{d,(i)}) = m/n$ holds, $F_{d,i}$ for short (here, $z_{d,(0)} = 0$ and $z_{d,(i,n+1)} = \infty$; these values are not used for estimation). For $z \in (z_{d,(i)}, z_{d,(i+1)})$, $F_d(z) = F_{d,i}$, but we will not use this.

   Alternatively, $F_d(z), d = 1, ..., D$, are fitted to standard distributions.

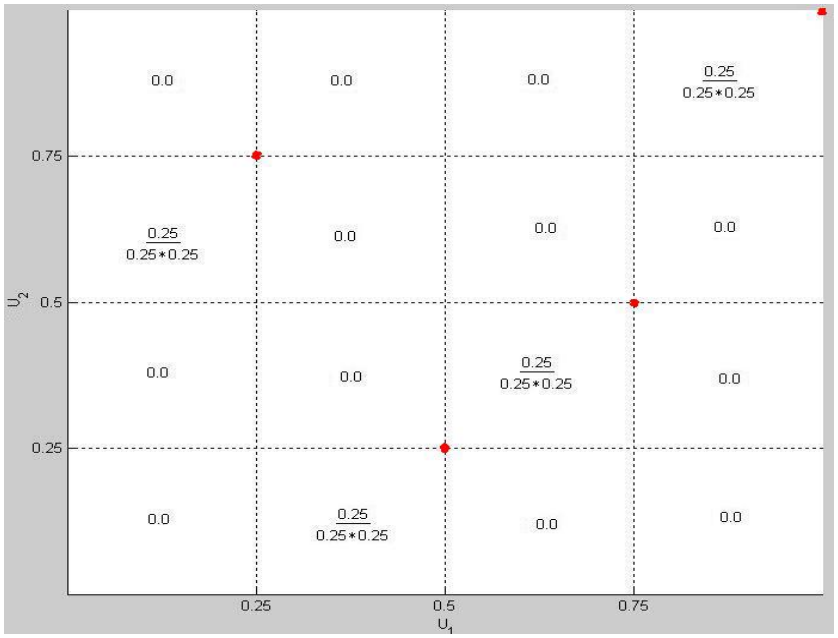

Figure 7.3: The values of the density in the subspaces

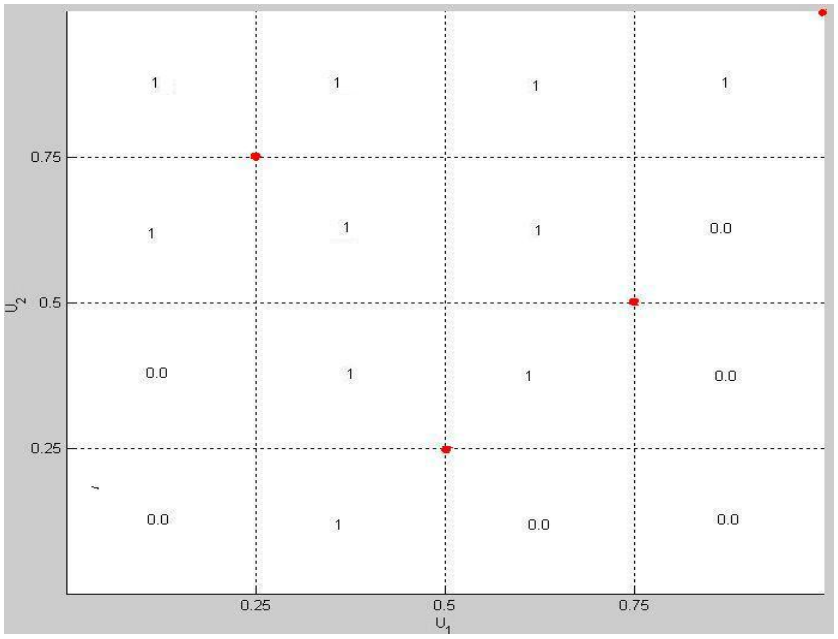

Figure 7.4: The values of the conditional probabilities $P\{U_2 \leq u_2 | U_1 = u_1\}$ in the different subspaces for the frequency distribution C_n

2. The observed sample points $\mathbf{z}_i,\ i = 1, ..., n$, are transformed into points $\mathbf{u}_i$ of the unit D-cube $[0, 1]^D$ by means of the marginal distribution functions. This means, $\mathbf{u}_i = (u_{1,i}, ..., u_{D,i}) \in [0, 1]^D$ where $u_{d,i} = F_{d,i},\ d = 1, ..., D,\ i = 1, ..., n$, are the transformed points.

3. The copula, more specific the density of the $\mathbf{u}_i$, is estimated. To this end, the D-cube is partitioned into sub-cubes. In each sub-cube, the density of the copula is estimated from the number of points $\mathbf{u}_i$ in the sub-cube, divided by n and the volume of the sub-cube. This gives $\mathcal{S}_\mathbf{j} = S_{1,j_1} \times ... \times S_{D,j_D},\ \mathbf{j} \in \mathcal{K} = \{1, ..., K_1\} \times ... \times \{1, ..., K_D\}$, are the sub-cubes of $[0, 1]^D$ where $S_{d,j} = [(j-1)\delta_d,\ j\delta_d),\ j = 1, ..., K_d - 1$, $S_{d,K_d} = [(K_d - 1)\delta_d,\ K_d\delta_d]$, and $\delta_d = 1/K_d,\ d = 1, ..., D$.

This partition $\mathcal{S}_\mathbf{j},\ \mathbf{j} \in \mathcal{K}$, induces a partition $\mathcal{T}_\mathbf{j} = T_{1,j_1} \times ... \times T_{D,j_D},\ \mathbf{j} \in \mathcal{K}$, in the original space $\mathcal{Z}$ of the observed random vectors $\mathbf{z}_i$ by means of $\mathbf{u}_i \in \mathcal{S}_\mathbf{j} \Leftrightarrow \mathbf{z}_i \in \mathcal{T}_\mathbf{j} \Leftrightarrow \forall d = 1, ..., D : z_{d,i} \in T_{d,j_d}$. This induced partition is unique only if the marginal distribution functions $F_d(z)$ are strictly increasing.

The approximate density of the copula is constant in each sub-cube $S_\mathbf{j},\ \mathbf{j} \in \mathcal{K}$. With the number $N_\mathbf{j}$ of points $u_\mathbf{j}$ in the sub-cubes $\mathcal{S}_\mathbf{j}$ and $H_\mathbf{j} = N_\mathbf{j}/n$, the density has the value $H_\mathbf{j}/(\delta_1 \cdot ... \cdot \delta_D)$. $H_\mathbf{j},\ \mathbf{j} \in \mathcal{K}$, is a frequency distribution for tuples $\mathbf{j}$.

The reader may note that these approximations are not really a copula: The marginal distributions are only approximately uniform. However, the empirical copulas, and thus their derivatives, the frequency copulas, converge to true copulas. See van den Goorbergh et al. (2005).

This copula, together with the pseudo-inverses of the marginal distributions, defines the approximate distribution $\mathcal{A}$.

The computational cost for the method is $O(n \log n + nK_1 \cdot ... \cdot K_D)$. In our examples, calculated with MATLAB on a 1GHz PC, the computing times were seconds or a few minutes.

7.1.1 Generating Random Vectors

Using the fitted approximate distribution $\mathcal{A}$, dependent uniform random vectors $\mathbf{u}' \in [0, 1]^D$ can be generated. From this, random vectors $\mathbf{z}'$ are obtained by means of the pseudo-inverses

$$F_d^{-1}(u) = \begin{cases} \inf\{z |\, F_d(z) \geq u\} & u > 0 \\ \sup\{z |\, F_d(z) = 0\} & u = 0 \end{cases}$$

of the estimated marginal distribution functions or the inverses of fitted standard distributions.

In the following we indicate how bi-dimensional random vectors $\mathbf{z}'$ are generated form the fitted approximate distribution $\mathcal{A}$. The generalization to higher dimensions is straightforward.

1. First a sub-cube $\mathcal{S}_{\mathbf{I}}$ is selected: Its index $\mathbf{I}$ is generated randomly from the frequency distribution of points $\mathbf{u}_i$ in the sub-cubes.

2. One of these points, $\hat{\mathbf{u}} = (\hat{u}_1, ..., \hat{u}_D)$, is selected according to its probability from the sub-cube $\mathcal{S}_{\mathbf{I}}$.

3. $\hat{\mathbf{z}}_d = F_d^{-1}(\hat{u}_d), d = 1, ..., D$, are the elements of the generated random vector $\hat{\mathbf{z}}$.

7.1.2 Examples

In the numerical examples, correctness and accuracy are verified via goodness-of-fit tests, scatter diagrams, and comparison of some statistical moments like the mean, the variance, and linear correlations. For this purpose, the statistical moments and the diagrams are calculated for the original sample $\mathbf{z}$, and the sample generated from the fitted approximate distribution $\mathcal{A}$, $\mathbf{z}'$. We also compare our results with results got from other already existing methods, when necessary.

Example 1: An Artificial Stochastic Process

We consider a sliding window over an artificially generated stochastic process. $Z_{1,i} = A_i,\ Z_{2,i} = A_{i+1},\ i = 1, ..., n$, where $A_{i+1},\ i = ..., -1, 0, 1, 2, ...$, is the stationary stochastic process defined by $A_{i+1} = 0.5\Big(1-4(A_i-0.5)^2\Big)+0.5X_i$, where the X_i are independent and uniformly distributed over $[0, 1]$.

We consider a sample of the stochastic process of size $n = 4000$. The number of sub-intervals in each of the two dimensions are chosen to be $K_1 = K_2 = 40$. The scatter diagrams of the original sample and the sample generated from the fitted approximate distribution shown in figure 7.5 indicate that there are regions where no points can exist, and that these regions are observed by the generated process with good accuracy.

For comparison, the artificially generated sample can be fitted to a stochastic process from the autoregressive type. We fit the sample to a linear nGAR process with lag equal to 1. Taking a sample of the fitted nGAR process, we see obviously that there are many points which lie in the impossible region as seen in figure 7.6.

We may hint to that the extended Yule-Walker method explained in chapter 5 can be also used for the purpose of estimating the parameters of the

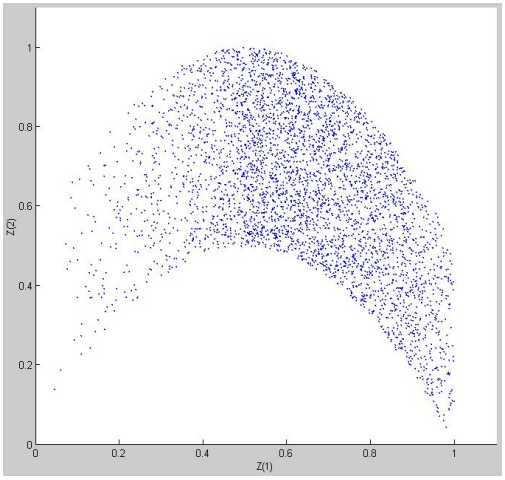

Figure 7.5: The original sample

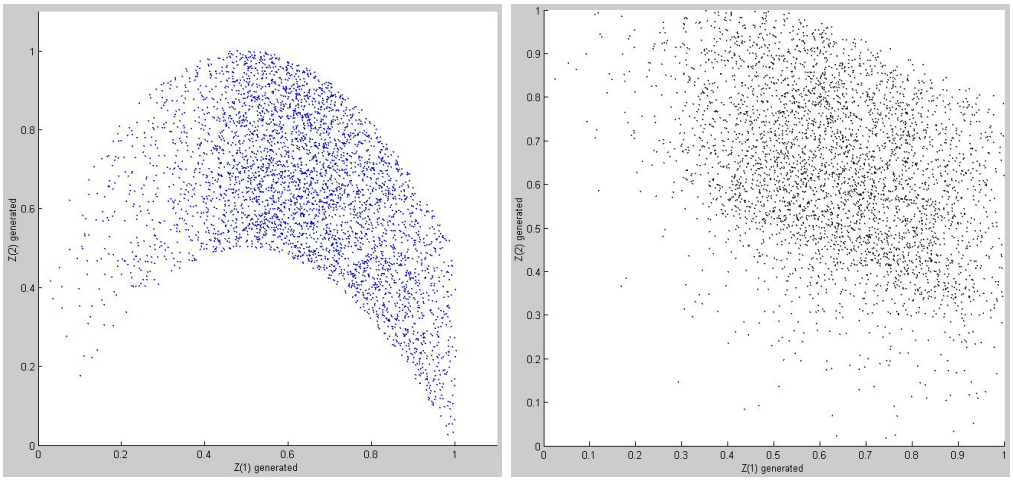

Figure 7.6: Left: Sample from the fitted approximate distribution utilizing a copula. Right: Sample from the fitted linear nGAR model

non-linear model. However, in this case, the model must be known in advance. If one has no idea about the model, the method of the empirical copula is suitable.

In addition to comparing the scatter plots of the original and the generated samples, we also compare in table 7.1 some statistical moments.

Table 7.1: Results summary of samples from the original artificial process, from a fitted approximate distribution, and from a fitted linear nGAR process

	Original	**Appr. Distr.**	**Linear nGAR**
[$\underline{Cov}$]	[-0.4488, 0.1505]	[-0.4652 , 0.1476]	[-0.4465 , 0.1750]
mean	0.6427	0.6436	0.6429
variance	0.0348	0.0340	0.0348
minimum	0.0327	0.0599	-0.1370
maximum	0.9989	0.9989	1.1334
MAR	-	0.0021	0.0080
LSS	-	8.65e-006	2.06e-004
KSS	-	0.0078	0.0235

As seen in Table 7.1, the fitted approximate distribution utilizing an approximate copula, and the fitted linear nGAR process, deliver good approximation of the statistical moments of the mean, the variance and the covariance, and small test statistics MAR, LSS, and KSS. The statistics MAR, LSS, and KSS are the mean absolute distance, the least-square, and the Kolmogorov-Smirnov test statistics (see section 4.5). Nevertheless, the fitted approximate distribution which utilizes the approximate empirical copula delivers better results and could capture the non-linear dependencies as seen in figure 7.6. We also see a problem in capturing the minimum and maximum value when fitting a linear nGAR process to the artificially generated data.

Example 2

We consider observed data by Klemm et al. (2002) at an Internet server. The data consists of interarrival times (A_i) and packet lengths (B_i). The original sample size is 150 000 bivariate samples. We consider only part of this sample for analysis (4 000).

Considering that current interarrival time depends on the previous interarrival time and the previous packet length, and considering that current packet length depends on the previous packet length and interarrival time,

we get the stochastic process: $Z_{1,i} = A_i,\ Z_{2,i} = B_i,\ Z_{3,i} = A_{i+1},\ Z_{4,i} = B_{i+1},\ i = 1, ..., n.$

Having a sample size of $n = 4000$ and number of sub-intervals of $K_1 = K_2 = K_3 = K_4 = 40$, we fit an approximate distribution which utilizes an approximate copula.

Some components of the original four-dimensional stochastic process $\mathbf{Z}_i = [Z_{1,i}, Z_{2,i}, Z_{3,i}, Z_{4,i}]'$ are plotted in scatter diagrams. We plot dimension 1 (interarrival times at time i) against dimension 2 (packet length at time i) as seen in figures 7.7 and we plot dimension 1 (interarrival times at time i) against dimension 3 (interarrival times at time i+1) as seen in figure 7.9.

We compare these plots with plots from a sample generated from the fitted approximate distribution and a sample generated from the BMAP model in figure 7.8 and in figure 7.10. For the given dimensions, the plots indicate good fitting of the sample generated from the approximate distribution. The plots of the sample generated from the fitted BMAP model indicate less accuracy than the sample from the approximate distribution. Plots from the other dimensions are similar to the presented ones and will not be shown here.

We have to mention here that we used in this example a fitted BMAP model that did not converge after a specific large number of iterations (100). This decision is due to the fact that the BMAP fitting procedure does not converge in most of the cases. Convergence of the BMAP procedure depends beside other conditions on the initial conditions which are taken randomly. However, even using a BMAP model which converged to the desired one does not give much more similar plots to the original ones. The differences in the plots are due to the nature of the BMAP process and not due to the non-convergence.

In table 7.2 we compare some statistical moments of the original sample, a sample generated from the fitted approximate distribution which utilizes a fitted approximate copula, and a sample generated from the BMAP model of Klemm et al. (2002). The statistics MAR, LSS, and KSS are the mean absolute distance, the least-square, and the Kolmogorov-Smirnov test statistics described in section 4.5.

We see from table 7.2 that the sample from the fitted approximate distribution fits better the original sample than the sample generated from the BMAP model. However, the "common" distribution might be more interesting. A statistical test that can indicate how well the fitted process and the original process are close is the number of bytes arriving at the server per time unit. For this purpose we plot the number of bytes against time for the time scales 0.1 seconds, as also done in Klemm et al. (2002). The plots from the sample generated from approximate distribution is more similar to the original one than that of the BMAP sample. This is easy to see in figures

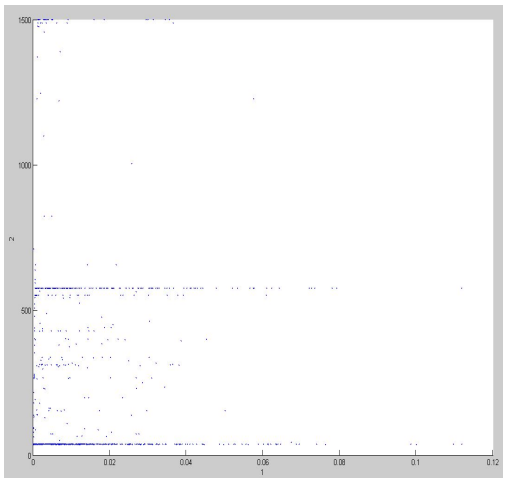

Figure 7.7: The original sample; Dimension 1 & 2

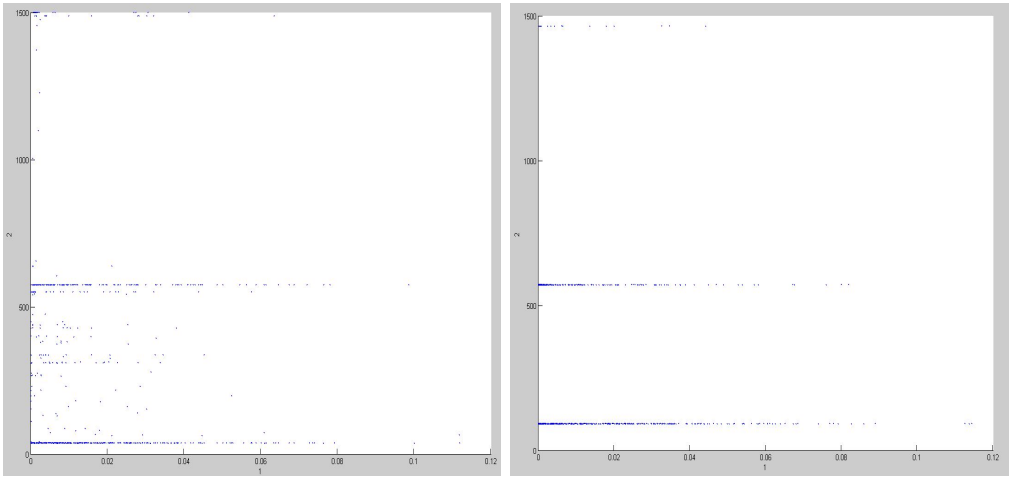

Figure 7.8: Left: Sample from the approximate distribution. Right: Sample from the BMAP model; Dimension 1 & 2

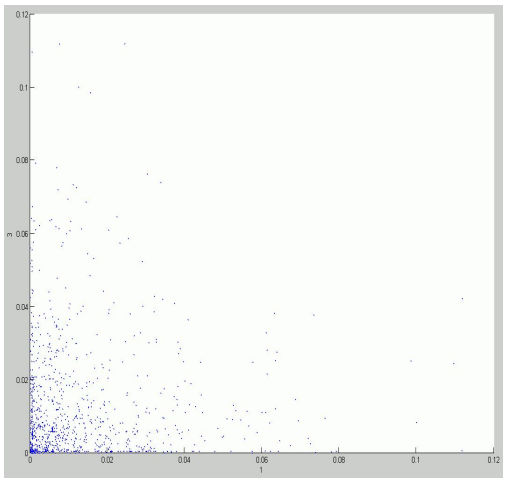

Figure 7.9: The original sample; Dimension 1 & 3

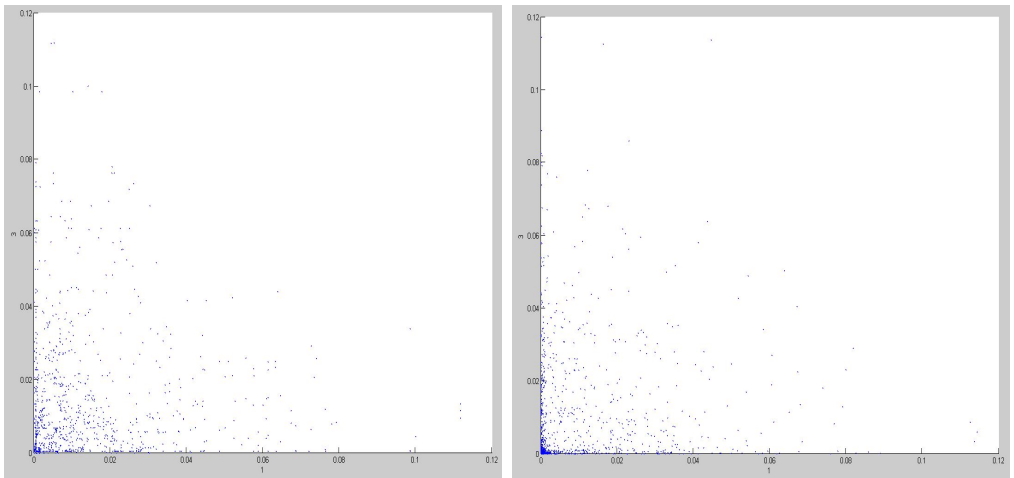

Figure 7.10: Left: Sample from the approximate distribution. Right: Sample from the BMAP model; Dimension 1 & 3

Table 7.2: Results summary of the original sample, a sample from a fitted approximate distribution, a fitted BMAP process

	Original	**Appr. Distr.**	**BMAP**
[Correlation]	[0.0884, 0.2388]	[0.0791, 0.2432]	[0.05671, 0.2297]
[mean]	[0.0108, 311]	[0.0109 306]	[0.0111 , 322]
[variance]	[2.0e-4, 1.128e5]	[1.98e-4, 1.125e5]	[2.928e-4 , 0.81709e5]
[minimum]	[6.40e-5, 40]	[6.50e-5, 40]	[0 , 94]
[maximum]	[0.1123, 1500]	[0.1123, 1500]	[0.1695, 1465]
[MAR]	-	[1.2826-04, 3.981]	[20.2e-04, 63.43]
[LSS]	-	[1.5549-7, 1067]	[1.08256e-05, 17972]
[KSS]	-	[0.011, 0.005]	[0.205 , 0.50]

7.11 and 7.12.

7.1.3 Where to Take Care

Difficulties might occur if one or more marginal distributions of the stochastic process of interest is discrete. Considering a bi-dimensional copula, a problem that might occur is that one or more columns of the approximate empirical copula contain(s) no sample points as shown in figure 7.13.

This problem appears in the following cases:

1. The number of sub-intervals chosen for the approximate copula is bigger than the number of available sample points. In this case, zero points will lie in one or more columns as shown in the left part of figure 7.13.

2. Choosing a kind of marginal distributions that maps real values to the space [0, 1] depending on the value of the sample points and not on the position of the sample points in the sample ordered statistics. This is illustrated in the following example.

   Consider the bi-dimensional vector (X_i, Y_i), $i = 1, 2, 3, 4$. Let (x_1, y_1)=(0.1, 1), (x_2, y_2)=(0.3, 4), (x_3, y_3)=(0.7, 4), and (x_4, y_4)=(0.9, 4). To fit an approximate distribution to this sample, all $\mathbf{x}$ and $\mathbf{y}$ must be first mapped separately to points in $[0, 1]$ by means of marginal distributions.

Choosing a marginal distribution that maps random variates with similar values to the same point in the [0, 1] space means that the above points

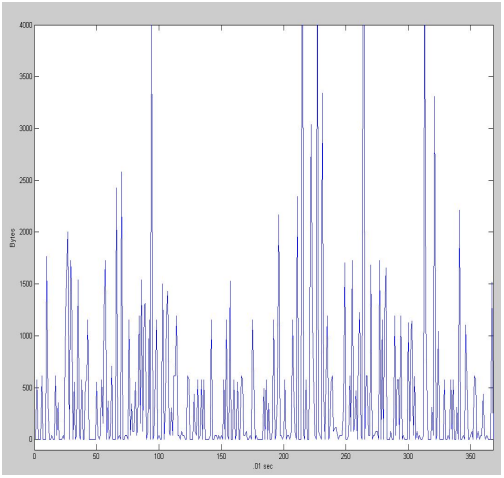

Figure 7.11: Number of bytes against time for the original process. Time scale 0.01 sec

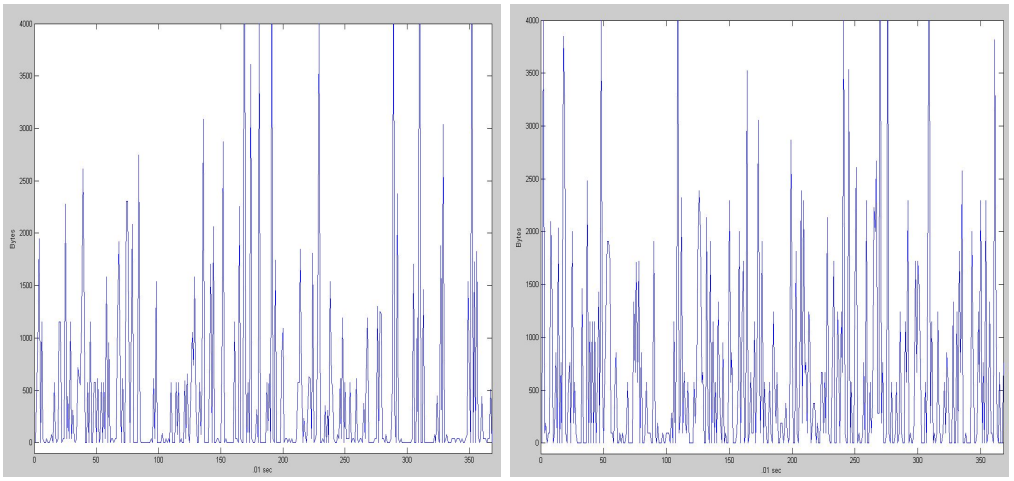

Figure 7.12: Number of bytes against time for a process from the approximate distribution (left) and for the BMAP process (right). Time scale 0.01 second

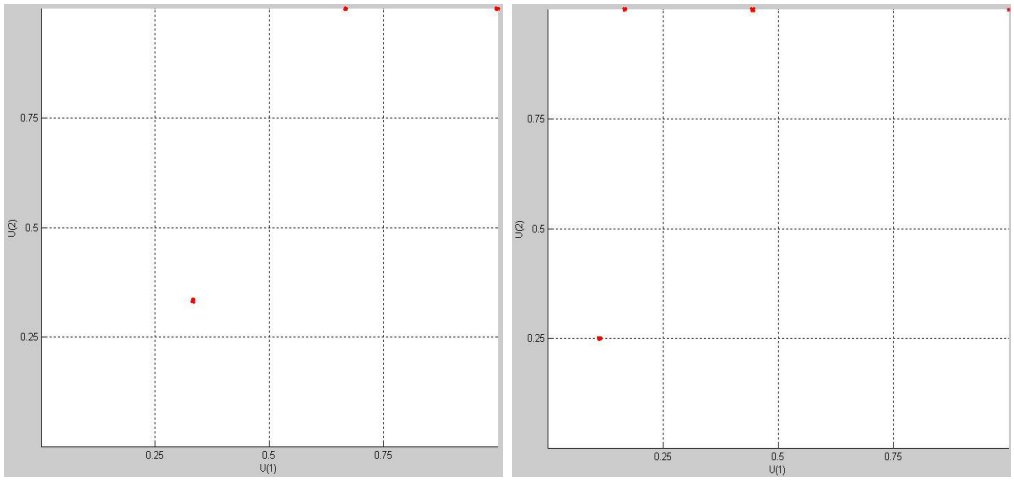

Figure 7.13: Some columns of the copula have no entries

will be the following points in the space [0,1]: **u**= [(0.1, 0.25), (0.3, 1), (0.7, 1), (0.9, 1)], as the $x_i = [0.1, 0.3, 0.7, 0.9]$ are mapped to the vector $u_{1,i} = [0.1, 0.3, 0.7, 0.9]$ and the $y_i = [1, 4, 4, 4]$ are mapped to the vector $u_{2,i} = [1/4, 1, 1, 1]$. Here we assume that the range of the real values of vector y_i is [0,4]. The cumulative distribution function of $U_{2,i}$ has a jump at $i = 4$ and is shown in figure 7.14

We see also in figure 7.14 the approximate copula resulting of the process. We notice a column in which no sample points lie.

Such an approximate copula which contains empty columns cause a problem when using them to generate random vectors. As described in section 7.1.1, generating a two dimensional sample requires first generating a uniform random number u_1 and secondly, depending on the conditional probabilities resulting from the approximate copula, generating u_2, provided $U_1 = u_1$. However, if u_1 lies in column 2 or 3, no u_2 can be generated, as all probabilities to choose one of the sub-intervals equals to zero. We call this problem as the problem of the deadend.

A solution of this problem lies first in choosing no more sub-intervals than the number of the available sample points. Moreover, instead of choosing marginal distributions that depend on the values of the sample points when mapping to the [0,1] space, distributions which depend on the position of the sample points in the sample ordered statistics of the samples are used.

Using such a procedure for the example mentioned above will result in the following process whose marginal distributions are in the [0,1] space: $\mathbf{u} = [(\frac{1}{4}, \frac{1}{4}), (\frac{2}{4}, \frac{2}{4}), (\frac{3}{4}, \frac{3}{4}), (1, 1)]$, as the the $\mathbf{x}_i$=[0.10, 0.3, 0.7, 0.9] are mapped to the vector $\mathbf{u}_{1,i} = [\frac{1}{4}, \frac{2}{4}, \frac{3}{4}, 1]$ and the $\mathbf{y}_i$=[1, 4, 4, 4] are mapped to the

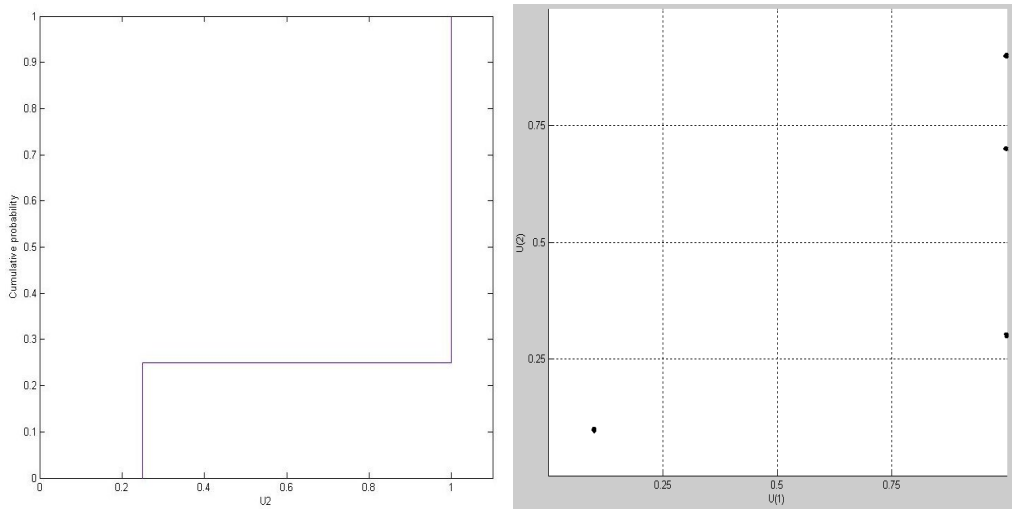

Figure 7.14: An empirical marginal distribution with a jump and an approximate copula of it

vector $\mathbf{u}_{2,i} = [\frac{1}{4}, \frac{2}{4}, \frac{3}{4}, 1]$.

We mentioned that different kinds of marginal distributions can be used. This is possible because a marginal distribution and its inverse will be used in the same procedure. In other words, if the sample is Z if the marginal distribution is F_1, one will use at one point $F_1(Z)$ and at a later point of the procedure $F_1^{-1}(F_1(Z))$ which results in Z always, whatever of which kind of marginal distributions F_1 is.

The second problem that might occur is that the generated points for time series end in a cycle of points which recur again and again. Some other points do not occur even once. This problem occurs with some probability if the sample is very small, in bigger samples this probability becomes smaller and smaller. We observed these problems only when very small samples were used for the distribution $\mathcal{A}$, ten points or so.

7.2 Fitting a multivariate distribution utilizing a real empirical copula

We explain in this section how to efficiently fit a multivariate distribution which utilizes a real empirical copula to a given D-dimensional sample of size n, and how to generate random vectors out of the fitted multivariate distribution.

We explain first how to estimate multivariate distribution functions, which

consist of marginal (empirical) distribution functions in the form of step functions, and a real empirical copula, to the sample. Then we give an efficient algorithm for the generation of D-variate samples out of the fitted multivariate distribution. For the purpose of fitting a real empirical copula, an integer K which controls the accuracy and the space and time complexities, must be defined. The space and time complexities are increasing with K.

In this section, after we describe basic facts about copulas, we present examples implemented in MATLAB program. The examples indicate accurate statistical properties of our new technique.

Once we have the multivariate distribution function estimated, multivariate random vectors can be generated. In the following subsections, we investigate in more detail the ideas behind fitting a multivariate distribution function utilizing the empirical copulas to data, and how to generate random vectors from the fitted distribution function.

7.2.1 Continuous Piecewise Multi-Linear Empirical Copulas

We consider continuous empirical copulas $C(\mathbf{u})$, $\mathbf{u} = (u_1, ..., u_D)$, which are linear in each variable u_d, $d = 1, 2, ..., D$, if the other variables u_e, $e = 1, 2, ...D$, $e \neq d$, are fixed. This linearity holds in each subspace of the unit D-space $[0, 1]^D$, which means that the density $c(\mathbf{u})$ of the copula is constant in each of these subspaces. However, the slopes in general are different in different subspaces. Therefore, we call these empirical copulas as *continuous piecewise multi-linear*.

In this section, we point out how a multivariate distribution which consists of a copula and empirical distribution functions can be estimated from a given sample, and we prove that the estimated continuous empirical copulas meet the requirements of real (empirical) copulas. Then we present an algorithm for computing the continuous empirical copulas and their conditional distribution functions needed for the later generation of the desired random vectors. Moreover, we consider time and space complexity of the used algorithm.

The subspaces are defined with the partitions $S_1, ..., S_K$ of the interval $[0, 1]$, $S_1 = [0, \delta]$ where $\delta = 1/K$ and K is a positive integer, $S_j = \Big((j-1)\delta, j\delta\Big]$, $j = 2, ..., K$. The D-dimensional subspaces are $\mathcal{S}_\mathbf{j} = S_{j_1} \times ... \times S_{j_D}$, $\mathbf{j} = (j_1, ..., j_D) \in \mathcal{K} = \{1, ..., K\}^D$.

We consider a sample $\mathbf{z}_i = (z_{1,i}, ..., z_{D,i})$, $i = 1, ..., n$, where n is the sample size and D is the dimension of the sample. We assume that the

different sample vectors $\mathbf{z}_i$ are independent and identically distributed (IID) for different i. For the definition of the copula, we use in each dimension $d = 1, ..., D$ the order statistics $z_{d,(i)},\ i = 1, ..., n$, of the elements of dimension d. Hence $z_{d,(i)} \leq z_{d,(i+1)},\ i = 1, 2, ...n-1$. Let $I_d : \{1 : n\} \to \{1 : n\}$ denote the function which maps the old place l of $z_{d,l}$ in the sample to the new place $I_d(l) = i$ in the order statistics $z_{d,(i)},\ i = 1, ..., n$.

With this mapping, we define $u_{d,l} = I_d(l)/n,\ l = 1, ..., n,\ d = 1, ..., D$, and $\mathbf{u}_l = (u_{1,l}, ..., u_{D,l}),\ l = 1, ..., n$. $\mathbf{u}_l$ are image points of the sample points $\mathbf{z}_l$ in the unit D-space $[0, 1]^D$.

Now we explain that the values $u_{d,l}$ are closely related to the empirical distribution function of the $z_{d,l}$. We use a specific kind of empirical distribution functions, namely step functions, one in each dimension $d = 1, ..., D$. The empirical distribution functions are defined with the order statistics as follows:

$$F_d(z) = \begin{cases} 0 & z < z_{d,(1)} \\ i/n & z_{d,(i)} \leq z < z_{d,(i+1)},\ i = 1, ..., n-1 \\ 1 & z \geq z_{d,(n)} \end{cases} \tag{7.3}$$

If in the order statistics there is $z_{d,(i)} = z_{d,(i+1)}$, then there is no z for which $F_d(z) = i/n$ which means that i/n is not in the range of F_d. In other words, if all sample values $z_{d,l},\ l = 1, ..., n$, are different, then $F_d(z_{d,l}) = u_{d,l}$ holds. If some sample values are not unique, then $F_d(z_{d,l}) = u_{d,l}$ holds only for values $z_{d,l}$ which occur only once, and for just one of the equal values $z_{d,l_1} = z_{d,l_2} = ...$ Namely for multiple values $z_{d,l_1}, z_{d,l_2}, ..., z_{d,l_x}$, x is the number of all $z_{d,l}$ which equal one value, the value of the distribution function at all the points z_{d,l_i}, for all $i = 1, 2, ..., x$, is $F_d(z_{d,l_i}) = \max\{u_{d,l_1}, u_{d,l_2}, ..., u_{d,l_x}\}$.

Let $N_{\mathbf{j}}$ denote the number of points in $\mathcal{S}_{\mathbf{j}},\ \mathbf{j} \in \mathcal{K}$. Then we define the density $c(\mathbf{u}) \equiv c_D(\mathbf{u})$ of the copula as

$$c(\mathbf{u}) = f_{\mathbf{j}},\ \ \mathbf{j} \in \mathcal{K},$$

where we use the notation $f_{\mathbf{j}} = N_{\mathbf{j}}/n/\delta^D$. For $u_d \in S_j \setminus \{0\}, j = \lceil u_d K \rceil$ holds, and for $u_d = 0,\ \ j = 1$. Hence, with the up-operator $\uparrow$: $[0, 1] \to \{1, ..., K\}$, $\uparrow u = \max\{1, \lceil uK \rceil\}$, we can write

$$c(\mathbf{u}) = f_{\uparrow u_1, ..., \uparrow u_D},\ \ \mathbf{u} \in [0, 1]^D. \tag{7.4}$$

Let $c(\mathbf{u}) \equiv c_D(\mathbf{u})$ denote the density of the copula, and $c_d(\mathbf{u}),\ d = 1, ..., D-1$, the marginal densities. Then we can write

$$c_1(u_1) = \int_{u_2=0}^{1} ... \int_{u_D=0}^{1} c(\mathbf{u}) du_2 ... du_D,$$

$$c_2(u_1,u_2)=\int_{u_3=0}^{1}...\int_{u_D=0}^{1}c(\mathbf{u})du_3...du_D,$$

$$...$$

$$c_d(u_1,...,u_d)=\int_{u_{d+1}=0}^{1}...\int_{u_D=0}^{1}c(\mathbf{u})du_{d+1}...du_D. \tag{7.5}$$

From the conditional probability we know that $P(B|A)=\frac{P(A\cap B)}{P(A)}$, where P is the probability and A and B are events. Then we have the copula conditional distribution function as

$$C_d(u_d|u_1,...,u_{d-1})=\frac{\int_{u=0}^{u_d}c_d(u_1,...,u_{d-1},u)du}{c_{d-1}(u_1,...,u_{d-1})}, d=2,...,D. \tag{7.6}$$

where $c_{d-1}(u_1,...,u_{d-1})>0$. Please note that C_d is a conditional distribution function.

Now we are ready to prove that the marginal copula densities can be expressed from (7.5) as

$$c_d(u_1,...,u_d)=\delta^{D-d}f^{(d)}_{\uparrow u_1,...,\uparrow u_d},\ \ d=1,...,D-1, \tag{7.7}$$

where

$$f^{(d)}_{j_1,...,j_d}=\sum_{(j_{d+1},...,j_D)=(1,...,1)}^{K\times...\times K}f_{\mathbf{j}},$$
$$(j_1,...,j_d)\in\{1,...,K\}^d,$$

and the conditional distribution functions can be expressed from (7.6) as

$$C_2(u_2|u_1)=\delta^{D-1}\sum_{j_2=1}^{\downarrow u_2}f^{(2)}_{\uparrow u_1,j_2}+(u_2-\downarrow u_2\delta)\delta^{D-2}f^{(2)}_{\uparrow u_1,\uparrow u_2},$$
$$C_d(u_d|u_1,...,u_{d-1})=\frac{\delta\sum_{j_d=1}^{\downarrow u_d}f^{(d)}_{\uparrow u_1,...,\uparrow u_{d-1},j_d}+(u_d-\downarrow u_d\delta)f^{(d)}_{\uparrow u_1,...,\uparrow u_d}}{\delta f^{(d-1)}_{\uparrow u_1,...,\uparrow u_{d-1}}},$$
$$d=3,...,D, \tag{7.8}$$

$\downarrow u=\uparrow u-1$, for positive denominators.

From the definition of the marginal densities we can write

$$c_{D-1}(u_1,...,u_{D-1}) = \int_{u=0}^{1} c_D(u_1,...,u_{D-1},u)\ \ du, \quad u_d \in [0,1], d = 1,...,D-1$$

$$= \int_{u=0}^{\delta} c_D(u_1,...,u_{D-1},u)\ \ du + \int_{u=\delta}^{2\delta} c_D(u_1,...,u_{D-1},u)\ \ du$$

$$+ ... + \int_{u=(K-1)\delta}^{K\delta} c_D(u_1,...,u_{D-1},u)\ \ du$$

$$= \sum_{j=1}^{K} \int_{u=(j-1)\delta}^{j\delta} c_D(u_1,...,u_{D-1},u)\ \ du$$

$$= \sum_{j=1}^{K} \int_{u=(j-1)\delta}^{j\delta} f_{\uparrow u_1,...,\uparrow u_{D-1},j}\ \ du$$

$$= \sum_{j=1}^{K} f_{\uparrow u_1,...,\uparrow u_{D-1},j}\ \delta$$

$$= \delta \sum_{j=1}^{K} f_{\uparrow u_1,...,\uparrow u_{D-1},j}$$

This means that the marginal density $c_D(u_1,...,u_{D-1})$ is constant for all $u_d \in S_j,\ d = 1,...D-1$. And accordingly we can write for all $d = 1,...D$:

$c_{d-1}(u_1,...,u_{d-1}) = \delta \sum_{j_d=1}^{K} \delta \sum_{j_{d+1}=1}^{K} ... \delta \sum_{j_D=1}^{K} f_{\uparrow u_1,...,\uparrow u_{d-1},j_d,...,j_D}$ which gives equation (7.7).

Now we derive the formula for the conditional copula distribution function. We know from the probability theory that

$$C_2(u_2|u_1) = \frac{\int_{u=0}^{u_2} c_2(u_1,u)du}{c_1(u_1)}$$

where $u_1, u_2 \in [0,1]$ and u_1 is fixed. We notice that the denominator $=1$, because C is a copula. Hence

$$C_2(u_2|u_1) = \int_{u=0}^{\delta} c_2(u_1,u)du + ... + \int_{u=(\downarrow u_2-1)\delta}^{\downarrow u_2 \delta} c_2(u_1,u)du + \int_{u=\downarrow u_2 \delta}^{u_2} c_2(u_1,u)du.$$

As c_2 is constant in the integrals we can write

$$C_2(u_2|u_1) = \sum_{j_2=1}^{\downarrow u_2} \int_{u=(j_2-1)\delta}^{j_2\delta} \delta^{D-2} f^{(2)}_{\uparrow u_1, j_2}\, du + \int_{u=\downarrow u_2\delta}^{u_2} \delta^{D-2} f^{(2)}_{\uparrow u_1, \uparrow u_2}\, du$$

$$C_2(u_2|u_1) = \sum_{j_2=1}^{\downarrow u_2} \delta^{D-2} f^{(2)}_{\uparrow u_1, j_2}\delta + \delta^{D-2} f^{(2)}_{\uparrow u_1, \uparrow u_2}(u_2 - \downarrow u_2\delta)$$

this gives the first part of equation 7.8.

The second part of formel (7.8) can be derived in a similar manner, with the difference that the denominator is generally not equal to 1. So that

$$C_d(u_d|u_1, ..., u_{d-1}) = \frac{1}{\text{denom}} \int_{u=0}^{u_d} c_d(u_1, ..., u_{d-1}, u)du$$

$$= \frac{1}{\text{denom}} \sum_{j_d=1}^{\downarrow u_d} \int_{u=(j_d-1)\delta}^{j_d\delta} \delta^{D-d} f^{(d)}_{\uparrow u_1, ..., \uparrow u_{d-1}, j_d}\, du + \frac{1}{\text{denom}} \int_{u=(u_d)\delta}^{u_d} \delta^{D-d} f^{(d)}_{\uparrow u_1, ..., \uparrow u_d}\, du$$

$$= \frac{1}{\text{denom}} \sum_{j_d=1}^{\downarrow u_d} \delta^{D-d} f^{(d)}_{\uparrow u_1, ..., \uparrow u_{d-1}, j_d}\delta + \frac{1}{\text{denom}} \delta^{D-d} f^{(d)}_{\uparrow u_1, ..., \uparrow u_d}(u_d - \downarrow u_d\delta), d = 3, ..., D$$

where denom $= c_{d-1}(u_1, ...u_{d-1}) = \delta^{D-(d-1)} f^{(d-1)}_{\uparrow u_1, ..., \uparrow u_{d-1}}$ which gives the second part of formula (7.8).

Now we are prepared to prove

Theorem 7.2.1 *If the sample size n and K are such that K divides n, $C(\mathbf{u})$ is a copula.*

Proof It suffices to prove that $c(\mathbf{u})$ is a density in the D-space $[0,1]^D$, and that $c_1(u)$ is the uniform density over $[0,1]$.

From (7.4), we conclude $c(\mathbf{u}) \geq 0$. Integrating the $c(\mathbf{u})$ over the D-space, one obtains $\delta^D \sum_{\mathbf{j}\in\mathcal{K}} f_{\mathbf{j}} = \delta^D n/n/\delta^D = 1$. Hence $c(\mathbf{u})$ is a density.

From the definition of $I_d(l)$, one can conclude that for fixed $d \in \{1, ..., D\}$, $I_d(l)$ for all $l = 1, ..., n$ assumes all values $i = 1, ..., n$, each value once. Hence, the $u_{d,l}$ for all $l = 1, ..., n$ assume all values $1/n, ..., (n-1)/n, 1$, each value once. Hence, in each subspace $S_{j_1} \times [0,1]^{D-1}$ are n/K image points $\mathbf{u}_l$. From (7.7) we find $f^{(1)}_{j_1} = n/K/n/\delta^D = 1/\delta^{D-1}$ and $c_1(u_1) = 1$. $c_1(u)$ is the uniform density over $[0,1]$. This concludes the proof.

Algorithm 1 below can be applied to calculate the arrays $f^{(d)}$ which we use for the calculation of the conditional distribution functions of the copula.

Algorithm 1

1. Calculate the empirical marginal distribution functions of the sample with (7.3) and the image points $\mathbf{u}_i$ of the sample points $\mathbf{z}_i$, $i = 1, ..., n$.

2. Set all elements of the arrays $f^{(d)}$ to 0.

3.
for $i := 1$ **to** n **do**
  for $d := 1$ **to** D **do**
    $j_d :=\uparrow u_{d,i}$;
  end
  for $d := 2$ **to** D **do**
    $f^{(d)}_{j_1,...,j_d}$ **plus** $1/n/\delta^D$;
  end
end

Each of the D empirical distributions need $\mathrm{O}(n \log n)$ steps for sorting n numbers, so this is done in $\mathrm{O}(Dn \log n)$ time.

For the space complexity, we remark that the arrays can be expected to be sparse. The biggest one, $f^{(D)}$, has K^D elements, most of which will be zero if $K^D \gg n$. Therefore we propose to store the big arrays $f^{(d)}$ in a sparse manner where elements with value zero are not stored. Hence, if K^D is not small, the needed storage is of order $O(Dn)$ since at most $(D-1)n$ times an array element changes its value from zero to nonzero. So we conclude the space complexity $O(Dn)$.

But, when an array is stored in a sparse manner, the index values must be stored together with each array element, and each time when an element is accessed, its index values are read. This results in a space complexity of $O(D^2n)$ and time complexity of $O(D^2n \ log \ n)$. Now we can state

Theorem 7.2.2 *Algorithm 1 has the time complexity $O(D^2 n \log n)$, and the data structures have the space complexity $O(D^2 n)$.*

7.2.2 The Generation Algorithm

Once we have the multivariate distribution function estimated, multivariate random vectors can be generated in two steps:

1. Generate a random D-vector with the empirical copula.

2. Transform its elements with the inverses of the marginal distribution functions of the sample. For this we propose three different alternatives:

   a) The estimated empirical distribution functions of the components in form of a step function. Here one can obtain only values which occur in the sample. This can be sensible for integer random variables, in particular.

   b) Some kind of linear interpolation. Here one can obtain values which lie between zero and the largest value of the sample, or something similar.

   c) Fitted standard distributions if feasible, where feasible means: The empirical distribution function of the sample is similar to a standard distribution function, and this standard distribution function can be inverted in a sufficiently simple manner.

In step 1, a random number u_1 is generated from the uniform distribution on the interval $(0, 1]$. The zero values are excluded to make the implementation of the quasi-inverse of distribution functions easier. Then $u_d,\ d = 2, ..., D$, are generated with the conditional distribution functions by (7.8).

Step 1 of the generation can be accomplished as follows:

u is a uniform random variable on [0,1]. With this random variable a uniform random variable u_d must be generated, where $u_1, u_2, ..., u_{d-1}$ are all previously generated.

For the generation we use the following transformation method:

1. search for j such that $C_d(j\delta|u_1, ..., u_{d-1}) \leq u \leq C_d((j+1)\delta|u_1, ..., u_{d-1})$.

2. calculate u_d from $u = C_d(u_d|u_1, ..., u_{d-1})$.

   In case of $d = 2$ we can write: $u = \delta^{D-1} \sum_{j_2=1}^{\downarrow u_2} f^{(2)}_{\uparrow u_1, j_2} - (u_2 - \downarrow u_2\delta)\delta^{D-2} f^{(2)}_{\uparrow u_1, \uparrow u_2}$ which gives the piecewise linear copula.

   In case of general d we can write:

$$u = \frac{\delta \sum_{j_d=1}^{\downarrow u_d} f^{(d)}_{\uparrow u_1, \uparrow u_2, ..., \uparrow u_{d-1}, j_d} - (u_d - \downarrow u_d\delta) f^{(d)}_{\uparrow u_1, \uparrow u_2, ... \uparrow u_d}}{\delta f^{(d-1)}_{\uparrow u_1, \uparrow u_2, ... \uparrow u_{d-1}}}$$

   which gives the piecewise linear copula.

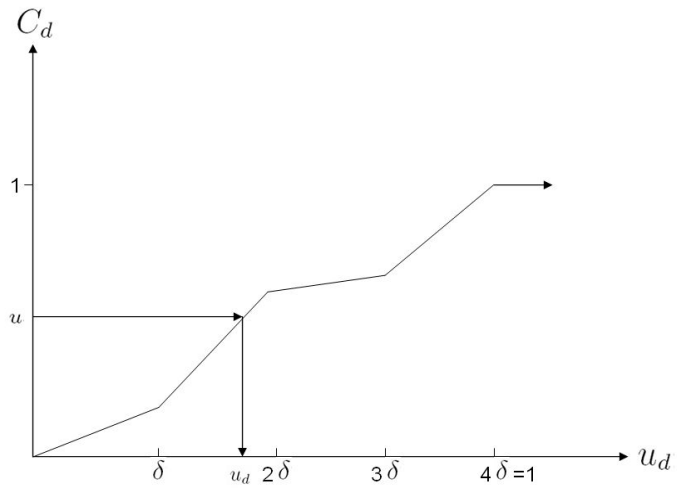

Figure 7.15: The generation of a random variable u_d from a conditioned CDF

To this end, for each $d = 2, ..., D$, a random number u is generated from the distribution $U(0, 1]$, and $u = C_d(u_d|u_1, ..., u_{d-1})$ is solved for u_d.

For the u_d we obtain the following formulas:

$$u_2 =\downarrow u_2\delta + \frac{u - \delta^{D-1}\sum_{j_2=1}^{\downarrow u_2} f^{(2)}_{\uparrow u_1,j_2}}{\delta^{D-2} f^{(2)}_{\uparrow u_1,\uparrow u_2}} \tag{7.9}$$

where $\downarrow u_2$ is the smallest integer in $\{0, ..., K-1\}$ for which

$$uK^{D-1} \leq \sum_{j_2=1}^{\downarrow u_2+1} f^{(2)}_{\uparrow u_1,j_2} \tag{7.10}$$

holds. For $d = 3, ..., D$, the formula is

$$u_d =\downarrow u_d\delta + \delta\frac{uf^{(d-1)}_{\uparrow u_1,...,\uparrow u_{d-1}} - \sum_{j_d=1}^{\downarrow u_d} f^{(d)}_{\uparrow u_1,...,\uparrow u_{d-1},j_d}}{f^{(d)}_{\uparrow u_1,...,\uparrow u_d}} \tag{7.11}$$

where $\downarrow u_d$ is the smallest integer in $\{0, ..., K-1\}$ for which

$$uf^{(d-1)}_{\uparrow u_1,...,\uparrow u_{d-1}} \leq \sum_{j_d=1}^{\downarrow u_d+1} f^{(d)}_{\uparrow u_1,...,\uparrow u_{d-1},j_d} \tag{7.12}$$

holds.

If the random vector $\mathbf{u}^{(\mathrm{gen})}$ is generated with these formulas, the calculation of the sums in (7.9) and (7.11) consist in $\mathrm{O}(DK)$ steps for calculating the sum with sparse storage organization; the same holds for the search of $\downarrow u_d$ with (7.10) and (7.12), for each dimension d. Hence this needs computing time of order $\mathrm{O}(D^2K)$.

This can be reduced to $\mathrm{O}(D^2 \log K)$ as follows, which is worthwile for large K. For example, if K=1000, the number of steps is about $10D$ instead of $500D$, in average. In a setup phase of the generation algorithm, the cumulative sums

$$s^{(d)}_{j_1,\ldots,j_d} = \sum_{j=1}^{j_d} f^{(d)}_{j_1,\ldots,j_{d-1},j}, \tag{7.13}$$

$$(j_1, \ldots, j_{d-1}) \in \{1, \ldots, K\}^{d-1}, \tag{7.14}$$

$$j_d = 0, \ldots, K, \quad d = 2, \ldots, D, \tag{7.15}$$

are calculated. In the formulas, the sums are replaced by a single element of the array s, and the $\downarrow u_d$ are determined with binary search. So we found

Theorem 7.2.3 *A random vector $\mathbf{u}^{(gen)}$ can be generated in $O(D^2 \log K)$ time.*

It must be remarked that the price for the logarithmic time complexity is a high space need for the s array. But in the next subsection we propose a tailored data structure for storing the f- and s-arrays which realizes the favourable space complexity $\mathrm{O}(D^2 n)$ and provides fast access.

Now we discuss the transformation of a random vector $\mathbf{u}^{(\mathrm{gen})}$ into a random vector $\mathbf{z}^{(\mathrm{gen})}$ with the original distribution as defined by the sample.

For the transformation, the elements of $\mathbf{u}^{(\mathrm{gen})}$ are transformed with the inverses of the marginal distribution functions of the sample. We propose three different approaches:

1. Using the estimated empirical distribution functions (7.3). Here one can obtain only values which occur in the sample. This can be sensible for integer random variables, in particular.

2. Using a linear interpolation. The flat steps of the step distribution function are replaced by straight lines above them with positive slope. Here one can obtain values which lie between zero and the largest value of the sample - different highest and lowest values can be defined similarly.

3. Using fitted standard distributions if feasible, where feasible means: The empirical marginal distribution (7.3) of the sample is similar to a standard distribution function, and this standard distribution function can be inverted in a sufficiently simple manner.

Method 1. The elements of $\mathbf{z}^{(\mathrm{gen})}$ are defined by

$$z_{d,l}^{(\mathrm{gen})} = z_{d,(i)}, \quad l = 1, ..., n, \quad d = 1, ..., D, \tag{7.16}$$

where $i = \max\{1, \lceil u_{d,l} n \rceil\}$.

Method 2. The elements of $\mathbf{z}^{(\mathrm{gen})}$ are defined by

$$\begin{aligned} & z_{d,l}^{(\mathrm{gen})} = \\ & \quad \begin{cases} u_{d,l} n z_{d,(1)} & i = 0 \\ z_{d,(i)} + (u_{d,l} n - i)(z_{d,(i+1)} - z_{d,(i)}) & i > 0 \end{cases} \\ & \quad \text{where} \quad i = \max\{1, \lceil u_{d,l} n \rceil\} - 1, \\ & \quad l = 1, ..., n, \quad d = 1, ..., D. \end{aligned} \tag{7.17}$$

Method 3. For a fixed $d \in \{1, ..., D\}$, the sample $z_{d,1}, ..., z_{d,D}$ is fitted to a suitable standard distribution with invertible distribution function, say $F^{(\mathrm{standard})}$, and the random variates are transformed as follows:

$$z_{d,l}^{(\mathrm{gen})} = F^{(\mathrm{standard})-1}(u_{d,l}), \quad l = 1, ..., n. \tag{7.18}$$

A Taylored Data Structure

When arrays or even sparse matrices are used for storing the array elements $f_{j_1,...,j_d}^{(d)}$ and their cumulative sums $s_{j_1,...,j_d}^{(d)}$, the applicability of our method is restricted with respect to the precision number K and the dimension D of the random vectors. Something like $K \leq 1000$ and $D \leq 3$, or $K \leq 30$ and $D \leq 6$ must be observed with full arrays, and with sparse matrices in MATLAB, $K \leq 1000$ and $D \leq 6$, or $K \leq 30$ and $D \leq 12$. Therefore we devised a hash-based data structure which realizes the more favourable space and time complexity and therefore makes bigger K and D possible.

We tried our program with $K = 4000$ and $D = 4$, with $K = 1000$ and $D = 40$, and with $K = 100$ and $D = 100$, for example. Moreover, the algorithm became much faster. For big values K, we observed 30...300 times shorter CPU times for the setup phase and 24...90 times faster generation of random vectors, compared with a MATLAB program which relies on sparse matrices.

We only sketch the basic ideas of the data structure which includes many details and is quite tricky. In a first phase, the $f^{(d)}$-values are included one after the other into a hash table. The hash address depends on d and on the index tuple $(j_1, ..., j_{d-1})$.

In a second phase, the data structure is reorganized to allocate sequentially the cumulative sums $s_{j_1,...,j_d}^{(d)}$ for given $(j_1, ..., j_{d-1})$. This allows for

binary search as will be seen in the sequel. After that, the array elements $f^{(d)}_{j_1,...,j_d}$ and $s^{(d)}_{j_1,...,j_d}$ are accessed as follows.

1. Given d and $(j_1, ..., j_d)$, an entry is searched in the hash table. If none is found, the $f^{(d)}$-element or the $s^{(d)}$-element is 0. Otherwise, in the entry are two pointers, $begin(d, j_1, ..., j_{d-1})$ and $end(d, j_1, ..., j_{d-1})$ which point to triples $(j_d, f^{(d)}_{j_1,...,j_d}, s^{(d)}_{j_1,...,j_d})$ in a list.

2. In the list, between the pointers, all positive f-values for the given d and $(j_1, ..., j_{d-1})$ are stored. The triples are sorted according to increasing j_d which includes increasing cumulative sums $s^{(d)}_{j_1,...,j_d}$. If there is no triple for the given j_d-value, the array element $f^{(d)}_{j_1,...,j_d}$ is 0.

When the triples are searched linearly between the pointers, the access time is $O(K)$ plus the access time to find the pointers in the hash table which is $O(D)$. With binary search, the access time is only $O(logK) + O(D)$.

Due to the sparsity which is often high-grade, there are generally only a couple of $f^{(d)}_{j_1,...,j_d} > 0$, given d and $(j_1, ..., j_{d-1})$. Therefore quite often binary searching the triples is not better, but sequential search is faster. Therefore we use sequential search in the MATLAB program Strelen (2007).

Time Series

The technique for random vectors can be applied for time series as follows: Consider a stationary time series $\mathbf{t}_i$, $i = 1, ..., n + m - 1$, of D'-dimensional random vectors, $\mathbf{t}_i = (t_{1,i}, ..., t_{D',i})$. A moving window with m vectors $\mathbf{t}_{i-m+1}, \mathbf{t}_{i-m+2}, ..., \mathbf{t}_i$ is taken as sample vectors $\mathbf{z}_i = (t_{i-m+1,1}, ..., t_{i-m+1,D'}$, $t_{i-m+2,1}, ..., t_{i-m+2,D'}$, $t_{i,1}, ..., t_{i,D'})$, $i = 1, ..., n$, hence $D = mD'$ is the dimensionality of the random vectors $\mathbf{z}_i$. With this sample, the marginal distributions and the copula are estimated as described in subsection 7.2.1. The reader may realize that there are only D' different marginal distributions.

The idea of this is as follows. The dependency between all $\mathbf{t}_i$ may be defined completely between two succeeding vectors, for example $\mathbf{t}_i = \alpha\mathbf{t}_{i-1} + (1 - \alpha)\mathbf{x}_i$, where $0 < \alpha < 1$, and the $\mathbf{x}_i$ are independent random vectors. In fact, $\mathbf{t}_i$ and $\mathbf{t}_{i+\delta}$ are dependent for $1 \leq \delta$, but this dependency is completely considered with window width $m = 2$. But the dependency between the $\mathbf{t}_i$ may not be defined completely between two succeeding vectors, for example $\mathbf{t}_i = \alpha\mathbf{t}_{i-1} + \beta\mathbf{t}_{i-2} + (1-\alpha-\beta)\mathbf{x}_i$ where $0 < \alpha, \beta, \alpha+\beta < 1$. Here the window width m must be greater than 2 in order to cover the complete dependency.

Remark. There is one exception from this. Namely when the accuracy K and the sample size n are equal, the complete dependency is considered automatically even with window width $m = 2$. The reason is as follows. When the random vectors $\mathbf{t}^{(gen)}_i$ are generated under these circumstances,

the subspaces appear in the same order as in the original given sample.

The generation of a time series $\mathbf{t}_i^{(\mathrm{gen})}, i = 1, 2, ...,$ is different. In each generation step i, only the last D' elements are newly generated, and not the whole vector $\mathbf{z}_i^{(\mathrm{gen})}$. The first $(m-1)D'$ elements for the new $\mathbf{z}_i^{(\mathrm{gen})}$ are taken from $\mathbf{z}_{i-1}^{(\mathrm{gen})}$ instead, namely its last $(m-1)D'$ elements.

$$\mathbf{z}_{i-1}^{(\mathrm{gen})} = \begin{pmatrix} \mathbf{t}_{i-1}^{(\mathrm{gen})} \\ \mathbf{t}_{i}^{(\mathrm{gen})} \\ \cdots \\ \mathbf{t}_{i+m-3}^{(\mathrm{gen})} \\ \mathbf{t}_{i+m-2}^{(\mathrm{gen})} \end{pmatrix} \begin{matrix} \nearrow \\ \cdots \\ \nearrow \\ \nearrow \end{matrix} \begin{pmatrix} \mathbf{t}_{i}^{(\mathrm{gen})} \\ \mathbf{t}_{i+1}^{(\mathrm{gen})} \\ \cdots \\ \mathbf{t}_{i+m-2}^{(\mathrm{gen})} \\ \mathbf{t}_{i+m-1}^{(\mathrm{gen})} \end{pmatrix} = \mathbf{z}_{i}^{(\mathrm{gen})}$$

The same holds for the $\mathbf{u}_i^{(\mathrm{gen})}$-vectors. The last D elements of the generated $\mathbf{z}_i^{(gen)}$ series are the desired generated time series $\mathbf{t}_i^{(gen)}$, $i = 1, 2, ...$

The first vector $\mathbf{z}_1^{(\mathrm{gen})}$ must be initialized somehow, since no older random vector is available. For this purpose, the whole vector can be generated. Generally, there is a transient phase in the beginning, therefore, some generated vectors should be skipped.

It must be remarked that for this generation method of time series, two subtle problems must be solved:

1. All parts $\mathbf{t}_j$ of the vectors $\mathbf{z}_i$ must have the same empirical marginal distributions.

2. For each $\mathbf{z}_i = (\mathbf{t}_i, ..., \mathbf{t}_{i+m-1})$, there must be another vectors $\mathbf{z}_j$ with $(\mathbf{t}_{i+1}, ..., \mathbf{t}_{i+m-1})$ in the lower places, like $\mathbf{z}_j = (\mathbf{t}_i, ...)$, $\mathbf{z}_j = (..., \mathbf{t}_i, ...)$.

Both postulations are not immediately true and must be forced explicitly. If they are not fulfilled, the generation algorithm may run into dead ends. In our MATLAB program Strelen (2007), these problems are solved. The implementation of the same empirical marginal distributions is not difficult. In the program, we omit the second problem as follows: In $\mathbf{z}_i = (\mathbf{t}_i, ..., \mathbf{t}_{i+m-1})$, $i = 1, ..., n$, each vector $\mathbf{t}_j$ with $n \leq j$ is replaced with $\mathbf{t}_{j-n}$.

7.2.3 Examples

We present numerical examples where we apply our technique which utilizes empirical real copulas. In all but the second example, random vectors $\mathbf{z}_i$ are

generated as samples with known distributions and dependence structure. We call these samples as artificial samples. In the second example we consider measured IP traffic as data-sample.

We determined the empirical marginal distributions and the copula with the methods described in subsection 7.2.1 and after the multivariate distribution is fitted, we generate random vectors $\mathbf{z}_i^{(\mathrm{gen})}$ as described in subsection 7.2.2. We verify the accuracy by comparing the statistical properties of the original sample and the generated sample, and visually, with scatter diagrams which again compare the original sample and the generated one. Scatter diagrams show clearly if there are regions in the D-dimensional space in which no original sample points are present, and if the same holds for the generated sample. Moreover, they give some visual impression of the frequency distribution of sample points and generated points.

Statistics and diagrams were calculated for both, the original sample $\mathbf{z}_i$ and the sample $\mathbf{z}_i^{(\mathrm{gen})}$ generated with the new technique. The statistics are means, coefficients of variation, correlations of the $z_{d,i}$, the latter between $z_{d,i}$ and $z_{d',i}$, $d \neq d'$, and correspondingly of the generated random vectors u_i. Moreover, we calculate the MAR, LSS and KSS statistics described in section 4.5.

We calculated the relative differences of means of the original and generated samples, the relative differences of the coefficients of variation of the original and generated samples, and relative differences of the correlations of the original and generated samples. We give the maximum of the absolute values of these differences, the *maximum statistics difference*.

Each generation of random vectors was repeated independently six times. We present the interval of the observed maximum statistics differences. In many examples, we found that the generated random vectors seem to have very similar statistical properties, compared to the samples. The examples were calculated with the MATLAB program Strelen (2007). The program can be adapted easily to different samples.

Example 1

We consider a multivariate distribution with dimension $D = 2$. The random variables of the distribution $Z_{1,i}$ have a Weibull(3,1) distribution, and the random variables $Z_{2,i}$ are correlated according to the formula $Z_{2,i} = Z_{1,i}(1 + Y_i)$, $i = 1, ..., n$. All $Z_{1,i}$ and Y_i are independent random variables. The Y_i are $U(0, 1)$-distributed.

We consider a sample size is $n = 4000$, and accuracy of $K = 1000$. After fitting a multivariate distribution utilizing an empirical copula, random variables $\mathbf{u}_i^{(\mathrm{gen})}$-vectors can be generated by equations (7.9) and (7.10).

As inverse transformation we apply method 2 of subsection 7.2.2 with the formula (7.17) and method 3 with the formula (7.18). We generate $\mathbf{z}^{(\text{gen})}$ using these two methods: Method 2: the interpolated empirical distribution function, and method 3 with the distribution Weibull(3.06, 0.983) for dimension 1 and Weibull (2.65, 1.48) for dimension 2.

For 4000 generated random vectors, the maximum statistics difference when considering method 2 is 0.003 ± 0.002. Where as the maximum statistics difference when considering method 3 is 0.022 ± 0.003. We get obviously better results when using method 2. The detailed statistical properties of the two fittings is shown in table 7.3

Table 7.3: Results summary of the original sample, of a fitted distribution utilizing real empirical copula

	Original	**multivariate Distr. 1**	**multivariate Distr. 2**
correlation coef	0.872	0.868	0.878
mean	0.889	0.898	0.887
Variation coeff.	0.367	0.358	0.348
minimum	0.0581	0.0723	0.078
maximum	2.16	2.13	2.024
MAR	-	0.01	0.019
LSS	-	0.0001	0.0005
KSS	-	0.018	0.027
KS rejected?	-	No	yes

The scatter diagrams of the sample and the generated points, figures 7.16 and 7.17, are similar for both inverse transformation methods. They indicate that there are obviously regions where no points can be, and that these regions are observed by the generated points quite accurate.

Example 2: Measured IP-Data

We consider data from Klemm et al. (2002). The sample vectors are $z_{1,i} = a_i,\ z_{2,i} = b_i,\ z_{3,i} = a_{i+1},\ z_{4,i} = b_{i+1},\ i = 1, ..., n-1$, and $z_{1,n} = a_n,\ z_{2,n} = b_n,\ z_{3,n} = a_1,\ z_{4,n} = b_1,\ i = 1, ..., n-1$, where a_i and b_i are observed inter-arrival times and packet sizes, respectively.

The values of $z_{3,n}$ and $z_{4,n}$ are a little strange; this setting is according to the remark 2 at the end of subsection time series.

Sample size $n = 4000$. The $\mathbf{u}_i^{(\text{gen})}$-vectors where generated with formulas (7.9) - (7.12). As inverse transformation we apply method 1 with the formula

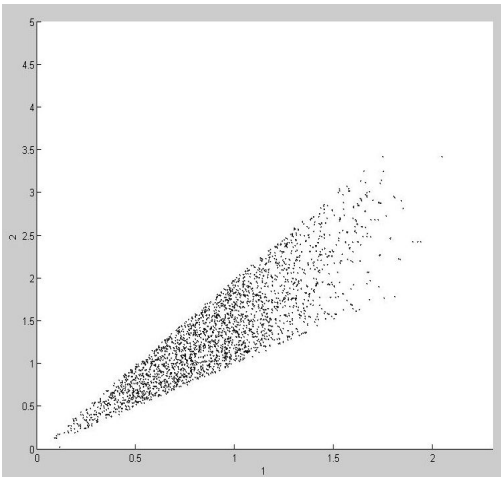

Figure 7.16: Original Sample

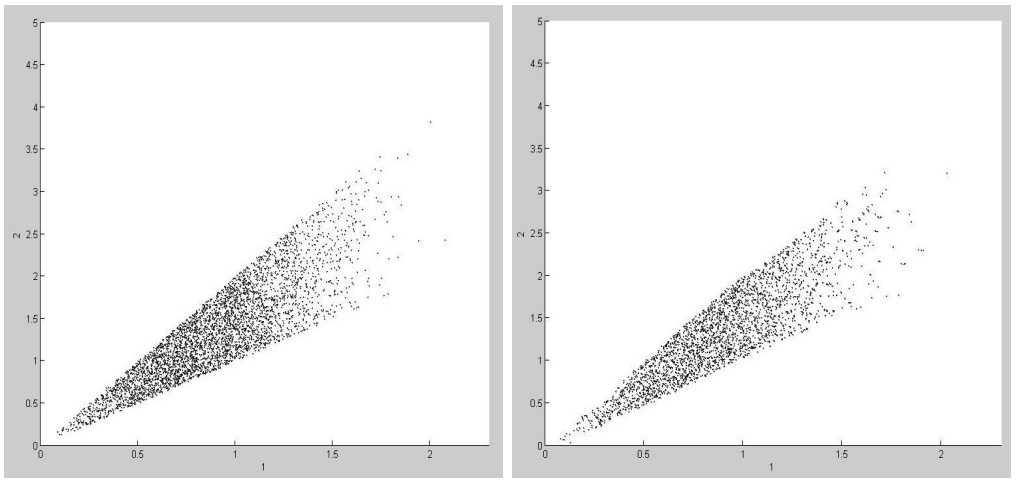

Figure 7.17: Left: The Generated Points from method 2. Right: The Generated Points from method 3

(7.16), the empirical distribution function, for the integer-valued packet sizes, and method 2 with the formula (7.17), the interpolated empirical distribution function, for the interarrival times.

For the accuracy $K = 1000$, random vectors were generated with maximum statistics difference 0.028 ± 0.016. The results summary of the fitting algorithm is shown in table 7.4.

Table 7.4: Results summary of fitting a distribution utilizing real empirical copula to IP-data

	Original	**multivariate Distr. 1**
corr. coef.	[-0.12, 0.09, -0.18, 0.25]	[-0.09, 0.077, -0.16, 0.24]
mean	[0.0114, 312.2828]	[0.0116 310.76]
variation coeff.	[1.3494, 1.0523]	[1.3791, 1.0715]
minimum	[0.0001, 40]	[0.0001, 40]
maximum	[0.1, 1500]	[0.1, 1500]
MAR	-	[0.0003, 6.304]
LSS	-	[0.0000, 680.716]
KSS	-	[0.0130, 0.0115]
KS rejected?	-	No

The correlation coefficients (corr. coef.) pointed in table 7.4 are those between the random variables $(z_{1,i}, z_{2,i})$, $(z_{1,i}, z_{3,i})$, $(z_{2,i}, z_{3,i})$, and $(z_{2,i}, z_{4,i})$, respectively.

The scatter diagrams, figures 7.18, 7.19, and 7.20, show a very irregular dependency structure. The comparison of the sample and the generated points indicate here also good accuracy.

Example 3

We consider the stochastic process where A_1 and all Y_k are $U(0,1)$-distributed, $A_{k+1} = 0.5(1-4(A_k-0.5)^2)+0.5Y_k,\ k = 2, ..., n0+n$, and the sample vectors are $Z_{1,i} = A_{i+n0},\ Z_{2,i} = A_{i+n0+1},\ i = 1, ..., n-1$, and $Z_{1,n} = A_{n+n0},\ Z_{2,n} = Z_{1,1}$. The first random variables A_1, A_2, ... are not stationary; this is why we skip $n0 > 0$ realizations, actually $n0 = 100$. We hope that the stochastic process is then nearly stationary. Here, the dimension is $D = 2$.

Sample size is $n = 4000$. For the accuracy $K = 1000$, 64000 random vectors were generated in 5.1 seconds CPU time with maximum statistics differences 0.0024 ± 0.0012. For the accuracy $K = 4000$, 64000 random

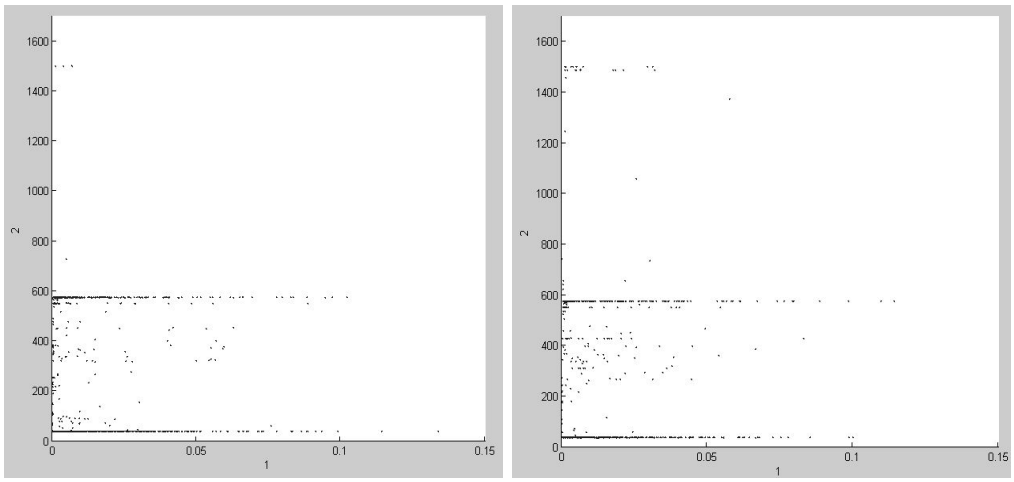

Figure 7.18: Left: Original Sample. Right: The Generated Points. Dimensions 1 and 2

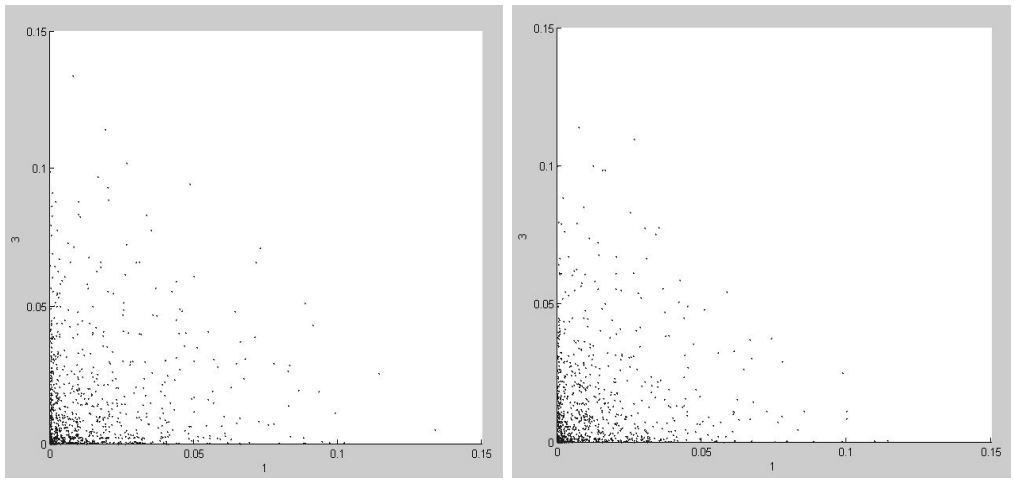

Figure 7.19: Left: Original Sample. Right: The Generated Points. Dimensions 1 and 3

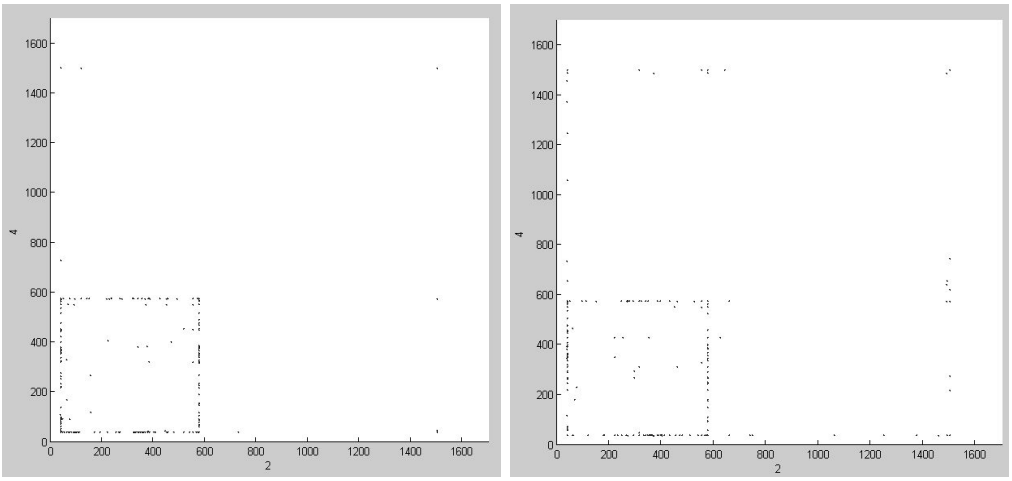

Figure 7.20: Left: Original Sample. Right: The Generated Points. Dimensions 2 and 4

vectors were generated in 7.8 seconds CPU time with maximum statistics differences 0.0002 ± 0.00002, which is very accurate.

Example 4

Here we consider random vectors with high dimensions D, namely $D = 5, D = 40$, and $D = 100$. $Z_{1,i} = Y_{1,i}$, $Z_{d,i} = Z_{\lceil d/2 \rceil,i}(1 - Z_{\lceil d/2 \rceil,i}) + Y_{d,i}$, $d = 2, ..., D$, $i = 1, ..., n$, and the $Y_{d,i}$ are independent and $U(0, 1)$-distributed.

With sample size $n = 4000$, dimension $D = 5$, and accuracy $K = 4000$, 64000 random vectors were generated in 29 seconds CPU time with maximum statistics differences of 0.007 ± 0.002. With sample size $n = 1000$, dimension $D = 40$, and accuracy $K = 1000$, 16000 random vectors were generated in 74 seconds CPU time with maximum statistics differences 0.028 ± 0.004. With sample size $n = 1000$, dimension $D = 100$, and accuracy $K = 100$, 16000 random vectors were generated in 210 seconds CPU time with maximum statistics differences 0.031 ± 0.0035.

Chapter 8

Conclusions and Further Work

Input modeling for simulation is a topic of current research. Especially when talking about modeling dependencies. Many input models used in professional modeling software can not model dependencies. Nevertheless, there are input modeling approaches that can capture dependencies lying among the data. Each of these approaches has its strong and weak points.

The relatively early approach of autoregressive to anything (ARTA) of Cario and Nelson (1996) is thought in principle for the generation of stochastic processes, provided that modeler knows which linear correlation and distribution must the output process have. This approach is extended to generate vector stochastic processes and called vector autoregressive to anything (VARTA) described in Biller and Nelson (2003). The NORTA principle is again used to generate vectors of random variables. The vectors are independent, but the random variables lying in each vector are linearly correlated. This approach is called the normal to anything (NORTA) (Cario and Nelson (1997)).

For the purpose of simulation input modeling, the modeler has often only a sample that must be fitted to a stochastic process. The ARTA approach has been modified so that an ARTA process can be fitted to such a sample. However, there is no technique for the VARTA or NORTA processes to given samples. Moreover, these approaches can model only linear dependencies, and they still need to be extended to consider heavy-tailed processes.

Another modern approach depends on the batch markovian arrival processes (BMAP) and the software utilizing this approach and developed by Klemm et al. (2002) is called IP2BMAP, because it fits a BMAP process to IP data. This approach shows high accuracy in capturing the most significant statistical moments of IP data. However, the approach is thought only for this purpose and therefore can not be used for example for the purpose of modeling tri-dimensional random vectors. Other weak points of this

approach are described in chapter 6.

We describe in the thesis three main approaches, that try to avoid most of the disadvantages of the previous approaches. So we develop the non-Gaussian autoregressive (nGAR) approach in chapter 4 for generally distributed marginal distributions, we extend Yule-Walker method so that non-linear correlations can be modeled in chapter 5, and we develop an approach that depends on the empirical copulas in chapter 7 which considers the complete dependence structure.

The nGAR approach is a flexible approach. It can make use of already existing approaches like the MLE, it can model non-linearly dependent data, and can generate heavy tailed processes. The approach depends on capturing the dependencies among the data and their distribution separately. We show in chapter 4 how this approach works very well for some examples. However, the approach can model only autoregressive data. We show in an example that this approach is not appropriate if the data can not be modeled as an autoregressive process.

The analytic approach of the extended Yule-Walker method described in chapter 5. This approach, like the nGAR approach, can be used only for fitting autoregressive models. Moreover, the nGAR approach and the approach of the extended Yule-Walker method depends on having assumptions about the kind of dependency, when non-linearity is assumed. In other words, the type of model must be known.

The nGAR approach and the approach of the extended Yule-Walker method are used to estimate the parameters of this known or assumed model, or they can be used to check which type of models fits the available data better. For example, this approach can answer the question, when non-linear dependencies are suggested: which parameters must have a first degree polynomial, so that it fits the data best. Or they can answer the question: does a first or second degree polynomial fits the data better?

If absolutely no idea about the model to be fitted is available, the approach which depends on copulas and which is described in chapter 7 can be used. In this approach, the dependencies among the data can be captured by means of the empirical copulas. Capturing the dependencies using other classes of theoretical copulas is also possible, but again leads to the restriction of assuming a specific type of models that can model the dependencies. The empirical copulas can capture any type of dependencies without having any idea about them.

In a future work, modifying the nGAR approach so that the process fitted satisfies special conditions can be done. For example, the variates generated from a fitted nGAR model might be all positive or negative.

The extended Yule-Walker method might be applied for other types of

models than those mentioned in this thesis. However, non-linear models that are common or useful in the real life are not known. There has been up to now less research about non-linear models that might be interesting for simulation input modeling.

The method which utilizes copulas has proved to be powerful. Copulas are common in finance and insurance. They should be useful for simulation as well.

Bibliography

Aas, K.: Modelling the dependence structure of financial assets: A survey of four copulas. by Norwegian Computing Center, Oslo, Norway (2004).

Baeck, T.; Fogel, D.; Z., M.: *Handbook of evolutionary computation*. Bristol, New York: Oxford University Press (1997).

Biller, B.; Ghosh, S.: Dependence modeling for stochastic simulation. In: *Winter Simulation Conference*, pages 153–161 (2004).

Biller, B.; Nelson, B. L.: Advanced input modeling: Parameter estimation for arta processes. In: *Winter Simulation Conference*, pages 255–262 (2002).

Biller, B.; Nelson, B. L.: Modeling and generating multivariate time-series input processes using a vector autoregressive technique. *ACM Transactions on Modeling and Computer Simulation*, 13, 3, (2003), 211–237.

Blum, P.; Dias, A.; Embrechts, P.: The art of dependence modelling: the latest advances in correlation analysis (2002).

Box, G. E. P.; Jenkins, G. M.: *Time Series Analysis, Forecasting and Control*. Holden-Day, Inc. (1976).

Cario, M. C.; Nelson, B. L.: Autoregressive to anything—time-series input processes for simulation. *Operations Research Letters*, 19, 2, (1996), 51–58.

Cario, M. C.; Nelson, B. L.: Modeling and generating random vectors with arbitrary marginal distributions and correlation matrix. Department of Industrial Engineering and Management Sciences (1997).

Chatfield, C.: *The Analysis of Time Series: An Introduction, 5th Edition*. Chapman and Hall, New York (1996).

Chipperfield, A.; Fleming, P.; Pohlheim, H.; Fonseca, C.: Genetic algorithm toolbox user's guide. ACSE Research Report No. 512, University of Sheffield (1994).

De Hoon, M. J. L.; Van der Hagen, T. H. J. J.; Schoonewelle, H.; Dam, H. V.: Why yule-walker should not be used for autoregressive modelling. *Annals of Nuclear Energy*, 23, 15, (1996), 1219–1228.

Embrechts, P.; Lindskog, F.; McNeil, A.: Modelling dependence with copulas and applications to risk management. Report: Department of Mathematics, ETHZ, Zrich (2001).

Fishman, G. S.: *Principles of Discrete Event Simulation*. John Wiley and Sons, Inc. (1978).

Ghosh, S.; Henderson, S. G.: Chessboard distributions. In: *Winter Simulation Conference*, pages 385–393 (2001).

Ghosh, S.; Henderson, S. G.: Chessboard distributions and random vectors with specified marginals and covariance matrix. *Operations Research Letters*, 50, 5, (2002a), 820–834.

Ghosh, S.; Henderson, S. G.: Properties of the norta method in higher dimensions. In: *Winter Simulation Conference*, pages 263–269 (2002b).

Jagerman, D.; Melamed, B.: The transition and autocorrelation structure of tes processes; part i: General theory. *Stochastic Models*, 8, 2, (1992a), 193–219.

Jagerman, D.; Melamed, B.: The transition and autocorrelation structure of tes processes; part ii: Special cases. *Stochastic Models*, 8, 3, (1992b), 499–527.

Johnson, M. E.: *Multivariate Statistical Simulation*. John Wiley and Sons, Inc., New York, NY, USA (1987).

Klemm, A.; Lindemann, C.; Lohmann, M.: Traffic modeling of ip networks using the batch markovian arrival process. In: *Computer Performance Evaluation / TOOLS*, pages 92–110 (2002).

Klemm, A.; Lindemann, C.; Lohmann, M.: Modeling ip traffic using the batch markovian arrival process. *Performance Evaluation*, 54, 2, (2003), 149–173.

Law, A. M.; Kelton, D. W.: *Simulation Modeling and Analysis, 3rd edition*. New York: McGraw-Hill (2000).

Lehn, J.; Wegmann, H.: *Einfuehrung in die Statistik*. Teubner, B.G. Teubner Stuttgart (1992).

Li, H.: Tail dependence comparison of survival marshall-olkin copulas. Technical report: Department of Mathematics and Department of Statistics Washington State University (2006).

Livny, M.; Melamed, B.; Tsiolis, A. K.: The impact of autocorrelation on queuing systems. *Management Science*, 39, 3, (1993), 322–339.

Lucantoni, D. M.; Gagan, L. C.; Whitt, W.: The transient bmap/g/1 queue. *Stochastic Models*, 10, 1, (1994), 145–182.

MathWork, T.: Matlab help. The Language of Technical Computing. Release 12.1 (2001).

Miket, M. J.: Chi-square procedures. Lecture notes: the Practice of Statistics, Chapter 11. At the department of Mathematics and Statistics, University of Saskatchewan. http://math.usask.ca/~miket/Sullivan_PP/Chapter_11/sec11_3.ppt (2006).

Mikosch, T.: Copulas: Tales and facts. Technical Report: Laboratory of Actuarial Mathematics, University of Copenhagen. Will appear in the journal *Extremes* (2005).

Nassaj, F.; Strelen, J. C.: Dependence input modeling with the help of non-Gaussian AR models and genetic algorithms. In: *Modelling and Simulation 2005, Proceedings of the European Simulation and Modelling Conference*, pages 146–153, Eurosis-ETI, Porto (2005).

Nassaj, F.; Strelen, J. C.: Generating simulation input data with approximate copulas. In: *Modelling and Simulation 2006, Proceedings of the European Simulation and Modelling Conference*, pages 88–93, Eurosis-ETI, Toulouse (2006).

Nelsen, R. B.: *An Introduction to Copulas*. Springer Verlag (1998).

Neuts, M. F.: *Structured Stochastic Matrices of M/G/1 Type and Their Applications*. Marcel Dekker, New York (1989).

Neuts, M. F.: *Matrix-Geometric Solutions in Stochastic Models*. Courier Dover Publications (1995).

of Aveiro, U.: Trace of ip packets, captured at the internet access of university of aveiro. http://www.av.it.pt/nmc/traces/UAcapture19.zip (2006).

Priestley, M. B.: Spectral analysis and time series. Academic Press, London. (1982).

Riska, A.: Aggregate matrix-analytic techniques and their applications. A dissertation presented at the computer science department, college of William & Mary (2002).

Sklar, A.: Fonctions de rpartition n dimensions et leures marges. *de l'Institut de Statistique de L'Universit de Paris*, 8, (1959), 229–231.

Strelen, J. C.: The genetic algorithm is useful to fitting input probability distributions for simulation models. In: *Business and Industry Symposium - ASTC*, pages 8–13 (2003).

Strelen, J. C.: Generating random vectors with copulas - MATLAB program pwlCopula_D_with_hashing (2007).

Strelen, J. C.; Nassaj, F.: Analysis and generation of random vectors with copulas. In: *Winter Simulation Conference* (2007).

van den Goorbergh, R. W.; Genest, C.; Werker, B. J.: Bivariate option pricing using dynamic copula models. *Insurance: Mathematics and Economics*, 37, 1, (2005), 101–114.

West, R. W.; Ogden, R. T.: The duration for eruptions of the old faithful geyser in yellowstone national park. http://www.amstat.org/publications/jse/v6n3/west.html (1998).