Fitting Simulation Input Models for Correlated Traffic Data

Dissertation

zur Erlangung des Grades eines

Doktors der Naturwissenschaften

der Technischen Universität Dortmund

an der Fakultät für Informatik

von

Jan Kriege

Dortmund

2012

Jan Kriege
Lehrstuhl IV - Modellierung und Simulation
Fakultät für Informatik
Technische Universität Dortmund
Martin-Schmeißer-Weg 18
44227 Dortmund

Tag der mündlichen Prüfung: 30.01.2012

Dekanin	**Prof. Dr. Gabriele Kern-Isberner**
Gutachter	**Prof. Dr. Peter Buchholz** (TU Dortmund, Fakultät für Informatik) **Prof. Dr. Claus Weihs** (TU Dortmund, Fakultät Statistik)

Abstract

The adequate representation of input models is an important step in building valid simulation models. Modeling independent and identically distributed data is well established in simulation, but for some application areas like computer and communication networks it is known, that the assumption of independent and identically distributed data is violated in practice and that for example interarrival times or packet sizes exhibit autocorrelation over a large number of lags. Moreover, it is known that negligence of these correlations can result in a serious loss of validity of the simulation model.

Although different stochastic processes, which can model these autocorrelations, like e.g. Autoregressive-To-Anything (ARTA) processes and Markovian Arrival Processes (MAPs), have been proposed in the past and more recently fitting algorithms to set the parameters of these processes such that they resemble the behavior of observations from a real system have been developed, the integration of correlated processes into simulation models is still a challenge.

In this work ARTA processes are extended in several ways to account for the requirements when simulating models of computer and communication systems. In a first step ARTA processes are extended to use an Autoregressive Moving Average (ARMA) process instead of a pure Autoregressive (AR) base process to be able to capture a large number of autocorrelation lags, while keeping the model size small. In a second step they are enabled to use the flexible class of acyclic Phase-type distributions as marginal distribution.

To support the usage of these novel processes in simulation models a fitting algorithm is presented, software for fitting and simulating these processes is developed and the tools are integrated into the toolkit ProFiDo, which provides a complete framework for fitting and analyzing different stochastic processes.

By means of synthetically generated and real network traces it is shown that the presented stochastic processes are able to provide a good approximation of the marginal distribution as well as the correlation structure of the different traces and result in a compact process description.

Acknowledgment

Many people supported and encouraged me during the work on this thesis. First and foremost, I would like to thank my thesis advisor Prof. Dr. Peter Buchholz for his support, ideas and valuable advice during the course of this work. I owe him a debt of gratitude for providing the right balance of scientific guidance and freedom to do my research, for funding various conference trips and for introducing me to established researchers. In this regard I would like to thank Professor Dr. Miklos Telek and his group from the Technical University of Budapest, which I was allowed to visit for several weeks.

Special thanks also go to my co-advisor Prof. Dr. Claus Weihs for his hints and suggestions that helped to improve this work. Furthermore, I would like to thank Prof. Dr. Heiko Krumm and Dr. Stefan Dißmann for agreeing to serve as members of my defense committee.

I am grateful to Dr. Falko Bause for the interesting scientific discussions and his advice over the course of the years. I also want to thank the other members of the Modeling and Simulation group for the friendly (always) and productive (most of the time) atmosphere.

Last but not least, I would like to thank my family and friends for their support, and in particular Nils for proofreading parts of this work.

Contents

Chapter 1

Introduction

The performance analysis of computer and communication systems has a long history. Whenever a new system is designed or an existing system is altered one is interested in forecasting the behavior of the system in a quantitative way. Because of the increasing complexity of these systems it is not possible or feasible to run experiments with the real system to obtain performance measures, which makes use of stochastic models necessary. Depending on the performance measures of interest and on the level of detail of the model different types of stochastic models and analysis techniques are applicable. In the past these systems have for example been modeled using queueing networks or stochastic Petri nets [71]. If the model respects certain requirements, i.e. the used probability distributions are (combinations of the) exponential distribution, numerical techniques can be applied for analysis [143]. However, because of the aforementioned complexity of the systems the models can only be analyzed by simulation [104] in many cases, which does not assume any restrictions for the model. In any case these models have to represent the relevant behavior that is observed from the real system and thus, an important step in constructing the models is the definition of accurate input models for interarrival times, failure or repair times, packet sizes and other specific properties of the modeled system. Often measurements from the real system are available in form of traces and one is looking for an input model that captures characteristics of these traces.

Usually these input models are described by general distributions, implying independent and identically distributed (iid) interarrival times or packet sizes. The theory of fitting distributions to observations from a real system is well understood [104, 158] and several (commercial) tools like `ExpertFit` [105] or the `Arena Input Analyzer` [134] are available for this task. However, in some application areas like computer and communication systems it is known that the assumption of independent and identically distributed interarrival times is violated in practice. It has been shown that network traffic [54, 114, 130], file or disk access patterns [138, 160] and failures in software [69] or hardware systems exhibit correlation that can be observed over large periods of time. Moreover, it is known that negligence of these correlations can result in a serious loss of validity of the simulation model [109].

In the past several stochastic processes have been proposed that are able to capture these autocorrelations. Known for a long time are Autoregressive Moving Average

(ARMA) Processes [35] and Markovian Arrival Processes (MAPs) [120].

MAPs are described by a Continuous Time Markov Chain (CTMC) where some transitions indicate an arrival event and are a flexible class of stochastic processes that can be applied to capture a wide range of different stochastic behaviors. Because of their interpretation as a CTMC MAPs can be used for queueing networks that should be analyzed numerically [143] or with matrix analytical methods [120]. But, of course, models using MAPs as input models can also be analyzed by simulation [104], which has for example been done in [146].

ARMA processes specify stationary time-dependent input processes and are modeled as the weighted sum of previous observations and a random term, called innovation. A generalization of ARMA processes, called Autoregressive Integrated Moving Average (ARIMA) Process, can also be used to describe non-stationary behavior. AR(I)MA models are usually used as discrete-time processes [107]. For example in [165] the number of arrivals in a given interval is modeled by an ARIMA process. However, ARIMA models can also be used to model interarrival time processes, which has for example been done in [154]. A major drawback of ARIMA models is that they result in marginal normal distributions, which might not be adequate for many applications.

More recent approaches try to overcome this limitation of ARIMA processes by transforming the process such that it can express other marginal distributions, most notably the Autoregressive-To-Anything (ARTA) process [47], which combines an AR base process with an arbitrary marginal distribution and relies on the inversion of the cumulative distribution function of the marginal distribution. Models using ARIMA or ARTA processes can only be analyzed by simulation.

Since these stochastic processes should capture the behavior of observations that have been measured in a real system, fitting algorithms have to be developed that set the parameters of the processes such that they exhibit the desired behavior. For ARIMA processes such algorithms have been incorporated into software for statistical computing like `R` [149]. For ARTA processes and MAPs first prototype implementations have been developed only recently. Numerical methods to construct the AR base process of an ARTA model with a given marginal distribution have been proposed in [48]. A more recent approach that can fit both, AR base process and a marginal distribution from the Johnson system, is presented in [25]. Parameter fitting for MAPs is a nonlinear optimization problem, which becomes even more complex because the matrices describing the MAP are known to contain redundant information [147] and a canonical representation is only available for MAPs of order two [31]. Nevertheless, several approaches have been proposed to fit the parameters of a MAP according to a trace, which can basically be divided into two classes. Expectation Maximization (EM) algorithms like the one from [40] use the complete information of the trace to maximize the likelihood. EM algorithms have a slow convergence and are not really feasible for fitting MAPs with a high order to large traces. The second class of approaches uses some derived properties from the trace like empirical moments, joint moments or lag-k autocorrelation coefficients. These algorithms are much faster than EM algorithms, but do not use the complete information from the trace for fitting. Examples for such approaches are given in [42, 84].

Although several stochastic processes exist and fitting methods are at least available as prototype implementations, the incorporation of these processes into simula-

tion models is still a challenge [22]. Common simulation tools like Arena [94, 134], Extend [98, 99] or AutoMod [7, 139] only offer distributions as input models. But even in simulation frameworks that are dedicated to network modeling [161] there is only little support for stochastic processes. The network simulation framework OMNeT++ [78] has only built-in support for random number generation from different distributions. More elaborate traffic generators have been proposed recently, e.g. [33] generates Voice over IP traffic from real sound files and the HTTP extensions presented in [91] allow for the specification of activity periods for traffic generation, but there is no native support for stochastic processes. For other network simulators like ns-2 and its successor ns-3 [75] the picture is similar.

In summary, the situation with stochastic processes as input models for simulation is not really satisfactory. Although the modeling of correlation is considered as one of the most important issues in simulation [23], the tool support for actually using stochastic processes in simulation models is insufficient. Even though several different fitting tools exist, they all require a different handling and use different input and output formats, which makes it cumbersome to fit stochastic processes of different types and to integrate them into simulation models. These issues have partially been addressed with the development of ProFiDo [12], which is a flexible toolkit for fitting distributions and stochastic processes, that makes the different fitting tools accessible in a coherent way by establishing an XML format for process descriptions. Yet missing is the possibility for easy integration of these processes into simulation models.

These observations clearly motivate the work done in this thesis. In an empirical study the existing stochastic processes will be assessed regarding their suitability as input models in simulation. After some of the weaknesses and disadvantages have been pointed out, several extensions and improvements for the ARTA approach are proposed. These theoretical ideas result in a fitting tool for the newly proposed types of stochastic processes, which is integrated into the ProFiDo framework. To support the easy integration of stochastic processes into simulation models a module for OMNeT++ is developed, which is integrated into ProFiDo as well. In particular, the following issues will be addressed in this work.

1.1. Outline and Contribution

In the remainder of this chapter some basic concepts of probability theory are summarized that will be helpful to understand the following chapters. Chapter 2 introduces different stochastic processes that have been proposed in the past for traffic modeling, summarizes their properties and presents some of the available fitting methods. The main focus of this chapter is on ARMA Processes, ARTA Processes and MAPs, which are the most commonly used types of processes and which are relevant to the concepts developed in this thesis. In Chapter 3 the three mentioned classes of stochastic processes are assessed in an empirical study. This study points out some drawbacks of these processes to motivate the work in the following chapters. In Chapters 4, 5 and 6 ARTA processes are extended in several directions. In Chapter 4 ARTA processes are enabled to use an ARMA base process instead of an AR process. For the interesting class of Phase-type distributions [119] the ARTA approach is not feasible, since in general the inverse cumulative distribution function cannot be computed ef-

ficiently for this type of distributions. In Chapter 5 a new approach for combining Hyper-Erlang distributions with an ARMA base process is proposed, which does not rely on the computation of the inverse cumulative distribution function as the original ARTA approach does. In Chapter 6 these ideas are generalized for acyclic Phase-type distributions. Chapter 7 incorporates the ideas from Chapters 4, 5 and 6 into one algorithmic framework. In Chapter 8 another empirical study is presented that evaluates the fitting quality of the newly developed processes and compares them with existing approaches. As already mentioned before there is only little support in existing simulation tools for stochastic processes. Chapter 9 presents several tools for fitting and simulation of the proposed processes, which are integrated into the toolkit `ProFiDo` and provide a complete framework for fitting and analyzing of stochastic processes. This thesis ends with the conclusions in Chapter 10.

The main theoretical contributions of this thesis can be found in Chapters 5 and 6 where a novel approach is presented to combine acyclic Phase-type distributions with an ARMA base process to model correlated data with Phase-type distribution. Chapter 5 outlines the ideas for a special class of Phase-type distributions, namely Hyper-Erlang distributions, and Chapter 6 generalizes these ideas for acyclic Phase-type distributions. Additionally, Chapter 6 presents transformations for the PH distributions that can be applied to increase the range of autocorrelation that can be modeled by the newly developed stochastic processes. A minor contribution can be found in Chapter 4 where the ARTA approach is extended to use an ARMA base process instead of an AR process. The practical contributions of this work are presented in Chapter 9. In this chapter a fitting tool for the stochastic processes from Chapters 4-6 is presented. Additionally, a module for the simulation framework `OMNeT++` is introduced, which allows for an easy integration of stochastic processes into simulation models. Both tools are integrated into the framework `ProFiDo`. Chapter 7 links the theoretical results to the practical implementation and outlines an algorithm using the ideas from Chapters 4-6 to fit the proposed stochastic processes. The work is accompanied by two empirical studies in Chapters 3 and 8 to assess the quality of existing and the newly developed stochastic processes.

Some parts of this work have already been published before. Most of the empirical work from Chapter 3 appeared in [11]. Parts of the results from Chapter 8 are taken from [100]. An earlier version of the software described in Section 9.3 was sketched in [101].

1.2. Notations

Throughout this work the following notations will be used. Matrices and vectors will both be typeset in bold face. Matrices will usually be denoted with capital letters (e.g. $\boldsymbol{M}$), while lower-case letters are used for vectors (e.g. $\boldsymbol{v}$). $\boldsymbol{v}^T$ denotes the transposed vector $\boldsymbol{v}$. We will use $\mathbf{1}$ to describe a column vector of 1's, i.e. $(1, 1, \ldots, 1)^T$, and $\boldsymbol{I}$ to denote the identity matrix. Random variables are printed in capital letters, e.g. U for an uniformly distributed random variable. For discrete-time stochastic processes and sequences of random variables the time index is denoted as t, e.g. $\{U_t\}, t = 1, 2, \ldots$ describes a sequence of random variables with uniform distribution. A continuous-time stochastic process is denoted as $\{X(t)\}$. A list with symbols that are used frequently

throughout this thesis can be found in Appendix A for reference.

1.3. Basic Definitions

In the following a brief reminder on basic concepts of probability theory is given to introduce some ideas and definitions that are helpful to understand the approaches presented in this thesis. A more elaborate overview can be found in any textbook about probability theory and statistics like for example [155].

1.3.1. Random Variables

A random variable is a function $X : \mathcal{S} \to \mathbb{R}$ assigning a real number to each point of the sample space $\mathcal{S}$. The sample space is a set of all possible outcomes of a stochastic experiment. The outcomes themselves are called sample points. An event is a subset of the sample space, i.e. a collection of sample points. Equivalently conditions defining this subset are called an event.
The (cumulative) distribution function of the random variable X is defined as

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

and has the following properties:

- $0 \leq F(x) \leq 1 \quad \forall x$,
- $F(x)$ is nondecreasing,
- $\lim_{x\to\infty} F(x) = 1$, $\lim_{x\to-\infty} F(x) = 0$.

A random variable X can be either discrete or continuous. In the discrete case X can take on at most a countable number of values $x_1, x_2, \ldots$. The probability mass function given by $p(x_i) = P(X = x_i)$ denotes the probability that a discrete random variable X has the value x_i. Of course, $\sum_{i=1}^{\infty} p(x_i) = 1$ has to hold. The distribution function for a discrete random variable can be calculated using the probability mass function:

$$F(x) = \sum_{x_i \leq x} p(x_i) \qquad \text{for all } -\infty < x < \infty.$$

In the continuous case a random variable can take on an uncountably infinite number of values. Probabilities of continuous random variables are defined in terms of the probability density function $f(x)$ and for $\mathcal{B} \subseteq \mathcal{S}$ we have that $P(X \in \mathcal{B}) = \int_{\mathcal{B}} f(x)dx$ and $\int_{-\infty}^{\infty} f(x)dx = 1$. For the distribution function we have that

$$F(x) = P(X \in [-\infty, x]) = \int_{-\infty}^{x} f(y)dy \qquad \text{for all } -\infty < x < \infty.$$

The conditional probability of $\mathcal{A} \subseteq \mathcal{S}$ given $\mathcal{B} \subseteq \mathcal{S}$ is $P(\mathcal{A}|\mathcal{B}) = \frac{P(\mathcal{A}\cap\mathcal{B})}{P(\mathcal{B})}$. If the probability of event $\mathcal{A}$ does not change regardless of whether $\mathcal{B}$ has occurred or not, $\mathcal{A}$ and $\mathcal{B}$ are independent and we have that $P(\mathcal{A}|\mathcal{B}) = P(\mathcal{A})$ and $P(\mathcal{A}\cap\mathcal{B}) = P(\mathcal{A})P(\mathcal{B})$.

1.3.2. Joint Distributions of Random Variables

Considering more than one random variable simultaneously one can define the joint probability mass function in the discrete case and the joint probability density function in the continuous case, respectively.
The joint probability mass function for two discrete random variables X and Y is defined by $p(x, y) = P(X = x, Y = y)$ for all x, y. If one is concerned with more than one random variable, the probability mass function of a single variable is called marginal probability mass function. X and Y are independent if $p(x, y) = p(x)p(y)$ for all x, y where the marginal probability mass functions of X and Y are defined as $p(x) = \sum_y p(x, y)$ and $p(y) = \sum_x p(x, y)$, respectively. For continuous random variables we have

$$P(X \in \mathcal{A}, Y \in \mathcal{B}) = \int_{\mathcal{B}} \int_{\mathcal{A}} f(x, y)dxdy.$$

Here X and Y are independent if $f(x, y) = f(x)f(y)$ for all x, y and the marginal probability density functions of X and Y are defined as $f(x) = \int_{-\infty}^{\infty} f(x, y)dy$ and $f(y) = \int_{-\infty}^{\infty} f(x, y)dx$. Of course, the above definitions can be generalized for more than two random variables.

1.3.3. Properties of Random Variables

Of particular interest are some characteristic properties of a random variable, the so-called moments. The first moment is the mean or expected value, denoted by μ, which is defined as

$$\mu = \mu_1 = E[X] = \begin{cases} \sum_{i=1}^{\infty} x_i p(x_i) & \text{if } X \text{ is discrete with values } x_1, x_2, \ldots \\ \int_{-\infty}^{\infty} xf(x)dx & \text{if } X \text{ is continuous.} \end{cases}$$

The mean is a measure of central tendency of a random variable. An alternative measure for this is the median $x_{0.5}$, which is defined as the smallest value of x such that $F(x) \geq 0.5$.
Consider a random variable X and a second random variable Y that can be expressed as a function of X, i.e. $Y = \zeta(X)$. To compute $E[Y]$ one may use

$$E[Y] = E[\zeta(X)] = \begin{cases} \sum_{i=1}^{\infty} \zeta(x_i) p(x_i) & \text{if } X \text{ is discrete with values } x_1, x_2, \ldots \\ \int_{-\infty}^{\infty} \zeta(x) f(x)dx & \text{if } X \text{ is continuous.} \end{cases}$$

Of special interest is $\zeta(X) = X^k$ and for $k = 1, 2, 3, \ldots$, $\mu_k = E[X^k]$ is known as the k-th moment of the random variable X.
The mode of a random variable is the number x for that $p(x)$ (in the discrete case) or $f(x)$ (in the continuous case) reaches its maximum.
The variance of a random variable X is a measure of the dispersion of X about its mean. It is defined by $\sigma^2 = Var[X] = E[(X - \mu)^2] = E[X^2] - \mu^2$ where σ is called the standard deviation.
Further measures often used in performance analysis are the coefficient of variation defined by

$$C_X = \frac{\sigma}{E[X]}$$

and the squared coefficient of variation defined by

$$C_X^2 = \frac{Var[X]}{E[X]^2} = \frac{E[X^2]}{E[X]^2} - 1,$$

which are expressions for the variance of a random variable X relative to its average value. The range of variability of the squared coefficient of variation is a common measure to characterize the flexibility in approximating a given general distribution.
All the properties given above are properties of a single random variable. Next measures of dependence between two random variables are considered. The covariance C_{ij} is a measure of linear dependence between the random variables X_i and X_j with mean μ_i and μ_j, respectively:

$$C_{ij} = Cov[X_i, X_j] = E[(X_i - \mu_i)(X_j - \mu_j)] = E[X_i X_j] - \mu_i \mu_j.$$

If $C_{ij} = 0$ the random variables X_i and X_j are uncorrelated. If $C_{ij} > 0$ then X_i and X_j are positively correlated, which means that $X_i > \mu_i$ and $X_j > \mu_j$ as well as $X_i < \mu_i$ and $X_j < \mu_j$ tend to occur together. If X_i and X_j are negatively correlated ($C_{ij} < 0$) $X_i < \mu_i$ and $X_j > \mu_j$ as well as $X_i > \mu_i$ and $X_j < \mu_j$ tend to occur together.
Since the covariance C_{ij} is not dimensionless a better measure of dependence between X_i and X_j is the correlation

$$\rho_{ij} = Corr[X_i, X_j] = \frac{C_{ij}}{\sqrt{\sigma_i^2 \sigma_j^2}}$$

for which $-1 \leq \rho_{ij} \leq 1$ holds.

1.3.4. Stochastic Processes

In the previous sections on random variables and probability distributions we referred to a random variable X taking on some value x_1 but disregarded the time the variable took on this value. Including the notion of time leads to stochastic processes, which are families of random variables $\{X(t)\}$ defined over the same probability space. Stochastic processes can be classified depending on the state space and the time parameter of random variables $X(t)$ for different values of t.
If the number of values in the state space of $X(t)$ is finite or countable, we have a discrete-state process (also called chain), otherwise a continuous-state process. If the times at which we observe values of $X(t)$ are finite or countable, we have a discrete-time process. In this case the process is called a stochastic sequence and we may write X_t instead of $X(t)$.
Similar as we defined the covariance and the correlation as a measure of dependence between two random variables, we may define the autocovariance and the autocorrelation as a measure of dependence among the random variables of a (discrete or continuous time) stochastic process. For two time points t_1, t_2 and the random variables $X(t_1)$ and $X(t_2)$ from the stochastic process $\{X(t)\}$ with mean μ the autocovariance is defined as

$$\gamma(t_1, t_2) = Cov[X(t_1), X(t_2)] = E[(X(t_1) - \mu)(X(t_2) - \mu)].$$

In case of an stationary process only the difference (or lag) k between t_1 and t_2 is important, thus the autocovariance can be written as

$$\gamma(k) = \gamma_k = E[(X(t) - \mu)(X(t+k) - \mu)].$$

The autocorrelation function of lag k is then defined as

$$\rho_k = \frac{\gamma_k}{\gamma_0} = \frac{E[(X(t) - \mu)(X(t+k) - \mu)]}{\sigma^2}.$$

1.3.5. Markov Processes

Markov processes are stochastic processes $\{X(t)\}$ with the so-called Markov property stating that

$$\begin{aligned} & P(X(t) = x \,|\, X(t_n) = x_n, X(t_{n-1}) = x_{n-1}, \ldots, X(t_0) = x_0) \\ = \quad & P(X(t) = x \,|\, X(t_n) = x_n),\ t > t_n > t_{n-1} > \cdots > t_0. \end{aligned} \tag{1.1}$$

This means that the next state of the Markov process is only determined by the present state $X(t_n)$ but not by the previous states. Markov processes are the basis for the definition of Phase-type distributions and Markovian Arrival Processes, which are introduced in Section 2.3.

As already mentioned we can classify Markov processes depending on state space and time parameter: If the state space is discrete the process is called Markov chain. We can distinguish between discrete-time Markov chains (DTMC) and continuous-time Markov chains (CTMC). In the following we will only consider Markov chains, i.e. Markov processes with a discrete state space.

To satisfy the Markov property the sojourn time (i.e. the time spent in a state of a Markov chain) must exhibit the memoryless property, i.e. for the sojourn time Y we require that $P(Y > t + s \,|\, Y > t) = P(Y > s)$ for all $t, s \geq 0$ has to hold. Since the exponential and the geometric distribution are the only distributions possessing this property, the sojourn times are exponentially distributed in the continuous case and geometrically distributed in the discrete case.

A Markov process is called nonhomogeneous if the transitions out of state $X(t)$ depend on the time t. If they are independent of time, the process is called homogeneous [143].

Discrete-time Markov Chains

For DTMCs the Markov property from Equation 1.1 can be written as

$$P(X_{n+1} = x_{n+1} \,|\, X_n = x_n, \ldots, X_0 = x_0) = P(X_{n+1} = x_{n+1} \,|\, X_n = x_n).$$

The conditional probabilities $p_{ij}(n) = P(X_{n+1} = j \,|\, X_n = i)$, for all $n = 0, 1, \ldots$ are called (single-step) transition probabilities between states i and j of the DTMC. In case of a homogeneous Markov chain the time parameter n may be dropped resulting in $p_{ij} = P(X_{n+1} = j \,|\, X_n = i)$, for all $n = 0, 1, \ldots$, which we will assume in the following. The matrix $\boldsymbol{P}$ consisting of the p_{ij} in row i and column j for all i and j is called the transition probability matrix. We have that $0 \leq p_{ij} \leq 1$ and for all i, $\sum_j p_{ij} = 1$. The single-step transition probabilities may be generalized to n-step transition probabilities $p_{ij}^{(n)} = P(X_{m+n} = j \,|\, X_m = i)$. The n-step transition probabilities can be obtained

from the single-step transition probabilities using the so called Chapman-Kolmogorov equations [143], which yield $\boldsymbol{P}^n$ as matrix for n-step transition probabilities.
To study the long-run behavior the states of a Markov-Chain can be classified depending on whether they are visited infinitely often or a finite number of times [155]. A state i is transient (or nonrecurrent) if and only if there is a positive probability that the process will not return to this state. State i is recurrent if and only if starting from i the process eventually returns to i with probability 1. If $p_{ii} = 1$ the state i is absorbing. Let $f_{ij}^{(n)}$ be the conditional probability that the first visit to state j from state i occurs in exactly n steps. We have that [155]

$$p_{ij}^{(n)} = \sum_{k=1}^{n} f_{ij}^{(k)} p_{jj}^{(n-k)}, n \geq 1.$$

Let f_{ij} denote the probability, that, starting from i, we will ever visit state j, i.e. $f_{ij} = \sum_{n=1}^{\infty} f_{ij}^{(n)}$. Thus we have, that state i is recurrent if $f_{ii} = 1$ and transient if $f_{ii} < 1$.
The mean recurrence time of a state i with $f_{ii} = 1$ is defined as

$$\upsilon_i = \sum_{n=1}^{\infty} n f_{ii}^{(n)}.$$

If υ_i is finite the state i is recurrent nonnull (or positive recurrent) and if υ_i is infinite i is recurrent null.
Consider a recurrent state i and the set $\{n \,|\, p_{ii}^{(n)} > 0\}$ of positive integers. The greatest common divisor of that set is called the period of state i and is denoted by d_i. A recurrent state i is aperiodic if $d_i = 1$ and periodic if $d_i > 1$.
A finite Markov chain is irreducible if every state can be reached from all other states in a finite number of steps, i.e. $\forall\, i, j \quad \exists n \geq 1 : p_{ij}^{(n)} > 0$. It is known, that in an irreducible Markov chain all states are of the same type [65], e.g. if one state is recurrent or transient, then so are all states and we call the Markov chain recurrent or transient, respectively.
Consider an irreducible, aperiodic and finite Markov chain. As $n \to \infty$ the n-step transition probabilities $p_{ij}^{(n)}$ become independent of i and n, i.e.

$$\pi_j = \lim_{n\to\infty} p_j^{(n)} = \lim_{n\to\infty} P(X_n = j) = \lim_{n\to\infty} p_{ij}^{(n)}.$$

The vector $\boldsymbol{\pi}$ containing π_j at position j is called the steady-state or stationary probability vector and we have that $\boldsymbol{\pi} = \boldsymbol{\pi}\boldsymbol{P}$ and $\pi_j \geq 0, \sum_j \pi_j = 1$.

Continuous-time Markov Chains

We can define CTMCs along the lines of DTMCs. A CTMC is a stochastic process $\{X(t), t \geq 0\}$ that fulfills

$$P(X(t_{n+1}) = x_{n+1} \,|\, X(t_n) = x_n, \ldots, X(t_0) = x_0) = P(X(t_{n+1}) = x_{n+1} \,|\, X(t_n) = x_n)$$

for all $t_0 < \cdots < t_n < t_{n+1}$. The transition probabilities of a CTMC are given by $p_{ij}(s,t) = P(X(t) = j \,|\, X(s) = i), t \geq s$ in the nonhomogeneous case[1]. The state probabilities at time t are denoted by $p_j(t) = P(X(t) = j), j = 0, 1, 2, \ldots$ with $\sum_j p_j(t) = 1$.

[1] For a homogeneous CTMC the transition probabilities only depend on the difference $\varsigma = t - s$ between t and s resulting in $p_{ij}(\varsigma) = P(X(s+\varsigma) = j \,|\, X(s) = i), t \geq s$.

The state probabilities can be expressed in terms of the transition probabilities

$$p_j(t) = P(X(t) = j) = \sum_i P(X(t) = j \,|\, X(v) = i)P(X(v) = i) = \sum_i p_{ij}(v,t)p_i(v).$$

The vector $\boldsymbol{p}(0) = (p_1(0), p_2(0), \ldots)$ denotes the initial probability vector of the CTMC. A DTMC can be represented by a matrix of transition probabilities. For a CTMC one can state a matrix of transition rates in a similar way: For CTMCs the probability of a transition from a state depends on the state and on the length of the interval of observation Δt. $p_{ij}(t, t+\Delta t)$ is the probability that we observe a transition in $[t, t+\Delta t]$ from state i to j. For small intervals the probability that a transition occurs will also be very small ($p_{ij}(t, t+\Delta t) \to 0$ for $i \neq j, \Delta t \to 0$), implying that $p_{ii}(t, t+\Delta t) \to 1, \Delta t \to 0$. For larger Δt the probability that one or more transitions occur will increase. We would like to ensure, that Δt is so small that the probability of more than one transition within this period is negligible. More formally, the probability should be $o(\Delta t)$ where $\lim_{\Delta t \to 0} \frac{o(\Delta t)}{\Delta t} = 0$.
The transition rate $q_{ij}(t)$ denotes the number of transitions from state i to state j at time t per unit time and does not depend on the length of the observation interval Δt. However, the transition rate and the transition probability are related:

$$q_{ij}(t) = \lim_{\Delta t \to 0} \left\{ \frac{p_{ij}(t, t+\Delta t)}{\Delta t} \right\}, i \neq j.$$

This implies that $p_{ij}(t, t+\Delta t) = q_{ij}(t)\Delta t + o(\Delta t), i \neq j$. Additionally we have $p_{ii}(t, t+\Delta t) = 1 + q_{ii}(t)\Delta t + o(\Delta t)$. This follows from (cf. [143])

$$1 - p_{ii}(t, t+\Delta t) = \sum_{j \neq i} p_{ij}(t, t+\Delta t) = \sum_{j \neq i} q_{ij}(t)\Delta t + o(\Delta t)$$

implying that

$$q_{ii}(t) \equiv \lim_{\Delta t \to 0} \left\{ \frac{p_{ii}(t, t+\Delta t) - 1}{\Delta t} \right\} = \lim_{\Delta t \to 0} \left\{ \frac{-\sum_{j \neq i} q_{ij}(t)\Delta t + o(\Delta t)}{\Delta t} \right\}$$

and finally

$$q_{ii}(t) = -\sum_{j \neq i} q_{ij}(t).$$

Out of the elements $q_{ij}(t)$ we can construct the infinitesimal generator matrix for the CTMC:

$$\boldsymbol{Q}(t) = \lim_{\Delta t \to 0} \left\{ \frac{\boldsymbol{P}(t, t+\Delta t) - \boldsymbol{I}}{\Delta t} \right\}$$

where $\boldsymbol{P}(t, t+\Delta t)$ is the transition probability matrix consisting of the $p_{ij}(t, t+\Delta t)$ and $\boldsymbol{I}$ the identity matrix. As mentioned before, the sojourn times of the CTMC are exponentially distributed. The rate of this distribution is given by $-q_{ii}(t)$ for state i. For a homogeneous CTMC the transition rates q_{ij} are independent of time and we may simply write the matrix as $\boldsymbol{Q}$.
The classification of states in a CTMC works similar to the classification in a DTMC. A state i is absorbing if $q_{ij} = 0$ for all $i \neq j$. A state j is reachable from state i if for some s, t we have $p_{ij}(s, t) > 0$. A CTMC is irreducible if every state can be reached

from every other state.
For an irreducible, homogeneous CTMC with all states being recurrent nonnull the limits

$$\pi'_j = \lim_{t\to\infty} p_{ij}(t) = \lim_{t\to\infty} p_j(t)$$

exist and are independent of the initial state i [96]. The steady-state probability vector $\boldsymbol{\pi}' = (\pi'_1, \pi'_2, \ldots)$ is then determined by

$$\boldsymbol{\pi}'\boldsymbol{Q} = \mathbf{0} \quad \text{and} \quad \sum_j \pi'_j = 1.$$

Irreducible CTMCs with positive limiting probabilities π'_j are called recurrent nonnull or positive recurrent.

1.3.6. Traffic Processes

Simple traffic consisting of single arrivals of e.g. packets can be equivalently described by three different types of processes [87]: A point process consists of a sequence of times $\hat{t}_1, \hat{t}_2, \ldots, \hat{t}_n$, where each $\hat{t}_i$ denotes an arrival. A counting process $\{N(t)\}_{t=0}^{\infty}$ is a continuous-time stochastic process with $N(t) = \max\{n : \hat{t}_n \leq t\}, N(t) \geq 0$ being the number of arrivals in $(0, t]$. An interarrival time process $\{t_i\}_{i=1}^{\infty}$ is a real-valued sequence, where $t_i = \hat{t}_i - \hat{t}_{i-1}$ is the length of the time interval between two successive arrivals $\hat{t}_{i-1}$ and $\hat{t}_i$.
Several models for traffic processes will be given in Chapter 2. The main focus of this thesis is on modeling interarrival time processes for that a sequence of observations has been obtained from a real system in form of a trace (cf. Section 1.3.8).

1.3.7. Long-Range Dependence and Self-Similarity

A property that packet traffic might exhibit is self-similarity [18, 108], which describes the phenomenon that traffic looks statistically the same over a wide range of time scales and is related to long-range dependence. More formally a stochastic process $X = \{X(t)\}_{t=0}^{\infty}$ is long-range dependent if it has an autocorrelation function of the form

$$\rho_k \approx k^{-\eta} L_1(k), \text{ as } k \to \infty \tag{1.2}$$

where $0 < \eta < 1$ and L_1 is a function slowly varying at infinity, i.e. $\lim_{t\to\infty} L_1(tx)/L_1(t) = 1$ for all $x > 0$. The autocorrelation function of long-range dependent processes decays hyperbolically in the lag. Equation 1.2 implies that $\sum_{k=1}^{\infty} \rho_k = \infty$, i.e. even though the autocorrelations of high lags are individually small, their cumulative effect is of importance. This distinguishes those processes from short-range dependent processes, which have an autocorrelation function that decays geometrically, i.e. $\rho_k \approx r^k$, as $k \to \infty, 0 < r < 1$ and $0 < \sum_{k=1}^{\infty} \rho_k < \infty$.

The degree of self-similarity is defined in terms of the Hurst parameter H. Let $X^{(l)} = \{X_k^{(l)}\}_{k=1}^{\infty}$ for each $l \geq 1$ denote a covariance stationary time series with variance $\left(\sigma^{(l)}\right)^2$, autocorrelation function $\rho^{(l)}$ and

$$X_k^{(l)} = \frac{1}{l} \sum_{i=1}^{l} X_{kl-l+i}.$$

This aggregated process is obtained by averaging the original time-series X over non-overlapping blocks of size l. The relation of statistical properties of X and $X^{(l)}$ allows for propositions about the process X [87]: X is (exactly) self-similar with parameter $H = 1 - \eta/2$ if X and $\left\{l^{1-H}X_k^{(l)}\right\}_{k=1}^{\infty}$ have the same finite-dimensional distribution for all $l \geq 1$. X is (exactly) second-order self-similar with parameter $H = 1 - \eta/2$ if X and $\left\{l^{1-H}X_k^{(l)}\right\}_{k=1}^{\infty}$ have the same variance and autocorrelation functions for all $l \geq 1$. X is called asymptotically (second-order) self-similar with parameter $H = 1 - \eta/2$ if for all $k \geq 0$

$$\rho_k^{(l)} \to \frac{1}{2}\delta^2(k^{2-\eta}), \text{ as } l \to \infty$$

where $\delta^2(f(k)) = f(k+1) - 2f(k) + f(k-1)$ for the sequence $\{f(k)\}$.

Let $\{X_k\}_{k=1}^n$ be a sequence of observations with sample mean $\bar{X}(n)$ and sample variance $S^2(n)$. Then the rescaled adjusted range statistic or short R/S statistic is given by

$$\frac{R(n)}{S(n)} = \frac{\max\{0, W_1, \ldots, W_n\} - \min\{0, W_1, \ldots, W_n\}}{S(n)}$$

where $W_k = \sum_{i=1}^{k} X_i - k\bar{X}(n)$. According to [86] many naturally occurring time series can be modeled by

$$E\left[\frac{R(n)}{S(n)}\right] \approx cn^H, \text{ as } n \to \infty$$

with the positive constant c and the Hurst parameter H that has typical values around 0.7. If one assumes a process with short-range dependence the relation is given by [112]

$$E\left[\frac{R(n)}{S(n)}\right] \approx dn^{0.5}, \text{ as } n \to \infty$$

where d is a positive constant. The discrepancy between the two relations is called the Hurst effect.

1.3.8. Traces and their Characteristics

When trying to find appropriate input models one usually has a set of data points (called trace in the following) that has been observed from a real system and wants to find a model or process description that resembles the characteristics of the trace. A trace $\mathcal{T}$ is defined as a sequence of l chronologically ordered points in time $t_j > 0$ ($j = 1, \ldots, l$). Usually t_j describes the inter-arrival time of the j-th event, but a trace could as well contain service times, packet sizes or other data. Various characteristics can be estimated from the trace. The estimator for the i-th moment is given by

$$\hat{\mu}_i = \frac{1}{l}\sum_{j=1}^{l}(t_j)^i \tag{1.3}$$

and the estimator for the variance by

$$\hat{\sigma}^2 = \frac{1}{l-1}\sum_{j=1}^{l}(t_j - \hat{\mu}_1)^2.$$

If the trace exhibits dependencies between consecutive arrivals, the autocorrelation or the joint moments are of interest. The autocorrelation of arrivals that are lag k apart is estimated by

$$\hat{\rho}_k = \frac{1}{(l-k-1)\hat{\sigma}^2} \sum_{j=1}^{l-k} (t_j - \hat{\mu}_1)(t_{j+k} - \hat{\mu}_1). \tag{1.4}$$

The estimator for the joint moments $E[X_k^i X_{k+1}^j]$ of consecutive arrivals is given by

$$\hat{\nu}_{ij} = \frac{1}{l-1} \sum_{k=1}^{l-1} (t_k)^i (t_{k+1})^j.$$

Finally, the empirical distribution function of a trace is given by a step function with l steps:

$$F_{\mathcal{T}}(x) = \frac{\sum_{j=1}^{l} \delta(t_j \leq x)}{l}$$

where

$$\delta(b) = \begin{cases} 1 & \text{if } b = true, \\ 0 & \text{if } b = false. \end{cases}$$

Chapter 2

Traffic Models

This chapter gives a brief overview of important model classes that are used for traffic modeling, their properties and available fitting approaches. The main focus of this chapter is on Phase-type (PH) distributions and the related Markovian Arrival Processes (MAPs), Autoregressive Moving Average Processes and Autoregressive-To-Anything Processes, which are all heavily related to this work. A more elaborate overview can for example be found in [87].

In Section 2.1 Autoregressive Moving Average Processes and related models are introduced. These models can be used for modeling traffic data and moreover, can serve as a base process for several other approaches. An example for this are Autoregressive-To-Anything Processes, which are presented in Section 2.2. Section 2.3 summarizes results for PH distributions and MAPs and finally, Section 2.4 mentions further existing approaches for traffic modeling.

2.1. Autoregressive-Type Traffic Models

The class of Autoregressive Moving Average Processes is known since the work of Box and Jenkins [35]. These processes describe a time series with dependent successive values, which can be interpreted as a series of independent shocks ϵ_t that are distributed according to a normal distribution with zero mean and variance σ_ϵ^2, called white noise or innovations, which is transformed to the process Z_t by a linear filter [166] as shown in Figure 2.1. The linear filter takes a weighted sum of previous observations such that [35]

$$Z_t = \mu + \epsilon_t + \zeta_1\epsilon_{t-1} + \zeta_2\epsilon_{t-2} + \ldots = \mu + \zeta(B)\epsilon_t$$

where $\zeta(B) = 1 + \zeta_1 B + \zeta_2 B^2 + \ldots$ is the transfer function of the filter and B is the backward shift operator, i.e. $B^p\epsilon_t = \epsilon_{t-p}$. In case of a stable filter, i.e. the sequence is finite or infinite and convergent, μ is the mean of the process. Linear filter models are the basis for several stochastic processes belonging to the class of ARIMA processes.

An important special case of the linear filter model is the Autoregressive process of order p, denoted $AR(p)$, which is defined as [35]

$$Z_t = \alpha_1 Z_{t-1} + \alpha_2 Z_{t-2} + \ldots + \alpha_p Z_{t-p} + \epsilon_t \tag{2.1}$$

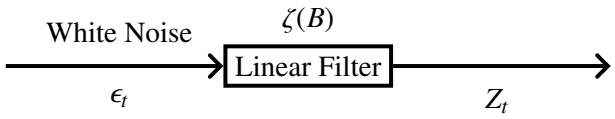


Figure 2.1.: Representation of a time series as the output from a linear filter (from [35])

where $Z_t = \tilde{Z}_t - \mu$ are deviations from the mean μ. Hence, it is assumed that the Z_t from the $AR(p)$ process describe a sequence with zero mean and the real time series $\tilde{Z}_t$ is obtained by adding μ to every Z_t. Defining $\alpha(B) = 1 - \alpha_1 B - \alpha_2 B^2 - \ldots - \alpha_p B^p$ the description can be abbreviated as $\alpha(B)Z_t = \epsilon_t$ or equivalently as $Z_t = \alpha^{-1}(B)\epsilon_t$. Observe, that an $AR(p)$ process has $p + 2$ unknown parameters, i.e. $\mu, \alpha_1, \ldots, \alpha_p, \sigma_\epsilon^2$.

Another important class is the Moving Average process of order q, abbreviated as $MA(q)$. While for an $AR(p)$ model Z_t is expressed as weighted sum of the p previous observations $Z_{t-i}, i = 1, \ldots, p$, for a $MA(q)$ model Z_t is constructed from q previous innovations $\epsilon_{t-i}, i = 1, \ldots, q$. Thus, the Moving Average process is defined as [35]

$$Z_t = \epsilon_t + \beta_1 \epsilon_{t-1} + \beta_2 \epsilon_{t-2} + \ldots + \beta_q \epsilon_{t-q}. \tag{2.2}$$

Defining $\beta(B) = 1 + \beta_1 B + \beta_2 B^2 + \ldots + \beta_q B^q$ we may write the Moving Average model as $Z_t = \beta(B)\epsilon_t$ with the $q + 2$ unknown parameters $\mu, \beta_1, \ldots, \beta_q, \sigma_\epsilon^2$.

A combination of Autoregressive and Moving Average processes results in Autoregressive Moving Average models, denoted $ARMA(p, q)$ and defined as

$$Z_t = \alpha_1 Z_{t-1} + \ldots + \alpha_p Z_{t-p} + \epsilon_t + \beta_1 \epsilon_{t-1} + \ldots + \beta_q \epsilon_{t-q}. \tag{2.3}$$

The equation can be abbreviated as $\alpha(B)Z_t = \beta(B)\epsilon_t$ and has the unknown parameters $\mu, \alpha_1, \ldots, \alpha_p, \beta_1, \ldots, \beta_q, \sigma_\epsilon^2$.

If a time series exhibits nonstationary behavior, it may be expressed by an Autoregressive Integrated Moving Average $ARIMA(p, d, q)$ model. ARIMA models add a third parameter d to an $ARMA(p, q)$ process to describe the difference between the observed values. They are especially helpful when the series does not vary about a fixed mean, but the behavior of the series is similar, if differences in the level are allowed [35]. An $ARIMA(p, d, q)$ model is defined by

$$W_t = \alpha_1 W_{t-1} + \ldots + \alpha_p W_{t-p} + \epsilon_t + \beta_1 \epsilon_{t-1} + \ldots + \beta_q \epsilon_{t-q} \tag{2.4}$$

with $W_t = \Delta^d \tilde{Z}_t$. The inverse of this equation is $\tilde{Z}_t = S^d W_t$ where S is the summation operator, i.e. $S W_t = \sum_{j=0}^{\infty} W_{t-j} = W_t + W_{t-1} + W_{t-2} + \ldots$. Observe, that the $ARMA(p, q)$ process in Equation 2.3 and the $ARIMA(p, d, q)$ process in Equation 2.4 are defined in a similar way, except that the Z_t and the W_t have a different interpretation. In Equation 2.3 all Z_t fluctuate around the mean zero of the time series, while in Equation 2.4 the W_t describe the d-th difference of the series. Hence, for an $ARIMA(p, d, q)$ process it is assumed that the d-th difference of the series can be modeled by an $ARMA(p, q)$ process [35]. In particular, an $ARIMA(p, d, q)$ becomes an $ARMA(p, q)$ model as defined in Equation 2.3 for $d = 0$ and by replacing the W_t with $Z_t = \tilde{Z}_t - \mu$.

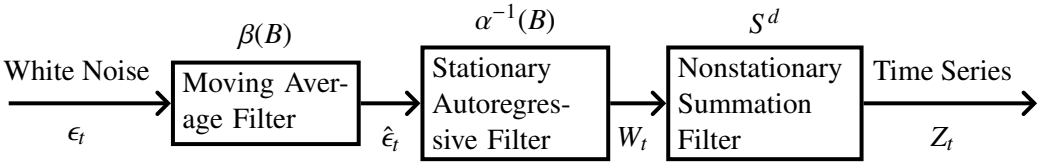


Figure 2.2.: Block diagram for $ARIMA(p, d, q)$ models (from [35])

An ARIMA model can be thought of as three consecutive filters as shown in Figure 2.2.

If non-integer values are allowed for d, we finally obtain the fractional Autoregressive Integrated Moving Average (f-ARIMA) processes [85], which are known to exhibit self-similarity.

2.1.1. Fitting of ARIMA Processes

For estimating the parameters of AR, MA, ARMA and also ARIMA processes several approaches are available. Maximum likelihood estimation [93] and least squares regression approaches [70, 111] are available for all types of processes. For AR processes the parameters can also be determined by solving Yule-Walker equations [159, 166] or by a spectral estimation procedure [46]. Moreover, the order p can be automatically determined by the so-called Akaike Information Criterion [1] for $AR(p)$ models.

Since all these fitting approaches are known for long time, they found their way into standard software, like e.g. the statistical software package `R` [51, 149], which supports all the fitting techniques mentioned above.

The main application area for ARIMA processes is modeling of discrete-time processes [107] and hence the data from a trace is usually interpreted as a count process for ARIMA models. For example in [165] the number of arrivals in given intervals are counted and f-ARIMA processes are used to model this process. Nevertheless, ARIMA processes have also been used to capture interarrival times in the past, e.g. [154] uses ARIMA processes for online prediction of the interarrival times of I/O requests. Further application examples for Autoregressive Processes can be found in [76] where $AR(2)$ processes are used to model VBR coded video and [133] where two $AR(1)$ processes were combined with other processes to achieve better results in capturing the empirical autocorrelation function of video traffic.

2.2. Autoregressive-To-Anything Processes

Autoregressive-To-Anything (ARTA) Processes have been introduced in [47] and combine an $AR(p)$ base process with an arbitrary marginal distribution F_Y and thus can model correlated input processes with a wide variety of shapes for the distribution. They are defined as a sequence

$$Y_t = F_Y^{-1}[\Phi(Z_t)], t = 1, 2, \ldots \tag{2.5}$$

where F_Y is the marginal distribution, Φ is the standard normal cumulative distribution function and $\{Z_t; t = 1, 2, \ldots\}$ is a stationary Gaussian $AR(p)$ process as described in Section 2.1. The $AR(p)$ base process has to be constructed in a way such that the distribution of the $\{Z_t\}$ is $N(0, 1)$ (cf. [47]), i.e. the variance of the innovations $\{\epsilon_t\}$, which are independent $N(0, \sigma_\epsilon^2)$ random variables, is set to $\sigma_\epsilon^2 = 1 - \alpha_1 Corr[Z_t, Z_{t+1}] - \alpha_2 Corr[Z_t, Z_{t+2}] - \cdots - \alpha_p Corr[Z_t, Z_{t+p}]$. Then the probability-integral transformation $U_t = \Phi(Z_t)$ ensures that U_t is uniformly distributed on $(0, 1)$ (cf. [59]) and the application of $Y_t = F_Y^{-1}[U_t]$ yields a time series $\{Y_t, t = 1, 2, \ldots\}$ with the desired marginal distribution F_Y. Note, that the last step, i.e. the transformation from U_t to Y_t is basically the inverse transform method used for random number generation (cf. e.g. [104]) with the exception that the U_t are correlated inducing a correlation in the sequence Y_t. Figure 2.3 shows the transformation steps that are necessary to construct an ARTA model from the $AR(p)$ base process. This approach works for any distribution F_Y

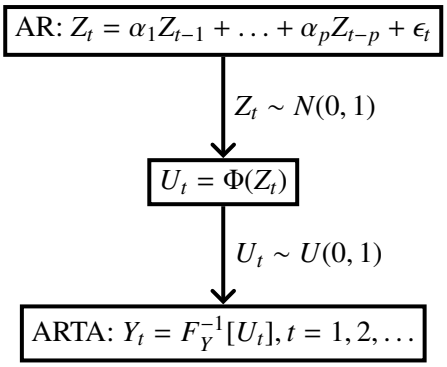


Figure 2.3.: ARTA process

for which F_Y^{-1} can be computed, either by a closed-form expression or by numerical methods.

In [47, 48] a relation between the autocorrelation structures of the ARTA process and the base process is established, which are related by

$$Corr[Y_t, Y_{t+h}] = Corr\left[F_Y^{-1}(\Phi(Z_t)), F_Y^{-1}(\Phi(Z_{t+h}))\right]. \tag{2.6}$$

Since

$$Corr[Y_t, Y_{t+h}] = \frac{E[Y_t Y_{t+h}] - (E[Y])^2}{Var[Y]}$$

and $E[Y]$ and $Var[Y]$ are given by F_Y, only $E[Y_t Y_{t+h}]$ has to be adjusted [47]. Furthermore (Z_t, Z_{t+h}) has a standard bivariate normal distribution with correlation ρ_h and thus, one obtains

$$\begin{aligned} E[Y_t Y_{t+h}] &= E[F_Y^{-1}(\Phi(Z_t)) F_Y^{-1}(\Phi(Z_{t+h}))] \\ &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} F_Y^{-1}(\Phi(z_t)) F_Y^{-1}(\Phi(z_{t+h})) \varphi_{\rho_h}(z_t, z_{t+h}) dz_t dz_{t+h}, \end{aligned} \tag{2.7}$$

where φ_{ρ_h} is the standard bivariate normal density function with correlation ρ_h. From Equation 2.7 it is apparent that the lag-h autocorrelations $\hat{\rho}_h$ of the ARTA process

$\{Y_t\}$ only depend on the lag-h autocorrelations ρ_h of the underlying Gaussian $AR(p)$ base process $\{Z(t)\}$, which appear in φ_{ρ_h}, i.e. they describe a function $\omega(\rho_h) = \hat{\rho}_h = Corr[Y_t, Y_{t+h}]$. Observe, that it is furthermore only possible to compute the ARTA autocorrelation $\hat{\rho}_h$ for a given base process autocorrelation ρ_h, but not the other way round. [47, 48] established properties of the relationship between the $\hat{\rho}_h$ and the ρ_h, i.e. they proved that $\omega(\rho_h)$ is continuous and nondecreasing in ρ_h and gave an efficient numerical procedure to find ρ_h such that the ARTA process has the desired autocorrelations $\hat{\rho}_h$. These ideas are picked up in a more detailed way in Chapter 4 where ARTA models are enabled to use an $ARMA(p, q)$ base process.

2.2.1. Fitting of ARTA Processes

For fitting ARTA processes two approaches are available.

The first approach is implemented in a tool called `ARTAFACTS` (ARTA Fitting Algorithm for Constructing Time Series) [47, 48] and assumes that the marginal distribution F_Y is given and constructs the $AR(p)$ base process such that the resulting ARTA model has the desired autocorrelation structure that has been estimated from a trace. Since it is only possible to compute the ARTA autocorrelation from a given base process autocorrelation using Equation 2.6 and not vice versa, the autocorrelation structure of the base process has to be determined by a numerical search procedure that tries several values for the base process correlation until $Corr[Y_t, Y_{t+h}]$ has the desired value save some given relative error [48]. Once the base process autocorrelation structure has been computed, the $AR(p)$ process can be fitted by solving Yule-Walker Equations. `ARTAFACTS` supports various marginal distributions like normal, Student's t, uniform, exponential, gamma, Weibull, lognormal, Johnson unbounded, discrete with finite support and empirical [48] (see Appendix B for a brief overview of the properties of these distributions), though no methods are provided for selecting an appropriate distribution and its parameters depending on the time series and thus additional tools are required for the estimation of the distribution, e.g. the tool `FITTR1` [56] could be used for the estimation of the parameters of a Johnson distribution or in general tools like `Expertfit` [105] can be used to choose a marginal distribution and its parameters.

The more recently developed second approach `ARTAFIT` [25, 26] is an advancement to `ARTAFACTS` that fits both the autocorrelation structure and the marginal distribution. The marginal distribution is assumed to be from the Johnson system, which is completely determined by four parameters that can be chosen such that the distribution can match any finite first four moments [56]. In an extensive empirical study [26] it was shown that even with the restriction to Johnson distributions ARTA models are still able to model a wide variety of processes.

2.3. Phase-Type Distributions and Markovian Arrival Processes

Markov processes introduced in Section 1.3.5 can be used to describe probability distributions derived from the exponential distribution. These distributions, called Phase-type or short PH distributions, can be seen as the time until absorption in a Markov process and are based on the method of stages, which has been introduced by A.K.

Erlang [61] and generalized by M.F. Neuts [120]. The basic idea of this technique is to model a random time interval as a number of exponentially distributed phases. In the following results for PH distributions and their generalization, Markovian Arrival Processes, are presented. Aside from statistical properties the relation between several related distributions belonging to the class of PH distributions is outlined and fitting methods for PH distributions and MAPs are summarized.

2.3.1. Phase-Type Distributions

Consider a Markov process with states $\{0, 1, \ldots, n\}$, initial probability vector $(\pi_0, \boldsymbol{\pi})$ and infinitesimal generator matrix

$$\boldsymbol{Q} = \begin{bmatrix} 0 & \mathbf{0} \\ \boldsymbol{t} & \boldsymbol{D}_0 \end{bmatrix} \tag{2.8}$$

where $\boldsymbol{\pi}$ is a row vector and $\boldsymbol{t}$ a column vector both of size n and $\boldsymbol{D}_0$ is $n \times n$ matrix. Keeping in mind that $\boldsymbol{Q}$ is the generator of a Markov process, we have that $\boldsymbol{D}_0(i,i) < 0, \boldsymbol{t}(i) \geq 0, \boldsymbol{D}_0(i,j) \geq 0$ for $1 \leq i \neq j \leq n$ and additionally $\boldsymbol{D}_0\mathbf{1} + \boldsymbol{t} = \mathbf{0}$ and $\pi_0 + \boldsymbol{\pi}\mathbf{1} = 1$. State 0 of the Markov process is an absorbing state, while all other states are transient. Thus, the nonsingular matrix $\boldsymbol{D}_0$ contains the transition rates from transient state i to j and the vector $\boldsymbol{t}$ describes the rates from a transient state into absorbing state 0. $\boldsymbol{D}_0$ is called a subgenerator of order n and $\boldsymbol{t}$ is called the killing-rate vector. The vector $\boldsymbol{\pi} = (\pi_1, \pi_2, \ldots, \pi_n)$ describes the initial probabilities for the transient states and π_0 the initial probability for the absorbing state 0, called point mass at zero. In most cases it is assumed that $\pi_0 = 0$, so that at least one transient state is visited before absorption. If for a state i we have that $\boldsymbol{\pi}(i) = \pi_i > 0$ the state is an entry state, if the row sum of the i-th row of $\boldsymbol{D}_0$ is negative, i.e. $\boldsymbol{t}(i) > 0$, it is an exit state. An example for a Phase-type distribution with 3 transient and 1 absorbing state, marked in gray, is shown in Figure 2.4.

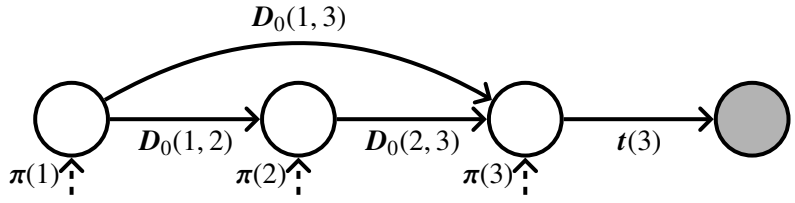


Figure 2.4.: Example for a PH distribution with 3 transient and 1 absorbing state

Definition 2.1. *The distribution of the time X until absorption in state* 0 *of a CTMC with generator matrix* $\boldsymbol{Q}$ *is called the continuous Phase-type distribution (CPH) of order n [120] with representation* $(\boldsymbol{\pi}, \boldsymbol{D}_0)$*, denoted as* $PH(\boldsymbol{\pi}, \boldsymbol{D}_0)$*, and we say X is* $PH(\boldsymbol{\pi}, \boldsymbol{D}_0)$*.*

Remark. Since $\boldsymbol{t}$ and π_0 are implicitly given by $\boldsymbol{D}_0$ and $\boldsymbol{\pi}$, they are omitted in the representation.

The distribution function of a Phase-type distributed variable X with representation $(\boldsymbol{\pi}, \boldsymbol{D}_0)$ is given by [103]

$$F(x) = 1 - \boldsymbol{\pi} \exp(\boldsymbol{D}_0 x)\mathbf{1} \quad \text{for } x \geq 0 \tag{2.9}$$

and its density function is given by

$$f(x) = \boldsymbol{\pi} \exp(\boldsymbol{D}_0 x)\boldsymbol{t} \quad \text{for } x > 0 \tag{2.10}$$

where the matrix exponential is defined as [19]

$$\exp(\boldsymbol{A}) = e^{\boldsymbol{A}} = \sum_{n \geq 0} \frac{1}{n!} \boldsymbol{A}^n.$$

The moments of a PH distribution are derived from the moment matrix $\boldsymbol{M} = -\boldsymbol{D}_0^{-1}$:

$$\mu_i = E[X^i] = i! \boldsymbol{\pi} \boldsymbol{M}^i \mathbf{1}. \tag{2.11}$$

PH distributions are a very flexible class of distributions that can represent every distribution with a strictly positive density in $(0, \infty)$ [124]. Depending on the structure of matrix $\boldsymbol{D}_0$ and the vector $\boldsymbol{\pi}$ one can distinguish several subclasses of Phase-type distributions. For most of these subclasses simpler expressions for the cumulative distribution function exist that do not require the computation of the matrix exponential.

Exponential Distribution

Since the phases of a PH distribution are exponentially distributed, the simplest Phase-type distribution, of course, is the exponential distribution itself consisting only of a single phase as shown in Figure 2.5. The exponential distribution is completely characterized by its rate parameter λ. It has density function

$$f(x) = \begin{cases} \lambda e^{-\lambda x} & \text{if } x \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

and distribution function

$$F(x) = \begin{cases} 1 - e^{-\lambda x} & \text{if } x \geq 0 \\ 0 & \text{otherwise.} \end{cases}$$

The mean is given by $E[X] = 1/\lambda$ and the variance by $Var[X] = 1/\lambda^2$.
The exponential distribution is the only continuous distribution with the so-called memoryless property, which states that $P(X > t + s \mid X > t) = P(X > s)$ for all $t, s \geq 0$.

Erlang Distribution

The Erlang distribution with parameters λ and n consists of n sequential phases that have identical exponential distribution with rate λ as shown in Figure 2.6. Its density function is given by

$$f(x) = \frac{\lambda^n}{(n-1)!} x^{n-1} e^{-\lambda x}, \quad x > 0, \lambda > 0.$$

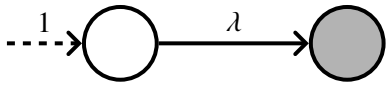


Figure 2.5.: Exponential distribution

The distribution function is given by

$$F(x) = 1 - \sum_{i=0}^{n-1} \frac{(\lambda x)^i}{i!} e^{-\lambda x}, \quad x \geq 0, \lambda > 0.$$

The i-th moment of an Erlang distributed random variable is given by

$$E[X^i] = \frac{(n+i-1)!}{(n-1)!} \frac{1}{\lambda^i}. \tag{2.12}$$

Therefore, the mean of an Erlang distributed random variable X is $E[X] = n/\lambda$ and its variance $Var[X] = n/(\lambda^2)$. The Erlang distribution can be described by matrix

$$\boldsymbol{D}_0 = \begin{bmatrix} -\lambda & \lambda & \dots & 0 & 0 \\ 0 & -\lambda & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & -\lambda & \lambda \\ 0 & 0 & \dots & 0 & -\lambda \end{bmatrix}$$

and initial probability vector $\boldsymbol{\pi} = (1, 0, \dots, 0)$.

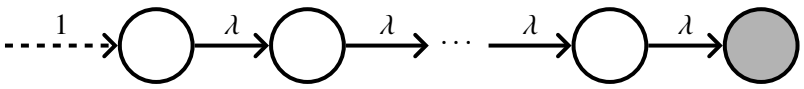


Figure 2.6.: Erlang distribution

Hypo-Exponential Distribution

The Hypo-Exponential distribution is a generalized Erlang distribution where the rates are not necessarily identical in each phase. Hence, it is characterized by the number of phases n and the rates $\boldsymbol{\Lambda} = (\lambda_1, \lambda_2, \dots, \lambda_n)$. Its density function is given by

$$f(x) = \sum_{i=1}^{n} a_i \lambda_i e^{-\lambda_i x}, \text{ where } a_i = \prod_{j=1, j\neq i}^{n} \frac{\lambda_j}{\lambda_j - \lambda_i}, \lambda_i \neq \lambda_j \text{ for } i \neq j.$$

A hypo-exponentially distributed random variable X has mean $E[X] = \sum_{i=1}^{n} 1/\lambda_i$ and variance $Var[X] = \sum_{i=1}^{n} 1/(\lambda_i^2)$. An example is shown in Figure 2.7. For the Hypo-

Exponential distribution matrix $\boldsymbol{D}_0$ has the form

$$\boldsymbol{D}_0 = \begin{bmatrix} -\lambda_1 & \lambda_1 & \dots & 0 & 0 \\ 0 & -\lambda_2 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & -\lambda_{n-1} & \lambda_{n-1} \\ 0 & 0 & \dots & 0 & -\lambda_n \end{bmatrix}$$

and the initial probability vector is given by $\boldsymbol{\pi} = (1, 0, \dots, 0)$.

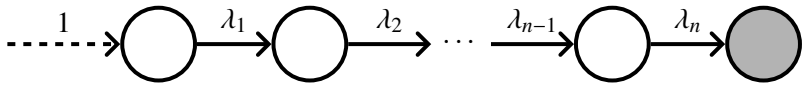


Figure 2.7.: Hypo-Exponential distribution

Hyper-Exponential Distribution

The Hyper-Exponential distribution is a convex mixture of n exponential distributions as shown in Figure 2.8. The density function of a n-phase Hyper-Exponential distribution is given by

$$f(x) = \sum_{i=1}^{n} \tau_i \lambda_i e^{-\lambda_i x}$$

where $x, \tau_i, \lambda_i > 0$ for all i and $\sum_{i=1}^{n} \tau_i = 1$. The distribution function is given by

$$F(x) = \sum_{i=1}^{n} \tau_i \left(1 - e^{-\lambda_i x}\right)$$

where $x \geq 0$. The mean of a hyper-exponentially distributed random variable X is given by $E[X] = \sum_{i=1}^{n} \tau_i / \lambda_i$ and its variance by

$$Var[X] = 2 \sum_{i=1}^{n} \frac{\tau_i}{\lambda_i^2} - \left(\sum_{i=1}^{n} \frac{\tau_i}{\lambda_i} \right)^2 .$$

A Hyper-Exponential distribution can be defined in terms of matrix

$$\boldsymbol{D}_0 = \begin{bmatrix} -\lambda_1 & 0 & 0 & \dots & 0 \\ 0 & -\lambda_2 & 0 & \dots & 0 \\ 0 & 0 & -\lambda_3 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & -\lambda_n \end{bmatrix}$$

and the initial probability vector $\boldsymbol{\pi} = (\tau_1, \tau_2, \dots, \tau_n)$.

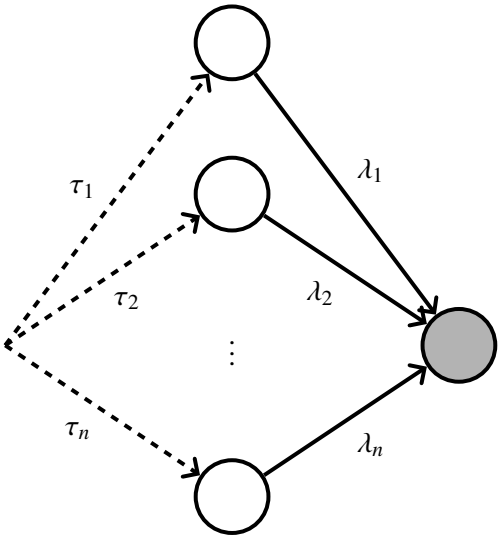


Figure 2.8.: Hyper-Exponential distribution

Hyper-Erlang Distribution

A Hyper-Erlang distribution (or short HErD) [63] is a mixture of m mutually independent Erlang distributions weighted with the probabilities $\tau_1, \ldots, \tau_m$ with $\tau_i \geq 0$ and $\sum_{i=1}^{m} \tau_i = 1$ as shown in Figure 2.9. Let S_i denote the number of phases and λ_i be the rate parameter of the i-th Erlang distribution.
The probability density function of a Hyper-Erlang random variable X is given by

$$f(x) = \sum_{i=1}^{m} \tau_i \frac{(\lambda_i x)^{S_i - 1}}{(S_i - 1)!} \lambda_i e^{-\lambda_i x}$$

and its cumulative distribution function by

$$F(x) = 1 - \sum_{i=1}^{m} \tau_i \sum_{j=0}^{S_i - 1} \frac{(\lambda_i x)^j}{j!} e^{-\lambda_i x}. \tag{2.13}$$

The i-th moment is given by

$$E[X^i] = \sum_{j=1}^{m} \tau_j \frac{(S_j + i - 1)!}{(S_j - 1)!} \frac{1}{\lambda_j^i}. \tag{2.14}$$

The overall number of states of a HErD is given by $n = \sum_{i=1}^{m} S_i$. For $m = 1$ the HErD becomes an Erlang distribution and for $S_i = 1$ for all $i = 1, \ldots, m$ a Hyper-Exponential distribution. The matrix representation of the Hyper-Erlang distribution is given by

$$\boldsymbol{D}_0 = \begin{bmatrix} \boldsymbol{G}_1 & 0 & \ldots & 0 \\ 0 & \boldsymbol{G}_2 & \ldots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & \ldots & \boldsymbol{G}_m \end{bmatrix}$$

and initial probability vector $\boldsymbol{\pi} = (\tau_1, 0, \ldots, 0, \tau_2, 0, \ldots, \tau_m, 0, \ldots, 0)$, where each of the $\boldsymbol{G}_i, 1 \leq i \leq m$ defines the transition rates between the transient states of one of the Erlang branches, i.e. they have the form

$$\boldsymbol{G}_i = \begin{bmatrix} -\lambda_i & \lambda_i & \ldots & 0 & 0 \\ 0 & -\lambda_i & \ldots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \ldots & -\lambda_i & \lambda_i \\ 0 & 0 & \ldots & 0 & -\lambda_i \end{bmatrix}.$$

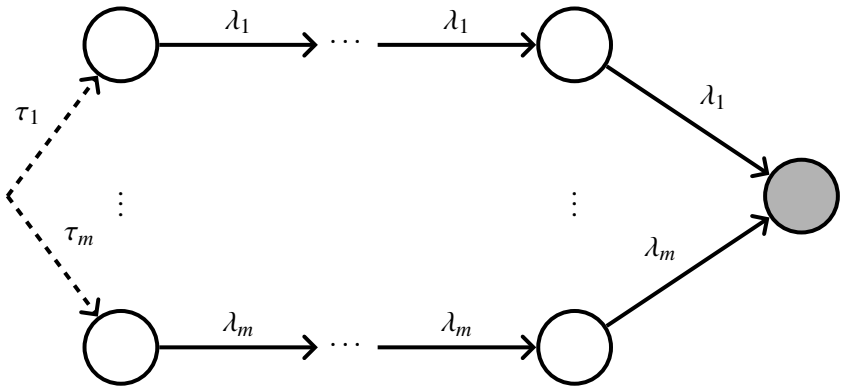


Figure 2.9.: Hyper-Erlang distribution

Cox Distribution

A Cox distribution is similar to the Hypo-Exponential distribution except that a probabilistic decision is made after every exponential phase deciding whether a next phase is taken or not. Thus, it consists of n exponential phases with rates $\lambda_i, i = 1, \ldots, n$. After phase i the next phase $i+1$ is taken with probability g_i and with probability $\bar{g}_i = 1 - g_i$ a transition to the absorbing state occurs. Cox distributions can approximate any other distribution with rational Laplace transform. An example for the Cox distribution is shown in Figure 2.10. For a Cox distribution matrix $\boldsymbol{D}_0$ is given by

$$\boldsymbol{D}_0 = \begin{bmatrix} -\lambda_1 & g_1\lambda_1 & 0 & \ldots & 0 & 0 \\ 0 & -\lambda_2 & g_2\lambda_2 & \ldots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \ldots & -\lambda_{n-1} & g_{n-1}\lambda_{n-1} \\ 0 & 0 & 0 & \ldots & 0 & -\lambda_n \end{bmatrix}$$

and the initial probability vector is given by $\boldsymbol{\pi} = (1, 0, \ldots, 0)$.

Acyclic Phase-Type Distribution

If $\boldsymbol{D}_0$ can be transformed into an upper (or lower) triangular matrix by symmetric permutations of rows and columns the PH distribution is acyclic (an APH distribution).

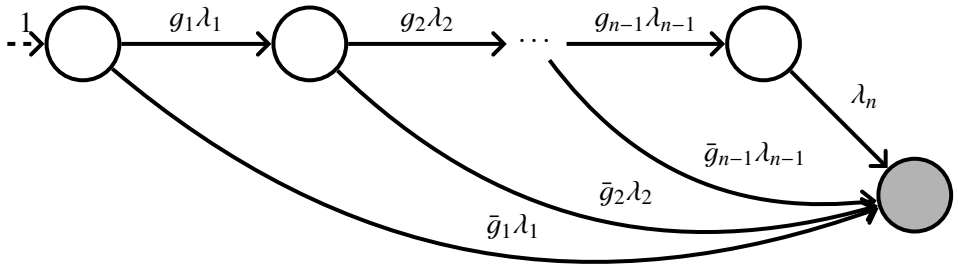


Figure 2.10.: Cox distribution

This structure of the matrix implies that a state i can only be connected with a state j if $j > i$ and that each state can be only visited at most once before absorption. The Markov Chain corresponding to an acyclic PH distribution is shown in Figure 2.11. In an APH all states may be entry or exit states. APHs are the largest subclass of PH distributions for which canonical representations exist (cf. Section 2.3.4).

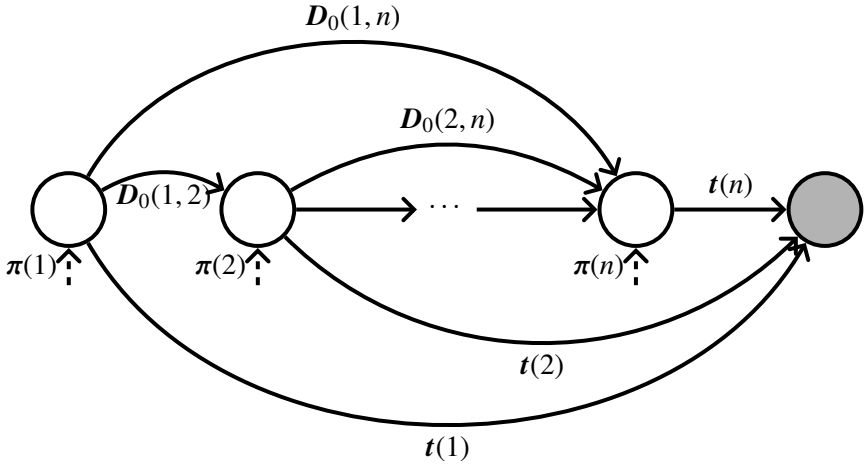


Figure 2.11.: Acyclic PH distribution

Matrix-Exponential Distribution

Phase-type distributions belong to the class of Matrix-Exponential (ME) distributions [2, 6, 62], which are defined in a similar way as PH distributions, but lack the descriptive interpretation as time until absorption of a Markov chain. Hence, they cannot be easily integrated into simulation models because random number generation is much more difficult than for PH distributions. In [38] an approach is presented to draw random numbers from a ME distribution, which basically relies on the inversion of the distribution function and therefore is very inefficient. Thus, we will only give a brief introduction on ME distributions to complete the description of PH distributions.

A nonnegative random variable X is distributed according to a ME distribution if it has a distribution function of the form

$$F(x) = \begin{cases} \pi_0, & x = 0 \\ 1 - \boldsymbol{\pi} \exp(\boldsymbol{D}_0 x)\mathbf{1}, & x > 0 \end{cases} \tag{2.15}$$

where $\boldsymbol{\pi}$ is a $1 \times n$ vector and $\boldsymbol{D}_0$ is a $n \times n$ matrix, both with possibly complex entries. If $\boldsymbol{D}_0$ and $\boldsymbol{\pi}$ respect the requirements mentioned at the beginning of Section 2.3.1 (i.e. $\boldsymbol{D}_0$ has non-negative off-diagonal elements, negative diagonal elements and non-positive row-sums and $\boldsymbol{\pi}$ is a probability vector) the ME distribution is a Phase-type distribution.
In general (and in contrast to a PH distribution) there are no restrictions on the elements of $\boldsymbol{D}_0$ and $\boldsymbol{\pi}$ except that Equation 2.15 must describe a valid distribution and $\boldsymbol{\pi}\mathbf{1} = 1$. Thus, for example matrix $\boldsymbol{D}_0$ may contain negative off-diagonal elements or non-negative elements in the diagonal and does in general not yield a generator matrix of a Markov chain. Furthermore, the vector $\boldsymbol{\pi}$ may contain negative entries and will not be a probability vector in general.

Phase-Type Renewal Processes

In Phase-type Renewal Processes the interarrival times have PH distribution. Thus, the interarrival times can be modeled by a continuous-time Markov process $J = \{J(t)\}_{t=0}^{\infty}$ with state space $\{0, 1, \ldots, n\}$, where state 0 is an absorbing state and states $1, \ldots, n$ are transient, initial distribution $\boldsymbol{\pi}$ and infinitesimal generator matrix $\boldsymbol{Q}$ as defined in Equation 2.8, which is associated with the PH distribution $F(\cdot)$. The interarrival time A_i is the time until absorption of J started with initial distribution $\boldsymbol{\pi}$. To determine A_{i+1} the process J is restarted with initial distribution $\boldsymbol{\pi}$ again. This leads to a sequence of times $\{0 = t_0 < t_1 < t_2 < \ldots\}$ at which the process is (re)started with its initial distribution and that have an interrenewal distribution $PH(\boldsymbol{\pi}, \boldsymbol{D}_0)$. If the state at t_0 is chosen with a different distribution $\hat{\boldsymbol{\pi}}$, we have a delayed PH renewal process.

A well known example for a PH Renewal process is the Poisson process [52] with rate λ that has exponentially distributed interarrival times A_i. A Poisson process has independent increments, i.e. the number of arrivals in disjoint intervals is independent, implying that a Poisson process is memoryless.

Poisson processes are often used for the modeling of traffic on main communication arteries, where a large number of independent traffic streams are multiplexed. Under certain conditions these multiplexed streams approach a Poisson process (the theoretical background for this observation is known as Palm's Theorem [102]).

Extensions of the standard Poisson process are the non-homogeneous (or time dependent) Poisson process, where the rate λ depends on time, and the Compound Poisson process, where a second distribution (independent of the A_i) specifies the batch size B_i. The superposition of independent Poisson processes with arrival rates $\lambda_1, \lambda_2, \ldots$ is itself a Poisson process with rate $\lambda_1 + \lambda_2 + \ldots$. The state diagram of a Poisson process is shown in Figure 2.12 where each state represents the number of arrivals.

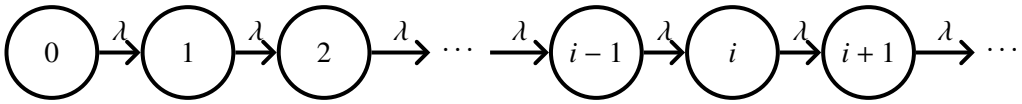


Figure 2.12.: State diagram of a Poisson process

2.3.2. Markovian Arrival Processes

Markovian Arrival Processes (MAPs) are a generalization of PH distributions that can model dependencies between consecutive arrivals. MAPs have been introduced in [119, 120] and have originally been developed as input processes for queuing systems that can be solved analytically [110], but may as well be used in simulation models.

Consider a Markov process $\{N(t), J(t)\}$ on the state space $\{(i, j), i \geq 0, 1 \leq j \leq n\}$. $N(t)$ is a counting process for the number of arrivals in $(0, t]$ and $J(t)$ is the state of the underlying Markov chain. Then the Markov process $\{N(t), J(t)\}$ has an infinitesimal generator matrix $\boldsymbol{Q}^*$ with the form

$$\boldsymbol{Q}^* = \begin{bmatrix} \boldsymbol{D}_0 & \boldsymbol{D}_1 & \boldsymbol{D}_2 & \boldsymbol{D}_3 & \dots \\ 0 & \boldsymbol{D}_0 & \boldsymbol{D}_1 & \boldsymbol{D}_2 & \dots \\ 0 & 0 & \boldsymbol{D}_0 & \boldsymbol{D}_1 & \dots \\ 0 & 0 & 0 & \boldsymbol{D}_0 & \dots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

where the $\boldsymbol{D}_k, k \geq 0$ are $n \times n$-matrices [103]. The $\boldsymbol{D}_k, k \geq 1$ are nonnegative matrices and $\boldsymbol{D}_0$ is defined in a similar way as for PH distributions. It has nonnegative off-diagonal elements and negative diagonal elements that are given by

$$\boldsymbol{D}_0(i, i) = -\left(\sum_{j, j \neq i} \boldsymbol{D}_0(i, j) + \sum_{k, k \geq 1} \left(\sum_j \boldsymbol{D}_k(i, j)\right)\right).$$

$\boldsymbol{D}_0$ contains the transition rates that do not correspond to an arrival (silent transitions). The matrices $\boldsymbol{D}_k, k \geq 1$ contain transition rates that are accompanied by an arrival, where the index k equals the batch size of the arrival.

The behavior of a MAP can be described as follows: The MAP stays in state i for an exponentially distributed time with rate $-\boldsymbol{D}_0(i, i)$. Afterwards one of the transitions from the matrices $\boldsymbol{D}_i, i = 0, \dots, k$ occurs. As already mentioned, transitions from $\boldsymbol{D}_0$ are not associated with an arrival and only change the state of the underlying Markov chain, i.e. with probability $\boldsymbol{D}_0(i, j) / (-\boldsymbol{D}_0(i, i))$ the MAP changes to state j. Transitions from the matrices $\boldsymbol{D}_k, k \geq 1$ are accompanied by an arrival, i.e. an entry $\boldsymbol{D}_k(i, j)$ specifies the rate of a transition from state i to state j that is associated with an arrival of batch size k. Consequently, with probability $\boldsymbol{D}_k(i, j) / (-\boldsymbol{D}_0(i, i))$ the MAP changes to state j and generates an arrival with batch size k.

If the number of arrivals $N(t)$ is omitted in the process description one obtains the Markov chain $\{J(t)\}$ with infinitesimal generator matrix [143]

$$\boldsymbol{Q} = \boldsymbol{D}_0 + \sum_{k \geq 1} \boldsymbol{D}_k.$$

The above definitions describe a very general form of the Markovian Arrival Process. It allows for the modeling of batch arrivals resulting in a Batch Markovian Arrival Process (BMAP) [97]. In the more common form the matrices $\boldsymbol{D}_k, k \geq 2$ are zero resulting in a MAP defined by the matrix pair $(\boldsymbol{D}_0, \boldsymbol{D}_1)$, which will be assumed for the following observations. A simple example for a $MAP(2)$, i.e. a MAP with 2 states,

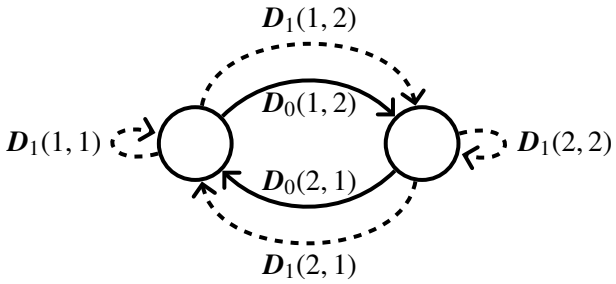


Figure 2.13.: Example MAP with 2 states

is shown in Figure 2.13 in which solid lines denote transitions not associated with an arrival and dashed lines transitions that are accompanied by an arrival.

As already mentioned, we have that $\boldsymbol{D}_0(i,i) < 0$, $\boldsymbol{D}_0(i,j) \geq 0$ for $i \neq j$ and $\boldsymbol{D}_1(i,j) \geq 0$ for all i, j. A further constraint on the row sums for a MAP is $\boldsymbol{D}_0\mathbf{1} = -\boldsymbol{D}_1\mathbf{1}$. Moreover, the matrix $\boldsymbol{D}_0$ is nonsingular. Recall, that matrix $\boldsymbol{Q} = \boldsymbol{D}_0 + \boldsymbol{D}_1$ is an irreducible infinitesimal generator matrix of the underlying Markov chain. Thus, the vector $\boldsymbol{\pi}'$ that is the solution of $\boldsymbol{\pi}'\boldsymbol{Q} = \mathbf{0}$ and $\boldsymbol{\pi}'\mathbf{1} = 1$ describes the stationary distribution of the underlying CTMC of a MAP. The arrival instances generated by a MAP form a DTMC with transition probability matrix $\boldsymbol{P} = (-\boldsymbol{D}_0)^{-1}\boldsymbol{D}_1$ [147]. Since the irreducible matrix $\boldsymbol{P}$ describes the transition probabilities of a DTMC, a steady state vector $\boldsymbol{\pi}$ exists for this embedded process that is the solution of $\boldsymbol{\pi} = \boldsymbol{\pi}\boldsymbol{P}$ and $\boldsymbol{\pi}\mathbf{1} = 1$ and describes the distribution just after an arrival. The steady state distribution of the underlying CTMC of a MAP and the steady state distribution of the DTMC embedded into the arrival instances are related as [84]

$$\boldsymbol{\pi}' = \frac{\boldsymbol{\pi}(-\boldsymbol{D}_0)^{-1}}{\boldsymbol{\pi}(-\boldsymbol{D}_0)^{-1}\mathbf{1}} \quad \text{or equivalently in [40]} \quad \boldsymbol{\pi} = \frac{\boldsymbol{\pi}'\boldsymbol{D}_1}{\boldsymbol{\pi}'\boldsymbol{D}_1\mathbf{1}}.$$

In steady-state the interarrival times of a MAP are Phase-type distributed with initial probability vector $\boldsymbol{\pi}$ and generator matrix $\boldsymbol{D}_0$. Then, the probability density function and the distribution function of the interarrival times of a MAP are given by

$$f(x) = \boldsymbol{\pi}e^{\boldsymbol{D}_0 x}\boldsymbol{D}_1\mathbf{1} \qquad \text{and} \qquad F(x) = 1 - \boldsymbol{\pi}e^{\boldsymbol{D}_0 x}\mathbf{1}.$$

Since the interarrival times of a MAP are distributed according to a PH distribution, the i-th moment μ_i of the interarrival times is computed according to Equation 2.11. Since the arrivals generated by a MAP are not independent one can state the joint density of the interarrival times $X_0, X_1, \ldots, X_k$ that is given by

$$f(x_0, x_1, \ldots, x_k) = \boldsymbol{\pi}e^{\boldsymbol{D}_0 x_0}\boldsymbol{D}_1 e^{\boldsymbol{D}_0 x_1}\boldsymbol{D}_1 \ldots e^{\boldsymbol{D}_0 x_k}\boldsymbol{D}_1\mathbf{1}$$

and the joint moments of the $j_0 = 0 < j_1 < j_2 < \ldots < j_k$-th interarrival times are

$$E[X_0^{i_0} X_{j_1}^{i_1} \ldots X_{j_k}^{i_k}] = \boldsymbol{\pi} i_0! (-\boldsymbol{D}_0)^{-i_0} \boldsymbol{P}^{j_1 - j_0} i_1! (-\boldsymbol{D}_0)^{-i_1} \ldots \boldsymbol{P}^{j_k - j_{k-1}} i_k! (-\boldsymbol{D}_0)^{-i_k} \mathbf{1}.$$

Aside from the joint moments the autocorrelation can be used to express the dependence between arrivals that are lag-k apart. For MAPs the autocorrelation is computed by [121]

$$\rho_k = \frac{\mu_1^{-2} \boldsymbol{\pi} (-\boldsymbol{D}_0)^{-1} \boldsymbol{P}^k (-\boldsymbol{D}_0)^{-1} \mathbf{1} - 1}{2\mu_1^{-2} \boldsymbol{\pi} (-\boldsymbol{D}_0)^{-1} (-\boldsymbol{D}_0)^{-1} \mathbf{1} - 1}.$$

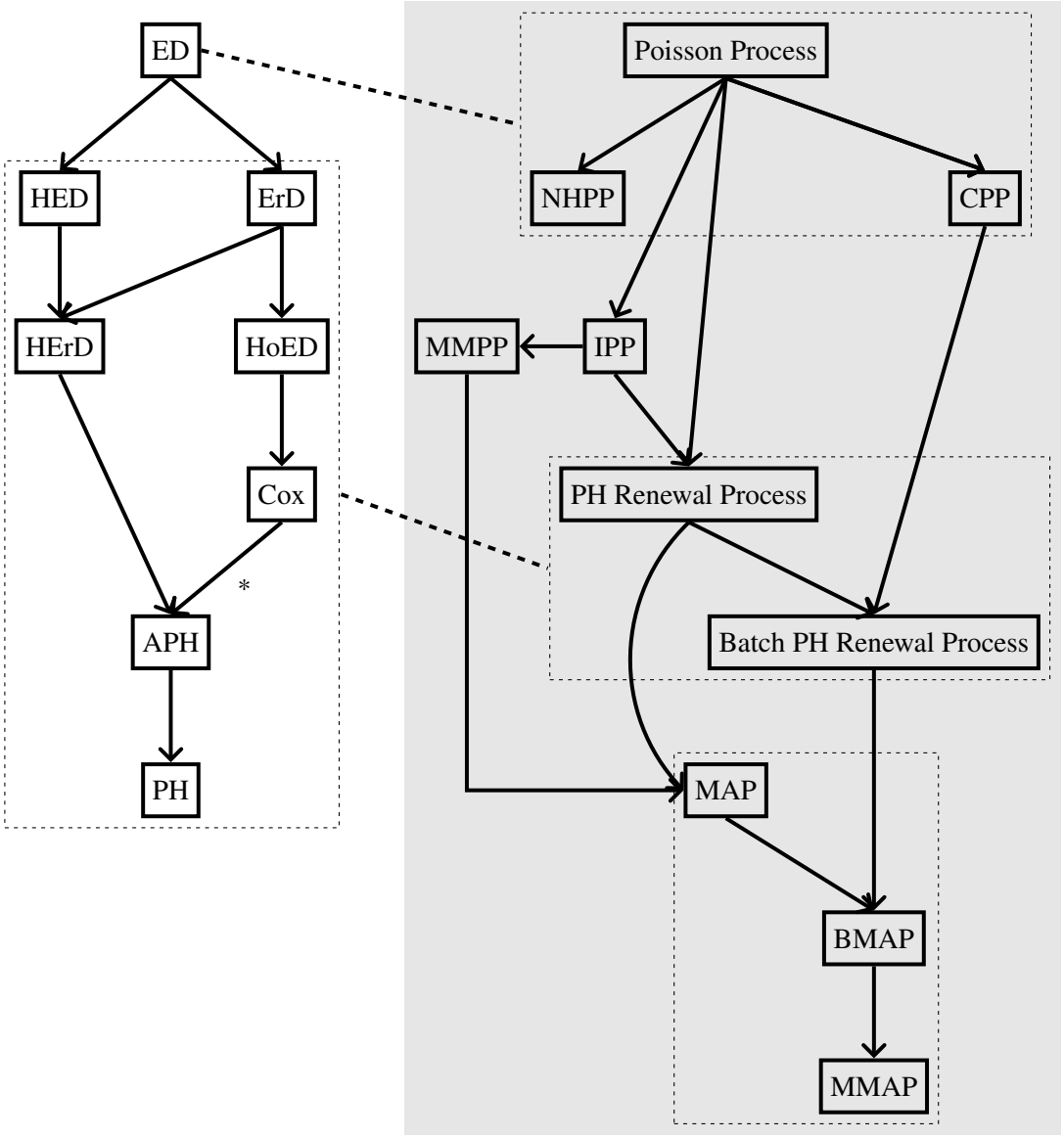
Although MAPs do not allow one to describe long range dependencies, they can be applied to approximate long range dependent processes on every finite time scale arbitrarily close [80].

MAPs are a very versatile class of processes containing several well-known stochastic processes as subclasses. Every Phase-type distribution $(\boldsymbol{D}_0', \boldsymbol{\pi})$ can be described by a MAP $(\boldsymbol{D}_0, \boldsymbol{D}_1)$ by setting $\boldsymbol{D}_0 = \boldsymbol{D}_0'$ and $\boldsymbol{D}_1 = (-\boldsymbol{D}_0'\mathbf{1})\boldsymbol{\pi}$. Then, $\boldsymbol{\pi}$ is the steady-state vector of matrix $\boldsymbol{P} = -\boldsymbol{D}_0^{-1}\boldsymbol{D}_1$. Another special case of MAPs are Markov-Modulated Poisson-Processes (MMPP) [66], which combine a Poisson process and an auxiliary Markov process $\{M(t)\}_{t=0}^{\infty}$ with states $1, 2, \ldots, n$ and generator matrix $\boldsymbol{Q}_{MMPP}$. When the Markov process is in state i arrivals occur according to the Poisson process at rate λ_i. When the state of the auxiliary process changes to state j arrivals occur at rate λ_j. Setting $\boldsymbol{D}_0 = \boldsymbol{Q}_{MMPP} - \Delta(\boldsymbol{\lambda})$ and $\boldsymbol{D}_1 = \Delta(\boldsymbol{\lambda})$, where $\Delta(\boldsymbol{\lambda})$ is the diagonal matrix with λ_i values in the diagonal, the MMPP can be represented by a MAP [103]. The interrupted Poisson process (IPP) is a special case of the MMPP with two phases and $\lambda_2 = 0$, i.e. in the second state no arrivals occur. However, the interrupted Poisson process can be seen as a PH renewal process with hyper-exponential renewal times as well.
On the other hand MAPs can be generalized in several ways. BMAPs have already been mentioned above. Directly related to BMAPs are MMAPs (Multiclass MAPs) with marked arrivals [4, 41, 72, 73]. For MMAPs the classes do not necessarily correspond to a batch size but can have an arbitrary meaning like e.g. different customer classes. Another different generalization are RAPs (Rational Arrival Processes) [3], also denoted as MEPs (Matrix Exponential Processes) [31, 32], for which the interarrival time is matrix exponential. MEPs share the drawbacks of these distributions and thus, lack the descriptive interpretation of an underlying Markov chain.

2.3.3. PH and MAP Hierarchy

In the previous sections some of the relations between different subclasses of PH distributions and MAPs have already been mentioned. In the following these relations will be established in a systematic way resulting in a hierarchy of the different classes as shown in Figure 2.14. On the left several distributions belonging to the class of Phase-type distributions and their relations are shown. The right part of Figure 2.14 shows the relationship of several processes belonging to the class of Markovian Arrival Processes.

As already mentioned before the exponential distribution (ED) is the simplest distribution belonging to the class of Phase-type distributions. It is generalized by the Hyper-Exponential distribution (HED), which consists of n alternate phases each being exponentially distributed, and the Erlang distribution (ErD), which consists of n



ED: Exponential distribution	HED: Hyper-Exponential distribution
ErD: Erlang distribution	HErD: Hyper-Erlang distribution
HoED: Hypo-Exponential distribution	Cox: Cox distribution
APH: Acyclic Phase-type distribution	PH: Phase-type distribution
NHPP: Non-homogenous Poisson Process	CPP: Compound Poisson Process
IPP: Interrupted Poisson Process	MMPP: Markov Modulated Poisson Process
BMAP: Batch MAPs	MMAP: Multiclass MAPs

Figure 2.14.: PH and MAP hierarchy

sequential phases. Thus, for $n = 1$ both the Hyper-Exponential and the Erlang become an exponential distribution.

Both, Hyper-Exponential and Erlang distribution, are generalized by the Hyper-Erlang distribution (HErD), which consists of m alternate branches each having $S_i, 1 \leq i \leq m$ sequential phases. Hence, a HErD is an Erlang distribution for $m = 1$ and an Hyper-Exponential distribution for $S_i = 1, i = 1, \ldots m$. Another generalization of the Erlang distribution is the Hypo-Exponential distribution (HoED). A HoED consists of n sequential phases like the ErD, but for a HoED each of the exponentially distributed phases may have a different parameter $\lambda_i, 1 \leq i \leq n$. Thus, if all the λ_i are equal the HoED becomes an ErD. The Cox distribution is similar to the HoED consisting of a sequence of exponentially distributed phases. The difference is, that for a Cox distribution not all the phases have to be taken, i.e. the absorbing state can be reached from every state. The most general Phase-type distributions are the acyclic Phase-type-distribution and the Phase-type distribution itself.

The relation between the Cox distribution and the APH distribution needs some further explanation: It is known, that the class of Cox distributions and the class of acyclic PH distributions are identical [55, 123]. More precisely the Cox distribution is equivalent to one of the canonical forms of APH distributions, which will be presented in Section 2.3.4. Nevertheless they are treated as two classes in Figure 2.14 with the Coxian distributions being a subclass of APH distributions to reflect the fact, that any representation of a Cox distribution is an APH representation by definition, while a representation of an APH distribution is not a valid Coxian representation by default, although applying a transformation into the canonical form one can construct an equivalent representation that is a Coxian representation.

The distributions on the left of Figure 2.14 serve as interarrival time distributions for the processes on the right. For example the Poisson process has exponentially distributed interarrival times. Generalizations of the Poisson process include the Compound Poisson process with batch arrivals (requiring a second distribution for the batch sizes) and the non-homogeneous Poisson process (NHPP) where the rate can change depending on the time.

Since the exponential distribution is a special case of the Phase-type distribution, the PH renewal process with Phase-type distributed interarrival times is a generalization of the Poisson process. The same holds for the Compound Poisson process and the batch Phase-type renewal process. Additionally, the batch Phase-type renewal process is of course a generalization of the Phase-type process with single arrivals.

The interrupted Poisson process (IPP) consists of two states: In the on-state it behaves like a normal Poisson process while in the off-state no arrivals occur. The IPP can be seen as a special case of a Phase-type renewal process with Hyper-Exponential interarrival times. However, the IPP is a special case of the Markov-Modulated Poisson-Process (MMPP) as well, which combines several Poisson processes and the active Poisson process is chosen by the state of an associated Markov chain. Then the IPP is a MMPP with two states and the rate $\lambda_2 = 0$.

Finally the Markovian Arrival Process (MAP) generalizes the Phase-type renewal process and the MMPP. The most general types of process forms are the Batch Markovian Arrival Process (BMAP) allowing for batch arrivals, which extends the MAP and the Batch PH renewal process, and the Multiclass MAP where the arrivals belong to an arbitrary class, which does not necessarily correspond to a batch size.

2.3.4. Canonical Representations for Phase-Type Distributions and Markovian Arrival Processes

Before a brief overview of the existing fitting approaches for PH distributions and MAPs is presented in the following two sections, canonical representations of those distributions and processes will be introduced.

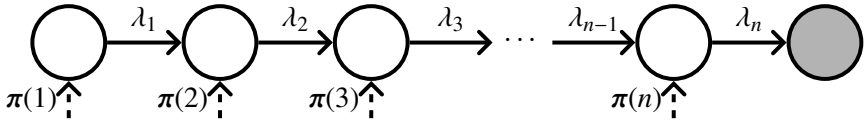
It is well known that the matrix representations of PH distributions and MAPs are not unique, i.e. there are different matrices (or matrices and vectors in the case of PH distributions) that describe the same distribution or the same MAP, respectively [123]. Moreover, it is known, that a PH distribution of order n is determined by at most $2n$ independent parameters[2] [124, 125] and a MAP of order n by at most n^2 independent parameters [147]. Since $(\boldsymbol{\pi}, \boldsymbol{D}_0)$ contains $n^2 + n$ free parameters for PH distributions[3] and $(\boldsymbol{D}_0, \boldsymbol{D}_1)$ contains $2n^2 - n$ free parameters for a MAP, these representations are highly redundant, which makes fitting those parameters cumbersome, because the fitting algorithm has to deal with more parameters than necessary and might switch between different representations of the same distribution. This motivates the search for canonical representations for PH distributions and MAPs that only consist of the minimal number of free parameters. Unfortunately canonical representations only exist for subclasses or lower order models of PH distributions or MAPs.

The oldest results in this area are due to Cumani [55] and cover canonical forms for acyclic PH distributions. [55] presents three types of equivalent canonical forms with $2n-1$ free parameters and an algorithm to transform any APH representation into canonical form. The different canonical forms are shown in Figure 2.15. In all three cases it is assumed that $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$. The series canonical form from Figure 2.15(a) has only one exit state, but all states may be entry states. Canonical form A from Figure 2.15(b) has only one entry state, but two exit states. The equivalence of the series canonical form and canonical form A becomes immediately apparent if one sets $g_i = \boldsymbol{\pi}(i)$. In canonical form B every state may be an exit state, but there is only one entry state. Observe, that the canonical form B from Figure 2.15(c) is a Cox distribution as shown in Figure 2.10. The crucial ideas for the transformation of an APH representation into canonical form stem from equivalent representations of the exponential distribution as shown in Figure 2.16 and the representation of an APH by a set of elementary series as shown in Figure 6.1[4]. Each elementary series has a probability proportional to the product of the transition rates along the corresponding path and to the initial probability of the first state of the path [55]. Using equivalent representations of the exponential distribution an elementary series containing a state with rate λ can be substituted by a mixture of two series, where one contains a state with rate $\mu > \lambda$ and one contains states with the rates λ and μ. Repeated application of this substitution results in a mixture of so-called basic series $BS_i = (\lambda_n, \lambda_{n-1}, \ldots, \lambda_i)$ that together with appropriate initial probabilities finally yield the canonical representation shown in Figure 2.15(a).

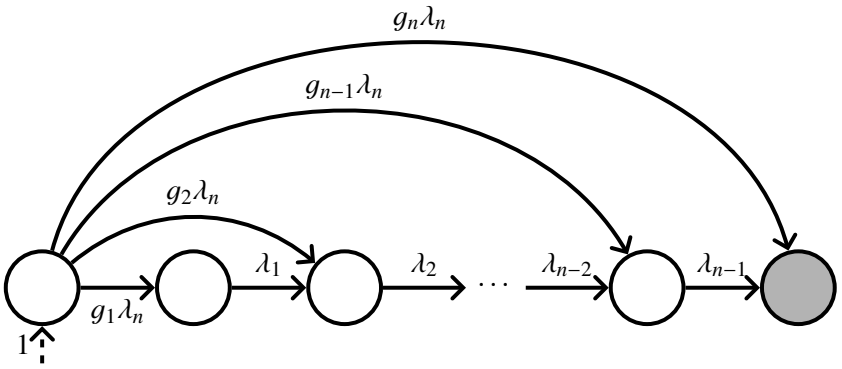
[2] If one does not allow a non-zero probability for the absorbing state, i.e. if $\boldsymbol{\pi}\mathbf{1} = 1$, the number of parameters reduces to $2n - 1$.

[3] Again, if one does not allow a non-zero probability for the absorbing state this number reduces to $n^2 + n - 1$.

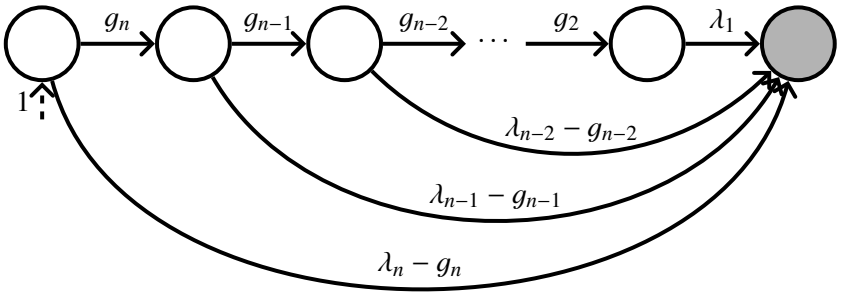
[4] Elementary series will become important for the construction of the stochastic process in Chapter 6 where the interested reader will also find the referenced figure and an example for the construction of the elementary series for a given APH.



(a) Series canonical form



(b) Canonical form A



(c) Canonical form B

Figure 2.15.: Canonical forms of acyclic PH distributions

For the general case of (cyclic) PH distributions canonical forms are only known up to order 3. PH distributions of order 2 can be transformed into an equivalent acyclic PH distribution [53], which makes the results from [55] applicable. For PH distributions of order 3 a canonical representation has been developed in [82, 83] that distinguishes three different cases.

Yet fewer results are available for MAPs. In fact, canonical forms are only known for the class of $MAP(2)$. In [31] it was shown that the classes $MAP(2)$ and $MEP(2)$ are equivalent and a canonical form was introduced that splits into two cases depending on the correlation parameter of the MAP.

While canonical forms based on the matrix representation are only available for some subclasses, it is possible to define all PH distributions and MAPs in terms of their moments and joint moments. Using some results from [157] it was shown in [32, 147] that a PH distribution is uniquely defined by the first $2n$ moments and a MAP by the

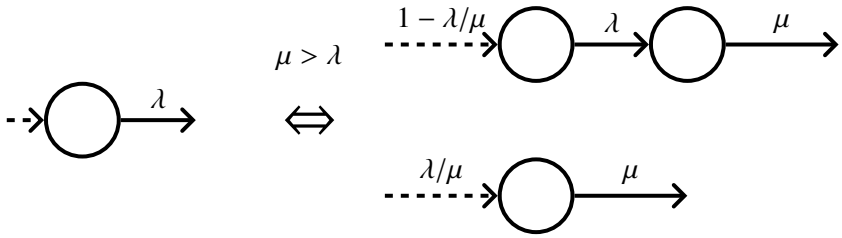


Figure 2.16.: Equivalent representations of the exponential distribution

first n^2 moments and joint moments. Higher moments can be computed from these moments [32]. Although the moments characterization is unique and minimal, it is not really suitable for application in models, which require the matrix representation to generate random numbers or to numerically analyze the model. [147] contains a optimization algorithm to obtain a matrix representation of a MAP or PH distribution from a given set of (joint) moments, but since the bounds for a feasible set of moments are not known in general for MAPs, one cannot tell if a failure of the algorithm really implies that the moments do not describe a MAP. Moreover, the algorithm is nonlinear such that it need not converge to a valid MAP representation even if one exists.

2.3.5. Fitting Methods for Phase-Type Distributions

For fitting Phase-type distributions basically two types of approaches exist. The first class are Expectation Maximization (EM) algorithms [28, 113], which try to maximize the likelihood and work on the complete trace. For the second class some characteristics like moments are derived from the trace and the PH distribution is fitted to these characteristics. Most of these approaches are tailored to a specific subclass of PH distributions to benefit from an available canonical representation or certain properties of that subclass.

The EM algorithm [57, 92, 135] is a general method for finding the maximum-likelihood estimate of parameters of an underlying distribution from a given dataset. For a given density function $f(\boldsymbol{x}|\boldsymbol{\Theta})$ with a set of parameters $\boldsymbol{\Theta}$ and a given trace $\mathcal{T} = \{t_1, t_2, \ldots, t_l\}$ of size l where the t_i are assumed to be independent and identically distributed with density f the function

$$\mathcal{L}(\boldsymbol{\Theta}|\mathcal{T}) = f(\mathcal{T}|\boldsymbol{\Theta}) = \prod_{i=1}^{l} f(t_i|\boldsymbol{\Theta}) \tag{2.16}$$

is called the likelihood function. In many cases the log-likelihood $log(\mathcal{L}(\boldsymbol{\Theta}|\mathcal{T}))$ is used instead of Equation 2.16, because the sum resulting from applying the logarithm to Equation 2.16 is analytically easier to compute than the product. The EM algorithm can be used to solve a maximum likelihood problem for which one aims at finding the $\boldsymbol{\Theta}^*$ that maximizes $\mathcal{L}$, i.e. one is interested in

$$\boldsymbol{\Theta}^* = \arg\max_{\boldsymbol{\Theta}} \mathcal{L}(\boldsymbol{\Theta}|\mathcal{T}).$$

The EM algorithm works in two steps, an expectation (E) step and a maximization (M) step, which are iteratively repeated, and increases the likelihood in each iteration.

In the past EM algorithms have been applied for fitting several subclasses of PH distributions. For the general class of PH distributions Equation 2.16 becomes

$$\mathcal{L}((\boldsymbol{D}_0, \boldsymbol{\pi})|\mathcal{T}) = \prod_{i=1}^{l} \left(\boldsymbol{\pi} e^{t_i \boldsymbol{D}_0}(-\boldsymbol{D}_0 \mathbf{1}\right).$$

In [5] an EM algorithm was used to fit a PH distribution to a trace. Since this approach is rather slow for general PH distributions, several modifications have been proposed that increase the performance of the algorithm [45] or that only consider a subclass of PH distributions for fitting. [95] tailored the EM algorithm for fitting Hyper-Exponential distributions. [137] used Hyper-Exponential distributions as well, but partitioned the entire range of values and fitted a Hyper-Exponential distribution to each partition using an EM algorithm. In [151] a variant of the EM algorithm was used for the fitting of Hyper-Erlang distributions. This approach was further extended in [129], such that not the complete trace has to be considered for fitting, but only an aggregated trace with a much smaller number of weighted elements. The key idea of this approach is to divide the empirical distribution of the trace data in a predefined number of intervals and represent the elements of an interval by their mean value and a weight given by the portion of elements in that interval. Hyper-Erlang distributions will be used as marginal distributions for the stochastic processes introduced in Chapter 5 and hence, the approach from [151] will be discussed in detail in Section 5.3.

Aside from EM approaches other heuristic techniques have been applied to fit the parameters of an acyclic PH distribution according to an empirical density function [29, 81].

[64] proposed an recursive heuristic approach to fit the parameters of a Hyper-Exponential distribution by first fitting the rightmost portion of the tail of the empirical distribution to a single exponential distribution. After that another exponential distribution is fitted to the rightmost portion of a new empirical distribution that is obtained by removing the part that was already fitted in the previous step and so on.

Finally, an acyclic PH distribution can be fitted to a set of moments estimated from a trace [42] by iteratively optimizing according to the initial probabilities $\boldsymbol{\pi}$ and the rates λ_i resulting in an APH in canonical form as shown in Figure 2.15(a). The approach from [42] will be used to fit the APH marginal distribution for the stochastic processes presented in Chapter 6 and is presented in a detailed manner in Section 6.2.

2.3.6. Fitting Methods for Markovian Arrival Processes

Similar to PH distributions fitting algorithms for MAPs can be distinguished according to the amount of data they use, i.e. the complete trace or only some characteristics computed from the trace, and according to the type of process they fit, i.e. a MAP or some subclass.

If the complete trace data is used for fitting, usually an EM algorithm is applied. For MAPs the likelihood is given as

$$\mathcal{L}((\boldsymbol{D}_0, \boldsymbol{D}_1)|\mathcal{T}) = \boldsymbol{\pi}\left(\prod_{i=1}^{l} e^{t_i \boldsymbol{D}_0} \boldsymbol{D}_1\right)\mathbf{1}.$$

EM algorithms have been used for fitting Markov Modulated Poisson Processes [142] and arbitrary MAPs. In the latter case approaches exist that fit the complete MAP in a single step [40] or that use two steps to fit a PH distribution with one EM algorithm and expand this distribution into a MAP, i.e. find a matrix $\boldsymbol{D}_1$ while preserving the given distribution, using a second EM algorithm [44]. EM algorithms have also been applied for fitting Batch Markovian Arrival Processes [36, 97]. While empirical observations indicate that EM algorithms yield the best results for MAP fitting currently available [100], they are also by far more costly than other approaches that only use some characteristics estimated from the trace and depending on the trace size and the MAP order they may run for several hours.

The faster algorithms, which do not use the complete information from the trace, are usually performed in two steps and separate distribution fitting and fitting of the dependence between consecutive arrivals. For distribution fitting any of the approaches introduced in Section 2.3.5 may be applied. From this first step one obtains a PH distribution $(\boldsymbol{\pi}, \boldsymbol{D}_0)$ and aims at finding a matrix $\boldsymbol{D}_1$ resulting in a MAP $(\boldsymbol{D}_0, \boldsymbol{D}_1)$ in the second step, such that $\boldsymbol{\pi}$ is the steady-state vector of the embedded Markov Chain after an arrival and that approximates some estimated measure of dependence between arrivals. Possible measurements for the dependence are the lag-k autocorrelation coefficients or the joint moments.

If the joint moments are used as a measure of dependence the resulting fitting problem may be expressed as [42]

$$\min_{\boldsymbol{D}_1 : \boldsymbol{D}_1 \geq \boldsymbol{0}, \boldsymbol{D}_1 \boldsymbol{1} = -\boldsymbol{D}_0 \boldsymbol{1}, \boldsymbol{\pi}(-\boldsymbol{D}_0)^{-1} \boldsymbol{D}_1 = \boldsymbol{\pi}} \left(\sum_{(i,j) \in \mathcal{J}} (\kappa_{i,j} \frac{\nu_{ij}}{\hat{\nu}_{ij}} - \kappa_{i,j})^2 \right)$$

where $\mathcal{J}$ is a set of joint moments, $\hat{\nu}_{ij}$ and ν_{ij} are the joint moments from the trace and the MAP, respectively, and $\kappa_{i,j}$ are weights, e.g. to privilege lower order joint moments. This type of equation can be solved by standard approaches for non-negative least squares problems [106] and thus, fitting of joint moments is much faster than other approaches and usually only takes up to a few seconds. This approach can be generalized to fit MMAPs as well by considering class-specific joint moments instead of general joint moments [41]. Another approach for MAP fitting according to joint moments was proposed in [15] where a special class of MAPs, called Structured MAPs [9], is used to combine several PH distributions and a Markov Chain that specifies which PH distribution generates the next arrival and is determined using the joint moments into a MAP.

If instead of joint moments autocorrelation coefficients are considered for fitting, the optimization is more complicated and cannot be expressed as a non-negative least squares problem. Several general purpose optimization algorithms have been applied for fitting according to autocorrelations in the past. In [84] and [100] the Nelder-Mead [118] algorithm was used for constructing matrix $\boldsymbol{D}_1$. Since the surface of the goal function contains various local optima, it is necessary to provide several different starting points for the optimization to obtain a good fitting result. These initial solutions can for example be constructed by a simplex algorithm that finds the maximal possible value for each element of $\boldsymbol{D}_1$ that still respects the constraints given by the PH distribution from the first step. For the approach presented in [127, 128] an evolutionary algorithm was combined with an EM algorithm to fit a MAP. The EM algorithm

constructs a PH distribution $(\boldsymbol{\pi}, \boldsymbol{D}_0)$, which is used as input for a multi-objective evolutionary algorithm (MOEA) that fits matrix $\boldsymbol{D}_1$ according to the autocorrelations of a trace. In contrast to the previous two-step fitting approaches where the vector $\boldsymbol{\pi}$ is used to establish constraints for fitting of $\boldsymbol{D}_1$, the MOEA is allowed to modify the vector $\boldsymbol{\pi}$ (but preserves matrix $\boldsymbol{D}_0$). Hence, the algorithm has a greater flexibility for finding $\boldsymbol{D}_1$ than the previous approaches, but has to consider two objective functions, which assess the quality of autocorrelation fitting and the quality of distribution fitting.

There are a few approaches that try to construct the MAP directly from characteristics computed from a trace. [74] proposes an inverse characterization of acyclic MAPs of order 2 where the elements of the matrices $(\boldsymbol{D}_0, \boldsymbol{D}_1)$ are computed directly from moments and an autocorrelation parameter estimated from the trace. In [147] and [79] moments and joint moments were used to construct the matrices of a MAP or MMAP, respectively, of an arbitrary order. A general drawback of this approaches is that the characteristics from the trace might not be feasible for MAPs, i.e. no MAP (of the desired order) exists that exhibits the estimated characteristics. In these cases for example the approach from [147] might yield matrices that describe a Matrix Exponential Process, if a MEP exists that describes the desired characteristics, or it might even result in matrices that do not describe a stochastic process at all.

For PH distributions several approaches have been presented in Section 2.3.5 that exploit the simpler structure of a restricted subclass to increase the performance of the fitting algorithm. Similarly, approaches for MAP fitting have been proposed that only use a simpler subclass for fitting. An example for this is [116] where Markov-Modulated Poisson Processes are used to model internet traffic.

Finally, for the approach presented in [49, 50] several smaller MAPs of order 2 for which fitting is easier are combined into one larger MAP by Kronecker Product Composition. Since for MAPs of order 2 canonical forms exist and the boundaries for moments and autocorrelation coefficients are known special fitting algorithms have been proposed that exploit these known properties. In [30] optimization procedures are introduced that decompose the bounding surface into components that are easier to handle to improve the efficiency of algorithms that minimize the moments distance. Additionally, it is possible to approximate larger MAPs by a MAP of order 2 by minimizing the distance of the joint distribution functions using efficiently computable matrix expressions for a $MAP(2)$ [30].

For another overview on PH and MAP fitting the reader is referred to [80].

2.4. Other Approaches

In the past various other approaches have been proposed for input models that can capture autocorrelations. Most of them are based upon the idea to model the autocorrelation structure by some type of base process and transform this process to yield a better approximation of the empirical distribution. In the following a brief overview of these techniques is given including some arguments why they are apparently inferior to the processes presented in the previous sections and hence are disregarded for the empirical work in Chapter 3.

TES (Transform-Expand-Sample) processes [114] use a technique related to the

ARTA approach described in Section 2.2, though they are a little older. Similar to ARTA processes a sequence of random numbers with standard uniform distribution is generated, which is transformed into an arbitrary marginal distribution. TES processes can have positive or negative lag-1 autocorrelations, thus the TES processes are divided in two classes, TES^+ and TES^-, respectively. TES models consist of two stochastic processes, a foreground sequence and an auxiliary background sequence. Foreground sequences have the form

$$Y_t^+ = D(U_t^+), \quad Y_t^- = D(U_t^-)$$

where the so-called distortion D is a transformation from $[0, 1)$ to the real numbers. Using the inverse transform method with $D = F_Y^{-1}$ the foreground sequence can be distributed according to any marginal distribution where the computation of the inverse cdf is possible. U_t^+, U_t^- are background sequences defined by

$$U_t^+ = \begin{cases} U_0, & t = 0 \\ \left\langle U_{t-1}^+ + V_t \right\rangle, & t > 0 \end{cases} \qquad U_t^- = \begin{cases} U_t^+, & t \text{ even} \\ 1 - U_t^+, & t \text{ odd} \end{cases}$$

where U_0 is distributed uniformly on $[0, 1)$, the innovation sequence $\{V_t\}_{t=1}^{\infty}$ is a sequence of iid. random variables (independent of U_0) and angular brackets denote the following operation: $\langle a \rangle = a - \max\{\text{integer } n : n \leq a\}$.
The marginal distributions of $\{U_t^+\}$ and $\{U_t^-\}$ are both uniform on $[0, 1)$. Since ARTA and TES processes both rely on the inverse transform method they support the same set of distributions, but controlling the autocorrelation structure is more straightforward for ARTA processes, since for TES processes a modification of the autocorrelation is done indirectly by modifying the innovation sequence and the relation between the innovation sequence and the resulting autocorrelation is not obvious, while for ARTA processes the autoregressive coefficients of the base process are directly related to the autocorrelation.

[164] discusses several linear and non-linear transformations and its properties to transform a time series Z_t generated by a f-ARIMA model into another time-series Y_t. Translations (i.e. $Y_t = Z_t + c$) and scaling (i.e. $Y_t = cZ_t$) can be used to shift the time series or to spread or compress the marginal distribution according to some constant c. Non-linear transformations of the type

$$Y_t = c_1^{\frac{1-c_2}{c_2+c_3} Z_t}$$

for $c_1 > 1$, $0 < c_2 < 1$ and some small constant c_3 allow for more elaborate modifications of the time series. Of course combinations of these transformations are possible. The main problem of this approach is, that the transformations have some impact on the characteristics of the trace and to the best of the author's knowledge, no automated approach exists to find parameters c and c_i and a series of transformations such that the f-ARIMA model preserves the characteristics of a set of given observations. Hence, application of this approach is not straightforward and would require a lot of expert knowledge.

Chapter 3

Empirical Comparison of Stochastic Processes

In the following an empirical comparison of the stochastic processes presented in Chapter 2 is carried out to assess the use of those processes in simulation models. For that matter some advantages and disadvantages of those processes are pointed out that motivate the work in the following chapters.

When building input models for simulation one usually has some observations from a real system and tries to find a stochastic model that resembles the behavior of the original observations by fitting a process to this data. Hence, we have three basic requirements for the stochastic processes:

1. The fitting procedure should be performed in an automated manner (or at least with little user interaction).

2. The processes have to provide a suitable fitting quality regarding the distribution and the autocorrelation structure.

3. Sampling from the process, i.e. generating random numbers, should be possible in an easy and efficient way.

For MAPs, ARTA and ARMA processes several fitting approaches have been introduced in Chapter 2 that fulfill the first requirement.

Sampling is possible for all three types of stochastic processes in an efficient way. For MAPs this can be done by simulating the underlying Markov chain, which requires storing the two matrices $(\boldsymbol{D}_0, \boldsymbol{D}_1)$ and the current state and drawing random numbers from a uniform distribution to determine the next state and from an exponential distribution to determine the transition times. For ARMA processes the vectors with the AR and MA coefficients have to be stored. Additionally, the previous p elements of the time series and the previous q innovations have to be saved. ARMA processes require the possibility to draw random numbers from a normal distribution to determine the next innovation. For simulation of ARTA processes the inverse cdf of the marginal distribution has to be computed. The simulation of the ARMA base process has already been described above. Random number generation from different stochastic processes will be further addressed in Section 9.3.

The second requirement (assessment of the fitting quality) will be subject of this chapter. In the following an overview of the experiment setup is given, proceeded by a

presentation of fitting results for the above-mentioned stochastic processes, which has already been published partially in [11].

3.1. Experiment Setup

For the empirical study to assess stochastic processes and existing fitting algorithms several of the approaches that have already been introduced briefly in Chapter 2 are compared.

For MAP fitting the Expectation Maximization algorithm from [40] and the two-step approach from [127], which combines an EM algorithm for fitting the distribution and an evolutionary algorithm to expand the distribution into a MAP, are used. The former will be denoted `MAP EM` and the latter `MAP MOEA` in the following.

ARTA fitting is done by the two tools `ARTAFACTS` [48] and `ARTAFIT` [25]. `ARTAFIT` is able to fit the complete ARTA process consisting of the marginal distribution and the base process, but is limited to distributions from the Johnson family. `ARTAFACTS` assumes the marginal distribution to be given and only finds a base process. Hence, whenever possible `ARTAFIT` is used for ARTA models. In cases where these results showed to be inapplicable, `ARTAFACTS` is used with a manually fitted marginal distribution, i.e. a distribution where the maximum likelihood estimators for the parameters can easily be computed like for the exponential distribution.

ARIMA models have been fitted using the common statistical software `R` [51, 149], which provides several fitting methods like solving Yule-Walker equations, Maximum likelihood estimation or least squares approximation, which have been briefly introduced in Section 2.1.

For the experiments six traces have been selected. Four traces are synthetically generated (two generated by ARTA processes and two by MAPs). The intention of the synthetically generated traces is to check whether processes of one class are able to capture the behavior of processes of another class. The remaining two traces have been observed in a real system and help to evaluate the suitability of the processes and corresponding fitting methods for practical use. These traces have been taken from the Internet Traffic Archive [148] and are common benchmark traces for assessing fitting methods. The trace *BC-pAug89* [108] contains a million packet arrivals observed at the Bellcore Morristown Research and Engineering facility in August 1989. The trace *LBL-TCP-3* [130] contains two hours of TCP traffic from the Lawrence Berkeley Laboratory and was recorded in January 1994. For the empirical study both traces were normalized to mean 1.0.

In application areas like computer and communication networks one is usually interested in modeling interarrival times or packet sizes. Consequently, the stochastic processes should only generate positive values. For MAPs the interarrival times have a PH distribution and thus, MAPs always fulfill this requirement. For ARTA processes it depends on the marginal distribution F_Y. The Johnson distribution returned by `ARTAFIT` can in general be specified, such that it is only defined for $x \geq 0$. If `ARTAFIT` fails to return such a distribution, one can select a more adequate distribution (e.g. exponential) and use `ARTAFACTS` for fitting, which was done in our experiments. ARIMA models always assume a normal distribution and for our experiments we tried several easy to implement strategies to avoid the generation of negative values. The

obvious (and easy to implement) choices are to either ignore those values in a simulation model, i.e. delete them from the generated observations, to replace them with some non-negative (fixed) value like 0, or to use absolute values only. Another ap-

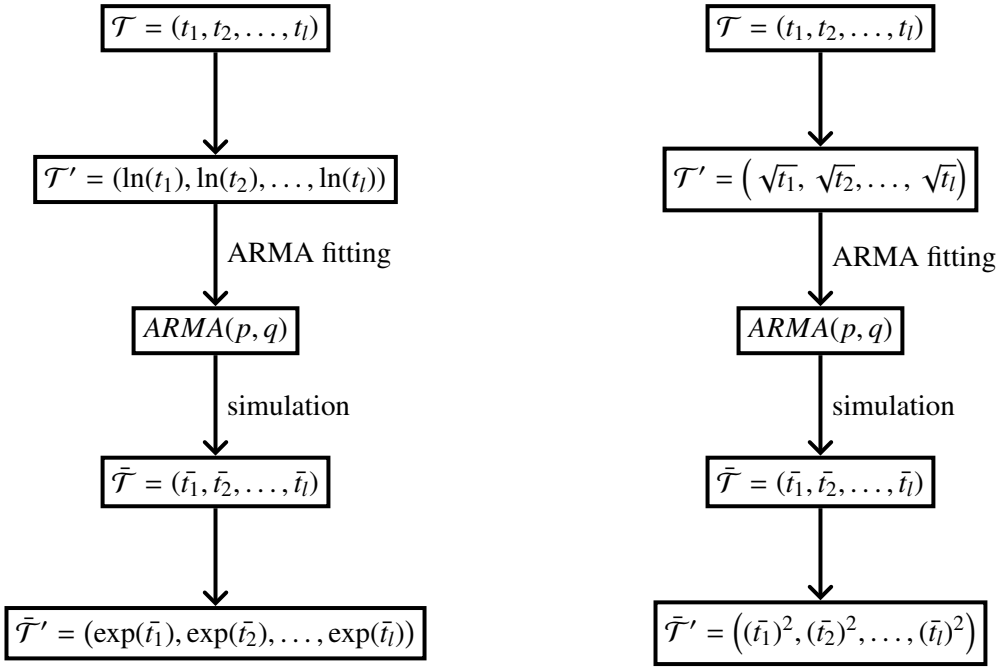


Figure 3.1.: Transformation steps for fitting ARMA processes

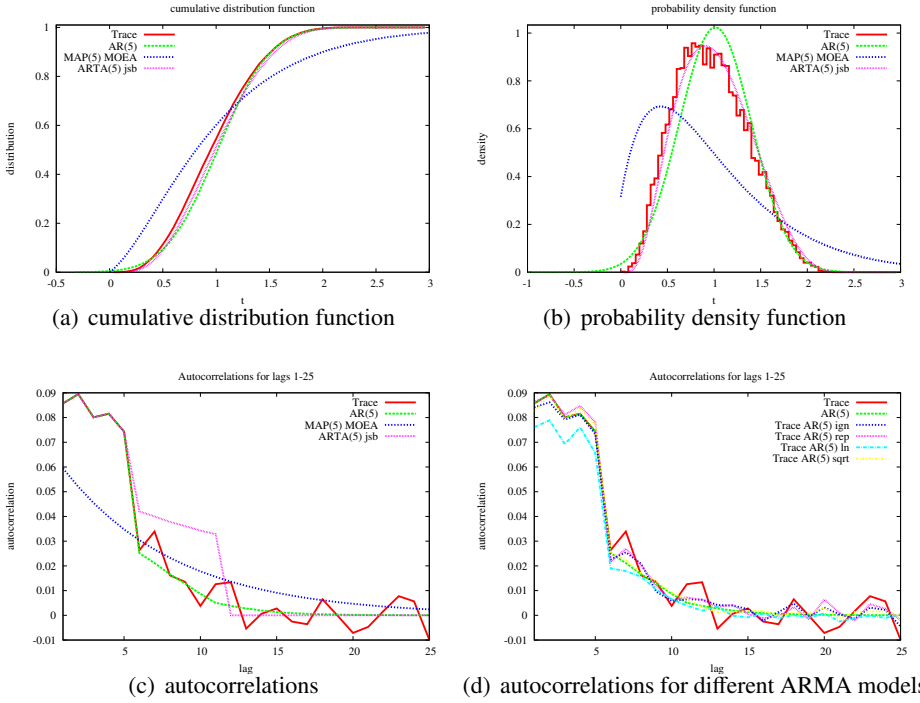
proach is a transformation of the time series as outlined in Figure 3.1. The original trace $\mathcal{T} = (t_1, t_2, \ldots, t_l)$ is transformed into $\mathcal{T}' = (\ln(t_1), \ln(t_2), \ldots, \ln(t_l))$ computing the natural logarithm of each trace element t_i. Then, an $ARMA(p, q)$ model is fitted to the transformed trace $\mathcal{T}'$. When simulating this ARMA model the generated trace $\bar{\mathcal{T}} = (\bar{t_1}, \bar{t_2}, \ldots, \bar{t_l})$ is again transformed into $\bar{\mathcal{T}}' = (\exp(\bar{t_1}), \exp(\bar{t_2}), \ldots, \exp(\bar{t_l}))$. Alternatively, the square root of each element in $\mathcal{T}$ is computed and the elements in $\bar{\mathcal{T}}'$ are obtained by computing the square of the elements in $\bar{\mathcal{T}}$. Of course this treatment of negative values has impact on the autocorrelation coefficients and the distribution of the generated observations, which will be demonstrated by several examples.

3.2. Experimental Results

In the following the results of the empirical comparison of the different stochastic processes are summarized. As already mentioned MAPs, ARMA and ARTA processes have been fitted to six different traces. To assess the fitting quality plots of the distribution and the autocorrelation structure for the traces and the fitted models are provided.

3.2.1. Traces generated by an ARTA Process

The first trace considered in the evaluation contains 20,000 observations from an ARTA process with $AR(5)$ base process and a Johnson bounded marginal distribution with shape parameters $\gamma = 0.6$ and $\delta = 1.4$, location parameter $\xi = 0.0$ and scale parameter $\lambda = 2.5$. The parameters of the Johnson distribution are chosen such that the distribution only yields positive values, which is, as already mentioned above, the case for most application areas of stochastic processes like packet sizes or interarrival or service times when simulating computer or communication networks.



(a) cumulative distribution function (b) probability density function

(c) autocorrelations (d) autocorrelations for different ARMA models

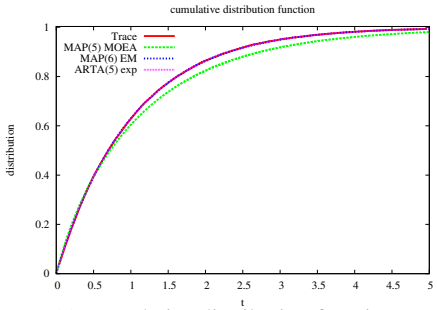
Figure 3.2.: Fitting results for a trace generated by an ARTA model with Johnson distribution

Since an $AR(5)$ base process was used for trace generation, an autoregressive process of order 5 was chosen for AR fitting. Because the original model has a marginal distribution from the Johnson family, `ARTAFIT` was used for obtaining an ARTA model, because it also tries to fit a Johnson marginal distribution and thus should provide good results for this type of trace. MAP fitting was done with `MAP MOEA` for a MAP of order 5. The fitting results are shown in Figure 3.2. As one can see `ARTAFIT` was able to recreate the original ARTA process and thus provided a good approximation of the distribution (Figures 3.2(a) and 3.2(b)) and the autocorrelation structure (Figure 3.2(c)). The AR model provides a good estimation of the autocorrelations too. Since AR models always assume normal marginal distribution, which is related to the Johnson distribution that was used for trace generation, the AR model captures the distribution as well. But in contrast to the Johnson distribution, which can be bounded such that $F(x) = 0$ for $x < 0$, a normal distribution is unbounded. Thus, negative val-

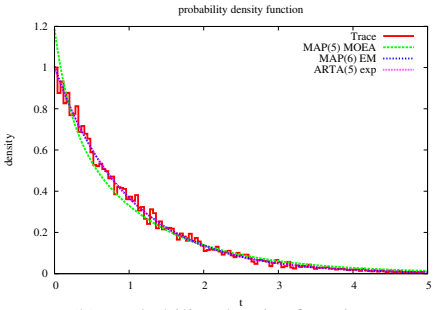
ues may be obtained if the fitted $AR(5)$ process is used in a simulation model as one can see from Figure 3.2(b). Figure 3.2(d) contains the curves for the autocorrelation of the original $AR(5)$ process and two traces generated from the AR model when ignoring negative values (Trace $AR(5)$ *ign*) or replacing them with 0 (Trace $AR(5)$ *rep*). Two additional traces have been generated by fitting an $AR(5)$ model to a transformed trace and by reversing the transformation when simulating the $AR(5)$ model as shown in Figure 3.1. For the trace $AR(5)$ *ln* the natural logarithm was used for transformation and the exponential function to reverse the transformation, for the trace $AR(5)$ *sqrt* the square root and the square have been used. For this example the effect on the autocorrelations is negligible, because only few negative values are generated by the model. In the following examples this problem will become more noticeable.

For MAP fitting this type of trace is actually a difficult task, because it has an untypical shape for a PH distribution. Although PH distributions can approximate the shape of normal-like distributions reasonably well [151], fitting becomes more sophisticated when autocorrelations have to be considered as well. While for ARTA processes the autocorrelation is independent of the distribution (as long as the desired autocorrelation is within the possible bounds for that distribution), the marginal distribution of a MAP has an immediate impact on the possible autocorrelation, i.e. matrix $\boldsymbol{D}_0$ and vector $\boldsymbol{\pi}$ determine the possible structure of matrix $\boldsymbol{D}_1$. If the sum of row i of $\boldsymbol{D}_0$ is equal to zero all elements in the corresponding row of $\boldsymbol{D}_1$ have to be zero as well. If an element $\pi_j = 0$ the j-th column of $\boldsymbol{D}_1$ is equal to zero. Although other representations $(\boldsymbol{\pi}', \boldsymbol{D}_0')$ for the marginal distribution that might be better suited for fitting the autocorrelation can be obtained by a transformation of the distribution [43], it might be difficult for the fitting algorithm to find this better representation, if the MAP is fitted in one step. Hence, for the example from Figure 3.2 one has to accept a trade-off between good distribution fitting and good fitting of the autocorrelation structure.

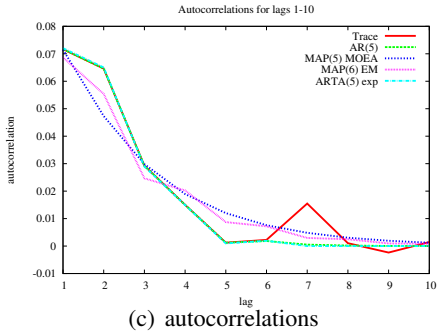
The second synthetically generated trace contains $20,000$ observations generated from an ARTA model with $AR(5)$ base process and exponential marginal distribution with rate parameter $\lambda = 1.0$. For AR and MAP fitting an autoregressive process of order 5 and MAPs of order 5 and 6 have been used, respectively. Since `ARTAFIT` had problems providing a good approximation for the exponential distribution in this case, `ARTAFACTS` was used, which can work with an exponential marginal distribution for autocorrelation fitting, and the exponential marginal distribution for the ARTA model was fitted manually. The fitting results for the distribution and the autocorrelation structure are shown in Figure 3.3. The curves for the distribution of the AR process have been omitted in Figures 3.3(a) and 3.3(b), since its marginal normal distribution is obviously not an adequate approximation of the empirical distribution of the trace. Instead, the cumulative distribution functions of different traces generated from transformed ARMA processes are shown in Figure 3.3(e). For the MAPs an exponential distribution is easy to fit and, of course, the ARTA model with manually fitted distribution provides a good approximation for the cumulative distribution and probability density functions. Regarding the autocorrelations of the trace all models provide good results as one can see from Figure 3.3(c), although the MAPs underestimate the correlation at lag 2. As already mentioned the AR model does not fit the distribution of the trace and similar to the previous example the simulation of the AR model yields negative values. Figure 3.3(d) shows the autocorrelation of traces generated from the AR



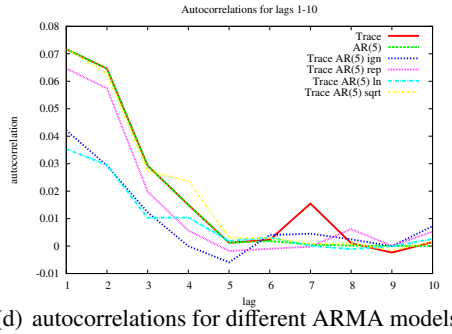
(a) cumulative distribution function



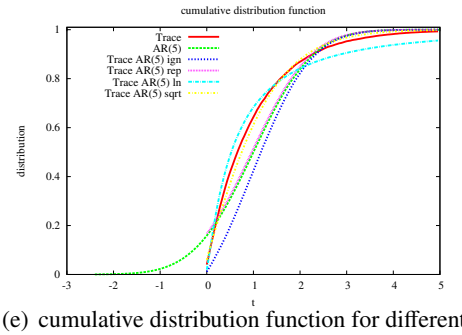
(b) probability density function



(c) autocorrelations



(d) autocorrelations for different ARMA models



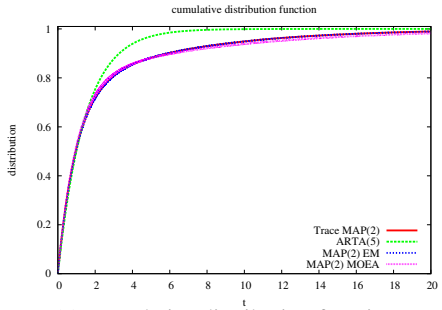
(e) cumulative distribution function for different ARMA models

Figure 3.3.: Fitting results for a trace generated by an ARTA model with exponential distribution

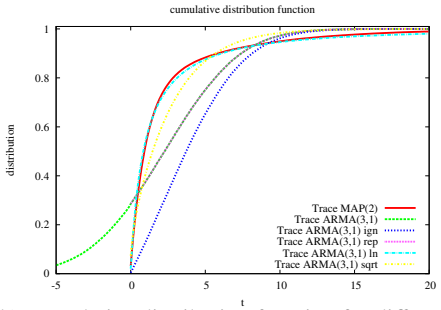
model when ignoring or replacing negative values or when transforming the trace according to Figure 3.1. While for the previous example the effect was insignificant, it is visible that the treatment of negative values has some serious impact on the autocorrelation in this case. The approach to fit an ARMA process to a trace that is transformed by computing the square root of each element preserves the autocorrelations best for this example.

3.2.2. Traces generated by a MAP

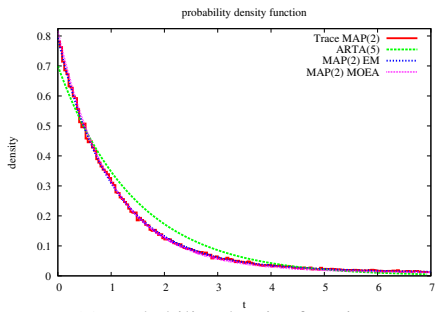
In the previous paragraph it was shown that it can be difficult for MAPs to capture the characteristics of an ARTA process, especially if the ARTA process has a marginal



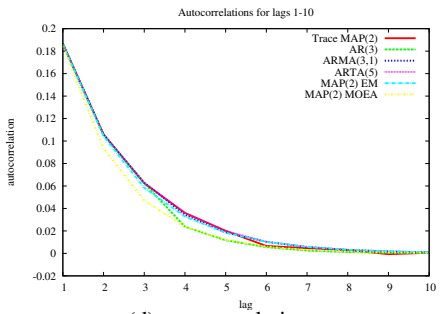
(a) cumulative distribution function



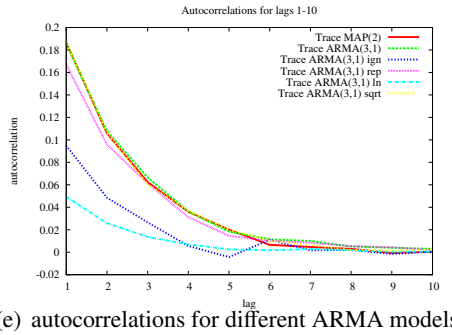
(b) cumulative distribution function for different ARMA models



(c) probability density function



(d) autocorrelations



(e) autocorrelations for different ARMA models

Figure 3.4.: Fitting results for a trace generated by a $MAP(2)$

distribution that is nontypical for a MAP. With the following two examples the opposite case is examined.

The first trace contains 200,000 elements and was generated from a $MAP(2)$ with matrices

$$\boldsymbol{D}_0 = \begin{bmatrix} -1.00 & 0.03 \\ 0.05 & -0.16 \end{bmatrix}, \qquad \boldsymbol{D}_1 = \begin{bmatrix} 0.90 & 0.07 \\ 0.01 & 0.10 \end{bmatrix}$$

and has non-zero autocorrelations for smaller lags only.

The results for the fitted MAPs and ARTA and ARMA models are shown in Figure 3.4. Since the original MAP does not exhibit a complicated autocorrelation structure, all fitted models were able to provide a good approximation for the autocorrelation (cf. Figure 3.4(d)). Furthermore, the MAPs captured the distribution as well,

while this was problematic for both ARTA and ARMA processes. The distribution of the ARMA models is, of course, normal again and the models yield negative values when used in simulation. The Johnson distribution returned by `ARTAFIT` has similar problems, though in general the Johnson distribution can take forms that only yield positive values. But this would require adding further constraints for the distribution fitting. Instead `ARTAFACTS` was used for autocorrelation fitting with a manually fitted exponential distribution. For the ARMA models it is not possible to fall back to different distributions and thus, one has to deal with the negative values in simulation. Figure 3.4(e) shows that ignoring negative values has a larger impact on the autocorrelation while replacing these values has much less effect. Again, fitting an ARMA model to the trace that is transformed using the square root has the smallest effect on the autocorrelation, although the trace resulting from the simulation and transformation of the corresponding ARMA process does not provide a good approximation of the distribution as one can see from Figure 3.4(b), which contains the cumulative distribution functions for the transformed traces resulting from the fitted $ARMA(3, 1)$ process. On the other hand, the alternative transformation of the trace using the natural logarithm resulted in a negative effect on the autocorrelation, while it provided a good approximation of the distribution.

The next trace contains $200,000$ elements generated by a $MAP(3)$ with autocorrelation up to lag 20 from [40]. The MAP is defined by the two matrices

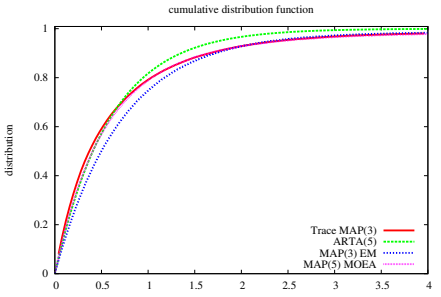
$$\boldsymbol{D}_0 = \begin{bmatrix} -3.721 & 0.500 & 0.020 \\ 0.100 & -1.206 & 0.005 \\ 0.001 & 0.002 & -0.031 \end{bmatrix}, \qquad \boldsymbol{D}_1 = \begin{bmatrix} 0.200 & 3.000 & 0.001 \\ 1.000 & 0.100 & 0.001 \\ 0.005 & 0.003 & 0.020 \end{bmatrix}.$$

The fitting results shown in Figure 3.5 are similar to the results of the previous trace generated from the $MAP(2)$. The MAP fitting methods were able to capture both distribution and autocorrelation again. ARTA fitting, which was done using `ARTAFACTS` with a manually specified distribution, provided a good approximation for the autocorrelations as well as one can see from Figure 3.5(c). Figures 3.5(d) and 3.5(e) show the impact of the different strategies to avoid negative values on the autocorrelation and the distribution function, respectively.

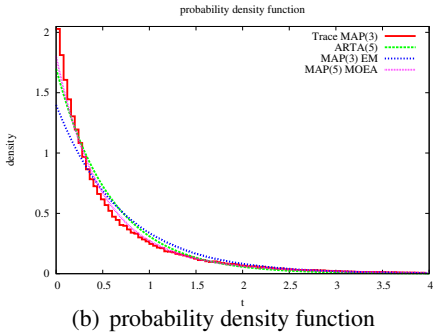
3.2.3. Real Traces

In the last step of this empirical evaluation the ability to approximate the behavior of traces that have been observed in a real system will be addressed. These traces are much harder to fit than the synthetically generated ones, since their empirical distribution function is more complicated and they exhibit autocorrelations over a large number of lags.

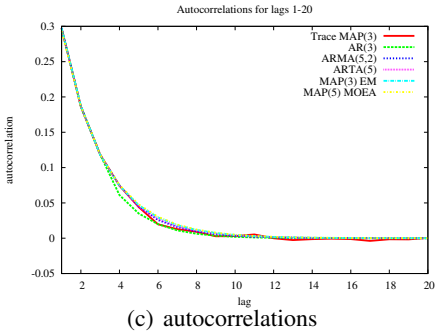
The first results have been obtained for the trace *BC-pAug89* and are shown in Figure 3.6. Figures 3.6(a) and 3.6(b) show the cumulative distribution function and the probability density function of the trace, of two MAPs fitted with the two different approaches mentioned before and of an ARTA model with manually chosen exponential distribution. For the same reasons as before we omitted curves for ARMA models in the plots. Although the shape of the empirical distribution function of the trace is difficult to fit, all three models provide a good approximation in terms of the cdf.



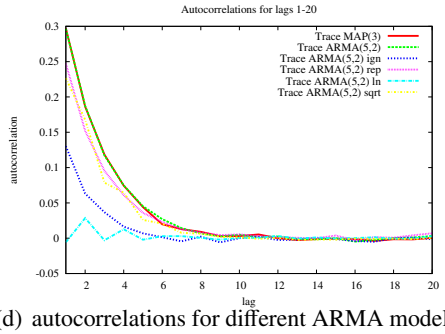
(a) cumulative distribution function



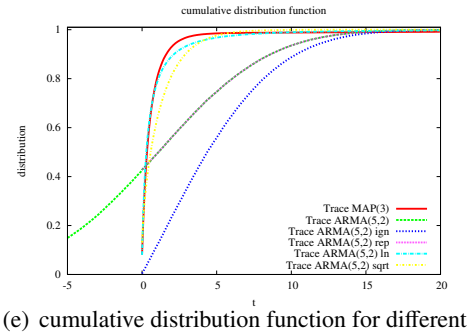
(b) probability density function



(c) autocorrelations



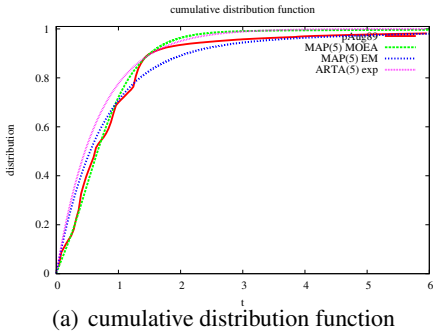
(d) autocorrelations for different ARMA models



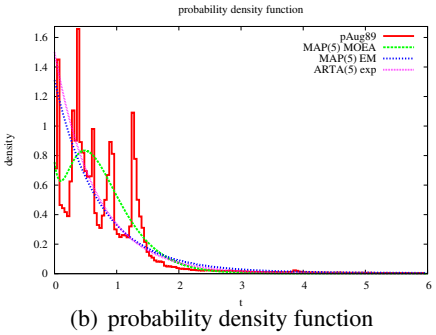
(e) cumulative distribution function for different ARMA models

Figure 3.5.: Fitting results for a trace generated by a $MAP(3)$

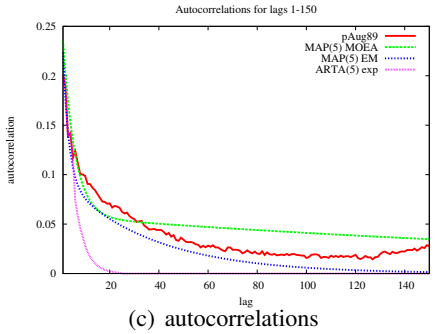
Figure 3.6(c) shows the autocorrelation for the MAPs and the ARTA process and Figure 3.6(d) the autocorrelation for different AR and ARMA models. The two MAPs fitted with the two different approaches either underestimated (`MAP EM`) or overestimated (`MAP MOEA`) the higher lag autocorrelations, but provide a good approximation for the lower lags. From Figure 3.6(d) one can see that the pure AR models ($AR(3)$ and $AR(9)$) only capture the first 3 and 9 lags, respectively. In general, an $AR(p)$ model can capture p lags exactly but for lags $> p$ the autocorrelation of the model converges to zero very fast. For fitting traces with a larger number of significant lags of autocorrelation this would result in large models with a huge amount of AR coefficients. In contrast, allowing for only few additional moving average terms in the model results in a vast improvement of the fitting quality for higher lags, while still keeping the model



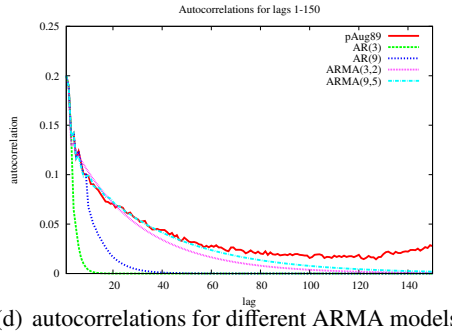
(a) cumulative distribution function



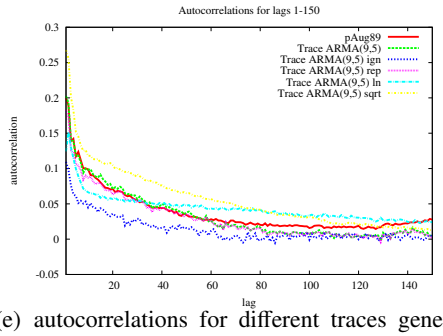
(b) probability density function



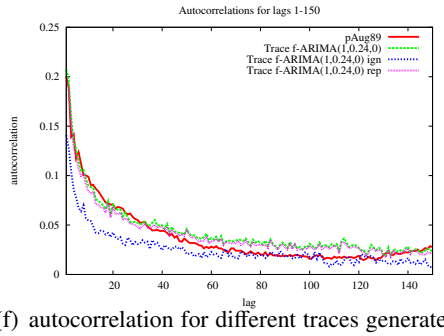
(c) autocorrelations



(d) autocorrelations for different ARMA models



(e) autocorrelations for different traces generated from ARMA models



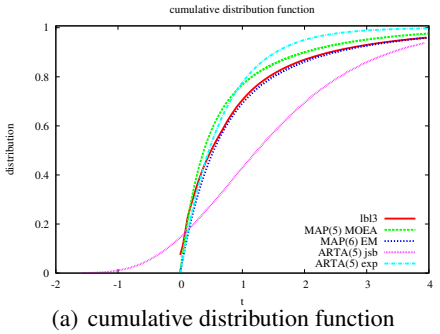
(f) autocorrelation for different traces generated from f-ARIMA models

Figure 3.6.: Fitting results for the trace *BC-pAug89*

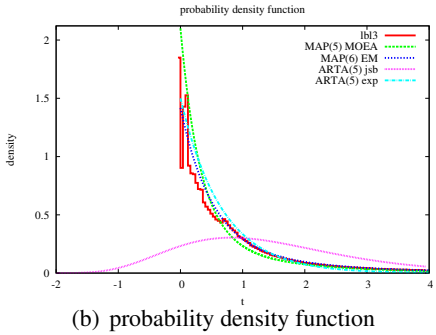
size small. This becomes visible by the curves for $ARMA(3,2)$ and the $ARMA(9,5)$ model in Figure 3.6(d). `ARTAFACTS` only supports autocorrelation for up to five lags and thus, the resulting ARTA model only captures the first five autocorrelations. But since ARTA models use an AR model as base process it is obvious from the previous observations on AR models that capturing autocorrelations up to a reasonable lag would result in a very large base process for the ARTA model.

Again, we have to deal with negative values that the ARMA models might yield when simulated. The impact of the different strategies (ignoring, replacing and the two transformations) to avoid negative values on the autocorrelation of the simulated $ARMA(9,5)$ model is shown in Figure 3.6(e). Plots for the other ARMA models have been omitted because they show similar results.

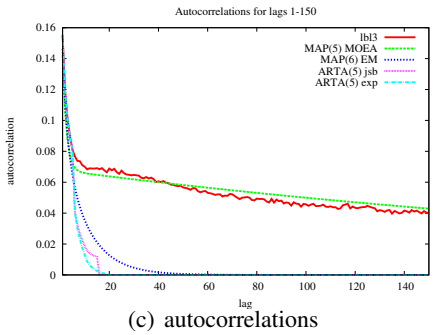
Figure 3.6(f) shows the autocorrelation of a fitted f-ARIMA model. f-ARIMA processes can model self-similar behavior and from the figure it becomes apparent that even a very small f-ARIMA process can provide a good approximation of the autocorrelation of the original trace. But as a drawback most characteristics like e.g. autocorrelation cannot be computed directly for f-ARIMA processes but have to be estimated from a simulation run of the model. Furthermore, f-ARIMA models do not overcome the problems with negative values caused by the marginal normal distribution that has been mentioned before for ARMA processes. Figure 3.6(f) also shows the traces that result from ignoring or replacing these values and one can see that the curves look similar to those of the $ARMA(9, 5)$ model in Figure 3.6(e).



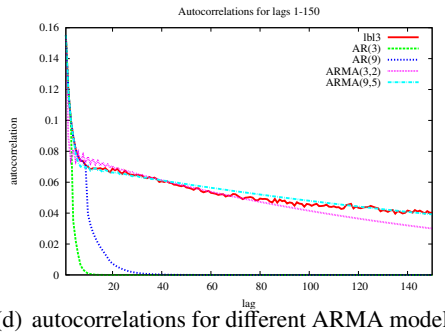
(a) cumulative distribution function



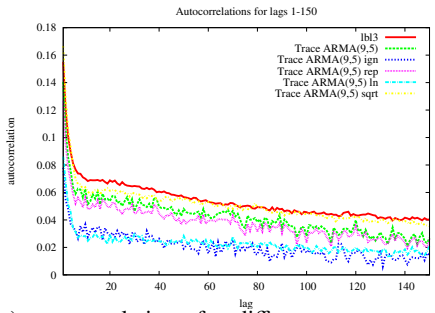
(b) probability density function



(c) autocorrelations



(d) autocorrelations for different ARMA models



(e) autocorrelations for different traces generated from ARMA models

Figure 3.7.: Fitting results for the trace *LBL-TCP-3*

Figure 3.7 shows the results for the second trace observed from a real system, *LBL-TCP-3*, which confirm the observations made for the trace *BC-pAug89*. Figures 3.7(a) and 3.7(b) show the distribution of the original trace and several fitted MAPs and ARTA models. As one can see `ARTAFIT`, which tries to find the complete ARTA model consisting of a Johnson distribution and the base process, was not able to return a Johnson distribution with $F(x) = 0, x < 0$. Using this distribution would lead to similar results as described before for ARMA models. Hence, a second ARTA process with a manually fitted exponential distribution was obtained using `ARTAFACTS` for capturing the autocorrelations only. Regarding the autocorrelations the ARTA models and one MAP only fitted the lower lags as shown in Figure 3.7(c). Only the MAP resulting from `MAP MOEA` was able to capture higher lag autocorrelations. Figure 3.7(d) shows again, that adding a few moving average terms to an AR process can help to improve the fitting of autocorrelations significantly. In Figure 3.7(e) the impact of the different strategies to deal with negative values on the autocorrelations is summarized.

3.3. Summary

From the observations in the previous paragraphs several drawbacks of the existing approaches for modeling traffic data that exhibits autocorrelations became apparent that directly motivate the processes that will be developed in Chapters 4, 5 and 6.

First, ARMA processes always result in a marginal normal distribution, which is not suitable in many cases. Furthermore, since the normal distribution is unbounded a transformation of the process is necessary to avoid negative values when the simulation requires positive values only. It was shown that the different strategies to deal with negative values have impact on the distribution and the autocorrelation, but usually none of the transformations resulted in an appropriate approximation for both measures. One should note, that when transforming a trace $\mathcal{T} = (t_1, t_2, \ldots, t_l)$ to $\mathcal{T}' = (\ln(t_1), \ln(t_2), \ldots, \ln(t_l))$ or to $\mathcal{T}' = \left(\sqrt{t_1}, \sqrt{t_2}, \ldots, \sqrt{t_l}\right)$ and fitting an ARMA model to the transformed trace, the fitted ARMA model does not completely capture the characteristics, especially the distribution, of this trace. Thus, when simulating that model and transforming the generated trace into $\bar{\mathcal{T}}' = (\exp(\bar{t_1}), \exp(\bar{t_2}), \ldots, \exp(\bar{t_l}))$ or $\bar{\mathcal{T}}' = \left((\bar{t_1})^2, (\bar{t_2})^2, \ldots, (\bar{t_l})^2\right)$, respectively, the approximation error of the ARMA model is transformed as well. For the transformation using the natural logarithm this resulted in a good approximation of the cdf in many cases while it usually had negative impact on the autocorrelation. On the other hand, the transformation using the square root preserved the fitting of the autocorrelation much better, but could not capture the distribution.

In contrast to the problems with fitting the distribution ARMA models are very flexible in capturing the autocorrelations. While $AR(p)$ processes with a reasonable model size are only adequate if few lag-k correlation coefficients are considered, adding a few MA terms resulted in small models that could fit a large number of autocorrelations.

ARTA processes overcome the distribution related issues of ARMA models and benefit from their abilities in autocorrelation fitting by combining an arbitrary marginal distribution with an $AR(p)$ base process. But it became also apparent, that the Johnson distribution, which is used by the fully automated `ARTAFIT`, is not sufficient in all cases, and one has to fall back to another distribution that has to be fitted by other tools.

Moreover, ARTA models can only fit a smaller number of autocorrelation lags with a reasonable model size, which is a problem inherited from its $AR(p)$ base process. This issue will be addressed in Chapter 4 where ARTA processes are extended to use an $ARMA(p,q)$ base process.

ARTA models require that the cdf of the marginal distribution is invertible, which excludes interesting distributions like Phase-type distributions (except exponential and Erlang), which could provide a better approximation of the empirical distribution of a trace than the distributions suitable for ARTA models. MAPs on the other hand are stochastic processes with Phase-type marginal distribution, but the distribution has a strong effect on the possible autocorrelation that the MAP could exhibit. From the examples one could see, that MAPs have problems to capture both distribution and dependence, if the distribution is untypical for Phase-type (e.g. has a normal-like shape). In Chapters 5 and 6 alternative stochastic processes are introduced, which combine a Phase-type distribution with an $ARMA(p,q)$ base process to allow for more flexible processes with Phase-type marginal distribution. While these processes pick up the basic idea of ARTA models to combine a distribution with a base process the combination of distribution and base process requires a different approach, because as already mentioned the cdf of PH distributions cannot be inverted efficiently in general.

Chapter 4

Extended ARTA Processes

As already mentioned several new stochastic processes for simulation models are developed in this and the following chapters that can capture correlated data with Phase-type marginal distribution. These processes aim at overcoming the drawbacks presented in Chapter 3 and will cover acyclic Phase-type distributions and all its subclasses as marginal distribution. The processes will be introduced stepwise. In this section ARTA processes from Section 2.2 will be extended such that they can capture autocorrelation for a higher number of lags while preserving a small model size. These extended ARTA models can be employed for all marginal distributions for that the inverse cdf can be computed and hence, cover the cases of exponential and Erlang distributions from the class of Phase-type distributions, but of course can be used with various other non-PH distributions like for example normal, Johnson, lognormal, Weibull or uniform. The stochastic processes presented in the following Chapters 5 and 6 will finally deal with more elaborate subclasses of PH distributions for which the inverse cdf cannot be computed efficiently.

Recall from the results of Chapter 3 that an $AR(p)$ model can provide an exact fitting for the first p lags of autocorrelation but falls short of capturing more than p lags. ARTA models inherit this property from its $AR(p)$ base process. Figure 4.1 shows three ARTA models with different base processes for $p = 5, 10$ and 15 that have been fitted to the trace *LBL-TCP-3*. From the curves it is apparent that an ARTA model that should fit a significant number of lag-k autocorrelations as it is necessary to capture the trace characteristics, would require a very large base process, i.e. a large number of AR coefficients. While in general the input model size is not of major importance in simulation, it is obvious that $AR(p)$ processes with e.g. 100 or more autoregressive coefficients are both cumbersome to fit and to simulate. On the other hand it was shown in Chapter 3 that $ARMA(p, q)$ processes can model a large number of autocorrelations with a much smaller model size. While they do not provide an exact matching as $AR(p)$ processes the approximation was still very close, which is sufficient, since the autocorrelations that should be matched are usually obtained from a trace and hence, they are in fact only estimates within some confidence interval.

These considerations motivate the idea to replace the $AR(p)$ base process of an ARTA model with an $ARMA(p, q)$ process to make these types of processes more suitable for the requirements that occur when modeling data from computer networks.

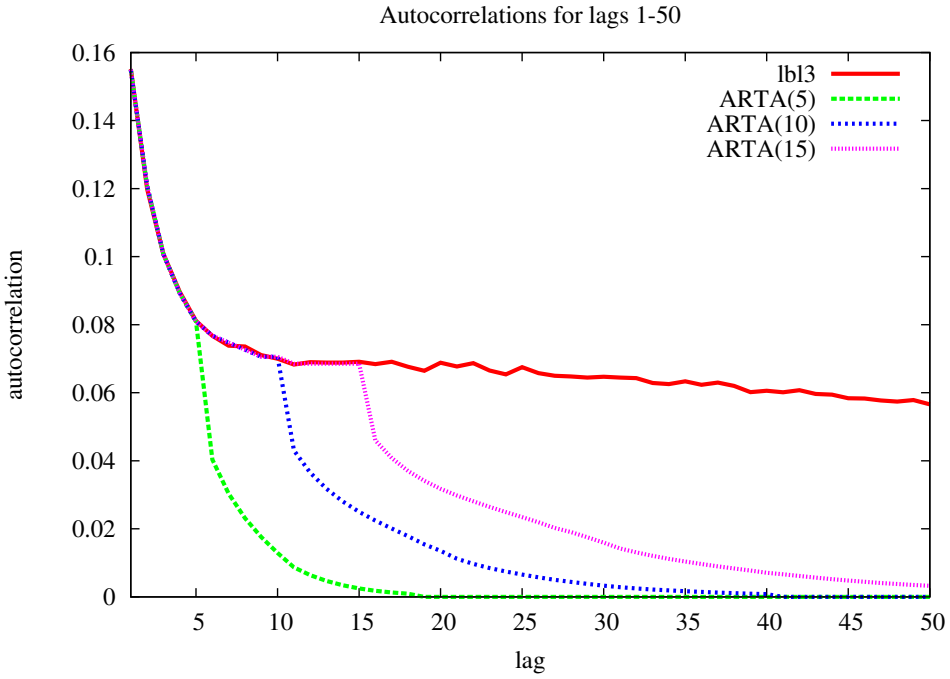


Figure 4.1.: Autocorrelations for different ARTA models fitted to trace *LBL-TCP-3*

In the following the concepts are introduced that are necessary to extend the ARTA approach from Section 2.2 with an Autoregressive Moving Average process and it is shown that most of the definitions and properties for ARTA processes stated in [47] and summarized in Section 2.2 still hold for the extended ARTA process.

An extended ARTA model consists of an arbitrary marginal distribution F_Y and an $ARMA(p, q)$ base process:

Definition 4.1 (Extended ARTA Process)**.** *An extended ARTA process is defined by a marginal distribution F_Y for which the inverse cdf F_Y^{-1} can be computed and a stationary Autoregressive Moving Average $ARMA(p, q)$ base process*

$$Z_t = \alpha_1 Z_{t-1} + \alpha_2 Z_{t-2} + \ldots + \alpha_p Z_{t-p} + \beta_1 \epsilon_{t-1} + \beta_2 \epsilon_{t-2} + \ldots + \beta_q \epsilon_{t-q} + \epsilon_t$$

as defined in Section 2.1. The base process is constructed such that the generated time series $\{Z_t; t = 1, 2, \ldots\}$ has standard normal distribution $N(0, 1)$. Then the extended ARTA process describes a time series

$$Y_t = F_Y^{-1}[\Phi(Z_t)], t = 1, 2, \ldots \tag{4.1}$$

where Φ is the standard normal cumulative distribution function.

Of course, for $q = 0$ the model becomes an ARTA model with an $AR(p)$ base process as described in Section 2.2. If the $\{Z_t; t = 1, 2, \ldots\}$ have standard normal distribution as required in Definition 4.1 the probability-integral transformation $\Phi(Z_t)$ generates

a sequence with uniform distribution on $(0, 1)$ (cf. [59]) and the inverse transform method using F_Y^{-1} generates a time series $\{Y_t, t = 1, 2, \ldots\}$ with the desired marginal distribution F_Y (cf. Section 2.2 and Figure 2.3).

In the following some properties of the extended ARTA process are established that relate statistical properties of the base process Z_t to properties of the ARTA process Y_t. For these observations we assume that the base process is constructed such that the $\{Z_t; t = 1, 2, \ldots\}$ have standard normal distribution as mentioned above. After that it is explained how the base process can be constructed to fulfill these requirements.

4.1. Properties of Extended ARTA Processes

To use the extended ARTA processes from Definition 4.1 for fitting traffic data a relation between the autocorrelation structure of the extended ARTA process and the autocorrelation structure of the base process has to be established. For given autocorrelations $\hat{\boldsymbol{\rho}} = (\hat{\rho}_1, \hat{\rho}_2, \ldots, \hat{\rho}_r)$ that have been estimated from a trace and a given marginal distribution F_Y one has to construct an $ARMA(p, q)$ base process with autocorrelations $\boldsymbol{\rho} = (\rho_1, \rho_2, \ldots, \rho_r)$ such that the extended ARTA process has autocorrelations $\hat{\boldsymbol{\rho}}$.

[47] introduced such a relation for ARTA processes with an $AR(p)$ base process (cf. Equation 2.6) and proved that an autocorrelation $\rho_h \{<, =, >\} 0$ for the base process implies an autocorrelation $\hat{\rho}_h \{<, =, >\} 0$ for the ARTA process. In the following it is shown that the same properties hold for extended ARTA models with an $ARMA(p, q)$ base process as well.

The autocorrelation of an ARTA process and the base process are related by (cf. Equation 2.6) $Corr[Y_t, Y_{t+h}] = Corr\left[F_Y^{-1}(\Phi(Z_t)), F_Y^{-1}(\Phi(Z_{t+h}))\right]$, regardless of whether the base process is $AR(p)$ or $ARMA(p, q)$. Since (cf. [47])

$$Corr[Y_t, Y_{t+h}] = \frac{E[Y_t Y_{t+h}] - (E[Y])^2}{Var[Y]} \tag{4.2}$$

and $E[Y]$ and $Var[Y]$ are determined by the marginal distribution F_Y the important term for establishing the relation between base process and (extended) ARTA process is the joint moment $E[Y_t Y_{t+h}]$. Recall the requirement for the $\{Z_t; t = 1, 2, \ldots\}$ resulting from the base process to have standard normal distribution. Then any two elements of the time series (Z_t, Z_{t+h}) have a standard bivariate normal distribution with density function φ_{ρ_h} and correlation $\rho_h = Corr[Z_t, Z_{t+h}]$. Now we have

$$\begin{aligned} E[Y_t Y_{t+h}] &= E[F_Y^{-1}(\Phi(Z_t)) F_Y^{-1}(\Phi(Z_{t+h}))] \\ &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} F_Y^{-1}(\Phi(z_t)) F_Y^{-1}(\Phi(z_{t+h})) \varphi_{\rho_h}(z_t, z_{t+h}) dz_t dz_{t+h}. \end{aligned} \tag{4.3}$$

Observe from Equations 4.2 and 4.3 that the ARTA correlation $\hat{\rho}_h = Corr[Y_t, Y_{t+h}]$ is a function of the base process autocorrelation ρ_h that appears in $\varphi_{\rho_h}(z_t, z_{t+h})$. This function is denoted by $\omega(\rho_h) = \hat{\rho}_h$. Note, that it is possible to compute a $\hat{\rho}_h$ from a given base process autocorrelation ρ_h using Equation 4.3, but for actually fitting ARTA processes it is necessary to determine the base process autocorrelation ρ_h for a given (i.e. estimated from a trace) $\hat{\rho}_h$. This is to be done numerically by a search algorithm and [47] established several properties of $\omega(\rho_h)$ that allow one to use such algorithms for ARTA processes with an $AR(p)$ base process. The crucial requirement

for proving the properties of $\omega(\rho_h)$ for ARTA processes is the fact, that the $\{Z_t; t = 1, 2, \ldots\}$ generated by the base process have standard normal distribution. Since this is required for the $ARMA(p, q)$ base process of extended ARTA models as well, the proofs from [47] still work for extended ARTA processes. In the following the main results from [47] that are necessary for applying a search algorithm are summarized.

Proposition 4.1. *For any extended ARTA process with marginal distribution F_Y and $ARMA(p, q)$ base process we have that*

- $\omega(0) = 0$,
- $\rho_h \leq 0 \Rightarrow \omega(\rho_h) \leq 0$,
- $\rho_h \geq 0 \Rightarrow \omega(\rho_h) \geq 0$.

Proof. This follows from the proof of [47, Proposition 1].

- $\rho_h = 0$ implies that Z_t and Z_{t+h} are independent. Then

$$\begin{aligned} E[Y_t Y_{t+h}] &= E\left[F_Y^{-1}(\Phi(Z_t))F_Y^{-1}(\Phi(Z_{t+h}))\right] \\ &= E\left[F_Y^{-1}(\Phi(Z_t))\right] E\left[F_Y^{-1}(\Phi(Z_{t+h}))\right] = E[Y_t]E[Y_{t+h}] = E[Y]^2 \end{aligned}$$

  and

$$Corr[Y_t, Y_{t+h}] = \frac{E[Y_t Y_{t+h}] - (E[Y])^2}{Var[Y]} = 0.$$

- According to [153] $\rho_h \leq (\geq)\ 0$ implies that $Cov[g_1(Z_t, Z_{t+h}), g_2(Z_t, Z_{t+h})] \leq (\geq)\ 0$ for all nondecreasing functions g_1 and g_2. Since $F_Y^{-1}(\Phi(\cdot))$ is a nondecreasing function, the result immediately follows by setting $g_1(Z_t, Z_{t+h}) = F_Y^{-1}(\Phi(Z_t))$ and $g_2(Z_t, Z_{t+h}) = F_Y^{-1}(\Phi(Z_{t+h}))$.

□

Theorem 4.1. *For any extended ARTA process with marginal distribution F_Y and $ARMA(p, q)$ base process the function $\omega(\rho_h)$ is nondecreasing for $-1 \leq \rho_h \leq 1$.*

Proof. For positive correlations $0 \leq \rho_h \leq 1$ this follows from [153, Theorem 5.3.10], which states that for two normal variables Z_1 and Z_2 with correlation ρ_h we have $Corr[g(Z_1), g(Z_2)]$ is nondecreasing in ρ_h for all functions $g(\cdot)$. Setting $Y_t = g(Z_t) = F_Y^{-1}(\Phi(Z_t))$ and $Y_{t+h} = g(Z_{t+h}) = F_Y^{-1}(\Phi(Z_{t+h}))$ we obtain that $\omega(\rho_h) = Corr[Y_t, Y_{t+h}]$ is nondecreasing for $0 \leq \rho_h \leq 1$. A similar result has been obtained for negative correlations $-1 \leq \rho_h \leq 0$ in [47, Theorem 1]. □

Theorem 4.2. *For any extended ARTA process with marginal distribution F_Y and $ARMA(p, q)$ base process the function $\omega(\rho_h)$ is continuous, if there exists $\epsilon > 0$ such that $E[|Y_t Y_{t+h}|^{1+\epsilon}] < \infty$ for all $-1 \leq \rho_h \leq 1$.*

Proof. In [47] Theorem 4.2 has been proven for ARTA processes with $AR(p)$ base process. We will summarize the basic ideas and show that the proof also holds for an

$ARMA(p, q)$ base process.
First observe, that requiring $E[|Y_t Y_{t+h}|^{1+\epsilon}] < \infty$ for all $-1 \leq \rho \leq 1$ is equivalent to

$$\int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \sup_{\rho \in [-1,1]} \left\{ |F_Y^{-1}[\Phi(z_1)] F_Y^{-1}[\Phi(z_2)]|^{1+\epsilon} \times \varphi_\rho(z_1, z_2) \right\} dz_1 dz_2 < \infty. \tag{4.4}$$

Now, let Z_1 and Z_3 be iid standard normal random variables and assume that $\rho \in [-1, 1]$ is fixed. Furthermore, let $\{\rho_n\}_{n=1}^{\infty}$ be a sequence with $\rho_n \in [-1, 1], n = 1, 2, \ldots$ and $\rho_n \to \rho$ as $n \to \infty$. Define

$$Z_{1n} \equiv Z_1, \qquad Z_{2n} \equiv \rho_n Z_1 + \left(\sqrt{1 - \rho_n^2} \right) Z_3, \qquad Z_2 \equiv \rho Z_1 + \left(\sqrt{1 - \rho^2} \right) Z_3.$$

Observe, that Z_1 and Z_2 are standard normal random variables with correlation ρ and Z_{1n} and Z_{2n} are standard normal random variables with correlation ρ_n [90]. Now, let

$$Y_{1n} \equiv F_Y^{-1}[\Phi(Z_{1n})] \qquad \text{and} \qquad Y_{2n} \equiv F_Y^{-1}[\Phi(Z_{2n})]$$

and define

$$h\begin{pmatrix} z_1 \\ z_2 \end{pmatrix} \equiv F_Y^{-1}[\Phi(z_1)] F_Y^{-1}[\Phi(z_2)].$$

For fixed z_2 the function h is monotone in z_1 and vice versa. Therefore h has only a countable number of discontinuities and from

$$\begin{pmatrix} Z_{1n} \\ Z_{2n} \end{pmatrix} \Rightarrow \begin{pmatrix} Z_1 \\ Z_2 \end{pmatrix} \text{ as } n \to \infty$$

we get by application of the mapping theorem (cf. [27, Theorem 29.2]) the following convergence in distribution

$$h\begin{pmatrix} Z_{1n} \\ Z_{2n} \end{pmatrix} \Rightarrow h\begin{pmatrix} Z_1 \\ Z_2 \end{pmatrix} \text{ as } n \to \infty.$$

For $Y_1 \equiv F_Y^{-1}[\Phi(Z_1)]$ and $Y_2 \equiv F_Y^{-1}[\Phi(Z_2)]$ this is equivalent to

$$Y_{1n} Y_{2n} \Rightarrow Y_1 Y_2 \qquad \text{as } n \to \infty. \tag{4.5}$$

It then follows from Equations 4.4 and 4.5 and [27, Theorem 25.12] that $E[Y_{1n} Y_{2n}] \to E[Y_1 Y_2]$ as $n \to \infty$, implying that $\omega(\rho)$ is a continuous function since $\omega(\rho_n) \to \omega(\rho)$ as $n \to \infty$. Setting $Z_1 = Z_t, Z_2 = Z_{t+h}, Y_1 = Y_t, Y_2 = Y_{t+h}$ and $\rho = \rho_h$ proves Theorem 4.2. Observe, that the proof holds for any Z_i that have standard normal distribution. Therefore, the presented proof from [47] is valid for an ARTA process with ARMA base process as well, as long as the observations generated by the ARMA process have standard normal distribution, which we required in Definition 4.1. □

Since the autocorrelation structure of the extended ARTA process is a nondecreasing and continuous function according to Theorems 4.1 and 4.2 any search procedure can be applied to find the base process autocorrelation that results in the desired extended ARTA autocorrelation. From Proposition 4.1 the initial bounds can be obtained, i.e. for a positive ARTA autocorrelation only base process autocorrelations $0 \leq \rho_h \leq 1$

have to be considered and for a negative ARTA autocorrelation only $-1 \leq \rho_h \leq 0$ has to be treated.

Aside from the properties above, which are all related to the autocorrelation of an extended ARTA model, several other characteristics are of interest to describe a stochastic process or to assess its fitting quality. The probability density function, the cumulative distribution function and moments, variance, etc. are all completely determined by the marginal distribution of the extended ARTA model and thus, cannot be given in general. Appendix B contains a brief description of several distributions that can be used as marginal distribution for an extended ARTA model. Another measure of a stochastic process that is related to the dependence of consecutive arrivals are joint moments, which have for example been successfully used for MAP fitting (cf. Section 2.3.6). For extended ARTA models Equation 4.3 can be generalized to compute arbitrary joint moments, i.e.

$$\begin{aligned} E[Y_t^k Y_{t+h}^l] &= E\left[\left(F_Y^{-1}(\Phi(Z_t))\right)^k \left(F_Y^{-1}(\Phi(Z_{t+h}))\right)^l\right] \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \left(F_Y^{-1}(\Phi(z_t))\right)^k \left(F_Y^{-1}(\Phi(z_{t+h}))\right)^l \varphi_{\rho_h}(z_t, z_{t+h}) dz_t dz_{t+h}. \end{aligned} \tag{4.6}$$

Recall from Definition 4.1 that we required the $ARMA(p,q)$ base process to be stationary, i.e. that Z_t are invariant under time shifts. This of course implies, that the (extended) ARTA process is stationary as well, because the Y_t are only a transformation of the Z_t.

The previous observations on the relation of the autocorrelation structure of the base process and the autocorrelation of the extended ARTA process can be employed to construct the base process, which will be described in the following section.

4.2. Constructing the ARMA Base Process

For the $ARMA(p,q)$ base process three requirements have been pointed out previously. First, the generated time series $\{Z_t; t = 1, 2, \ldots\}$ must have standard normal distribution ($Z_t \sim N(0,1)$), second, the base process should have an autocorrelation structure $\boldsymbol{\rho} = (\rho_1, \rho_2, \ldots, \rho_k)$ such that the extended ARTA process has autocorrelations $\hat{\boldsymbol{\rho}} = (\hat{\rho}_1, \hat{\rho}_2, \ldots, \hat{\rho}_k)$ and third, the base process must be stationary.

The $\{Z_t; t = 1, 2, \ldots\}$ having standard normal distribution implies that the variance (or autocovariance at lag 0) of the $ARMA(p,q)$ process has to be 1. The autocovariance function of a stationary $ARMA(p,q)$ satisfies [37]

$$\gamma(k) = \sigma_\epsilon^2 \sum_{j=0}^{\infty} \psi_j \psi_{j+|k|},$$

where

$$\psi(z) = \sum_{j=0}^{\infty} \psi_j z^j = \beta(z)/\alpha(z) \qquad \text{for } |z| \leq 1$$

and

$$\beta(z) = 1 + \beta_1 z + \ldots + \beta_q z^q, \qquad \alpha(z) = 1 - \alpha_1 z - \ldots - \alpha_p z^p.$$

Since the autocorrelation of a stationary $ARMA(p,q)$ process is given by

$$\rho_k = \gamma(k)/\gamma(0)$$

the fraction can be reduced by eliminating σ_ϵ^2. Thus, the ρ_k are independent of σ_ϵ^2 and one can set σ_ϵ^2 such that the $\{Z_t; t = 1, 2, \ldots\}$ have a standard normal distribution, i.e. are $N(0,1)$, without modifying the autocorrelation structure of the base process.

[37] provides another formula for the autocovariance that is more appropriate for actually computing the autocovariances:

$$\gamma(k) - \alpha_1\gamma(k-1) - \ldots - \alpha_p\gamma(k-p) = \sigma_\epsilon^2 \sum_{k \le j \le q} \beta_j \psi_{j-k}, \qquad 0 \le k < \max(p, q+1) \quad (4.7)$$

and

$$\gamma(k) - \alpha_1\gamma(k-1) - \ldots - \alpha_p\gamma(k-p) = 0, \qquad k \ge \max(p, q+1). \quad (4.8)$$

Defining $\beta_0 = 1, \beta_j = 0, j > q$ and $\alpha_j = 0, j > p$ the ψ_i can be computed from

$$\psi_j - \sum_{0<k\le j} \alpha_k \psi_{j-k} = \beta_j, \qquad 0 \le j < \max(p, q+1)$$

and

$$\psi_j - \sum_{0<k\le p} \alpha_k \psi_{j-k} = 0, \qquad j \ge \max(p, q+1).$$

Thus, for a given $ARMA(p,q)$ process with $\tilde{\sigma}_\epsilon^2$ being the variance of the white noise one can solve Equations 4.7 and 4.8 to obtain the autocovariance at lag 0, $\tilde{\gamma}(0)$, and then set the new variance to

$$\sigma_\epsilon^2 = \tilde{\sigma}_\epsilon^2/\tilde{\gamma}(0) \quad (4.9)$$

resulting in a new $N(0,1)$ process with the same autocorrelations as the old process.

Finding suitable base process autocorrelations $\boldsymbol{\rho} = (\rho_1, \rho_2, \ldots, \rho_r)$ such that the extended ARTA process has autocorrelations $\hat{\boldsymbol{\rho}} = (\hat{\rho}_1, \hat{\rho}_2, \ldots, \hat{\rho}_r)$ requires a numerical search procedure. Observe from Equation 4.3 that it is only possible to compute $\hat{\rho}_h$ from a given ρ_h, but not the other way round. Since in our case $\hat{\rho}_h$ is given (i.e. estimated from the trace) and we have to determine the corresponding ρ_h, this has to be done numerically by a search algorithm. More precisely, we are searching for ρ_h that minimizes

$$(\hat{\rho}_h - \omega(\rho_h))^2 \quad (4.10)$$

where $\hat{\rho}_h$ is the desired ARTA correlation and $\omega(\rho_h)$ is the ARTA correlation that results from a base process correlation ρ_h. The minimum of Equation 4.10 can be determined by a golden section search [131]. This search algorithm always maintains three points x_1, x_2, x_3. x_1 and x_3 are the lower and upper bound of the interval under consideration and x_2 is a point within this interval. The initialization values for x_1 and x_3 are set according to Proposition 4.1, i.e. x_1 and x_3 define the interval $[-1, 0]$ if $\hat{\rho}_h < 0$ or $[0, 1]$ if $\hat{\rho}_h > 0$. For $\hat{\rho}_h = 0$ no search is necessary, since this implies $\rho_h = 0$. The optimal value for x_2 divides the interval $[x_1, x_3]$ such that x_2 has a fractional distance of 0.38197 to x_1 and 0.61803 to x_3 (cf. [131]). The search procedure then iteratively selects a new point x_4 that divides the interval $[x_2, x_3]$ into the fractions mentioned above, evaluates Equation 4.10 at x_2 and x_4 and depending on the outcome continues

to search in the interval $[x_1, x_4]$ with mid-point x_2 or in the interval $[x_2, x_3]$ with mid-point x_4 until we have found an interval of a width less than a given ϵ. The mid-point of that interval is the base process autocorrelation ρ_h that results in a correlation for the extended ARTA model that is close enough to $\hat{\rho}_h$. For autocorrelation coefficients estimated from real traces it is very likely that coefficients that are only a few lags apart have a similar value, e.g. the difference between $\hat{\rho}_h$ and $\hat{\rho}_{h+1}$ might be small in many cases. For these lags the intervals examined by the search algorithm are identical for the first iterations of the procedure. To improve the performance of the search algorithm already computed pairs $(\rho_h, \omega(\rho_h))$ should be saved in a table such that they only have to be computed once and can be looked up if they are required again.
This way the autocorrelation structure can be determined numerically up to an arbitrary accuracy. A similar search procedure for ARTA models is given in [48].

Once the base process autocorrelation structure $\boldsymbol{\rho} = (\rho_1, \rho_2, \ldots, \rho_r)$ has been determined, an $ARMA(p, q)$ process has to be constructed that exhibits this structure. This can be done by using a general purpose optimization algorithm that minimizes the difference between the autocorrelations of the ARMA model that is constructed during the minimization process and $\boldsymbol{\rho} = (\rho_1, \rho_2, \ldots, \rho_r)$.
Recall, that we required the $ARMA(p, q)$ to be stationary and thus, we additionally have to penalize non-stationary solutions.
The stationarity of an $ARMA(p, q)$ model only depends on the AR coefficients α_i of the process, i.e. the process is stationary if the zeroes of the polynomial $\alpha(z) = 1 - \alpha_1 z - \alpha_2 z^2 - \ldots - \alpha_p z^p$ lie outside the unit circle [35]. More formal, the $ARMA(p, q)$ process is stationary if and only if $\alpha(z) \neq 0$ for all $z \in \mathbb{C}$ with $|z| \leq 1$.
Hence, we use the following goal function to fit an $ARMA(p, q)$ process to a set of given autocorrelations:

$$\underset{\alpha_1,\ldots,\alpha_p,\beta_1,\ldots,\beta_q}{\arg\min} \sum_{i=1}^{r} \left(\frac{\rho_i^*}{\rho_i} - 1 \right)^2 + \varrho \sum_{\xi,\alpha(\xi)=0} \min\left(0, (\xi - (1+\varepsilon))\right)^2 . \tag{4.11}$$

The first term is the objective function to minimize the difference of the autocorrelation coefficients, the second term is the penalty function. The ρ_i in Equation 4.11 are the autocorrelation coefficients to be achieved and the ρ_i^* are computed from the ARMA process as described in Equations 4.7 and 4.8. Thus, if the $\alpha_1, \ldots, \alpha_p, \beta_1, \ldots, \beta_q$ describe a stationary ARMA process the ρ_i^* are the autocorrelation coefficients of the $ARMA(p, q)$ model that is constructed during the minimization process. If the ARMA process is not stationary the values can be computed anyway, but do not have the interpretation as autocorrelation coefficients. In this case the penalty function is used to obtain a larger value for the goal function to force the optimization algorithm to leave the non-stationary region. For the penalty function all roots ξ of $\alpha(z)$ are considered. Let $\varepsilon > 0$ be some small constant, i.e. for an implementation of Equation 4.11 ε should be the smallest value such that $1 + \varepsilon \neq 1$. Then the term $\xi - (1 + \varepsilon)$ is non-negative, if the roots are outside the unit circle, i.e. the model is stationary, and $\min\left(0, (\xi - (1 + \varepsilon))\right)^2$ is zero. For non-stationary models at least one root lies inside the unit circle and $\min\left(0, (\xi - (1 + \varepsilon))\right)^2 > 0$. The penalty function is multiplied with some factor ϱ, which is increased in each iteration of the minimization to ensure that the algorithm leaves the area with non-stationary solutions. For a stationary solution the goal function only depends on the difference between the autocorrelation coeffi-

cients and the optimal solution that exactly matches the desired autocorrelation will result in a goal function value of zero.

Hence, finding an appropriate $ARMA(p, q)$ base process for a given marginal distribution F_Y and given autocorrelations $\hat{\boldsymbol{\rho}} = (\hat{\rho}_1, \hat{\rho}_2, \ldots, \hat{\rho}_r)$ for the extended ARTA process consists of three steps:

1. For each $\hat{\rho}_h$ find a ρ_h such that $Corr[Y_t, Y_{t+h}] = \hat{\rho}_h$ using Equation 4.2.
2. Fit an $ARMA(p, q)$ model to the autocorrelations $\boldsymbol{\rho} = (\rho_1, \rho_2, \ldots, \rho_r)$ determined in the first step.
3. Adjust the variance of the innovations of the $ARMA(p, q)$ base process resulting from the second step according to Equation 4.9.

Extended ARTA models can have any marginal distribution for which the inverse cdf can be computed. This includes uniform, triangular, normal and lognormal distributions. Moreover, of course distributions from the Johnson system, which have been used for the original ARTA process from Section 2.2, can be applied. From the class of Phase-type distributions exponential and Erlang distributions could serve as marginal distribution. Furthermore, gamma and χ^2 distributions, which are related to the Erlang distribution but are not Phase-type, can be used. See e.g. [104] for a definition and closed-form expressions for most of the mentioned distributions. For some distributions no closed-form expression for the inverse cdf exists. In these cases fast numerical algorithms are available. The inverse cdf of normal, lognormal, and Johnson distribution can be computed using [163], for Erlang, gamma and χ^2 distributions the algorithm from [20] can be used. A brief overview of the mentioned distributions and their properties can be found in Appendix B.

For the above considerations the marginal distribution F_Y is assumed to be given. Furthermore, the determination of the order p, q of the base process has been left open. These issues will be addressed in Chapter 7 where the ideas for extended ARTA models and further process types developed in the following chapters are integrated into an algorithmic framework.

Chapter 5

Correlated Hyper-Erlang Processes

In the following the basic ideas of ARTA processes are extended for the specification of processes that have a Hyper-Erlang marginal distribution (cf. Section 2.3.1). Since these considerations are more straightforward and easier to compute for HErDs than for the more general class of acyclic Phase-type distributions, the concept is first developed for HErDs in this chapter. The general case for APHs is treated in Chapter 6.

ARTA processes rely on the inversion of the cumulative distribution function and thus, they are suitable for distributions for which a closed-form expression for the inverse cdf exists. For these types of distributions random numbers are usually generated using the inverse transform method and thus, the additional effort for the ARTA approach in comparison to the generation of iid random numbers results from the simulation of the base process. Once an uniformly distributed random number has been obtained (either from the transformed base process for ARTA processes or from a random number generator for iid random numbers) the final step for generating the random number with the desired distribution is the same in both cases.

If F_Y^{-1} can be approximated numerically, the ARTA approach is also applicable, although both fitting of the process and sampling from the process can be cumbersome in this case. For example in [68] the ARTA approach is combined with Matrix Exponential marginal distributions, which results in a very computation-intensive procedure to determine the base process autocorrelation and to sample from the distribution, since the inverse cdf has to be computed numerically various times. In general the ARTA approach is not feasible for the interesting classes of ME and Phase-type distributions, since the cdf contains a matrix exponential (cf. Equation 2.9) and the inverse cdf cannot be computed efficiently. Exceptions are the exponential and the Erlang distributions, for which a closed-form expression or a numerical procedure [20] exist, respectively. For these reasons random numbers from a PH distribution are usually not generated using the inverse transform method, but by simulating the transitions of the underlying CTMC and by adding the transition times until absorption [136]. Hence, the ARTA approach does not conform to the usual way of random number generation for PH distributions.

Fortunately, APH distributions can be characterized by a mixture of finite sequences of exponential distributions, which allows for a different combination of the ARMA base process and the distribution that does not rely on the inversion of the cumulative

distribution function and respects the above considerations on random number generation for PH distributions resulting in a more efficient approach to describe correlated random numbers with PH distribution.

To clarify the benefits of using PH distributions several Hyper-Erlang distributions with increasing number of states and several other distributions for which the ARTA approach is applicable are fitted to three different traces. The two traces *BC-pAug89* [108] and *LBL-TCP-3* [130] have already been used in Chapter 3. The third trace *TUDo* [100] contains the interarrival times of one million packets that have been measured from the Squid proxy server at the Computer Science Department of TU Dortmund in 2006. For the experiments all traces have been scaled to have a mean value of 1. The distributions considered for fitting are exponential, lognormal, Johnson, Weibull and Hyper-Erlang. As fitting approach usually a likelihood-based technique was applied. For exponential and lognormal distributions maximum likelihood estimators are available in standard literature [104]. For the Weibull distribution a general purpose optimization algorithm for the maximization of the likelihood function was used and Hyper-Erlang distributions have been fitted using the EM approach from [151]. Since for distributions from the Johnson system maximum likelihood fitting is difficult and might result in unfeasible values [156] the quantile estimation from [162] was used for fitting Johnson distributions. Table 5.1 shows the log-likelihood values and the first three moments of the fitted distributions for the three traces. In addition to the moments the relative errors $(|\mu_i - \hat{\mu}_i|/\hat{\mu}_i) \cdot 100$ in percent with μ_i and $\hat{\mu}_i$ being the i-th moment of the distribution and the trace, respectively, are listed. As one can see from Table 5.1 fitting with Hyper-Erlang distributions resulted in the best log-likelihood values for all traces. For the trace *BC-pAug89* even a HErD with 2 states yielded a better likelihood value than the other distributions, which is further increased when using HErDs of higher order. Regarding the moments the lognormal distributions and the HErDs yielded results close to the moments of the trace.

For the trace *LBL-TCP-3* the results are similar. Again, for Hyper-Erlang distributions the largest log-likelihood values have been obtained. Only the likelihood value of the lognormal distribution is at least close to the one of a HErD(2), but except for the HErDs all other distributions provided poor results for the moments. Note, that the trace contains values for which the fitted Johnson distribution is not defined and thus, the log-likelihood is $-\infty$ for this distribution.

The trace *TUDo* was most difficult to fit. While the largest log-likelihood value can still be achieved with HErDs it requires a larger number of states in this case. If the moments are considered as well, only Hyper-Erlang distributions provided a good approximation of the first moment, but for higher moments none of the distributions was able to provide a good approximation.

In summary the results clearly indicate that PH distributions usually yield better approximations in terms of both likelihood and moments than the other considered distributions, especially the likelihood increases significantly when increasing the number of states.

Because the cumulative distribution function of the Hyper-Erlang distribution (cf. Equation 2.13) cannot be inverted efficiently, a different approach for the combination of the $ARMA(p,q)$ base process and the distribution has to be applied. The basic idea is to use the correlated random numbers obtained from the base process for the selection of a branch from the Hyper-Erlang distribution. Of course, this requires the HErD

	Distribution	Log-Likelihood	Moment 1	Moment 2	Moment 3
pAug89	Exponential	−999999	1.0 (0.0%)	2.0 (52.7%)	6.0 (90.7%)
	Johnson SU	−959863	0.89 (11.0%)	1.8 (57.4%)	8.1 (87.5%)
	Weibull	−990007	0.95 (5.0%)	2.1 (50.3%)	7.3 (88.7%)
	Lognormal	−953799	1.05 (5.0%)	4.1 (2.9%)	56.9 (12.1%)
	HErD(2)	−911558	1.0 (0.0%)	4.5 (5.9%)	66.3 (2.4%)
	HErD(3)	−911135	1.0 (0.0%)	3.5 (16.9%)	34.4 (46.9%)
	HErD(4)	−874270	1.0 (0.0%)	4.1 (2.9%)	51.2 (20.9%)
	HErD(5)	−847551	1.0 (0.0%)	4.1 (2.9%)	50.8 (21.5%)
	HErD(10)	−838863	1.0 (0.0%)	3.9 (7.8%)	44.6 (31.1%)
	HErD(15)	−816679	1.0 (0.0%)	3.8 (9.0%)	42.6 (34.3%)
LBL3	Exponential	$-1.790e+06$	1.0 (0.0%)	2.0 (32.0%)	6.0 (64.4%)
	Johnson SB	$-\infty$	0.96 (4.0%)	2.3 (21.8%)	8.6 (48.9%)
	Weibull	$-1.724e+06$	0.95 (5.0%)	2.4 (18.4%)	9.8 (41.8%)
	Lognormal	$-1.698e+06$	1.14 (14.0%)	8.1 (175.3%)	345.6 (1952%)
	HErD(2)	$-1.698e+06$	1.0 (0.0%)	2.8 (4.8%)	14.4 (14.5%)
	HErD(3)	$-1.697e+06$	1.0 (0.0%)	2.9 (1.4%)	16.1 (4.4%)
	HErD(4)	$-1.695e+06$	1.0 (0.0%)	2.9 (1.4%)	15.5 (8.0%)
	HErD(5)	$-1.672e+06$	1.0 (0.0%)	2.9 (1.4%)	14.9 (11.5%)
	HErD(10)	$-1.66527e+06$	1.0 (0.0%)	2.9 (1.4%)	15.9 (5.04%)
	HErD(15)	$-1.64763e+06$	1.0 (0.0%)	2.9 (1.4%)	15.1 (10.4%)
TUDo	Exponential	$-\infty$	1.0 (0.0%)	2.0 (96.9%)	6.0 (100%)
	Johnson SL	398952	0.64 (36.0%)	49.0 (24.2%)	426400.6 (3206%)
	Weibull	455191	0.48 (52.0%)	2.0 (96.9%)	24.6 (99.8%)
	Lognormal	603905	0.55 (45.0%)	21.9 (66.1%)	64009.4 (396%)
	HErD(2)	384320	1.0 (0.0%)	12.0 (81.4%)	236.8 (98.1%)
	HErD(3)	571616	1.0 (0.0%)	24.0 (62.9%)	1022.5 (92.1%)
	HErD(4)	595607	1.0 (0.0%)	31.1 (51.9%)	1818.7 (85.9%)
	HErD(5)	609339	1.0 (0.0%)	48.0 (25.8%)	4938.6 (61.7%)
	HErD(10)	626463	1.0 (0.0%)	40.0 (38.1%)	3034.1 (76.5%)
	HErD(15)	643982	1.0 (0.0%)	42.0 (35.0%)	3595.0 (72.1%)

Table 5.1.: Likelihood and moments for the fitted distributions

to have at least two different branches, which will be assumed for the remainder of this chapter. Actually this is not a real restriction on the class of Hyper-Erlang distributions, since a HErD with one branch only is an Erlang distribution, which is covered by the ARTA processes presented in Chapter 4. Additionally it is assumed, that each branch describes a different Erlang distribution. If two branches have the same Erlang distribution, one of the branches is redundant and the two branches could be merged together by adjusting the initial probability of the first branch and by deleting the second branch. Finally, we will assume that the Erlang branches are ordered according to the mean values of the Erlang distributions corresponding to the branches, i.e. let X_i and X_j be two random variables that have a Erlang distribution corresponding to the i-th and j-th branch of the Hyper-Erlang distribution. Then $i \leq j \Rightarrow E[X_i] \leq E[X_j]$. This is not a restriction, because reordering the branches has no effect on the overall distribution.

As preliminary consideration some type of process description has to be found to describe a series of arrivals generated by a Hyper-Erlang distribution. Then, this description can be used to incorporate an $ARMA(p, q)$ base process to introduce autocorrelations into the process. Consider the HErD from Figure 5.1, which has three

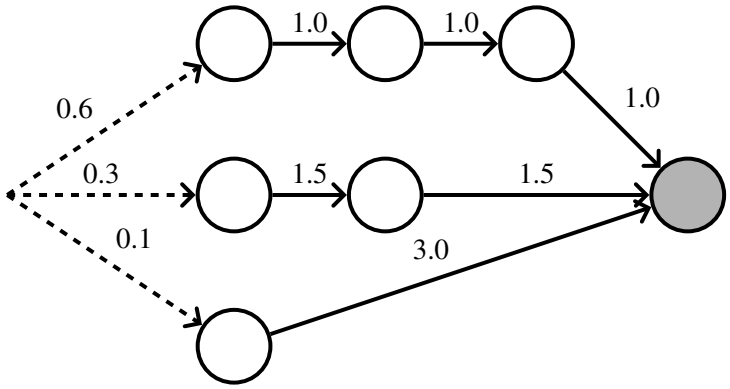


Figure 5.1.: Example for a Hyper-Erlang distribution with 3 branches

different Erlang branches with 3, 2 and 1 phase, respectively. When simulating such a HErD this is usually done by repeatedly drawing a random number U with uniform distribution on $[0, 1]$ to determine the Erlang branch i and by drawing S_i random numbers with exponential distribution, each with rate λ_i (or one random number from an Erlang distribution with S_i phases and rate λ_i), to determine the time until absorption. Define

$$\begin{aligned} \underline{b}_1 &= 0 \\ \bar{b}_i &= \underline{b}_i + \tau_i \qquad i = 1, \ldots, m \\ \underline{b}_i &= \bar{b}_{i-1} \qquad i = 2, \ldots, m \end{aligned}$$

as lower and upper limits for the probability to choose branch i, i.e. if $U \in [\underline{b}_i, \bar{b}_i)$ branch i is chosen. Let

$$\delta(U, i) = \begin{cases} 1, & U \in [\underline{b}_i, \bar{b}_i) \\ 0, & \text{otherwise.} \end{cases} \tag{5.1}$$

Moreover, let $\{U_t\}$ be a sequence of random numbers with uniform distribution on $(0, 1)$ and let $\{X_t^{(S_i,\lambda_i)}\}, i = 1, \ldots, m$ be m sequences of random numbers with Erlang distribution with S_i phases and rate λ_i. If the $\{U_t\}$ are independent

$$Y_t = \sum_{i=1}^{m} \delta(U_t, i) X_t^{(S_i,\lambda_i)} \tag{5.2}$$

describes a sequence of iid random variables with Hyper-Erlang distribution. Note, that for given U_t the function $\delta(U_t, i)$ is equal to 1 for one i only and 0 elsewhere. Hence, one would not use Equation 5.2 for actually simulating a Hyper-Erlang distribution, since it would require drawing $m - 1$ unused Erlang random numbers, but instead only draw one Erlang random number according to that branch i for which $\delta(U_t, i) = 1$. Anyway, for the following considerations this description will serve best.

Equation 5.2 can be easily modified to incorporate correlation between consecutive elements of the process. Let $\{Z_t; t = 1, 2, \ldots\}$ be a time series with standard normal distribution generated by an $ARMA(p, q)$ process as described in Section 2.1. Using the

same probability-integral transformation as for (extended) ARTA models (cf. Chapter 4) we get $U_t = \Phi(Z_t)$. Now, since the Z_t (and hence the U_t) are correlated, using $U_t = \Phi(Z_t)$ in Equation 5.2 results in a series Y_t of correlated random variables with Hyper-Erlang distribution and motivates the following definition:

Definition 5.1 (Correlated Hyper-Erlang Process)**.** *A Correlated Hyper-Erlang Process of order n, p, q, denoted $CHEP(n, p, q)$, is defined by a Hyper-Erlang distribution*

$$F(y) = 1 - \sum_{i=1}^{m} \tau_i \sum_{j=0}^{S_i - 1} \frac{(\lambda_i y)^j}{j!} e^{-\lambda_i y}$$

with n phases divided into m Erlang branches with the number of phases $S_i, i = 1, \ldots, m$ and rates $\lambda_i, i = 1, \ldots, m$ and a stationary Autoregressive Moving Average $ARMA(p, q)$ base process

$$Z_t = \alpha_1 Z_{t-1} + \alpha_2 Z_{t-2} + \ldots + \alpha_p Z_{t-p} + \beta_1 \epsilon_{t-1} + \beta_2 \epsilon_{t-2} + \ldots + \beta_q \epsilon_{t-q} + \epsilon_t.$$

The base process is constructed such that the generated time series $\{Z_t; t = 1, 2, \ldots\}$ has a standard normal distribution $N(0, 1)$. The branches of the HErD are ordered such that $i \leq j \Rightarrow (S_i/\lambda_i) \leq (S_j/\lambda_j)$ holds. Then the Correlated Hyper-Erlang Process describes a time series

$$Y_t = \sum_{i=1}^{m} \delta(\Phi(Z_t), i) X_t^{(S_i, \lambda_i)}, t = 1, 2, \ldots \tag{5.3}$$

where Φ is the standard normal cumulative distribution function, $\delta(U, i)$ is a function as defined in Equation 5.1 and $\{X_t^{(S_i, \lambda_i)}\}, i = 1, \ldots, m$ are sequences of independent and identically distributed random numbers with Erlang distribution with S_i phases and rate λ_i.

To make use of this approach several properties for the relation between the base process and the CHEP correlation have to be established like it was done in Chapter 4 for extended ARTA models.

5.1. Properties of Correlated Hyper-Erlang Processes

To use the process from Definition 5.1 a relation between the correlation of the CHEP, i.e. $Corr[Y_t, Y_{t+h}]$, and the base process correlation $Corr[Z_t, Z_{t+h}]$ has to be established, i.e. a process $\{Z_t\}$ with autocorrelations $Corr[Z_t, Z_{t+h}]$ has to be found such that $\{Y_t\}$ has the desired autocorrelations $Corr[Y_t, Y_{t+h}]$.
Recall, that the autocorrelations of $\{Y_t\}$ can be expressed as

$$Corr[Y_t, Y_{t+h}] = \frac{E[Y_t Y_{t+h}] - E[Y]^2}{Var[Y]}. \tag{5.4}$$

Again, $E[Y]$ and $Var[Y]$ are known, i.e. they can be computed using Equation 2.14. Hence, we can restrict ourselves to $E[Y_t Y_{t+h}]$.

$$\begin{aligned}
E[Y_t Y_{t+h}] &= E\left[\left(\sum_{i=1}^{m} \delta(U_t, i) X_t^{(S_i,\lambda_i)}\right)\left(\sum_{j=1}^{m} \delta(U_{t+h}, j) X_{t+h}^{(S_j,\lambda_j)}\right)\right] \qquad (5.5)\\
&= E\left[\sum_{i,j} \delta(U_t, i) X_t^{(S_i,\lambda_i)} \delta(U_{t+h}, j) X_{t+h}^{(S_j,\lambda_j)}\right], i, j = 1, \ldots, m\\
&= \sum_{i,j} E\left[\delta(U_t, i) X_t^{(S_i,\lambda_i)} \delta(U_{t+h}, j) X_{t+h}^{(S_j,\lambda_j)}\right]\\
&= \sum_{i,j} \left(E\left[\delta(U_t, i)\delta(U_{t+h}, j)\right] E[X_t^{(S_i,\lambda_i)}] E[X_{t+h}^{(S_j,\lambda_j)}]\right)\\
&= \sum_{i,j} \left(\frac{S_i}{\lambda_i}\frac{S_j}{\lambda_j} E\left[\delta(U_t, i)\delta(U_{t+h}, j)\right]\right)\\
&= \sum_{i,j} \left(\frac{S_i}{\lambda_i}\frac{S_j}{\lambda_j} E\left[\delta(\Phi(Z_t), i)\delta(\Phi(Z_{t+h}), j)\right]\right)\\
&= \sum_{i,j} \left(\frac{S_i}{\lambda_i}\frac{S_j}{\lambda_j} \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \delta(\Phi(z_t), i)\delta(\Phi(z_{t+h}), j)\varphi_{\rho_h}(z_t, z_{t+h}) dz_t dz_{t+h}\right)
\end{aligned}$$

where $\varphi_{\rho_h}(z_t, z_{t+h})$ is the standard bivariate normal probability density function with correlation $\rho_h = Corr[Z_t, Z_{t+h}]$.
Using the knowledge of the function $\delta(\cdot)$ Equation 5.5 can be simplified. Note from Equation 5.1 that $\delta(U, i)$ is 1 for $U \in [\underline{b}_i, \bar{b}_i)$ and 0 otherwise. Hence, this information can be used to determine the integration bounds in Equation 5.5 and omit $\delta(\cdot)$ in the double integral:

$$E[Y_t Y_{t+h}] = \sum_{i,j} \left(\frac{S_i}{\lambda_i}\frac{S_j}{\lambda_j} \int_{\Phi^{-1}(\underline{b}_j)}^{\Phi^{-1}(\bar{b}_j)} \int_{\Phi^{-1}(\underline{b}_i)}^{\Phi^{-1}(\bar{b}_i)} \varphi_{\rho_h}(z_t, z_{t+h}) dz_t dz_{t+h}\right). \qquad (5.6)$$

Thus, for the computation of $E[Y_t Y_{t+h}]$ for each combination of two branches of the HErD the product of the mean of the first branch, the mean of the second branch and the double integral of the bivariate standard normal density where the integration bounds are determined by the probabilities of the branches is computed. The only difficult part in this expression is the bivariate normal integral for whose computation fast numerical procedures exist [60].

Remark. Recall from the beginning of this chapter, that we required the Hyper-Erlang distribution to have at least two different branches, such that the uniformly distributed random number generated from the ARMA base process can be used to select between the branches. If the two branches describe different Erlang distributions, but have the same mean value, e.g. the first branch consists of one phase with rate 2 and the second branch consists of two phases each with rate 4, we have that $E[Y_t Y_{t+h}] = E[Y]^2$ according to Equation 5.6 and the correlation from Equation 5.4 is always zero. In fact, we require the Hyper-Erlang distribution to have at least two branches with different expected durations to be able to model

autocorrelations with the resulting CHEP. However, all the considerations in this chapter also hold for the special case of a HerD that does not have two branches with different expected durations. Moreover, in Section 6.4 transformations are presented that can modify the representation of such a HErD, so that it can be used for modeling non-zero autocorrelations anyway.

Remark. Note, that $\varphi_\rho(a,b) = \varphi_\rho(b,a)$ holds for the standard bivariate normal distribution (see also Equation B.2 in Appendix B). Thus, in Equation 5.6 the expressions for the branches $i = k, j = l$ and $i = l, j = k, k \neq l$ are equal and the term is computed twice. We may split Equation 5.6 in two terms, one for $i = j$ and one for $i < j$, to save some numerical approximations of the double integral:

$$\begin{aligned} E[Y_t Y_{t+h}] &= \sum_i \left(\frac{S_i}{\lambda_i} \frac{S_i}{\lambda_i} \int_{\Phi^{-1}(\underline{b}_i)}^{\Phi^{-1}(\bar{b}_i)} \int_{\Phi^{-1}(\underline{b}_i)}^{\Phi^{-1}(\bar{b}_i)} \varphi_{\rho_h}(z_t, z_{t+h}) dz_t dz_{t+h} \right) \\ &+ 2 \sum_{i,j,i<j} \left(\frac{S_i}{\lambda_i} \frac{S_j}{\lambda_j} \int_{\Phi^{-1}(\underline{b}_j)}^{\Phi^{-1}(\bar{b}_j)} \int_{\Phi^{-1}(\underline{b}_i)}^{\Phi^{-1}(\bar{b}_i)} \varphi_{\rho_h}(z_t, z_{t+h}) dz_t dz_{t+h} \right). \end{aligned} \tag{5.7}$$

For the theoretical considerations in this work we will use the shorter expression from Equation 5.6, but for implementations Equation 5.7 is recommended.

Using Equation 5.6 it is possible to compute the CHEP correlation for a given base process correlation. For fitting CHEPs one is usually interested in a relation that describes the opposite direction, i.e. given the desired CHEP correlation one wants to compute the corresponding base process correlation. Unfortunately, it is not possible to compute the latter analytically and thus, one has to use a search algorithm that starts with an arbitrary base process correlation and tries to find a base process correlation that yields the desired CHEP correlation as it was done for ARTA processes as well. For this search algorithm to work it is necessary to establish a few properties about the relation of the correlation functions of the base process and the CHEP, e.g. if the base process correlation increases, does the CHEP correlation increase as well? Does a base process correlation less than zero imply a CHEP correlation less than zero? These questions will be answered by the following propositions and theorems.

Observe from Equation 5.6 that the lag-h autocorrelation of the Y_t is a function of the lag-h correlation of the Z_t, which appears in φ_{ρ_h}. Let the function $\omega(\rho_h)$ denote the autocorrelation of the Y_t for a given autocorrelation ρ_h of the Z_t at lag h.

Proposition 5.1. *For any Hyper-Erlang distribution F_Y (with at least two ordered branches), we have that $\rho_h = 0 \Rightarrow \omega(\rho_h) = 0$.*

Proof. If $\rho_h = 0$ then Z_t and Z_{t+h} are independent. Consequently, one obtains for

Equation 5.5

$$\begin{aligned}
E[Y_t Y_{t+h}] &= \sum_{i,j}\left(\frac{S_i}{\lambda_i}\frac{S_j}{\lambda_j}E\left[\delta(\Phi(Z_t),i)\delta(\Phi(Z_{t+h}),j)\right]\right)\\
&= \sum_{i,j}\left(\frac{S_i}{\lambda_i}\frac{S_j}{\lambda_j}E\left[\delta(\Phi(Z_t),i)\right]E\left[\delta(\Phi(Z_{t+h}),j)\right]\right)\\
&= \sum_{i,j}\left(\frac{S_i}{\lambda_i}\frac{S_j}{\lambda_j}(\bar{b}_i-\underline{b}_i)(\bar{b}_j-\underline{b}_j)\right)\\
&= \sum_{i,j}\left(\frac{S_i}{\lambda_i}\tau_i\frac{S_j}{\lambda_j}\tau_j\right)\\
&= \sum_{i}\left(\frac{S_i}{\lambda_i}\tau_i\right)\sum_{j}\left(\frac{S_j}{\lambda_j}\tau_j\right)\\
&= E[Y]E[Y] = E[Y]^2
\end{aligned}$$

The last equation holds because $\sum_i(\tau_i S_i/\lambda_i)$ is the first moment of a Hyper-Erlang distribution (cf. Equation 2.14). Using this result for Equation 5.4 one gets

$$Corr[Y_t, Y_{t+h}] = \frac{E[Y_t Y_{t+h}] - E[Y]^2}{Var[Y]} = \frac{E[Y]E[Y] - E[Y]^2}{Var[Y]} = 0.$$

□

Proposition 5.1 states that an uncorrelated base process results in an uncorrelated CHEP.

The most important requirement that allows for applying a search algorithm is the monotonicity of $\omega(\rho_h)$, which is established by the following Theorem:

Theorem 5.1. *For any Hyper-Erlang distribution F_Y (with at least two ordered branches) $\omega(\rho)$ is a nondecreasing function for $-1 \le \rho \le 1$. I.e. for two base process autocorrelations with $\rho_{h_1} \ge \rho_{h_2}$, we have that $\omega(\rho_{h_1}) \ge \omega(\rho_{h_2})$.*

Proof. The proof is similar to the one for Theorem 4.1. The theorem will be proved separately for the cases $-1 \le \rho \le 0$ and $0 \le \rho \le 1$.

- $0 \le \rho \le 1$: This case follows from [153, Theorem 5.3.10], which states that for two normal variables Z_1 and Z_2 with correlation ρ we have $Corr[g(Z_1), g(Z_2)]$ is nondecreasing in ρ for all functions $g(\cdot)$. For

  $$Y_t = g(Z_t) = \sum_{i=1}^{m}\delta(\Phi(Z_t),i)X_t^{(S_i,\lambda_i)}$$

  and

  $$Y_{t+h} = g(Z_{t+h}) = \sum_{i=1}^{m}\delta(\Phi(Z_{t+h}),i)X_{t+h}^{(S_i,\lambda_i)}$$

  we obtain that $\omega(\rho_h) = Corr[Y_t, Y_{t+h}]$ is nondecreasing for $0 \le \rho_h \le 1$.

- $-1 \leq \rho \leq 0$: This case can be shown by a slightly modified version of the proof for Theorem 1 in [47], which uses ideas from the proof for Theorem 5.3.10 in [153]. Let (X_1, X_2) and (Z_1, Z_2) have standard bivariate normal distribution with common mean $\mu = 0$, common variance $\sigma^2 = 1$ and correlations ρ_X and ρ_Z with $0 \leq \rho_Z < \rho_X < 1$, respectively. Define

$$g(X) = \sum_{i=1}^{m} \delta(\Phi(X), i) X_t^{(S_i, \lambda_i)}$$

as in the previous case. It will be shown that this implies

$$Corr[g(X_1), g(-X_2)] \leq Corr[g(Z_1), g(-Z_2)]$$

or (since we have a standard bivariate normal distribution) equivalently

$$E[g(X_1)g(-X_2)] \leq E[g(Z_1)g(-Z_2)]. \tag{5.8}$$

From these results it will immediately follow that $\omega(\rho)$ is nondecreasing for $-1 \leq \rho \leq 0$.

The key idea of the proof is to represent the $(X_1, -X_2), (Z_1, -Z_2)$ with standard bivariate normal distribution by different combinations of standard normal random variables, such that these combinations have the same distribution as $(X_1, -X_2)$ and $(Z_1, -Z_2)$, respectively. Using these combinations one can derive expressions for $E[g(X_1)g(-X_2)]$ and $E[g(Z_1)g(-Z_2)]$ that prove Equation 5.8.
Note, that for the standard bivariate normal variable (Y_1, Y_2) with correlation ρ_Y we may write [90]

$$Y_i = (\sqrt{\rho_Y})\, N_0 + \left(\sqrt{1-\rho_Y}\right) N_i, \qquad i = 1, 2 \tag{5.9}$$

where N_0, N_1, N_2 are independent standard normal random variables. Moreover, for two standard normal variables N_a, N_b we have that $aN_a + bN_b$ has normal distribution with zero mean and variance $a^2 + b^2$. Hence, we may write Equation 5.9 as

$$Y_i = \left(\sqrt{\rho_Y - b}\right) N_a + \sqrt{b} N_b + \left(\sqrt{1-\rho_Y}\right) N_i, \qquad i = 1, 2 \tag{5.10}$$

if we replace N_0 in Equation 5.9 by a combination of N_a and N_b. If the N_i are replaced by a combination of the standard normal random variables N_{a_i} and N_{b_i}, we may write

$$Y_i = (\sqrt{\rho_Y})\, N_0 + \left(\sqrt{1-a}\right) N_{a_i} + \left(\sqrt{a-\rho_Y}\right) N_{b_i}, \qquad i = 1, 2. \tag{5.11}$$

Now, let T_1, T_2, V_1, V_2, W be random variables with standard normal distribution. Then $(X_1, -X_2)$ and

$$\begin{aligned} &\left(\left(\sqrt{1-\rho_X}\right) T_1 + \left(\sqrt{\rho_X - \rho_Z}\right) V_1 + \sqrt{\rho_Z} W, \right. \\ &\left. -\left(\sqrt{1-\rho_X}\right) T_2 - \left(\sqrt{\rho_X - \rho_Z}\right) V_1 - \sqrt{\rho_Z} W\right) \end{aligned}$$

are identically distributed according to Equation 5.10. The same holds for $(Z_1, -Z_2)$ and

$$\begin{aligned}&\left(\left(\sqrt{1-\rho_X}\right)T_1 + \left(\sqrt{\rho_X-\rho_Z}\right)V_1 + \sqrt{\rho_Z}W,\right.\\ &\left.-\left(\sqrt{1-\rho_X}\right)T_2 - \left(\sqrt{\rho_X-\rho_Z}\right)V_2 - \sqrt{\rho_Z}W\right),\end{aligned}$$

because of Equation 5.11. Note, that the Z_i depend on V_i while the X_i depend on the common V_1. Then, according to [47] we may write

$$\begin{aligned}&E[g(X_1)g(-X_2)]\\ =\;& E\left[E\left\{E\left[g\left(\left(\sqrt{1-\rho_X}\right)T_1 + \left(\sqrt{\rho_X-\rho_Z}\right)V_1 + \sqrt{\rho_Z}W\right)\right.\right.\\ &\left.\left.g\left(-\left(\sqrt{1-\rho_X}\right)T_2 - \left(\sqrt{\rho_X-\rho_Z}\right)V_1 - \sqrt{\rho_Z}W\right)|(V_1,W)=(v_1,w)\right]W=w\right\}\Big]\\ =\;& E\left[E\left\{\Psi_W(V_1)\Psi_{-W}(-V_1)|W=w\right\}\right]\end{aligned}$$

where the conditional expectation $\Psi_W(V_1)$ is given by

$$\Psi_w(v_1) = E\left[g\left(\left(\sqrt{1-\rho_X}\right)T + \left(\sqrt{\rho_X-\rho_Z}\right)V_1 + \sqrt{\rho_Z}W\right)|(V_1,W)=(v_1,w)\right].$$

Because T_1 and T_2 are independent $N(0,1)$ random variables and T_1 and $-T_2$ have the same distribution, $\Psi_w(v_1)$ is an independent $N(0,1)$ random variable with respect to T (cf. [47]).
Note, that we required the Erlang branches to be sorted according to their mean values and hence for a fixed W the function $\Psi_w(v_1) = E[g(\cdot)]$ is nondecreasing in v_1. Similarly, $-\Psi_{-w}(\cdot)$ is nonincreasing. Then (cf. [47]), for a random variable $U \sim U(0,1)$ and $V_1 = \Phi^{-1}(U)$ and $-V_1 = \Phi^{-1}(1-U)$ the variance

$$\begin{aligned}&Var[\Psi_W(V_1) - (-\Psi_{-W}(-V_1))]\\ =\;& Var[\Psi_W(V_1)] + Var[-\Psi_{-W}(-V_1)] - 2Cov[\Psi_W(V_1), -\Psi_{-W}(-V_1)]\end{aligned}$$

is minimized and thus, the covariance $Cov[\Psi_W(V_1), -\Psi_{-W}(-V_1)]$ is maximized [140]. This implies that the covariance $Cov[\Psi_W(V_1), \Psi_{-W}(-V_1)]$ is minimized. Then,

$$\begin{aligned}E\left\{\Psi_W(V_1)\Psi_{-W}(-V_1)|W=w\right\} &\leq E\left\{\Psi_W(V_1)|W=w\right\}E\left\{\Psi_{-W}(-V_1)|W=w\right\}\\ &= E\left\{\Psi_W(V_1)|W=w\right\}E\left\{\Psi_{-W}(-V_2)|W=w\right\}.\end{aligned}$$

The inequality holds because the minimum expected value is smaller than the expected value under independence and the last equation holds because V_1 and V_2 have the same distribution.
Since the above inequality holds for any w, this implies that

$$\begin{aligned}E[g(X_1)g(-X_2)] &= E\left[E\left\{\Psi_W(V_1)\Psi_{-W}(-V_1)|W\right\}\right]\\ &\leq E\left[E\left\{\Psi_W(V_1)|W\right\}\right]E\left[E\left\{\Psi_{-W}(-V_2)|W\right\}\right].\end{aligned}$$

Since V_1 and V_2 are independent one obtains for $E[g(Z_1)g(-Z_2)]$

$$\begin{aligned}E[g(Z_1)g(-Z_2)] &= E\left[E\left\{\Psi_W(V_1)\Psi_{-W}(-V_2)|W\right\}\right]\\ &= E\left[E\left\{\Psi_W(V_1)|W\right\}\right]E\left[E\left\{\Psi_{-W}(-V_2)|W\right\}\right].\end{aligned}$$

Thus, $E[g(X_1)g(-X_2)] \leq E[g(Z_1), g(-Z_2)]$.

Now, let (X_1', X_2') and (Z_1', Z_2') be standard bivariate normal distributed random variables with correlation $-1 < \rho_{X'} < \rho_{Z'} \leq 0$. Then, $(X_1', -X_2')$ and $(Z_1', -Z_2')$ have correlation $0 \leq \rho_{Z'} < \rho_{X'} < 1$ and using the above considerations it immediately follows, that

$$E[g(X_1')g(X_2')] \leq E[g(Z_1'), g(Z_2')].$$

□

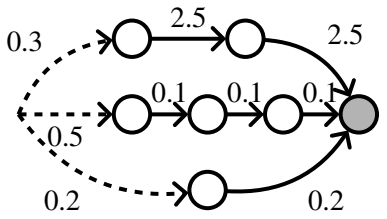
Remark. Recall, that we required the Erlang branches to be sorted according to their mean values in Definition 5.1. This requirement was used in Theorem 5.1 to prove that $\omega(\rho)$ is nondecreasing for $-1 \leq \rho \leq 0$. It should be noted, that this requirement is not just a technical requirement necessary for proving the theorem. Instead, ignoring the required order of the Erlang branches could lead to serious issues with the autocorrelation structure of a CHEP. Figure 5.2 visualizes the impact of the order of the Erlang branches on the autocorrelation structure of a CHEP. Figure 5.2(a) shows an example Hyper-Erlang distribution with 3 branches that are not ordered according to the mean values as required. If we assume that the upper branch is the first and the lower branch the last one, one sees that the mean value for the second branch is 30 and for the third branch 5. Thus, these two branches have to be switched to fulfill the requirement for CHEPs. The valid Hyper-Erlang distribution with ordered branches is shown in Figure 5.2(b). Figure 5.2(c) visualizes the impact of the sort order on the autocorrelation of a CHEP. The CHEP with invalid/unsorted HErD uses the distribution from Figure 5.2(a) and the CHEP with valid/sorted HErD the distribution from Figure 5.2(b), respectively. The plot shows the correlation of the CHEP $\omega(\rho)$ for different base process autocorrelations $-1 \leq \rho \leq 1$. As one can see, $\omega(\rho)$ is nondecreasing for $0 \leq \rho \leq 1$ in both cases (recall from the prove of Theorem 5.1 that ordering the branches was not necessary for the case $0 \leq \rho \leq 1$), but for $-1 \leq \rho \leq 0$ only the CHEP with sorted HErD has the desired property.

This example clearly shows that ordering the branches according to the mean values is not only necessary to avoid potential problems with the CHEP autocorrelation structure, but leads to a unique definition of the autocorrelation structure as well, since the order of branches has some impact on almost all $\omega(\rho)$ values.

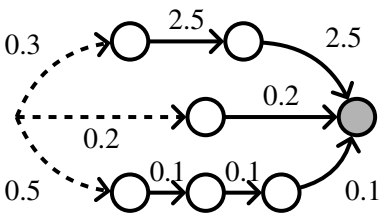
The results from Proposition 5.1 and Theorem 5.1 imply the following statement:

Proposition 5.2. *For any Hyper-Erlang distribution F_Y (with at least two ordered branches), we have that*

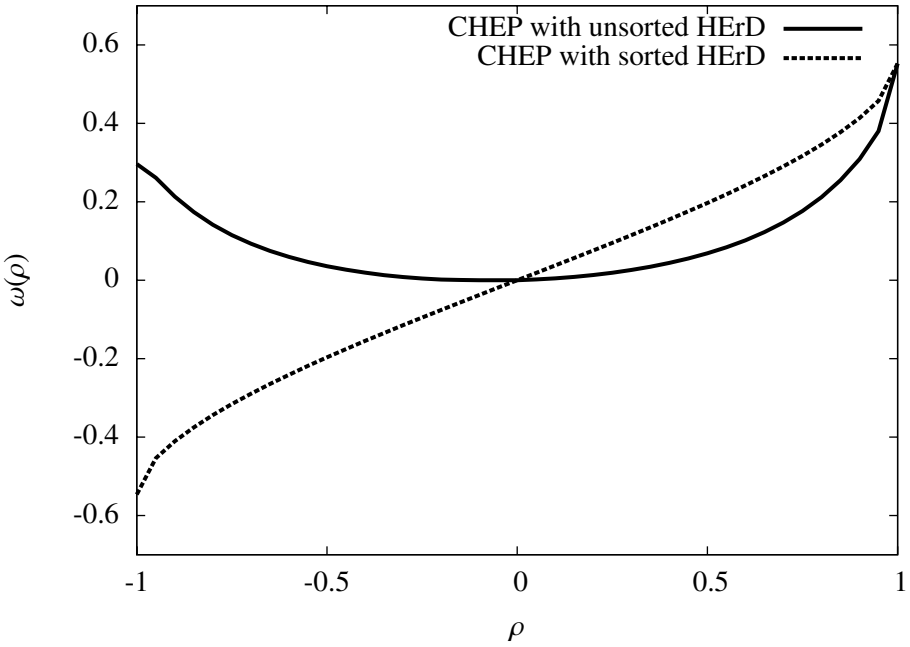
1. $\rho_h \leq 0 \Rightarrow \omega(\rho_h) \leq 0$ *and*
2. $\rho_h \geq 0 \Rightarrow \omega(\rho_h) \geq 0$.



(a) HErD with unsorted branches



(b) HErD with sorted branches



(c) autocorrelations

Figure 5.2.: Impact of the order of Erlang branches on CHEP autocorrelation

Proof. The proposition follows immediately from Proposition 5.1 and Theorem 5.1. Since $\omega(\rho)$ is nondecreasing and $\omega(0) = 0$, we have that $\omega(\rho) \leq 0$ for $\rho \leq 0$ and $\omega(\rho) \geq 0$ for $\rho \geq 0$, respectively. □

For a CHEP the minimal and maximal possible autocorrelation can be computed using the following proposition.

Proposition 5.3. *The maximal and minimal possible autocorrelations $\hat{\rho}_{max}$ and $\hat{\rho}_{min}$ for a CHEP with Hyper-Erlang marginal distribution F_Y are given by $\hat{\rho}_{max} = \omega(1)$ and $\hat{\rho}_{min} = \omega(-1)$, respectively.*

Proof. The proposition immediately follows from Theorem 5.1. □

If the minimal and maximal possible autocorrelations are known it is desirable to know, whether all values between $\hat{\rho}_{min}$ and $\hat{\rho}_{max}$ can be obtained:

Theorem 5.2. *For a Hyper-Erlang distribution F_Y (with at least two ordered branches) $\omega(\rho)$ is a continuous function.*

Proof. Since the mean $E[Y]$ and the variance $Var[Y]$ from Equation 5.4 are independent of the base process autocorrelation ρ we only have to show that the joint moment $E[Y_t Y_{t+h}]$ from Equation 5.6 is a continuous function regarding ρ. Note, that the sum and the product of continuous functions result in a continuous function and moreover, that the mean value of branch i, i.e. S_i/λ_i is of course independent of ρ. Thus, we only have to show that the standard bivariate normal distribution

$$\int_{\Phi^{-1}(\underline{b}_j)}^{\Phi^{-1}(\bar{b}_j)} \int_{\Phi^{-1}(\underline{b}_i)}^{\Phi^{-1}(\bar{b}_i)} \varphi_\rho(z_1, z_2) dz_1 dz_2$$

is a continuous function for arbitrary but fixed $\underline{b}_i, \bar{b}_i, \underline{b}_j, \bar{b}_j$, which follows from [152, Lemma 1.1]. □

Equation 5.5 given above can be used to compute the first joint moment, which is necessary for the autocorrelation. The equation can be generalized for computing arbitrary joint moments:

$$E[Y_t^k Y_{t+h}^l] = E\left[\left(\sum_{i=1}^{m} \delta(U_t, i) X_t^{(S_i,\lambda_i)}\right)^k \left(\sum_{j=1}^{m} \delta(U_{t+h}, j) X_{t+h}^{(S_j,\lambda_j)}\right)^l\right]. \tag{5.12}$$

Equation 5.12 can be simplified by exploiting the properties of $\delta(U, i)$. By application of the multinomial theorem one gets

$$\begin{aligned} & \left(\sum_{i=1}^{m} \delta(U, i) X^{(S_i,\lambda_i)}\right)^k \\ = \; & \left(\delta(U,1)X^{(S_1,\lambda_1)} + \delta(U,2)X^{(S_2,\lambda_2)} + \ldots + \delta(U,m)X^{(S_m,\lambda_m)}\right)^k \\ = \; & \sum_{k_1,k_2,\ldots,k_m} \left(\frac{k!}{k_1!k_2!\ldots k_m!}(\delta(U,1)X^{(S_1,\lambda_1)})^{k_1}(\delta(U,2)X^{(S_2,\lambda_2)})^{k_2}\ldots \right. \\ & \left. (\delta(U,m)X^{(S_m,\lambda_m)})^{k_m}\right) \end{aligned} \tag{5.13}$$

for $k_1 + k_2 + \ldots + k_m = k$. Observe, that $\delta(U, i) = 1$ for exactly one i and $\delta(U, j) = 0$ for all $j \neq i$ and a given U. Hence, all terms in Equation 5.13 that contain $\delta(U, i)$ and $\delta(U, j)$ for $i \neq j$ in the product are zero. Consequently, only those terms for which $k_i = k$ and $k_j = 0$, $j \neq i$ have to be considered. Furthermore, since $\delta(U, i)$ is either zero or one, we have $\delta(U, i)^k = \delta(U, i)$. Thus,

$$\left(\sum_{i=1}^{m} \delta(U, i) X^{(S_i,\lambda_i)}\right)^k = \sum_{i=1}^{m} \left(\delta(U, i) X^{(S_i,\lambda_i)}\right)^k = \sum_{i=1}^{m} \left(\delta(U, i) X^{(S_i,\lambda_i)^k}\right). \tag{5.14}$$

Substituting Equation 5.14 in Equation 5.12 one obtains

$$\begin{aligned}
E[Y_t^k Y_{t+h}^l] &= E\left[\sum_{i=1}^{m}\left(\delta(U_t,i)X_t^{(S_i,\lambda_i)^k}\right)\sum_{j=1}^{m}\left(\delta(U_{t+h},j)X_{t+h}^{(S_j,\lambda_j)^l}\right)\right] \\
&= \sum_{i,j} E\left[\delta(U_t,i)X_t^{(S_i,\lambda_i)^k}\delta(U_{t+h},j)X_{t+h}^{(S_j,\lambda_j)^l}\right] \\
&= \sum_{i,j}\left(E\left[\delta(U_t,i)\delta(U_{t+h},j)\right]E\left[X_t^{(S_i,\lambda_i)^k}\right]E\left[X_{t+h}^{(S_j,\lambda_j)^l}\right]\right) \\
&= \sum_{i,j}\left(E\left[\delta(\Phi(Z_t),i)\delta(\Phi(Z_{t+h}),j)\right]\mu_k^{(erl)}(S_i,\lambda_i)\mu_l^{(erl)}(S_j,\lambda_j)\right) \\
&= \sum_{i,j}\Big(\mu_k^{(erl)}(S_i,\lambda_i)\mu_l^{(erl)}(S_j,\lambda_j) \\
&\qquad \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\delta(\Phi(z_t),i)\delta(\Phi(z_{t+h}),j)\varphi_{\rho_h}(z_t,z_{t+h})dz_t dz_{t+h}\Big) \qquad (5.15)
\end{aligned}$$

where $\mu_k^{(erl)}(S_i,\lambda_i)$ is the k-th moment of an Erlang distribution with S_i phases and rate λ_i as given in Equation 2.12. Again, the known properties of $\delta(U,i)$ can be used to determine the integration bounds in Equation 5.15 resulting in

$$\begin{aligned}
E[Y_t^k Y_{t+h}^l] &= \sum_{i,j}\Big(\mu_k^{(erl)}(S_i,\lambda_i)\mu_l^{(erl)}(S_j,\lambda_j) \qquad (5.16) \\
&\qquad \int_{\Phi^{-1}(\underline{b}_j)}^{\Phi^{-1}(\bar{b}_j)}\int_{\Phi^{-1}(\underline{b}_i)}^{\Phi^{-1}(\bar{b}_i)}\varphi_{\rho_h}(z_t,z_{t+h})dz_t dz_{t+h}\Big).
\end{aligned}$$

5.2. An Alternative Definition of Correlated Hyper-Erlang Processes

Observe from Equation 5.3 that there are basically two possibilities to incorporate the transformed base process $\Phi(Z_t)$ into the process Y_t. The first one, which is the $\delta(\cdot)$ function for selecting the branches of the HErD, was used in Definition 5.1. The second possibility is somewhat hidden in the process description, but of course another uniformly distributed random variable can be used to describe the $X_t^{(S_i,\lambda_i)}$. Let F_i be the Erlang distribution described by the i-th branch of the HErD. Then we may write $X_t^{(S_i,\lambda_i)} = F_i^{-1}(\Phi(Z_t))$ resulting in the process description

$$Y_t = \sum_{i=1}^{m}\delta(U_t,i)F_i^{-1}(\Phi(Z_t)), t = 1,2,\ldots.$$

Now the U_t in $\delta(U_t,i)$ are assumed to be a sequence of independent and identically distributed random numbers from an uniform distribution and the correlated ARMA base process is used to describe the random variables for the Erlang branches using the inverse transform method. This results in the following process:

Definition 5.2 (Correlated Hyper-Erlang Process of Type 2)**.** *A Correlated Hyper-Erlang Process of Type 2 with order n, p, q, denoted $CHEP_2(n, p, q)$, is defined by a Hyper-Erlang distribution*

$$F(y) = 1 - \sum_{i=1}^{m} \tau_i \sum_{j=0}^{S_i-1} \frac{(\lambda_i y)^j}{j!} e^{-\lambda_i y}$$

with n phases divided into m Erlang branches with the number of phases $S_i, i = 1, \ldots, m$, rates $\lambda_i, i = 1, \ldots, m$ and distribution $F_i(y)$ and a stationary Autoregressive Moving Average $ARMA(p, q)$ base process

$$Z_t = \alpha_1 Z_{t-1} + \alpha_2 Z_{t-2} + \ldots + \alpha_p Z_{t-p} + \beta_1 \epsilon_{t-1} + \beta_2 \epsilon_{t-2} + \ldots + \beta_q \epsilon_{t-q} + \epsilon_t.$$

The base process is constructed such that the generated time series $\{Z_t; t = 1, 2, \ldots\}$ has standard normal distribution $N(0, 1)$. Then the Correlated Hyper-Erlang Process of Type 2 describes a time series

$$Y_t = \sum_{i=1}^{m} \delta(U_t, i) F_i^{-1}(\Phi(Z_t)), t = 1, 2, \ldots \tag{5.17}$$

where Φ is the standard normal cumulative distribution function, $\delta(U, i)$ is a function as defined in Equation 5.1 and $U_t, t = 1, 2, \ldots$ is a sequence of independent and identically distributed random numbers with uniform distribution.

Then, for $E[Y_t Y_{t+h}]$ one obtains

$$\begin{aligned}
E[Y_t Y_{t+h}] &= E\left[\left(\sum_{i=1}^{m} \delta(U_t, i) F_i^{-1}(\Phi(Z_t))\right)\left(\sum_{j=1}^{m} \delta(U_{t+h}, j) F_j^{-1}(\Phi(Z_{t+h}))\right)\right] \qquad (5.18)\\
&= E\left[\sum_{i,j} \delta(U_t, i) F_i^{-1}(\Phi(Z_t)) \delta(U_{t+h}, j) F_j^{-1}(\Phi(Z_{t+h}))\right], i, j = 1, \ldots, m\\
&= \sum_{i,j} E\left[\delta(U_t, i) F_i^{-1}(\Phi(Z_t)) \delta(U_{t+h}, j) F_j^{-1}(\Phi(Z_{t+h}))\right]\\
&= \sum_{i,j} \left(E\left[\delta(U_t, i)\right] E\left[\delta(U_{t+h}, j)\right] E[F_i^{-1}(\Phi(Z_t)) F_j^{-1}(\Phi(Z_{t+h}))]\right)\\
&= \sum_{i,j} \left((\tau_i \tau_j) E[F_i^{-1}(\Phi(Z_t)) F_j^{-1}(\Phi(Z_{t+h}))]\right)\\
&= \sum_{i,j} \left((\tau_i \tau_j) \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} F_i^{-1}(\Phi(z_t)) F_j^{-1}(\Phi(z_{t+h})) \varphi_{\rho_h}(z_t, z_{t+h}) dz_t dz_{t+h}\right).
\end{aligned}$$

Observe from Equation 5.18 that the double integral is almost identical to the double integral in the expression for ARTA processes in Equation 4.3. The only difference is in the distributions, which is the marginal distribution F_Y for ARTA models in both cases and the Erlang distributions of the branches F_i, F_j for $CHEP_2$ models. Equation 5.18 is much more complicated to compute than the similar expression from Equation 5.5, because the double integral contains arbitrary Erlang distributions (and not only the bivariate normal density) and F_i^{-1} has to be computed numerically for Erlang distributions, and hence, only the CHEPs from Definition 5.1 will be considered in the following.

5.3. Fitting of the Hyper-Erlang Marginal Distribution

For the CHEP approach the marginal distribution is assumed to be Hyper-Erlang and to have at least two branches. From the existing fitting algorithms presented in Section 2.3.5 the approaches that fit a Hyper-Exponential or a Hyper-Erlang distribution could be used.

In [150, 151] a fast Expectation Maximization algorithm was presented for fitting Hyper-Erlang distributions to a trace, which is implemented in the software `GFIT` and which is the first choice for fitting the marginal distribution for a CHEP in this work. The fitting time of this algorithm is independent of the number of states of the Hyper-Erlang distribution, it only depends on the number of Erlang branches. Furthermore, the authors show in [150] that any probability density function of a nonnegative random variable can be approximated arbitrarily close by a HErD. In the following the basic ideas of the algorithm from [150, 151] will be outlined.

Let $f(x; m, \boldsymbol{S}, \boldsymbol{\tau}, \boldsymbol{\lambda})$ be the density function of a HErD consisting of m Erlang branches (cf. Section 2.3.1). $\boldsymbol{S} = (S_1, S_2, \ldots, S_m) \in \mathbb{N}^m$ is a vector containing the number of phases of each branch, $\boldsymbol{\tau} = (\tau_1, \tau_2, \ldots, \tau_m) \in \mathbb{R}^m$ gives the initial probabilities and $\boldsymbol{\lambda} = (\lambda_1, \lambda_2, \ldots, \lambda_m) \in \mathbb{R}^M$ the rates of each branch. Keeping in mind that $\sum_{i=1}^{m} \tau_i = 1$ a HErD with m Erlang branches has $2m - 1$ continuous (given by the τ_i and λ_i) and m discrete (given by the S_i) parameters. The approach from [151] assumes that the S_i and m are given and fits the τ_i and λ_i for one setting. To obtain values for the discrete parameters the proposed approach tries all possible combinations of m and S_i for a given overall number of states n and selects the best configuration. Since the EM algorithm for a single configuration is very efficient this is possible as long as n is not too large (i.e. $n \leq 10$).

For a single HErD with given m and S_i the EM algorithm has to solve a mixture density parameter estimation problem [28] with parameters $\boldsymbol{\Theta} = (\tau_1, \ldots, \tau_m, \lambda_1, \ldots, \lambda_m)$ and the given observations $\mathcal{T} = (t_1, t_2, \ldots, t_l)$. Mixture density parameter estimation problems are one of the most common applications for the EM algorithm. The general setup assumes that m component densities are mixed together with coefficients τ_i, i.e.

$$p(t_j|\boldsymbol{\Theta}) = \sum_{i=1}^{m} \tau_i p_i(t_j|\lambda_i).$$

For HErDs the components have Erlang distribution

$$p_i(t_j|\lambda_i) = \frac{(\lambda_i t_j)^{S_i - 1}}{(S_i - 1)!} \lambda_i e^{-\lambda_i t_j}.$$

The log-likelihood function for the data and the density expression can be determined as

$$\log(\mathcal{L}(\boldsymbol{\Theta}|\mathcal{T})) = \log \prod_{j=1}^{l} p(t_j|\boldsymbol{\Theta}) = \sum_{j=1}^{l} \log \left(\sum_{i=1}^{m} \tau_i p_i(t_j|\lambda_i) \right). \tag{5.19}$$

To avoid the optimization of an expression that contains the logarithm of a sum Equation 5.19 can be simplified by considering $\mathcal{T}$ as incomplete data, i.e. the existence of unobserved data $\mathcal{Y} = (y_1, \ldots, y_l)$ is assumed. The y_j denote which component density

generated an observation of $\mathcal{T}$, i.e. $y_j = i$ if t_j was generated by the i-th Erlang density. Then Equation 5.19 can be simplified to

$$\log(\mathcal{L}(\mathbf{\Theta}|\mathcal{T}, \mathcal{Y})) = \sum_{j=1}^{l} \log(\tau_{y_j} p_{y_j}(t_j|\lambda_{y_j})). \tag{5.20}$$

Let $\hat{\mathbf{\Theta}} = (\hat{\tau}_1, \hat{\tau}_2, \ldots, \hat{\tau}_m, \hat{\lambda}_1, \hat{\lambda}_2, \ldots, \hat{\lambda}_m)$ be an initial guess for the parameters. Then the $p_i(t_j|\hat{\lambda}_i)$ can be easily computed. Applying Bayes' Rule the probability mass function of the unobserved data $\mathcal{Y}$ given the observed data $\mathcal{T}$ and the estimates $\hat{\mathbf{\Theta}}$ can be computed [151]

$$q(y_j|t_j, \hat{\mathbf{\Theta}}) = \frac{\hat{\tau}_{y_j} p_{y_j}(t_j|\hat{\lambda}_{y_j})}{\sum_{i=1}^{m} \hat{\tau}_i p_i(t_j|\hat{\lambda}_i)}$$

and

$$q(\boldsymbol{y}|\mathcal{T}, \hat{\mathbf{\Theta}}) = \prod_{j=1}^{l} q(y_j|t_j, \hat{\mathbf{\Theta}}), \tag{5.21}$$

where $\boldsymbol{y} = (y_1, \ldots, y_l)$ is an instance of the unobserved data independently drawn from $\mathcal{Y}$. Then the E-step of the EM algorithm is given by [28, 151]

$$Q(\mathbf{\Theta}, \hat{\mathbf{\Theta}}) = E\left[\log(\mathcal{L}(\mathbf{\Theta}|\mathcal{T}, \mathcal{Y}))|\mathcal{T}, \hat{\mathbf{\Theta}}\right] = \sum_{\boldsymbol{y} \in \mathcal{Y}} \log(\mathcal{L}(\mathbf{\Theta}|\mathcal{T}, \boldsymbol{y})) q(\boldsymbol{y}|\mathcal{T}, \hat{\mathbf{\Theta}}). \tag{5.22}$$

By inserting Equations 5.20 and 5.21 into Equation 5.22 an rearranging one obtains [28, 151]

$$Q(\mathbf{\Theta}, \hat{\mathbf{\Theta}}) = \underbrace{\sum_{i=1}^{m} \sum_{j=1}^{l} \log(\tau_i) q(i|t_j, \hat{\mathbf{\Theta}})}_{Q_1} + \underbrace{\sum_{i=1}^{m} \sum_{j=1}^{l} \log(p_i(t_j|\lambda_i)) q(i|t_j, \hat{\mathbf{\Theta}})}_{Q_2}, \tag{5.23}$$

which is maximized in the M-step of the algorithm. Note, that the terms Q_1 and Q_2 in Equation 5.23 are not related and can be maximized separately.

An expression for τ_i can be found applying a Lagrange multiplier resulting in [28]

$$\tau_i = \frac{1}{l} \sum_{j=1}^{l} q(i|t_j, \hat{\mathbf{\Theta}}). \tag{5.24}$$

The λ_i can be maximized according to [151]

$$\lambda_i = \frac{S_i \sum_{j=1}^{l} q(i|t_j, \hat{\mathbf{\Theta}})}{\sum_{j=1}^{l} q(i|t_j, \hat{\mathbf{\Theta}}) t_j}. \tag{5.25}$$

The complete EM algorithm for parameter estimation of a HErD presented in [151] starts with an initial parameter estimate $\hat{\mathbf{\Theta}} = (\hat{\tau_1}, \ldots, \hat{\tau_m}, \hat{\lambda_1}, \ldots, \hat{\lambda_m})$ and performs the E-step from Equation 5.22 and the M-step from Equations 5.24 and 5.25 iteratively until convergence is reached.

5.4. Constructing the ARMA Process

Since the $ARMA(p,q)$ base process has to fulfill the same requirements for CHEPs as for extended ARTA processes, i.e. it has to exhibit an autocorrelation structure $\boldsymbol{\rho} = (\rho_1, \rho_2, \ldots, \rho_r)$ such that the CHEP has autocorrelations $\hat{\boldsymbol{\rho}} = (\hat{\rho}_1, \hat{\rho}_2, \ldots, \hat{\rho}_r)$, it has to ensure that $Z_t \sim N(0,1)$ and it has to be stationary, the approach described in Section 4.2 can be applied for CHEPs as well with only slight modifications.

The adjustment of the variance of the white noise that results in a standard normal distribution for the Z_t can be performed as presented in Section 4.2, i.e. by applying Equation 4.9.

The search algorithm that finds $\boldsymbol{\rho}$ for a given $\hat{\boldsymbol{\rho}}$ of course has to use Equation 5.6 now to compute the CHEP autocorrelation that corresponds to a base process autocorrelation.

For fitting an $ARMA(p,q)$ model to $\boldsymbol{\rho}$ again Equation 4.11 can be minimized.

Note, that the construction of the base process is independent of the number of states of the Hyper-Erlang distribution. However, Equation 5.6 (and thus the search algorithm for determining the base process autocorrelation) depends on the number of branches of the HErD. Once the base process autocorrelation has been computed, the fitting procedure for the $ARMA(p,q)$ model is completely independent of the distribution and only depends on the base process model order (p,q) and the number of autocorrelations to consider.

Chapter 6

Correlated Acyclic Phase-Type Processes

In the following the ideas and concepts presented in Chapter 5 for Hyper-Erlang distributions will be generalized for the case where the marginal distribution is an arbitrary acyclic Phase-type distribution.

For CHEPs the Hyper-Erlang distribution and the ARMA base process have been fitted separately in two steps and are combined when selecting the Erlang branch, i.e. the sequence U_t of uniformly distributed random numbers resulting from a transformed ARMA process is used to choose the Erlang branch for each realization of Y_t obtained from the CHEP. Since acyclic PH distributions cannot be split into branches, another way to incorporate the base process has to be found, which will result in a different representation of the acyclic PH distribution than the usual matrix notation.

An acyclic Phase-type distribution can be represented as a set of elementary series [55]. Each series is one path from an initial state to the absorbing state of the APH distribution and has a probability proportional to the product of the transition rates along the path and to the initial probability of the first state of the path. An example of an APH and its elementary series is shown in Figure 6.1. The APH consists of one absorbing and 3 transient states. All three transient states are entry and exit states. The transition rate matrix, the killing-rate vector and the initial probabilities of the PH distribution are given by

$$\boldsymbol{D}_0 = \begin{bmatrix} -3.0 & 1.0 & 0.5 \\ 0.0 & -2.0 & 0.0 \\ 0.0 & 0.0 & -1.0 \end{bmatrix}, \quad \boldsymbol{t} = \begin{bmatrix} 1.5 \\ 2.0 \\ 1.0 \end{bmatrix} \quad \text{and} \quad \boldsymbol{\pi} = \left[\frac{1}{2}, \frac{1}{3}, \frac{1}{6} \right],$$

respectively. Starting from an entry state there are 5 possibilities (or paths) to reach the absorbing state. Consequently, the PH distribution can be represented by 5 elementary series as shown in Figure 6.1. The rate between two states of an elementary series is equal to the exit rate of the first of the two states (i.e. it is equal to the corresponding entry in the diagonal of $|\boldsymbol{D}_0|$). The probability of the elementary series is computed from the initial probability of the first state of the series and the transition rates along the path. Let $i_1, i_2, \ldots, i_k$ be k states that form the j-th elementary series of a PH distribution. Then, the initial probability of the elementary series τ_j is given by

$$\tau_j = \boldsymbol{\pi}(i_1) \frac{\boldsymbol{D}_0(i_1, i_2)}{-\boldsymbol{D}_0(i_1, i_1)} \frac{\boldsymbol{D}_0(i_2, i_3)}{-\boldsymbol{D}_0(i_2, i_2)} \cdots \frac{\boldsymbol{D}_0(i_{k-1}, i_k)}{-\boldsymbol{D}_0(i_{k-1}, i_{k-1})} \frac{\boldsymbol{t}(i_k)}{-\boldsymbol{D}_0(i_k, i_k)}. \tag{6.1}$$

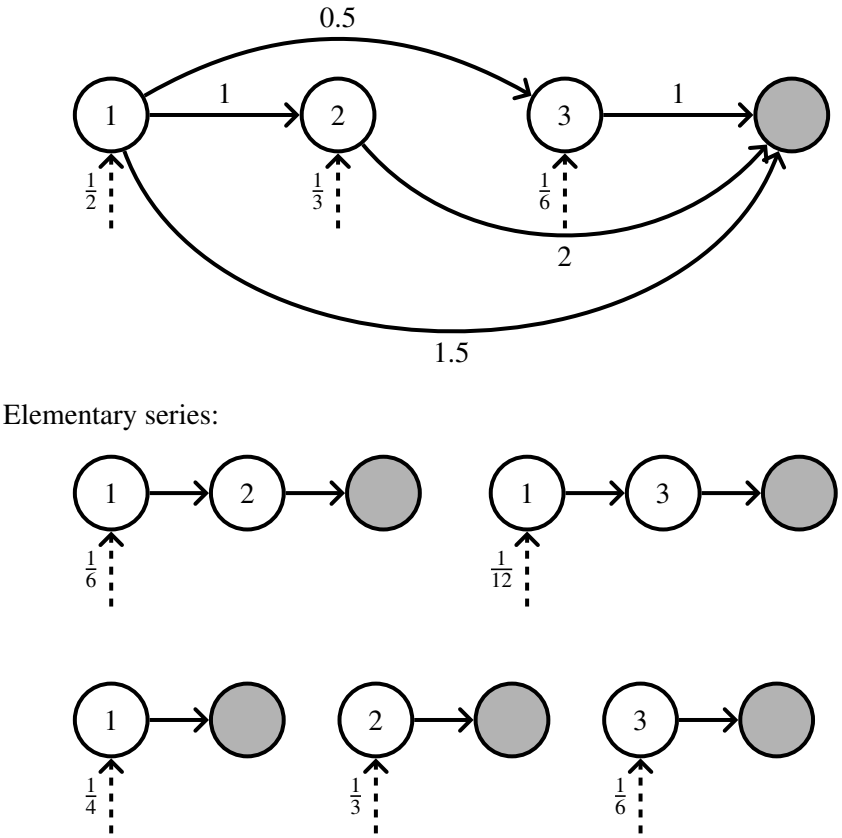


Figure 6.1.: An acyclic PH distribution and its elementary series

For the first elementary series from Figure 6.1 consisting of the states 1 and 2 this would result in

$$\tau_1 = \frac{1}{2}\frac{1}{3}\frac{2}{2} = \frac{1}{6}.$$

The probabilities of the remaining elementary series are computed similarly. Note, that each elementary series describes a Hypo-Exponential distribution (cf. Section 2.3.1) consisting of a number of exponentially distributed phases with potentially different rates.

Representing an acyclic PH distribution by means of its elementary series one can establish similar definitions for an APH with ARMA base process as it was done for Hyper-Erlang distributions in the previous chapter. Observe, that the elementary series of an Hyper-Erlang distribution are equivalent to its Erlang branches. In this way the following considerations are the natural generalization of the ideas presented in Chapter 5.

Now, let m denote the number of elementary series of an acyclic PH distribution and τ_i be the probability of the i-th elementary series ($i = 1, \ldots, m$). Furthermore, let S_i ($i = 1, \ldots, m$) be the number of phases in elementary series i and let $\mathbf{\Lambda_i}$ be a vector of length S_i that contains the transition rates of the i-th series. Again, we assume that the elementary series are ordered according to their mean values, i.e. for two random

variables X_i and X_j that have a Hypo-Exponential distribution corresponding to the i-th and j-th elementary series of the APH distribution we require $i \leq j \Rightarrow E[X_i] \leq E[X_j]$.

Define

$$\begin{aligned} \underline{b}_1 &= 0 \\ \bar{b}_i &= \underline{b}_i + \tau_i \qquad i = 1, \ldots, m \\ \underline{b}_i &= \bar{b}_{i-1} \qquad i = 2, \ldots, m \end{aligned} \tag{6.2}$$

$$\tag{6.3}$$

to be the bounds of the probabilities to chose the different elementary series of an APH distribution and define for a random number U with uniform distribution

$$\delta(U, i) = \begin{cases} 1, & U \in [\underline{b}_i, \bar{b}_i) \\ 0, & \text{otherwise} \end{cases} \tag{6.4}$$

to be a function that selects one elementary series according to U. Assume that $\{X_t^{(\mathbf{\Lambda}_i)}\}, i = 1, \ldots, m$ are sequences of iid random numbers from Hypo-Exponential distributions, each with a vector of rates $\mathbf{\Lambda}_i$. Let $\{U_t\}$ be a sequence of random numbers with uniform distribution on $(0, 1)$. If the $\{U_t\}$ are independent the process

$$Y_t = \sum_{i=1}^{m} \delta(U_t, i) X_t^{(\mathbf{\Lambda}_i)} \tag{6.5}$$

uses the elementary series to describe a sequence of iid random variables with the same acyclic Phase-type distribution that the elementary series have been computed from. To introduce autocorrelation into the sequence from Equation 6.5 the U_t can be constructed in the same way as for CHEPs in Chapter 5, i.e. they are set to $U_t = \Phi(Z_t)$ where Φ is standard normal cdf and the $\{Z_t\}$ result from an $ARMA(p, q)$ process with σ_ϵ^2 set in a way such that the $\{Z_t\}$ have standard normal distribution. This leads to the following definition:

Definition 6.1 (Correlated Acyclic Phase-Type Process)**.** *A Correlated Acyclic Phase-Type Process of order n, p, q, denoted $CAPP(n, p, q)$, is defined by an acyclic Phase-type distribution $(\boldsymbol{D}_0, \boldsymbol{\pi})$ of order n that has m elementary series with S_i phases, probabilities τ_i, and rates $\mathbf{\Lambda}_i = (\mathbf{\Lambda}_i(1), \ldots, \mathbf{\Lambda}_i(S_i))$, $(i = 1, \ldots, m)$, and a stationary Autoregressive Moving Average $ARMA(p, q)$ base process*

$$Z_t = \alpha_1 Z_{t-1} + \alpha_2 Z_{t-2} + \ldots + \alpha_p Z_{t-p} + \beta_1 \epsilon_{t-1} + \beta_2 \epsilon_{t-2} + \ldots + \beta_q \epsilon_{t-q} + \epsilon_t.$$

The base process is constructed such that the generated time series $\{Z_t; t = 1, 2, \ldots\}$ has standard normal distribution $N(0, 1)$ and the elementary series are ordered such that $i \leq j \Rightarrow (\sum_{k=1}^{S_i} 1/\mathbf{\Lambda}_i(k)) \leq (\sum_{l=1}^{S_j} 1/\mathbf{\Lambda}_j(l))$ holds. Then the Correlated Acyclic Phase-Type Process describes a time series

$$Y_t = \sum_{i=1}^{m} \delta(\Phi(Z_t), i) X_t^{(\mathbf{\Lambda}_i)}, t = 1, 2, \ldots \tag{6.6}$$

where Φ is the standard normal cumulative distribution function, $\delta(U, i)$ is a function as defined in Equation 6.4 and $\{X_t^{(\mathbf{\Lambda}_i)}\}, i = 1, \ldots, m$ are sequences of independent and identically distributed random variables with Hypo-Exponential distribution with S_i phases and rates $\mathbf{\Lambda}_i$.

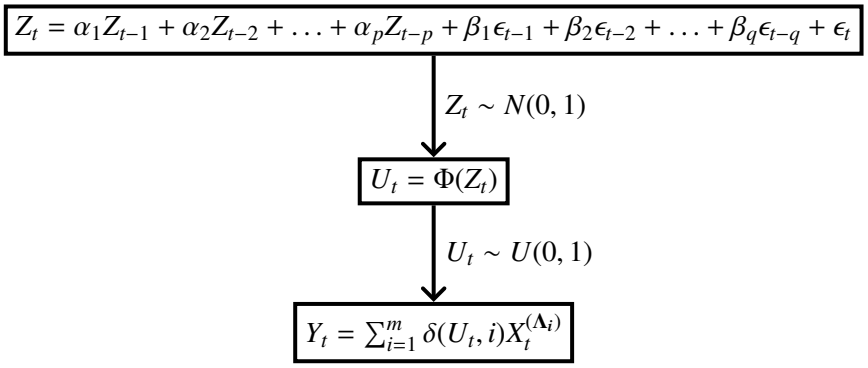


Figure 6.2.: Correlated Acyclic Phase-Type Process

The construction of the Correlated Acyclic Phase-Type Process using an ARMA base process is visualized in Figure 6.2.

Since we introduce the autocorrelation between the elementary series, we require the PH distribution to have at least two elementary series, implying that the distribution has at least two states and that the number of non-zero entries in the initial probability vector $\boldsymbol{\pi}$ and the killing rate vector $\boldsymbol{t}$ together is greater than or equal to 3 (i.e. the distribution has at least either two entry states and one exit state or one entry state and two exit states). This restriction excludes exponential, Erlang and Hypo-Exponential distributions. The former two are covered by extended ARTA processes from Chapter 4. However, applying the transformations that will be introduced in Section 6.4 these three distributions can be modified such that they can be used for a CAPP.

6.1. Properties of Correlated Acyclic Phase-Type Processes

We can define several properties of CAPPs along the lines of CHEPs (cf. Section 5.1). First, we need to relate the autocorrelation of the CAPP $Corr[Y_t, Y_{t+h}]$ and its base process $Corr[Z_t, Z_{t+h}]$. Recall from Section 5.1, that the autocorrelations of $\{Y_t\}$ can be expressed as

$$Corr[Y_t, Y_{t+h}] = \frac{E[Y_t Y_{t+h}] - E[Y]^2}{Var[Y]}. \tag{6.7}$$

Again, $E[Y]$ and $Var[Y]$ are known and can be computed using Equation 2.11. For $E[Y_t Y_{t+h}]$, which remains the only term of interest in Equation 6.7, we get a similar

expression as Equation 5.5:

$$\begin{aligned}
E[Y_t Y_{t+h}] &= E\left[\left(\sum_{i=1}^{m} \delta(U_t, i) X_t^{(\mathbf{\Lambda}_i)}\right)\left(\sum_{j=1}^{m} \delta(U_{t+h}, j) X_{t+h}^{(\mathbf{\Lambda}_j)}\right)\right] \\
&= E\left[\sum_{i,j} \delta(U_t, i) X_t^{(\mathbf{\Lambda}_i)} \delta(U_{t+h}, j) X_{t+h}^{(\mathbf{\Lambda}_j)}\right], i, j = 1, \ldots, m \\
&= \sum_{i,j} E\left[\delta(U_t, i) X_t^{(\mathbf{\Lambda}_i)} \delta(U_{t+h}, j) X_{t+h}^{(\mathbf{\Lambda}_j)}\right] \\
&= \sum_{i,j} \left(E\left[\delta(U_t, i)\delta(U_{t+h}, j)\right] E[X_t^{(\mathbf{\Lambda}_i)}] E[X_{t+h}^{(\mathbf{\Lambda}_j)}]\right) \\
&= \sum_{i,j} \left(\left(\sum_{s=1}^{S_i} \frac{1}{\mathbf{\Lambda_i}(s)}\right)\left(\sum_{s=1}^{S_j} \frac{1}{\mathbf{\Lambda_j}(s)}\right) E\left[\delta(U_t, i)\delta(U_{t+h}, j)\right]\right) \\
&= \sum_{i,j} \left(\left(\sum_{s=1}^{S_i} \frac{1}{\mathbf{\Lambda_i}(s)}\right)\left(\sum_{s=1}^{S_j} \frac{1}{\mathbf{\Lambda_j}(s)}\right) E\left[\delta(\Phi(Z_t), i)\delta(\Phi(Z_{t+h}), j)\right]\right) \\
&= \sum_{i,j} \left(\left(\sum_{s=1}^{S_i} \frac{1}{\mathbf{\Lambda_i}(s)}\right)\left(\sum_{s=1}^{S_j} \frac{1}{\mathbf{\Lambda_j}(s)}\right)\right. \\
&\qquad \left.\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \delta(\Phi(z_t), i)\delta(\Phi(z_{t+h}), j)\varphi_{\rho_h}(z_t, z_{t+h}) dz_t dz_{t+h}\right) \qquad (6.8)
\end{aligned}$$

where $\varphi_{\rho_h}(z_t, z_{t+h})$ is the bivariate standard normal probability density function with correlation $\rho_h = Corr[Z_t, Z_{t+h}]$.

Since $\delta(u, i)$ from Equation 6.4 is 1 for $u \in [\underline{b}_i, \bar{b}_i)$ and 0 otherwise, the integration bounds in Equation 6.8 can be simplified and $\delta(\cdot)$ can be omitted in the double integral:

$$\begin{aligned}
E[Y_t Y_{t+h}] &= \sum_{i,j} \left(\left(\sum_{s=1}^{S_i} \frac{1}{\mathbf{\Lambda_i}(s)}\right)\left(\sum_{s=1}^{S_j} \frac{1}{\mathbf{\Lambda_j}(s)}\right)\right. \\
&\qquad \left.\int_{\Phi^{-1}(\underline{b}_j)}^{\Phi^{-1}(\bar{b}_j)} \int_{\Phi^{-1}(\underline{b}_i)}^{\Phi^{-1}(\bar{b}_i)} \varphi_{\rho_h}(z_t, z_{t+h}) dz_t dz_{t+h}\right). \qquad (6.9)
\end{aligned}$$

In summary $E[Y_t Y_{t+h}]$ can be computed using the mean values of the elementary series (which are Hypo-Exponential distributions) and the double integral of the bivariate standard normal density where the integration bounds are determined by the probabilities of the elementary series. Comparing Equation 5.6 with Equation 6.9 confirms the observation that CAPPs are the natural generalization of the CHEPs. The mean values of Erlang distributions in Equation 5.6 have been replaced by mean values of Hypo-Exponential distributions and instead of branches we compute the sum over all combinations of elementary series. For the computation of the double integral of the bivariate standard normal density again [60] can be used.

Remark. Analog to the case of CHEPs with Hyper-Erlang distributions Equation 6.9 will result in $E[Y_t Y_{t+h}] = E[Y]^2$ if the APH distribution has only two elementary

series that have the same expected durations. Again, all the following considerations also hold for this special case and the distribution can be transformed (cf. Section 6.4) to be able to model non-zero autocorrelations.

Remark. Again, we may exploit the fact that $\varphi_\rho(a,b) = \varphi_\rho(b,a)$ holds for the standard bivariate normal distribution to save some approximations of the double integral, i.e. we can split the double sum in Equation 6.9 such that

$$\begin{aligned} E[Y_t Y_{t+h}] &= \sum_i \left(\left(\sum_{s=1}^{S_i} \frac{1}{\mathbf{\Lambda_i}(s)} \right) \left(\sum_{s=1}^{S_i} \frac{1}{\mathbf{\Lambda_i}(s)} \right) \right. \\ &\quad \left. \int_{\Phi^{-1}(\underline{b}_i)}^{\Phi^{-1}(\bar{b}_i)} \int_{\Phi^{-1}(\underline{b}_i)}^{\Phi^{-1}(\bar{b}_i)} \varphi_{\rho_h}(z_t, z_{t+h}) dz_t dz_{t+h} \right) \\ &+ 2 \sum_{i,j,i<j} \left(\left(\sum_{s=1}^{S_i} \frac{1}{\mathbf{\Lambda_i}(s)} \right) \left(\sum_{s=1}^{S_j} \frac{1}{\mathbf{\Lambda_j}(s)} \right) \right. \\ &\quad \left. \int_{\Phi^{-1}(\underline{b}_j)}^{\Phi^{-1}(\bar{b}_j)} \int_{\Phi^{-1}(\underline{b}_i)}^{\Phi^{-1}(\bar{b}_i)} \varphi_{\rho_h}(z_t, z_{t+h}) dz_t dz_{t+h} \right). \end{aligned} \tag{6.10}$$

In Section 5.1 several properties on the relation of the autocorrelation of a CHEP and its base process have been established. The same properties hold for the autocorrelation of CAPPs. Again, let the function $\omega(\rho_h)$ denote the autocorrelation of the Y_t for a given base process autocorrelation ρ_h of the Z_t at lag h. Then the following propositions and theorems, which will be stated without proofs, since the proofs are analog to the ones presented in Section 5.1, can be established for the autocorrelation function. The first proposition states, that an uncorrelated base process implies zero correlation for the CAPP.

Proposition 6.1. *For any acyclic Phase-type distribution F_Y (with at least two ordered elementary series), we have that $\rho_h = 0 \Rightarrow \omega(\rho_h) = 0$.*

For applying a search procedure to find a base process correlation for a desired CAPP autocorrelation $\omega(\rho_h)$ is required to be nondecreasing.

Theorem 6.1. *For any acyclic Phase-type distribution F_Y (with at least two ordered elementary series) $\omega(\rho)$ is a nondecreasing function for $-1 \leq \rho \leq 1$. I.e. for two base process autocorrelations with $\rho_1 \geq \rho_2$, we have that $\omega(\rho_1) \geq \omega(\rho_2)$.*

Proposition 6.1 and Theorem 6.1 allow for statements on the relation of a negative (or positive) base process autocorrelation and the corresponding CAPP autocorrelation and on the maximal and minimal possible autocorrelation of a CAPP.

Proposition 6.2. *For any acyclic Phase-type distribution F_Y (with at least two ordered elementary series), we have that*

1. *$\rho_h \leq 0 \Rightarrow \omega(\rho_h) \leq 0$ and*
2. *$\rho_h \geq 0 \Rightarrow \omega(\rho_h) \geq 0$.*

Proposition 6.3. *The maximal and minimal possible autocorrelations $\hat{\rho}_{max}$ and $\hat{\rho}_{min}$ for a CAPP with APH marginal distribution F_Y are given by $\hat{\rho}_{max} = \omega(1)$ and $\hat{\rho}_{min} = \omega(-1)$, respectively.*

Theorem 6.2. *For an acyclic Phase-type distribution F_Y (with at least two ordered elementary series) $\omega(\rho)$ is a continuous function.*

Finally, Equation 6.9 can be generalized to compute arbitrary joint moments, i.e.

$$E[Y_t^k Y_{t+h}^l] = \sum_{i,j} \left(\mu_k^{(hexp)}(S_i, \mathbf{\Lambda_i}) \mu_l^{(hexp)}(S_j, \mathbf{\Lambda_j}) \int_{\Phi^{-1}(\underline{b}_j)}^{\Phi^{-1}(\bar{b}_j)} \int_{\Phi^{-1}(\underline{b}_i)}^{\Phi^{-1}(\bar{b}_i)} \varphi_{\rho_h}(z_t, z_{t+h}) dz_t dz_{t+h} \right) \tag{6.11}$$

where $\mu_k^{(hexp)}(S_i, \mathbf{\Lambda_i})$ is the k-th moment of a Hypo-Exponential distribution with S_i phases and rate vector $\mathbf{\Lambda_i}$ that can be computed using Equation 2.11.

6.2. Fitting the APH Marginal Distribution

For fitting the APH marginal distribution for CAPPs the moment matching approach from [42] is used. Of course, any of the fitting approaches from Section 2.3.5 that yields an acyclic PH distribution could be applied, but moment fitting is much faster than the other presented PH fitting techniques that are in most cases EM algorithms. Additionally, an EM algorithm was already used for fitting the marginal distribution of CHEPs in Chapter 5 and hence, that approach could be used if the fitting quality of the moment matching is not good enough.

The approach from [42] was originally developed as part of a two-step MAP fitting approach and consists of a PH fitting step that results in a APH distribution in series canonical form and a subsequent transformation step to increase the flexibility for the final MAP fitting step. Here only the PH fitting step is summarized. Transformations are treated in the following Section 6.4. Note, that an APH with n states in canonical form has up to n entry states and 1 exit state. For MAP fitting it is desirable and usually necessary to have more exit states to increase the number of non-zero entries in Matrix $\boldsymbol{D}_1$. For CAPP fitting the series canonical form fulfills the requirement of having at least two elementary series if there are two entry states and a transformation is only required under the conditions described in Section 6.4.

Let $\hat{\mu}_i, i = 1, \ldots, K$ be a set of moments that are computed from a given trace according to Equation 1.3. The fitting algorithm from [42] tries to solve

$$\min_{(\boldsymbol{\pi}, \boldsymbol{D}_0)} \left(\sum_{i=1,\ldots,K} \left(\kappa_i \frac{\mu_i}{\hat{\mu}_i} - \kappa_i \right)^2 \right)$$

where μ_i are the moments of the APH distribution $(\boldsymbol{\pi}, \boldsymbol{D}_0)$ and κ_i are weights to privilege lower order moments. This general optimization problem is solved approximately by the repeated optimization of simpler problems. In canonical form the APH of order n has parameters $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_n)$ and $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_n)$. The first optimization step assumes that the λ_i are known and optimizes according to $\boldsymbol{\pi}$. In the second step $\boldsymbol{\pi}$ is assumed to be given and the algorithm optimizes according to λ_i.

In canonical form the matrix $\boldsymbol{D}_0$ has bidiagonal form resulting in the moment matrix

$$\boldsymbol{M} = -\boldsymbol{D}_0^{-1} = \begin{bmatrix} \frac{1}{\lambda_1} & \frac{1}{\lambda_2} & \cdots & \cdots & \frac{1}{\lambda_n} \\ 0 & \frac{1}{\lambda_2} & \cdots & \cdots & \frac{1}{\lambda_n} \\ \vdots & \ddots & \ddots & & \vdots \\ 0 & \cdots & \cdots & 0 & \frac{1}{\lambda_n} \end{bmatrix}.$$

Then $\bar{\boldsymbol{\mu}}_i = \boldsymbol{M}^i \mathbf{1}$ is the vector of the i-th conditional moment and the moments are computed from $\mu_i = \boldsymbol{\pi}\bar{\boldsymbol{\mu}}_i$. If the λ_i are assumed to be given, $\boldsymbol{M}$ and $\bar{\boldsymbol{\mu}}_i$ are known and the minimization problem according to $\boldsymbol{\pi}$ becomes [42]

$$\min_{\boldsymbol{\pi}:\boldsymbol{\pi}\mathbf{1}=1,\boldsymbol{\pi}\geq 0} \left(\sum_{i\in 1,\ldots,K} \left(\kappa_i \frac{\boldsymbol{\pi}\bar{\boldsymbol{\mu}}_i}{\hat{\mu}_i} - \kappa_i \right)^2 \right), \tag{6.12}$$

which is a non negative least squares (NNLS) problem with a single linear constraint that can be solved with standard approaches for NNLS problems [106].

Optimization according to $\boldsymbol{\lambda}$ is more difficult. Assume that $\boldsymbol{\pi}$ and all $\lambda_i, i \neq r$ are given. Then the approach from [42] aims at optimizing the single rate λ_r. If λ_r is modified such that $1/\lambda_r$ becomes $1/\lambda_r+\Delta$, the new moments matrix is $\boldsymbol{M}_{\Delta,r} = \boldsymbol{M}+\Delta\boldsymbol{M}_r$ where $\boldsymbol{M}_r$ is a $n \times n$ matrix with 1 in the positions $(1,r),\ldots,(r,r)$ and 0 elsewhere. Then the i-th conditional moment is $\mu_i(\Delta, r) = \boldsymbol{\pi}\left(\boldsymbol{M} + \Delta\boldsymbol{M}_r\right)^i \mathbf{1}$ and the minimization problem according to λ_r becomes

$$\min_{\Delta} \left(\sum_{i=1,\ldots,K} \left(\kappa_i \frac{\mu_i(\Delta, r)}{\hat{\mu}_i} - \kappa_i \right)^2 \right). \tag{6.13}$$

Equation 6.13 is not a least squares problem and has to be minimized using standard optimization techniques.

Starting from an initial guess for $\boldsymbol{\pi}$ and $\boldsymbol{\lambda}$ the fitting approach from [42] iteratively optimizes Equation 6.12 for $\boldsymbol{\pi}$ and Equation 6.13 for all λ_i until convergence is reached. This approach is very fast and usually only takes a few seconds for fitting an APH distribution.

6.3. Constructing the ARMA Process

Since CAPPs are a generalization of CHEPs, only a little modification is necessary for the construction of the $ARMA(p,q)$ base process as described in Section 4.2.

Because the base process has still to result in a sequence Z_t with $Z_t \sim N(0,1)$ the variance of the white noise can be adjusted according to Equation 4.9.

The search algorithm used to determine the base process autocorrelations $\boldsymbol{\rho}$ for given CAPP autocorrelations $\hat{\boldsymbol{\rho}}$ has to compute the correlations according to Equation 6.9, but no further changes are necessary. Then, minimizing Equation 4.11 will again result in an $ARMA(p,q)$ model that approximates the autocorrelations $\boldsymbol{\rho}$.

6.4. Transformation of the APH distribution

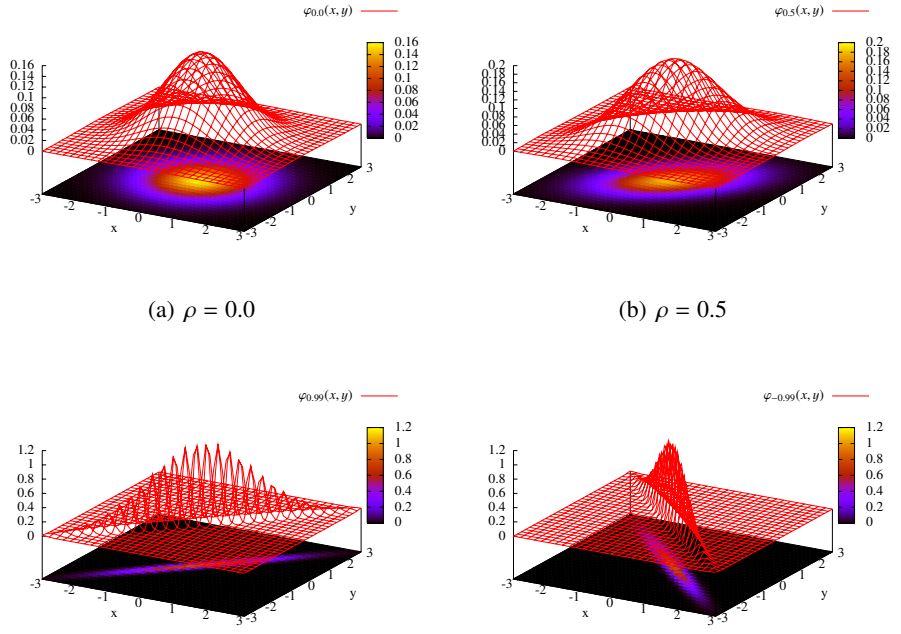
As already mentioned before in Section 2.3.1 the matrix representation of a PH distribution $(\boldsymbol{D}_0, \boldsymbol{\pi})$ is not unique. There are several different matrices and vectors that describe the same distribution. Depending on the purpose that the distribution is used for different representations might be preferable and equivalence transformations can be applied to transform one representation into another (see [43] for an overview of different transformations). For PH fitting one is interested in a distribution with the minimal number of parameters in $(\boldsymbol{D}_0, \boldsymbol{\pi})$. Section 2.3.1 already introduced canonical forms for this task and outlined a transformation for any acyclic PH distribution into canonical form [55]. On the other hand, PH distributions in canonical form are not really suitable for two-step MAP fitting algorithms that try to expand a given PH distribution into a MAP (cf. Section 2.3.6 for examples). These algorithms require a large number of entry and exit states to have more flexibility for finding a matrix $\boldsymbol{D}_1$. [42] and [84] present equivalence transformations to obtain such PH distributions. Finally, equivalence transformations can be applied to reduce the number of states of a PH distribution by removing unnecessary states [132].

For the processes presented in this work there are two cases that make necessary a transformation of the distribution. As already mentioned Hypo-Exponential distributions are the only subclass of acyclic PH distributions that are not covered by any of the presented process types. For these distributions the number of exit or entry states has to be increased such that they can be used as marginal distribution of a CAPP. Additionally a transformation might be necessary, if the maximal possible autocorrelation of the stochastic process is too small, i.e. the trace that should be fitted exhibits a higher autocorrelation than the maximal autocorrelation that can be modeled by an ARTA process, a CHEP or a CAPP for the given distribution. For (extended) ARTA processes this is not very likely, because the inverse transform approach results in a minimal and maximal possible autocorrelation that is close to the minimal and maximal theoretical values for a particular distribution. For CHEPs and CAPPs the correlated base process is only used to determine the elementary series, but the time until absorption from this series is drawn independently. The possible range for the autocorrelation can be computed using Equation 5.6 or Equation 6.9, respectively, and by setting the base process autocorrelations to -1 and 1. Hence, a CHEP or CAPP can in general not capture the full range of autocorrelations in the interval $[-1, 1]$. The experimental results in Chapter 8 suggest that the possible range for the autocorrelation is usually sufficient to model real traces, but in the rare cases where it is not sufficient a transformation of the PH distribution will help to increase the possible range for the autocorrelation.

The transformations proposed in the following will modify the transition rates between states and the initial probabilities of states but will preserve the distribution, i.e. only the representation of the distribution is modified. For Hyper-Erlang distributions a change in the transition rates will imply that the resulting representation is not Hyper-Erlang any longer, but an equivalent acyclic PH distribution. Thus, after the transformation the fitted process will always be a CAPP, no matter what the initial marginal distribution was.

6.4.1. The Boundary Cases for the Base Process Autocorrelation

Before several APH transformations are introduced that will allow us to increase the possible range of autocorrelation for a CAPP, some preceding considerations will help to prove that these transformations really perform as proposed. First of all note, that since the CAPP autocorrelation $\omega(\rho)$ is nondecreasing and continuous in the base process autocorrelation ρ (cf. Theorems 6.1 and 6.2), it is sufficient that the transformation increases the possible CAPP autocorrelation for the smallest and largest possible base process autocorrelation, i.e. for $\rho = \pm 1$. For these extreme cases we can derive an analytic expression for Equation 6.9 by exploiting known facts on the standard bivariate normal distribution and the integration bounds that occur for a CAPP. Figure 6.3



(a) $\rho = 0.0$ (b) $\rho = 0.5$

(c) $\rho = 0.99$ (d) $\rho = -0.99$

Figure 6.3.: Density function of the standard bivariate normal distribution with different correlations ρ

shows plots of the standard bivariate normal density function and the contours of equal density for different correlations ρ. It is obvious that all the (x, y) for which the density $\varphi_\rho(x, y)$ has the same value lie on ellipses (cf. e.g. [90]). The largest value for the density can of course be observed at the origin, while for increasing x and y values the density decreases. For $\rho = 0$ these ellipses become a circle (cf. Figure 6.3(a)) while for $\rho \to 1$ the ellipses almost reduce to a line (cf. Figure 6.3(c)), i.e. the density is zero except for $x \approx y$. For $\rho = 1$ the distribution becomes singular and the probability density function (c.f. Equation B.2) is no longer defined (though the distribution still is). In Figure 6.4 the bivariate normal distribution is split into four parts, i.e.

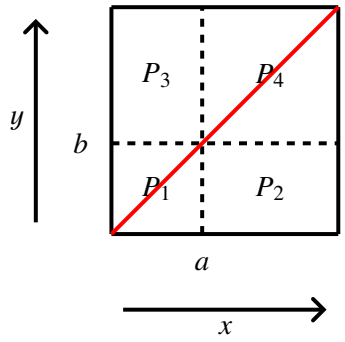


Figure 6.4.: The bivariate normal distribution for a correlation $\rho = 1$

$$\begin{aligned} P_1 &= P(x < a, y < b) = B(a, b, \rho) \\ P_2 &= P(x \geq a, y < b) \\ P_3 &= P(x < a, y \geq b) \\ P_4 &= P(x \geq a, y \geq b) = L(a, b, \rho). \end{aligned}$$

Of course, we have $P_1 + P_2 + P_3 + P_4 = 1$. Additionally Figure 6.4 shows the area for which $a = b$ as a red line. For $\rho = 1$ the terms $B(a, b, \rho)$ and $L(a, b, \rho)$ can be expressed using the standard normal distribution [141], i.e.

$$\begin{aligned} B(a, b, 1) &= \Phi(a) + \Phi(b) - \Phi(\max(a, b)) \\ L(a, b, 1) &= \Phi(-\max(a, b)). \end{aligned} \tag{6.14}$$

For $a = b$, obviously $P_2 = P_3 = 0$, since $P_1 + P_4 = B(a, a, 1) + L(a, a, 1) = \Phi(a) + \Phi(a) - \Phi(a) + \Phi(-a) = \Phi(a) + \Phi(a) - \Phi(a) + 1 - \Phi(a) = 1$. For the computation of the CAPP autocorrelation in Equation 6.9 this implies, that for $\rho = 1$ only those terms have to be considered for which $i = j$, i.e. those terms for which the two elementary series are identical. The probability for all other combinations with $i \neq j$ falls into the regions P_2 and P_3 and is zero. Hence, we obtain for Equation 6.9:

$$\begin{aligned} E[Y_t Y_{t+h}] &= \sum_i \left(\left(\sum_{s=1}^{S_i} \frac{1}{\boldsymbol{\Lambda_i}(s)} \right) \left(\sum_{s=1}^{S_i} \frac{1}{\boldsymbol{\Lambda_i}(s)} \right) \int_{\Phi^{-1}(\underline{b}_i)}^{\Phi^{-1}(\bar{b}_i)} \int_{\Phi^{-1}(\underline{b}_i)}^{\Phi^{-1}(\bar{b}_i)} \varphi_1(z_t, z_{t+h}) dz_t dz_{t+h} \right) \\ &= \sum_i \left(\left(\sum_{s=1}^{S_i} \frac{1}{\boldsymbol{\Lambda_i}(s)} \right) \left(\sum_{s=1}^{S_i} \frac{1}{\boldsymbol{\Lambda_i}(s)} \right) \right. \\ &\quad \left. B\left(\Phi^{-1}(\bar{b}_i), \Phi^{-1}(\bar{b}_i), 1\right) - B\left(\Phi^{-1}(\underline{b}_i), \Phi^{-1}(\underline{b}_i), 1\right) \right). \end{aligned}$$

Substitution with Equation 6.14 yields

$$\begin{aligned} E[Y_t Y_{t+h}] &= \sum_i \left(\left(\sum_{s=1}^{S_i} \frac{1}{\mathbf{\Lambda_i}(s)} \right) \left(\sum_{s=1}^{S_i} \frac{1}{\mathbf{\Lambda_i}(s)} \right) \right. \\ &\quad \left(\Phi(\Phi^{-1}(\bar{b}_i)) + \Phi(\Phi^{-1}(\bar{b}_i)) - \Phi(\Phi^{-1}(\bar{b}_i)) \right) \\ &\quad \left. - \left(\Phi(\Phi^{-1}(\underline{b}_i)) + \Phi(\Phi^{-1}(\underline{b}_i)) - \Phi(\Phi^{-1}(\underline{b}_i)) \right) \right) \\ &= \sum_i \left(\left(\sum_{s=1}^{S_i} \frac{1}{\mathbf{\Lambda_i}(s)} \right) \left(\sum_{s=1}^{S_i} \frac{1}{\mathbf{\Lambda_i}(s)} \right) \left(\bar{b}_i + \bar{b}_i - \bar{b}_i \right) - \left(\underline{b}_i + \underline{b}_i - \underline{b}_i \right) \right) \\ &= \sum_i \left(\left(\sum_{s=1}^{S_i} \frac{1}{\mathbf{\Lambda_i}(s)} \right)^2 \tau_i \right) \end{aligned} \tag{6.15}$$

where $\tau_i = \bar{b}_i - \underline{b}_i$ is the probability of the i-th elementary series as defined in Equation 6.2. Hence, for a base process autocorrelation $\rho = 1$ the CAPP autocorrelation can be computed using Equation 6.15 without any numerical approximation.

For $\rho = -1$ the ellipses also reduce to a line (cf. Figure 6.3(d)), but this time the density is zero except for $x = -y$. The expression for $B(a, b, -1)$ is more complicated this time, because two cases have to be distinguished. According to [141] we have

$$B(a, b, -1) = \begin{cases} \Phi(a) + \Phi(b) - 1, & a + b \geq 0 \\ 0, & a + b < 0. \end{cases} \tag{6.16}$$

For the term corresponding to the elementary series i and j in Equation 6.9 we are interested in computing the standard bivariate normal double integral for the bounds $[\underline{b}_i, \bar{b}_i], [\underline{b}_j, \bar{b}_j]$ and correlation $\rho = -1$, i.e.

$$\begin{aligned} & P\left[\Phi^{-1}(\underline{b}_i) < Z_1 < \Phi^{-1}(\bar{b}_i), \Phi^{-1}(\underline{b}_j) < Z_2 < \Phi^{-1}(\bar{b}_j) \right] \\ = \; & B\left(\Phi^{-1}(\bar{b}_i), \Phi^{-1}(\bar{b}_j), -1 \right) \\ & -B\left(\Phi^{-1}(\bar{b}_i), \Phi^{-1}(\underline{b}_j), -1 \right) \\ & -B\left(\Phi^{-1}(\underline{b}_i), \Phi^{-1}(\bar{b}_j), -1 \right) \\ & +B\left(\Phi^{-1}(\underline{b}_i), \Phi^{-1}(\underline{b}_j), -1 \right). \end{aligned} \tag{6.17}$$

Since in our case the parameters of $B(a, b, -1)$ result from the inverse standard normal cdf we may rewrite Equation 6.16 as

$$B(\Phi^{-1}(b_i), \Phi^{-1}(b_j), -1) = \begin{cases} b_i + b_j - 1, & b_i + b_j \geq 1 \\ 0, & b_i + b_j < 1. \end{cases} \tag{6.18}$$

Depending on the values of $\underline{b}_i, \bar{b}_i, \underline{b}_j$ and $\bar{b}_j$ we can derive the following simplified expressions for Equation 6.17 by substituting with Equation 6.18:

$$\begin{aligned} & \bar{b}_i + \bar{b}_j < 1 : \\ & \quad P\left[\Phi^{-1}(\underline{b}_i) < Z_1 < \Phi^{-1}(\bar{b}_i), \Phi^{-1}(\underline{b}_j) < Z_2 < \Phi^{-1}(\bar{b}_j) \right] \\ & \quad = 0 \end{aligned}$$

$$
\begin{aligned}
&\underline{b}_i + \underline{b}_j \geq 1: \\
&\quad P\left[\Phi^{-1}(\underline{b}_i) < Z_1 < \Phi^{-1}(\bar{b}_i), \Phi^{-1}(\underline{b}_j) < Z_2 < \Phi^{-1}(\bar{b}_j)\right] \\
&\quad = \left(\bar{b}_i + \bar{b}_j - 1\right) - \left(\bar{b}_i + \underline{b}_j - 1\right) - \left(\underline{b}_i + \bar{b}_j - 1\right) + \left(\underline{b}_i + \underline{b}_j - 1\right) \\
&\quad = 0 \\
&\underline{b}_i + \underline{b}_j < 1, \bar{b}_i + \bar{b}_j \geq 1, \underline{b}_i + \bar{b}_j < 1, \bar{b}_i + \underline{b}_j < 1: \\
&\quad P\left[\Phi^{-1}(\underline{b}_i) < Z_1 < \Phi^{-1}(\bar{b}_i), \Phi^{-1}(\underline{b}_j) < Z_2 < \Phi^{-1}(\bar{b}_j)\right] \\
&\quad = \bar{b}_i + \bar{b}_j - 1 \\
&\underline{b}_i + \underline{b}_j < 1, \bar{b}_i + \bar{b}_j \geq 1, \underline{b}_i + \bar{b}_j \geq 1, \bar{b}_i + \underline{b}_j < 1: \\
&\quad P\left[\Phi^{-1}(\underline{b}_i) < Z_1 < \Phi^{-1}(\bar{b}_i), \Phi^{-1}(\underline{b}_j) < Z_2 < \Phi^{-1}(\bar{b}_j)\right] \\
&\quad = \left(\bar{b}_i + \bar{b}_j - 1\right) - \left(\underline{b}_i + \bar{b}_j - 1\right) \\
&\quad = \bar{b}_i - \underline{b}_i = \tau_i \\
&\underline{b}_i + \underline{b}_j < 1, \bar{b}_i + \bar{b}_j \geq 1, \underline{b}_i + \bar{b}_j < 1, \bar{b}_i + \underline{b}_j \geq 1: \\
&\quad P\left[\Phi^{-1}(\underline{b}_i) < Z_1 < \Phi^{-1}(\bar{b}_i), \Phi^{-1}(\underline{b}_j) < Z_2 < \Phi^{-1}(\bar{b}_j)\right] \\
&\quad = \left(\bar{b}_i + \bar{b}_j - 1\right) - \left(\bar{b}_i + \underline{b}_j - 1\right) \\
&\quad = \bar{b}_j - \underline{b}_j = \tau_j \\
&\underline{b}_i + \underline{b}_j < 1, \bar{b}_i + \bar{b}_j \geq 1, \underline{b}_i + \bar{b}_j \geq 1, \bar{b}_i + \underline{b}_j \geq 1: \\
&\quad P\left[\Phi^{-1}(\underline{b}_i) < Z_1 < \Phi^{-1}(\bar{b}_i), \Phi^{-1}(\underline{b}_j) < Z_2 < \Phi^{-1}(\bar{b}_j)\right] \\
&\quad = \left(\bar{b}_i + \bar{b}_j - 1\right) - \left(\underline{b}_i + \bar{b}_j - 1\right) - \left(\bar{b}_i + \underline{b}_j - 1\right) \\
&\quad = 1 - \underline{b}_i - \underline{b}_j.
\end{aligned}
$$

Hence, for a base process autocorrelation of $\rho = -1$ Equation 6.9 becomes

$$
E[Y_t Y_{t+h}] = \sum_{i,j} \left(\left(\sum_{s=1}^{S_i} \frac{1}{\boldsymbol{\Lambda_i}(s)} \right) \left(\sum_{s=1}^{S_j} \frac{1}{\boldsymbol{\Lambda_j}(s)} \right) P_\varphi(i,j) \right) \qquad \text{where} \tag{6.19}
$$

$$
P_\varphi(i,j) = \begin{cases}
0, & \bar{b}_i + \bar{b}_j < 1 \\
0, & \underline{b}_i + \underline{b}_j \geq 1 \\
\bar{b}_i + \bar{b}_j - 1, & \underline{b}_i + \underline{b}_j < 1, \bar{b}_i + \bar{b}_j \geq 1, \underline{b}_i + \bar{b}_j < 1, \bar{b}_i + \underline{b}_j < 1 \\
\tau_i, & \underline{b}_i + \underline{b}_j < 1, \bar{b}_i + \bar{b}_j \geq 1, \underline{b}_i + \bar{b}_j \geq 1, \bar{b}_i + \underline{b}_j < 1 \\
\tau_j, & \underline{b}_i + \underline{b}_j < 1, \bar{b}_i + \bar{b}_j \geq 1, \underline{b}_i + \bar{b}_j < 1, \bar{b}_i + \underline{b}_j \geq 1 \\
1 - \underline{b}_i - \underline{b}_j, & \underline{b}_i + \underline{b}_j < 1, \bar{b}_i + \bar{b}_j \geq 1, \underline{b}_i + \bar{b}_j \geq 1, \bar{b}_i + \underline{b}_j \geq 1.
\end{cases}
$$

Figure 6.5 shows the standard bivariate normal distribution with correlation $\rho = -1$ and examples for the different cases listed in Equation 6.19, i.e.

$$
\begin{aligned}
&\text{a)} && \bar{b}_i + \bar{b}_j < 1 \\
&\quad \Leftrightarrow && \Phi^{-1}(\bar{b}_i) + \Phi^{-1}(\bar{b}_j) < 0 \\
&\text{b)} && \underline{b}_i + \underline{b}_j \geq 1 \\
&\quad \Leftrightarrow && \Phi^{-1}(\underline{b}_i) + \Phi^{-1}(\underline{b}_j) \geq 0 \\
&\text{c)} && \underline{b}_i + \underline{b}_j < 1, \quad \bar{b}_i + \bar{b}_j \geq 1, \quad \underline{b}_i + \bar{b}_j < 1, \quad \bar{b}_i + \underline{b}_j < 1
\end{aligned}
$$

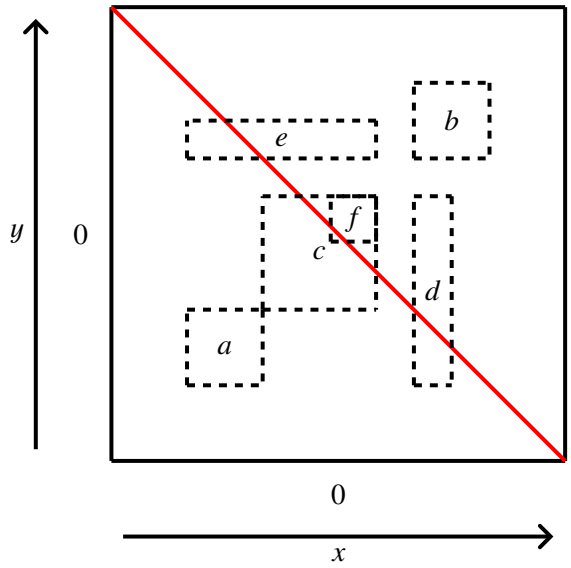


Figure 6.5.: The bivariate normal distribution for a correlation $\rho = -1$

$$
\begin{array}{lll}
 & \Leftrightarrow & \Phi^{-1}(\underline{b}_i) + \Phi^{-1}(\underline{b}_j) < 0, \quad \Phi^{-1}(\bar{b}_i) + \Phi^{-1}(\bar{b}_j) \geq 0, \\
 & & \Phi^{-1}(\underline{b}_i) + \Phi^{-1}(\bar{b}_j) < 0, \quad \Phi^{-1}(\bar{b}_i) + \Phi^{-1}(\underline{b}_j) < 0 \\
\text{d)} & & \underline{b}_i + \underline{b}_j < 1, \quad \bar{b}_i + \bar{b}_j \geq 1, \quad \underline{b}_i + \bar{b}_j \geq 1, \quad \bar{b}_i + \underline{b}_j < 1 \\
 & \Leftrightarrow & \Phi^{-1}(\underline{b}_i) + \Phi^{-1}(\underline{b}_j) < 0, \quad \Phi^{-1}(\bar{b}_i) + \Phi^{-1}(\bar{b}_j) \geq 0, \\
 & & \Phi^{-1}(\underline{b}_i) + \Phi^{-1}(\bar{b}_j) \geq 0, \quad \Phi^{-1}(\bar{b}_i) + \Phi^{-1}(\underline{b}_j) < 0 \\
\text{e)} & & \underline{b}_i + \underline{b}_j < 1, \quad \bar{b}_i + \bar{b}_j \geq 1, \quad \underline{b}_i + \bar{b}_j < 1, \quad \bar{b}_i + \underline{b}_j \geq 1 \\
 & \Leftrightarrow & \Phi^{-1}(\underline{b}_i) + \Phi^{-1}(\underline{b}_j) < 0, \quad \Phi^{-1}(\bar{b}_i) + \Phi^{-1}(\bar{b}_j) \geq 0, \\
 & & \Phi^{-1}(\underline{b}_i) + \Phi^{-1}(\bar{b}_j) < 0, \quad \Phi^{-1}(\bar{b}_i) + \Phi^{-1}(\underline{b}_j) \geq 0 \\
\text{f)} & & \underline{b}_i + \underline{b}_j < 1, \quad \bar{b}_i + \bar{b}_j \geq 1, \quad \underline{b}_i + \bar{b}_j \geq 1, \quad \bar{b}_i + \underline{b}_j \geq 1 \\
 & \Leftrightarrow & \Phi^{-1}(\underline{b}_i) + \Phi^{-1}(\underline{b}_j) < 0, \quad \Phi^{-1}(\bar{b}_i) + \Phi^{-1}(\bar{b}_j) \geq 0, \\
 & & \Phi^{-1}(\underline{b}_i) + \Phi^{-1}(\bar{b}_j) \geq 0, \quad \Phi^{-1}(\bar{b}_i) + \Phi^{-1}(\underline{b}_j) \geq 0.
\end{array}
$$

6.4.2. Transformation of APH Distributions that are not in Canonical Form

In the following two transformations for the APH marginal distribution of a CAPP are introduced. The first transformation can be applied to distributions that are not in canonical form and preserves the number of states. The second transformation adds an additional state to the APH distribution and requires the distribution to be in canonical form. The basic idea behind both transformations are the equivalent representations of the exponential distribution as shown in Figure 2.16, which allows one to split an elementary series with rate λ into two series, one with rate $\mu \geq \lambda$ and one with rates μ and λ. The following Lemma will be useful to prove that these transformations actually increase the possible range of autocorrelation.

Lemma 6.1. *Let $\lambda_{j_m} \geq \lambda_{j_{m-1}} \geq \cdots \geq \lambda_{j_1}$ be the rates of an elementary series of an*

APH distribution with probability $\tau_j > 0$. Furthermore, let $\lambda_x \geq (>)\lambda_{j_{m-k}}$. Then, for the two elementary series

$$\lambda_{j_m} \geq \cdots \geq \lambda_{j_{m-k+1}} \geq \lambda_x \geq \lambda_{j_{m-k}} \geq \cdots \geq \lambda_{j_1}$$

with probability $\tau_j(1 - \lambda_{j_{m-k}}/\lambda_x)$ and

$$\lambda_{j_m} \geq \cdots \geq \lambda_{j_{m-k+1}} \geq \lambda_x \geq \lambda_{j_{m-k-1}} \geq \cdots \geq \lambda_{j_1}$$

with probability $\tau_j\lambda_{j_{m-k}}/\lambda_x$ the following condition holds:

$$\begin{aligned} &\left(\frac{1}{\lambda_{j_m}} + \cdots \frac{1}{\lambda_{j_{m-k+1}}} + \frac{1}{\lambda_x} + \frac{1}{\lambda_{j_{m-k}}} + \cdots + \frac{1}{\lambda_{j_1}}\right)^2 \tau_j \left(1 - \frac{\lambda_{j_{m-k}}}{\lambda_x}\right) \\ &+ \left(\left(\frac{1}{\lambda_{j_m}} + \cdots \frac{1}{\lambda_{j_{m-k+1}}} + \frac{1}{\lambda_x} + \frac{1}{\lambda_{j_{m-k-1}}} + \cdots + \frac{1}{\lambda_{j_1}}\right)^2 \tau_j \frac{\lambda_{j_{m-k}}}{\lambda_x}\right) \\ \geq (>) \quad &\left(\left(\frac{1}{\lambda_{j_m}} + \cdots \frac{1}{\lambda_{j_{m-k+1}}} + \frac{1}{\lambda_{j_{m-k}}} + \cdots + \frac{1}{\lambda_{j_1}}\right)^2 \tau_j\right). \end{aligned} \tag{6.20}$$

Proof. The inequality can be shown by applying the multinomial theorem to all three terms and by further splitting the resulting expressions, which requires some lengthy computations that are omitted here but can be found in Appendix C.1. Finally, one obtains

$$\left(\left(\frac{1}{\lambda_{j_m}} + \cdots \frac{1}{\lambda_{j_{m-k+1}}} + \frac{1}{\lambda_{j_{m-k}}} + \cdots + \frac{1}{\lambda_{j_1}}\right)^2 \tau_j\right) + \frac{\tau_j}{\lambda_x}\left(\frac{1}{\lambda_{j_{m-k}}} - \frac{1}{\lambda_x}\right) \tag{6.21}$$

for the left-hand side of Equation 6.20. Observe, that the first term is equal to the right-hand side of Equation 6.20. The second term is zero for $\lambda_x = \lambda_{j_{m-k}}$ and greater than zero for $\lambda_x > \lambda_{j_{m-k}}$. □

Now, let F_{aph} be an acyclic PH distributions whose representation is not in canonical form (cf. Figure 2.15). [55] showed that every APH distribution can be transformed into series CF by exploiting the equivalent representations of the exponential distribution (cf. Figure 2.16). This transformation has already been briefly introduced in Section 2.3.4. Let $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$ be the rates of the n states of F_{aph}. [55] defined

$$\begin{array}{llll} \lambda_n, & \lambda_{n-1}, & \ldots & \lambda_2, \quad \lambda_1 \\ \lambda_n, & \lambda_{n-1}, & \ldots & \lambda_2 \\ \vdots & & & \\ \lambda_n, & \lambda_{n-1} & & \\ \lambda_n & & & \end{array}$$

to be the basic series of the APH distribution. Note, that the basic series are actually the elementary series of an APH distribution in series canonical form. Hence, the goal of the transformation is to modify each elementary series of F_{aph} such that it resembles one of the basic series. Let $\lambda_{j_m}, \lambda_{j_{m-1}}, \ldots, \lambda_{j_1}$ be the j-th elementary series of F_{aph} and assume that $\lambda_{j_k} \leq \lambda_{j_{k+1}}$. Starting with λ_{j_m} the elementary series is compared with the

basic series. Let $\lambda_{j_{m-k}}$ be the first rate that is different, i.e. $\lambda_{j_{m-k}} \neq \lambda_{n-k}$. Then the elementary series is replaced by two new series, one of the form

$$\lambda_{j_m}, \ldots, \lambda_{j_{m-k+1}}, \lambda_{n-k}, \lambda_{j_{m-k}}, \ldots \lambda_{j_1}$$

and one of the form

$$\lambda_{j_m}, \ldots, \lambda_{j_{m-k+1}}, \lambda_{n-k}, \lambda_{j_{m-k-1}}, \ldots \lambda_{j_1}$$

exploiting the herein before mentioned equivalent representations of the exponential distribution. Additionally the probabilities of the two new elementary series are adjusted as shown in Figure 2.16. Repeated application of this step transforms all elementary series in basic series [55]. Finally, the APH in series CF can be constructed from the transformed elementary series. The distribution consists of the n states as shown in Figure 2.15(a). The initial probabilities $\boldsymbol{\pi}(i)$ can be computed from the transformed elementary series, i.e. $\boldsymbol{\pi}(i)$ is the sum of the probabilities of all transformed elementary series of the form $\lambda_n, \ldots, \lambda_i$.

Transformation 1. *Let F_{aph} be an acyclic PH distribution that is not in canonical form. Then, the transformation converts F_{aph} into series canonical form as described above and proposed in [55]. The transformed distribution is denoted by F'_{aph}.*

Proposition 6.4. *Transformation 1 preserves the distribution of F_{aph}, i.e. F_{aph} and F'_{aph} describe the same distribution.*

Proof. The proof is given in [55]. □

Proposition 6.5. *Transformation 1 increases the possible range of positive autocorrelation for a CAPP with APH marginal distribution F_{aph} under the mild assumption that F_{aph} has at least one elementary series that does not resemble a basic series.*

Proof. Because Transformation 1 consists of a sequence of steps where each step splits one elementary series into two equivalent series, we will show that at least one of the single steps increases the possible range of autocorrelation for the CAPP. It is sufficient to prove that this range is increased for the maximal value of the base process autocorrelation $\rho = 1$, which implies that only the terms with $i = j$ contribute to $E[Y_t Y_{t+h}]$ and Equation 6.15 can be used for the computation. Assume, that for the current step elementary series j with rates $\lambda_{j_m} \geq \lambda_{j_{m-1}} \geq \cdots \geq \lambda_{j_1}$ and probability τ_j is considered that does not resemble a basic series. Recall, that we assumed the existence of at least one such elementary series. Furthermore assume, that $\lambda_{j_{m-k}}$ is the rate of the exponential distribution that is replaced by its equivalent representation. As described above, this series is then replaced by two elementary series

$$\lambda_{j_m} \geq \cdots \geq \lambda_{j_{m-k+1}} \geq \lambda_x \geq \lambda_{j_{m-k}} \geq \cdots \geq \lambda_{j_1}$$

with probability $\tau_j(1 - \lambda_{j_{m-k}}/\lambda_x)$ and

$$\lambda_{j_m} \geq \cdots \geq \lambda_{j_{m-k+1}} \geq \lambda_x \geq \lambda_{j_{m-k-1}} \geq \cdots \geq \lambda_{j_1}$$

with probability $\tau_j \lambda_{j_{m-k}}/\lambda_x$ and $\lambda_x \geq \lambda_{j_{m-k}}$. Since in this step all other elementary series remain unchanged, it is sufficient to show that the two new elementary series

contribute more to $E[Y_t Y_{t+h}]$ than the old untransformed series did, i.e.

$$\begin{aligned}
&\left(\left(\frac{1}{\lambda_{j_m}} + \cdots \frac{1}{\lambda_{j_{m-k+1}}} + \frac{1}{\lambda_x} + \frac{1}{\lambda_{j_{m-k}}} + \cdots + \frac{1}{\lambda_{j_1}}\right)^2 \tau_j \left(1 - \frac{\lambda_{j_{m-k}}}{\lambda_x}\right)\right) \\
&+\left(\left(\frac{1}{\lambda_{j_m}} + \cdots \frac{1}{\lambda_{j_{m-k+1}}} + \frac{1}{\lambda_x} + \frac{1}{\lambda_{j_{m-k-1}}} + \cdots + \frac{1}{\lambda_{j_1}}\right)^2 \tau_j \frac{\lambda_{j_{m-k}}}{\lambda_x}\right) \\
> &\left(\left(\frac{1}{\lambda_{j_m}} + \cdots \frac{1}{\lambda_{j_{m-k+1}}} + \frac{1}{\lambda_{j_{m-k}}} + \cdots + \frac{1}{\lambda_{j_1}}\right)^2 \tau_j\right).
\end{aligned}$$

But this follows from Lemma 6.1, because the elementary series j was not a basic series before the transformation step and therefore two rates $\lambda_{i_1} < \lambda_{i_2}$ exist, where λ_{i_1} is part of series j and λ_{i_2} is the rate that is missing at position i_1 for j to become a basic series. Then, in one transformation step these two rates are treated and we have $\lambda_{i_2} = \lambda_x > \lambda_{j_{m-k}} = \lambda_{i_1}$. □

Remark. For Proposition 6.5 it was required that at least one elementary series does not resemble a basic series, such that at least one transformation step has to be performed. Recall from Figure 2.15 that there are three equivalent canonical forms for an APH distribution. Observe, that all three canonical forms exhibit the same elementary series, i.e. the series have the form $\lambda_n, \lambda_{n-1}, \ldots, \lambda_i$ in all cases. Thus, if an APH distribution F_{aph} is available in canonical form A or B it can be transformed into series canonical form without applying a transformation step, i.e. by just computing the initial probabilities of the series canonical form from the probabilities of the elementary series of F_{aph}. Then the possible range of autocorrelation is of course not modified by Transformation 1. In these cases, Transformation 1 can be applied safely anyway to prepare F_{aph} for the transformation in the following section, which expects an APH distribution in series canonical form.

According to Proposition 6.5 Transformation 1 increases the maximal positive autocorrelation that can be achieved for a given APH distribution by changing its representation. For the negative autocorrelation a similar statement is much more difficult to prove. In contrast to the expression for the computation of the positive autocorrelation from Equation 6.15 the computation of the negative autocorrelation requires to distinguish between several cases depending on the interval bounds of the branches (cf. Equation 6.19). However, for APH distributions with a special structure it can be shown, that the transformations from this chapter increase the possible range of negative autocorrelation. The proof and some considerations for the general case are given in Section C.2. Note, that real network traces usually only exhibit positive autocorrelation (cf. Chapter 8) and thus, the more important case is completely covered by the proofs in this section.

6.4.3. Transformation of APH Distributions in Series Canonical Form

Since the transformation into series canonical form increases (or at least preserves) the possible range of positive autocorrelation for all APH distributions, this is the best representation we can achieve for a given number of states of the APH distribution.

If this representation is still not sufficient to model some estimated autocorrelation coefficient from a trace, it is necessary to increase the model size, i.e. to introduce an additional state.

Transformation 2. *Let F_{aph} be an acyclic PH distribution in series canonical form with n states, initial probabilities $\boldsymbol{\pi}(1), \ldots, \boldsymbol{\pi}(n)$ and bidiagonal matrix $\boldsymbol{D}_0$ with rates $\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$. Then, the transformation constructs the distribution F'_{aph} by adding a new state $n+1$ with rate $\mu > \lambda_n$. The initial probabilities of F'_{aph} are set to $\boldsymbol{\pi}'(i) = \boldsymbol{\pi}(i), i = 1, \ldots n-1, \boldsymbol{\pi}'(n) = \boldsymbol{\pi}(n)(1-\lambda_n/\mu)$ and $\boldsymbol{\pi}'(n+1) = \boldsymbol{\pi}(n)\lambda_n/\mu$. Matrix $\boldsymbol{D}'_0$ is set to*

$$\boldsymbol{D}'_0 = \begin{bmatrix} -\lambda_1 & \lambda_1 & 0 & \cdots & 0 & 0 & 0 \\ 0 & -\lambda_2 & \lambda_2 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & -\lambda_{n-1} & \lambda_{n-1}(1-\lambda_n/\mu) & \lambda_{n-1}\lambda_n/\mu \\ 0 & 0 & 0 & \cdots & 0 & -\lambda_n & \lambda_n \\ 0 & 0 & 0 & \cdots & 0 & 0 & -\mu \end{bmatrix}.$$

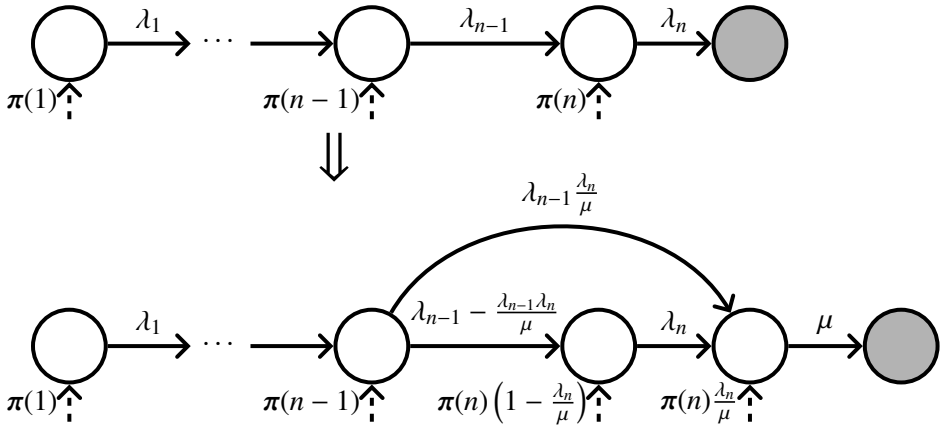


Figure 6.6.: Transformation 2

Figure 6.6 visualizes Transformation 2. The transformation splits the initial probability of state n and the transition from state $n-1$ to n to introduce an initial probability for the new state $n+1$ and a transition from $n-1$ to this new state.

A special case arises for the exponential distribution, which only has a single state with initial probability 1. In this case only the initial probability has to be split for introducing the new state as shown in Figure 6.7.

Proposition 6.6. *Transformation 2 preserves the distribution of F_{aph}, i.e. F_{aph} and F'_{aph} describe the same distribution.*

Proof. To prove the proposition it will be shown that F'_{aph} results from F_{aph} if λ_n is replaced by its equivalent representation with rates λ_n and $\mu > \lambda_n$ according to

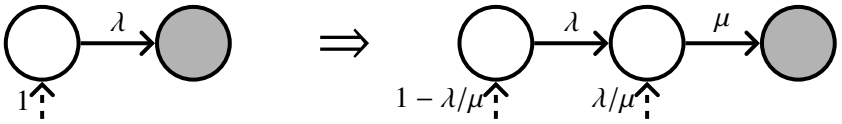


Figure 6.7.: Transformation 2 for the exponential distribution

Figure 2.16. F_{aph} has n elementary series

$$
\begin{array}{lll}
n: & \lambda_n & \text{with probability } \boldsymbol{\pi}(n) \\
n-1: & \lambda_n,\ \lambda_{n-1} & \text{with probability } \boldsymbol{\pi}(n-1) \\
 & \vdots & \\
i: & \lambda_n,\ \lambda_{n-1},\ \ldots,\ \lambda_i & \text{with probability } \boldsymbol{\pi}(i) \\
 & \vdots & \\
1: & \lambda_n,\ \lambda_{n-1},\ \ldots,\ \lambda_i,\ \ldots,\ \lambda_1 & \text{with probability } \boldsymbol{\pi}(1)
\end{array}
$$

Introducing $\mu > \lambda_n$ series i is split into two series i_1, i_2 and we obtain

$$
\begin{array}{lll}
n_1: & \mu,\ \lambda_n & \text{with probability } \boldsymbol{\pi}(n)(1-\lambda_n/\mu) \\
n_2: & \mu & \text{with probability } \boldsymbol{\pi}(n)\lambda_n/\mu \\
(n-1)_1: & \mu,\ \lambda_n,\ \lambda_{n-1} & \text{with probability } \boldsymbol{\pi}(n-1)(1-\lambda_n/\mu) \\
(n-1)_2: & \mu,\ \lambda_{n-1} & \text{with probability } \boldsymbol{\pi}(n-1)\lambda_n/\mu \\
 & \vdots & \\
i_1: & \mu,\ \lambda_n,\ \lambda_{n-1},\ \ldots,\ \lambda_i & \text{with probability } \boldsymbol{\pi}(i)(1-\lambda_n/\mu) \\
i_2: & \mu,\ \lambda_{n-1},\ \ldots,\ \lambda_i & \text{with probability } \boldsymbol{\pi}(i)\lambda_n/\mu \\
 & \vdots &
\end{array}
$$

Observe, that these series are identical with the elementary series of F'_{aph}, because for each state i there is one path taking all the phases from λ_i to μ and one path where λ_n is omitted, i.e. the transition from λ_{n-1} to μ is taken. Using Equation 6.1 it is easy to verify that the initial probabilities of the series are identical as well. □

Proposition 6.7. *Transformation 2 increases the possible range of positive autocorrelation for a CAPP with APH marginal distribution F_{aph} in series canonical form.*

Proof. The proof works similar to the one of Proposition 6.5. In the proof for Proposition 6.6 it was shown that Transformation 2 splits each elementary series into two equivalent series. In the following it is shown that the two new elementary series contribute more to $E[Y_t Y_{t+h}]$ (and thus, to the autocorrelation) than the old untransformed series did. Again, it is sufficient to consider the maximal value of the base process autocorrelation $\rho = 1$ only and use Equation 6.15 for the computation of $E[Y_t Y_{t+h}]$. Transformation 2 replaces each elementary series $\lambda_n, \lambda_{n-1}, \ldots, \lambda_i$ with probability $\boldsymbol{\pi}(i)$ by the two series $\mu, \lambda_n, \lambda_{n-1}, \ldots, \lambda_i$ with probability $\boldsymbol{\pi}(i)(1-\lambda_n/\mu)$ and $\mu, \lambda_{n-1}, \ldots, \lambda_i$

with probability $\boldsymbol{\pi}(i)\lambda_n/\mu$ and $\mu > \lambda_n$. It follows immediately from Lemma 6.1 that

$$\begin{aligned} & \left(\left(\frac{1}{\mu} + \frac{1}{\lambda_n} + \frac{1}{\lambda_{n-1}} + \cdots + \frac{1}{\lambda_i}\right)^2 \boldsymbol{\pi}(i)\left(1 - \frac{\lambda_n}{\mu}\right)\right) \\ + & \left(\left(\frac{1}{\mu} + \frac{1}{\lambda_{n-1}} + \cdots + \frac{1}{\lambda_i}\right)^2 \boldsymbol{\pi}(i)\frac{\lambda_n}{\mu}\right) \\ > & \left(\left(\frac{1}{\lambda_n} + \frac{1}{\lambda_{n-1}} + \cdots + \frac{1}{\lambda_i}\right)^2 \boldsymbol{\pi}(i)\right), \end{aligned}$$

which proves the proposition. □

Remark. Note, that after applying Transformation 2 the distribution is in general no longer in series canonical form. Transformation 1 could be used afterwards to obtain a distribution in series canonical form (and further increase the possible range of autocorrelation).

Transformation 2 increases the possible range of positive autocorrelation for every $\mu > \lambda_n$. However, recall from Equation 6.21 that the difference between an untransformed elementary series j and the two equivalent series that result from the transformation is given by

$$\frac{\tau_j}{\mu}\left(\frac{1}{\lambda_n} - \frac{1}{\mu}\right).$$

It is easy to verify, that this expression is maximized for $\mu = 2\lambda_n$, which is the best choice for μ.

Remark. For Transformation 2 the new state with rate μ was added as last state of the APH distribution, i.e. we required $\mu > \lambda_n$. Note, that of course the new state could be added at any other position for some $\lambda_i < \mu < \lambda_{i+1}$. However, at position i it might not always be possible to set $\mu = 2\lambda_i$, because $\mu < \lambda_{i+1}$ might be violated in this case.

Again, for the treatment of negative autocorrelations the reader is referred to Section C.2.

The two presented transformations cover all subclasses of acyclic PH distributions. For exponential, Erlang and Hypo-Exponential distributions, which only have a single elementary series, Transformation 2 can be applied to make them usable for a CAPP. In general, one could iterate between the two transformations to increase the possible range of autocorrelation (of course at the cost of an increased model size), i.e. apply Transformation 1 to bring the distribution into canonical form (if necessary), introduce a new state with Transformation 2, bring the distribution into canonical form again with Transformation 1 and so on. Appendix D shows the effect of the presented transformations on several different APH distributions.

Chapter 7

An Algorithmic Approach

In the previous chapters several stochastic processes have been presented that can combine various types of marginal distributions with an $ARMA(p,q)$ base process to model correlated traffic data. In the following the concepts of Chapters 4, 5 and 6 will be integrated into an algorithmic framework. For all types of processes distribution and correlation fitting can be performed separately in two steps. Thus, for the first part of this chapter several fitting approaches for the supported distributions are presented. The second part outlines algorithms for the construction of the base process.

7.1. Fitting the Marginal Distribution

Since distribution and autocorrelation fitting are separated for the stochastic processes from Chapters 4, 5 and 6, the marginal distribution may be fitted with any of the available tools that can provide a distribution that is suitable for the corresponding process.

For (extended) ARTA models every distribution for which the inverse cdf can be computed is applicable. Hence, tools like `Expertfit` [105], which can fit various different distributions to given data, can be used for determining F_Y, though those tools may require user interaction for the selection of the distribution. If a fully automated approach is preferred one could decide on one flexible type of distribution that can take a wide set of different shapes and for which fitting algorithms exist like the Johnson distribution [88] that was also used for the `ARTAFIT` approach in [25, 26]. For example the software `FITTR1` [56] provides fitting algorithms for the Johnson distribution. For other distributions like exponential, normal or lognormal simple maximum likelihood estimators exist [104] and these distributions can be fitted to a trace easily.

CHEPs have been tailored to Hyper-Erlang distributions and hence the tool `GFIT`, which implements the algorithm from [151] (cf. Section 5.3), is recommended for fitting the marginal distribution. Alternatively, the approaches from [64], [95] or [137] (cf. Section 2.3.5), which all fit an Hyper-Exponential distribution (which is a special case of the HErD with only one phase per branch), could be applied, but the Hyper-Exponential distribution is of course not as flexible as the HErD.

For CAPPs any of the fitting approaches from Section 2.3.5 that yields an acyclic PH distribution could be used. Since EM algorithms are much slower for general APH

distributions than for special subclasses like HErDs, the much faster moment matching algorithm from [42] is recommended.

7.2. Fitting the Base Process

Since the construction of the ARMA base process is similar for the three stochastic processes from Chapters 4, 5 and 6 only a single routine for fitting the base process is presented in the following, which will be divided into alternative parts whenever necessary.

Algorithm 7.1: Main routine

input : marginal distribution F_Y
Trace $\mathcal{T} = (t_1, t_2, \ldots, t_l)$
r: number of autocorrelation lags to match
p: number of AR coefficients
q: number of MA coefficients
output: ARTA, CHEP or CAPP (depending on F_Y)

1 compute desired autocorrelations $\hat{\boldsymbol{\rho}} = (\hat{\rho}_1, \hat{\rho}_2, \ldots, \hat{\rho}_r)$ from $\mathcal{T}$ according to Equation 1.4
`//depending on` F_Y `call one of the subroutines`
2 **if** F_Y^{-1} *can be computed* **then**
3 call Algorithm 7.2 for (extended) ARTA processes
4 **if** F_Y *is an APH* **then**
5 call Algorithm 7.3 for CHEPs and CAPPs
6 **return** *fitted process*

The main routine from Algorithm 7.1 expects a marginal distribution F_Y and a trace $\mathcal{T}$ as input. Furthermore, the order (p, q) of the base process and the number of autocorrelation lags r to consider for fitting have to be specified. The main routine first computes the first r lags from $\mathcal{T}$. After that depending on the type of F_Y a subroutine is called. If F_Y^{-1} can be computed, an (extended) ARTA model is fitted using Algorithm 7.2, for acyclic PH distributions (including Hyper-Erlang distributions) a CHEP or CAPP is fitted using Algorithm 7.3. Note, that for exponential and Erlang distributions actually a call to both subroutines is possible. However, for exponential distributions it is recommended to fit an (extended) ARTA model, since a closed-form expression for the inverse cdf exists. Erlang distributions can be used for CAPP fitting after a transformation step of the distribution, which is recommended here to avoid the numerical computation of the inverse cdf, which is necessary for those distributions when used as a marginal distribution for ARTA processes.

Algorithm 7.2 describes the subroutine for fitting (extended) ARTA models that implements the ideas from Chapter 4. Inputs are the marginal distribution F_Y (for which F_Y^{-1} has to be computable), a vector $\hat{\boldsymbol{\rho}}$ with the autocorrelation coefficients estimated from the trace and that the resulting ARTA process should have, and the order of the base process. In the first step the subroutine from Algorithm 7.4 is called to determine the base process autocorrelations $\boldsymbol{\rho}$ that yield the desired ARTA autocorrelations $\hat{\boldsymbol{\rho}}$. In

Algorithm 7.2: Routine for fitting extended ARTA processes

input : marginal distribution F_Y
$\hat{\boldsymbol{\rho}} = (\hat{\rho}_1, \hat{\rho}_2, \ldots, \hat{\rho}_r)$: desired autocorrelation for ARTA process
p: number of AR coefficients
q: number of MA coefficients
output: ARTA process

1 compute $ARMA(p,q)$ autocorrelations $\boldsymbol{\rho} = (\rho_1, \rho_2, \ldots, \rho_r)$ by calling Algorithm 7.4
2 minimize Equation 4.11 to find an $ARMA(p,q)$ model which approximates $\boldsymbol{\rho}$
3 set variance of $ARMA(p,q)$ according to Equation 4.9
4 **return** *ARTA model with base* $ARMA(p,q)$ *process and distribution* F_Y

a second step Equation 4.11 is minimized to find a stationary $ARMA(p,q)$ model that approximates $\boldsymbol{\rho}$. This is done applying the general purpose optimization algorithm of Nelder and Mead [118] using an implementation from [117]. Finally, the variance of the $ARMA(p,q)$ base process is modified according to Equation 4.9, such that the Z_t generated by the base process have standard normal distribution, and the resulting (extended) ARTA process consisting of the marginal distribution F_Y and the $ARMA(p,q)$ base process is returned. Note, that for $q = 0$ the algorithm fits an ARTA model with $AR(p)$ base process, while an extended ARTA model is fitted otherwise.

Algorithm 7.3 describes the routine that is used for fitting CHEPs and CAPPs.

Inputs are the marginal distribution, which is an arbitrary acyclic PH distribution (for CAPP fitting) or a Hyper-Erlang distribution (for CHEP fitting), the vector $\hat{\boldsymbol{\rho}}$ with the desired autocorrelation and the base process order. The routine then first determines the m elementary series of the distribution. For a general APH distribution they are parameterized by S_i that describes the number of phases of series i, the probability τ_i and a vector $\boldsymbol{\Lambda_i}$ that contains the rates for the phases of series i. For a HErD the series are directly given by the branches of the distribution and instead of a vector there is only a single rate λ_i for each series. As mentioned in Chapters 5 and 6 the elementary series have to be sorted according to their mean values. In the second step the algorithm checks whether the desired autocorrelation can be reached with the given marginal distribution, i.e. whether $\hat{\rho}_{min} \leq \min(\hat{\boldsymbol{\rho}})$ and $\hat{\rho}_{max} \geq \max(\hat{\boldsymbol{\rho}})$ where $\min(\cdot)$ and $\max(\cdot)$ return the smallest and largest value from the vector of desired autocorrelations. ρ_{min} and ρ_{max} are the minimal and maximal possible autocorrelation for the base process, respectively. In theory $\rho_{min} = -1$ and $\rho_{max} = 1$ are possible, but it might be difficult for the base process fitting algorithm to find such an $ARMA(p,q)$ process, in particular if other autocorrelation lags have to be matched as well. Hence, it is more reasonable to use values close to those theoretical values, i.e. to set $\rho_{min} = -0.9$ and $\rho_{max} = 0.9$. If the desired autocorrelation cannot be achieved with the given distribution the algorithm calls Algorithm 7.5 to perform a transformation of the distribution to increase the possible autocorrelation range. Since this changes the representation of the distribution (but of course not the distribution itself) a recomputation of the elementary series is necessary after that. The remaining steps are similar to Algorithm 7.2. The routine calls the search Algorithm 7.4 to determine the base process autocorrelation that yields the desired CHEP or CAPP autocorrelation, fits an $ARMA(p,q)$ to this au-

Algorithm 7.3: Routine for fitting CHEPs and CAPPs

input : acyclic Phase-type marginal distribution $APH(n)$ (or $HErD(n)$) with distribution function F_Y
$\hat{\boldsymbol{\rho}} = (\hat{\rho}_1, \hat{\rho}_2, \ldots, \hat{\rho}_r)$: desired autocorrelation for CAPP
p: number of AR coefficients
q: number of MA coefficients
output: $CAPP(n, p, q)$ (or $CHEP(n, p, q)$)

1 determine elementary series $i = 1, \ldots, m$ of PH distribution with S_i phases, probabilities τ_i, and rates vector $\boldsymbol{\Lambda_i}$ (or rates λ_i)
2 sort elementary series according to their mean values
3 determine possible range for autocorrelation $[\hat{\rho}_{min}, \hat{\rho}_{max}]$ by computing Equation 6.9 (or Equation 5.6) for base process correlation ρ_{min} and ρ_{max}
4 **while** $((\hat{\rho}_{min} > \min(\hat{\boldsymbol{\rho}})) \parallel (\hat{\rho}_{max} < \max(\hat{\boldsymbol{\rho}})))$ **do**
5 perform PH transformation by calling Algorithm 7.5
6 recompute the elementary series
7 recompute the possible range for autocorrelation $[\hat{\rho}_{min}, \hat{\rho}_{max}]$
8 compute $ARMA(p, q)$ autocorrelations $\boldsymbol{\rho} = (\rho_1, \rho_2, \ldots, \rho_r)$ by calling Algorithm 7.4
9 minimize Equation 4.11 to find an $ARMA(p, q)$ model which approximates $\boldsymbol{\rho}$
10 set variance of $ARMA(p, q)$ according to Equation 4.9
11 **return** *$CAPP(n, p, q)$ model with base $ARMA(p, q)$ process and distribution $APH(n)$ (or $CHEP(n, p, q)$ model with base $ARMA(p, q)$ process and distribution $HErD(n)$)*

tocorrelation and adjusts the variance of the base process. Depending on the type of the input marginal distribution the algorithm returns either a CHEP or a CAPP.

Algorithm 7.4 is a simple search algorithm that is used for finding the base process autocorrelation for a given main process (i.e. (extended) ARTA, CHEP or CAPP) autocorrelation. Inputs are the marginal distribution F_Y and the desired autocorrelation $\hat{\boldsymbol{\rho}}$ for the main process. The procedure returns the base process autocorrelation $\boldsymbol{\rho}$. The algorithm makes use of the properties that have been stated for the relation between the base process autocorrelation and the main process autocorrelation in Chapters 4, 5 and 6. In the simplest case the main process autocorrelation is zero and the base process autocorrelation is set to zero as well (cf. line 3) according to Propositions 4.1, 5.1 and 6.1. For a non-zero main process autocorrelation the procedure searches either in the interval $[lx, rx] = [-1.0, 0.0]$ for a negative main process autocorrelation or in $[lx, rx] = [0.0, 1.0]$ for a positive main process autocorrelation (lines 5 to 10). Additionally a midpoint mx in this interval is computed that divides the interval $[lx, rx]$ as described in Section 4.2 into two fractions with a width of about 38% and 62%, respectively. A second midpoint mx_2 is computed that divides the interval $[mx, rx]$ in a similar way. With this initial points a golden section search is started to find the minimum of Equation 4.10. The algorithm evaluates Equation 4.10 for the two midpoints mx and mx_2 and depending on the value for these two points continues to search in the interval $[lx, mx_2]$ or $[mx, rx]$, respectively, with adjusted midpoints that divide the new interval as described above. If finally the width of the interval is smaller than some

Algorithm 7.4: Search algorithm

input : marginal distribution F_Y
$\hat{\boldsymbol{\rho}} = (\hat{\rho}_1, \hat{\rho}_2, \ldots, \hat{\rho}_r)$: desired autocorrelation for CAPP
output: $\boldsymbol{\rho} = (\rho_1, \rho_2, \ldots, \rho_r)$

```
1  for (i = 1 to r) do
2      if (ρ̂_i == 0.0) then
3          ρ_i = 0.0
4      else
5          if (ρ̂_i < 0) then
6              lx = −1.0
7              rx = 0.0
8          else
9              lx = 0.0
10             rx = 1.0
11         mx = lx + 0.381966 (rx − lx)
12         mx_2 = mx + 0.381966 (rx − mx)
13         f1 = result of Equation 4.10 for ρ̂_i and mx
14         f2 = result of Equation 4.10 for ρ̂_i and mx_2
15         while (|rx − lx| > ε) do
16             if (f2 < f1) then
                   //continue in interval [mx, rx]
17                 lx = mx
18                 mx = mx_2
19                 mx_2 = (1.0 − 0.381966) mx + 0.381966 rx
20                 f1 = f2
21                 f2 = result of Equation 4.10 for ρ̂_i and mx_2
22             else
                   //continue in interval [lx, mx2]
23                 rx = mx_2
24                 mx_2 = mx
25                 mx = (1.0 − 0.381966) mx_2 + 0.381966 lx
26                 f2 = f1
27                 f1 = result of Equation 4.10 for ρ̂_i and mx
28         if (f1 < f2) then
29             ρ_i = mx
30         else
31             ρ_i = mx_2
32 return ρ = (ρ_1, ρ_2, ..., ρ_r)
```

predefined ϵ, the desired base process autocorrelation ρ_i that yields the main process autocorrelation $\hat{\rho}_i$ has been found. For computing Equation 4.10 the main process autocorrelation $\omega(x_i)$ needs to be evaluated and this is actually the only part of the procedure that depends on the marginal distribution and thus, on the type of the stochastic

process. For extended ARTA models $\omega(x_i)$ is computed according to Equation 4.3, for CHEPs Equation 5.6 is used and for CAPPs Equation 6.9 is applied. As mentioned in Section 4.2 a table should be used to store pairs $(x_i, \omega(x_i))$ to avoid the recomputation of $\omega(x_i)$ for values that have already been computed before.

Algorithm 7.5: Routine for APH transformation

input : $APH(n)$
output: $APH'(n)$

```
1 if (APH(n) is not in series canonical form) then
2 |   apply Transformation 1
  |   // APH'(n) now has series canonical form
3 else
4 |   apply Transformation 2
  |   // APH'(n) now has an additional state
5 return transformed APH distribution APH'(n)
```

For the APH transformation outlined in Algorithm 7.5 two cases have to be distinguished. If the distribution is not in series canonical form, Transformation 1 can be applied to obtain a distribution in series canonical form, otherwise Transformation 2 is used to add an additional state.

7.2.1. Selecting the Model Order

The selection of the model order of the base process is important for the quality of the fitted model. If p and q are too small for the number of autocorrelation lags to match, the model will provide a poor approximation of the autocorrelation. Since a run of Algorithm 7.1 only takes a few seconds we decided to fit an $ARMA(p, q)$ model for all combinations of $p \in [p_{min}, p_{max}]$ and $q \in [q_{min}, q_{max}]$ for given $p_{min}, p_{max}, q_{min}, q_{max}$ and select the model with the best result according to Equation 4.11.

Chapter 8

Experimental Results

In this chapter the ability of the stochastic processes introduced in Chapters 5 and 6 to capture the characteristics of different traces will be assessed. In the first part of this empirical study in Section 8.1 the processes are fitted to synthetically generated traces to evaluate the quality of the proposed approach in approximating characteristics from another stochastic process. We will neglect ARTA processes in this study and only fit CHEPs and CAPPs to the synthetically generated traces. Obviously, since all these processes use the same type of ARMA base process, no surprising results regarding the autocorrelation are to be expected from fitting ARTA processes. Moreover, the overall fitting quality of an ARTA process will heavily depend on the type of marginal distribution and no automated approach exists to select the best type of distribution from all possible distributions. Additionally it was already shown in Chapter 5 that PH distributions are flexible enough to capture a wide range of distributions. On the other hand, we can easily generate traces from ARTA models that exhibit nontypical shapes for the distribution, which are difficult to fit for PH fitting tools. Hence, we will only use ARTA processes to obtain some synthetically generated traces.

For the second part of the study in Section 8.2 the novel stochastic processes are fitted to real traces. Several characteristics of these models are compared with Markovian Arrival Processes that have been fitted to the same traces. In particular, we will compare the moments and the distribution and density functions to assess the fitting of the distribution and the joint moments and autocorrelation coefficients to assess the fitting of the dependence structure of the trace and the stochastic processes. For small and moderate sample sizes statistical tests could be applied to decide whether a trace might be drawn from a specific distribution or process. For example in [26] the two-dimensional KS test was used to evaluate the fitting quality of ARTA processes. However, for network traces with a million or more entries these tests tend to reject the hypothesis that the sample is drawn from the distribution or process even if it has been drawn from exactly this distribution [58]. Other measures like the joint density or the likelihood function of the correlated trace are hard to compute for the stochastic processes used in this work. Therefore, we used the queueing performance of the stochastic processes when used as input process to a simple queueing model instead as an additional measure for comparison.

For fitting the marginal distribution of a CHEP we use the tool `GFIT` (cf. [151] and

Section 5.3) and for fitting general acyclic PH distributions for CAPPs we use a tool called `Momfit` (cf. [42] and Section 6.2). Using these distributions a CHEP or CAPP is constructed as described in Chapter 7. Most of the MAPs presented in this study are taken from [100] where different MAP fitting algorithms have been compared.

8.1. Synthetically generated Traces

To assess whether CHEPs and CAPPs can capture the characteristics of other stochastic processes, several different processes have been simulated to obtain synthetically generated traces.

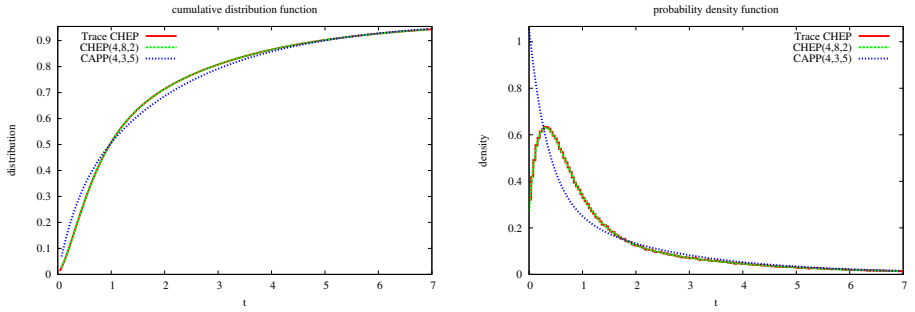
The first trace has been generated by a $CHEP(4,5,4)$ consisting of a Hyper-Erlang distribution with three branches, phases $\boldsymbol{S} = (1,1,2)$, rates $\boldsymbol{\lambda} = (0.12, 0.46, 2.97)$ and initial probabilities $\boldsymbol{\tau} = (0.08, 0.55, 0.37)$ and an $ARMA(5,4)$ base process with AR coefficients

$$\boldsymbol{\alpha} = (0.50, 1.09, -0.42, -0.25, 0.078)$$

and MA coefficients

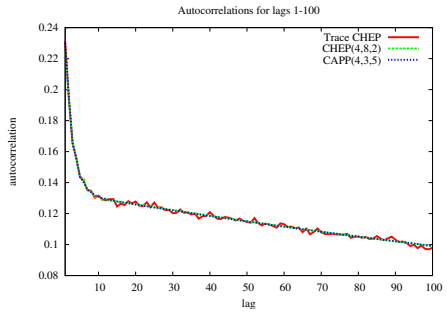
$$\boldsymbol{\beta} = (1.24, -0.69, -1.39, 0.03).$$

Figure 8.1 shows the results of a $CHEP(4,8,2)$ and a $CAPP(4,3,5)$ fitted to the



(a) cumulative distribution function

(b) probability density function

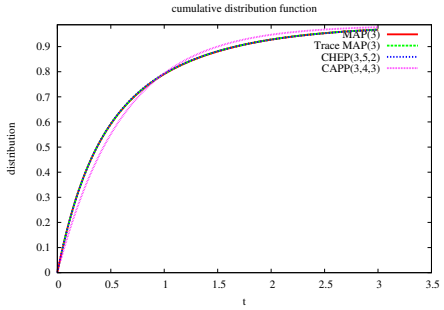


(c) autocorrelations

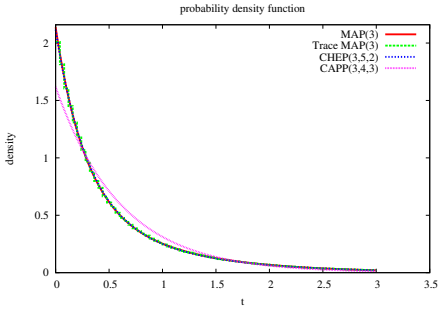
Figure 8.1.: Fitting results for a trace generated by a CHEP

trace. As one can see from Figures 8.1(a) and 8.1(b) `GFIT` was able to recreate the marginal distribution while `Momfit` resulted in an acyclic PH distribution with a similar

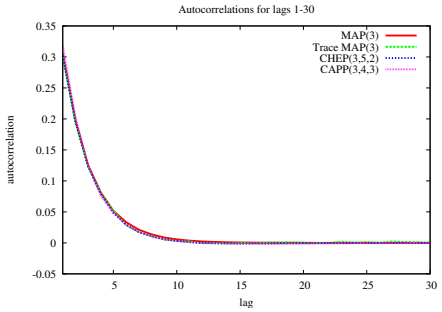
cdf but a different density function. In both cases the autocorrelation was captured almost exactly (cf. Figure 8.1(c)). Observe, that the fitted models have a different base process order than the original CHEP, which is caused by some slight variations in the autocorrelation structure between the original CHEP and the generated trace.



(a) cumulative distribution function



(b) probability density function

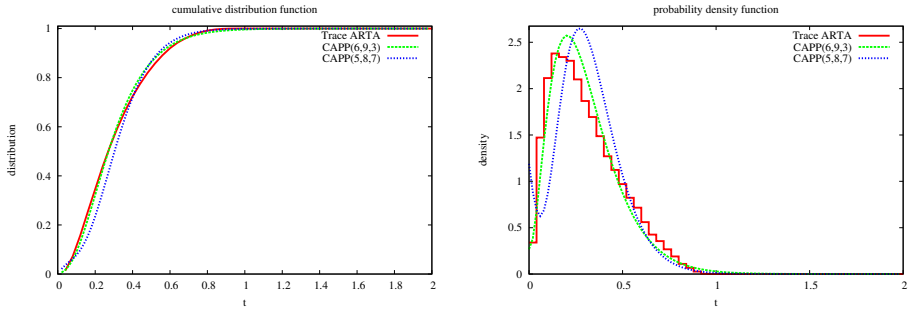


(c) autocorrelations

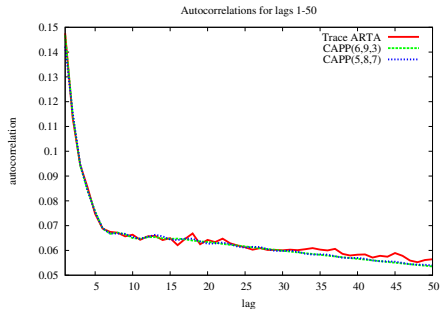
Figure 8.2.: Fitting results for a trace generated by a $MAP(3)$

As a second example a trace from the $MAP(3)$ that was already used in Section 3.2.2 was generated. The results are shown in Figure 8.2 and give a similar picture as for the first trace. Again, `GFIT` was able to deliver a slightly better fitting of the distribution than `Momfit`, but in both cases the autocorrelation was fitted almost exactly by the $CHEP(3,5,2)$ and the $CAPP(3,4,3)$, respectively.

To generate traces with a distribution that is nontypical for a PH distribution and therefore more difficult to fit than the two previous examples, ARTA processes with different marginal distributions have been used for trace generation. The first process has a Johnson bounded marginal distribution and an $ARMA(9,5)$ base process.
As one can see from Figure 8.3 both, `GFIT` and `Momfit` were able to provide a sufficient approximation of the distribution. Note, that the Hyper-Erlang distribution returned by `GFIT` was not adequate for capturing the autocorrelation structure and thus, a transformation was necessary to bring the distribution into series canonical form and add an additional state. Therefore, both processes in Figure 8.3 are CAPPs, although the $CAPP(6,9,3)$ resulted from a transformed Hyper-Erlang distribution with 5 states. For the general acyclic PH distribution returned by `Momfit` that was used for the second $CAPP(5,8,7)$ no transformation was necessary.



(a) cumulative distribution function (b) probability density function

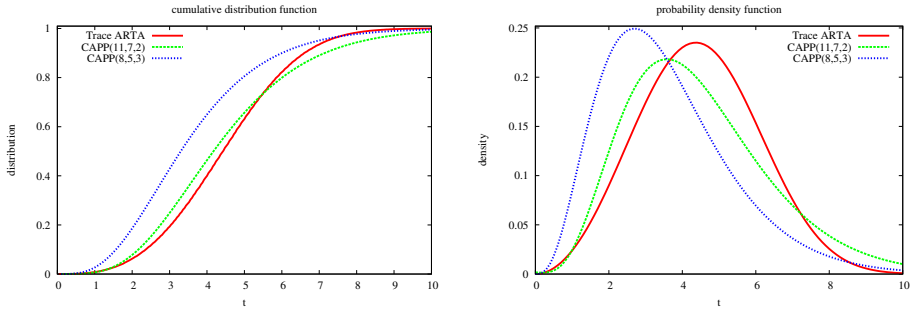


(c) autocorrelations

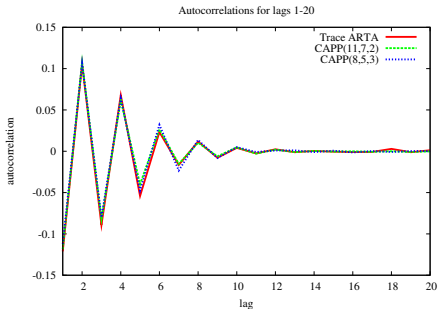
Figure 8.3.: Fitting results for a trace generated by an ARTA process with Johnson marginal distribution

The last synthetically generated trace was most difficult to fit. It was generated from an ARTA process with Weibull marginal distribution and a base process that exhibits positive and negative autocorrelation. The fitting results are shown in Figure 8.4. For both distributions, the Hyper-Erlang distribution returned by `GFIT` and the APH distribution fitted according to the moments by `Momfit`, several transformation steps were necessary to obtain a marginal distribution that could capture the negative correlation of the trace. Figures 8.4(a) and 8.4(b) show that the $CAPP(11, 7, 2)$ using the transformed Hyper-Erlang distribution provides a slightly better approximation of the distribution than the $CAPP(8, 5, 3)$ using an APH distribution fitted according to the moments. As one can see from Figure 8.4(c) both processes were able to provide a good fitting of the autocorrelation structure of the trace.

From the previous examples we have seen that CHEPs and CAPPs are able to model a variety of different distribution shapes with different autocorrelation structures. However, the choice of the basic parameters used for the fitting steps is not really done in a systematic way. Usually an exhaustive search is performed over a subset of the parameter region to find the best model for the available data. For example `GFIT` tries several combinations for the number of branches and number of phases for each branch and selects the best model from these combinations (cf. [151] and Sect. 5.3). A similar approach is proposed in Sect. 7.2 for selecting the order of the $ARMA(p, q)$ base process. This can be done if the parameter region is not too large, which is often the case,



(a) cumulative distribution function (b) probability density function



(c) autocorrelations

Figure 8.4.: Fitting results for a trace generated by an ARTA process with Weibull marginal distribution

but reaches its limits for cases where a large number of phases or AR and MA coefficients are required to model the distribution or autocorrelation structure, respectively. Additionally it is unclear which measures are most important for the fitting quality. For the analysis of technical systems and also in simulation models where several parameters have an unknown impact on the results often ideas from statistical design of experiments [34] are used to evaluate the influence of parameters in a systematic way in order to identify the parameters which have the main influence on the result measures. It seems promising to consider similar ideas for the parametrization of CHEPs and CAPPs. To the best of the author's knowledge, design of experiments has not been used before for assessing fitting approaches for stochastic processes. However, although the design of experiments has the potential to give new insights into the effect of different parameters on the fitting quality and may allow one to find good parameter settings with less effort, the identification of important factors and the experimental setup require some careful planning and preliminary considerations to yield meaningful results. Since the different measures and parameters are highly dependent, a straightforward application of factorial designs of experiments [115] does not result in a useful approach. Consequently, the application of more systematic approaches from experimental design for setting parameters of CHEPs and CAPPs is identified as a promising area which should be considered in future research.

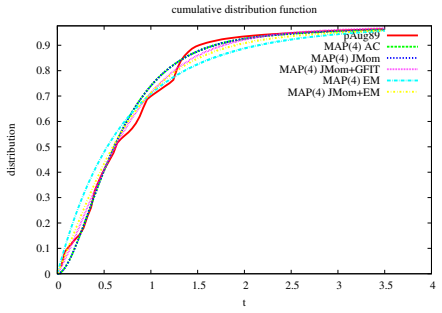
For MAP fitting one often uses common benchmark traces that have been measured from a real system and the queueing performance as a result measure. Since both MAPs and CHEPs/CAPPs have a PH marginal distribution we will use these real benchmark traces to compare these different process types in the following.

8.2. Real Traces

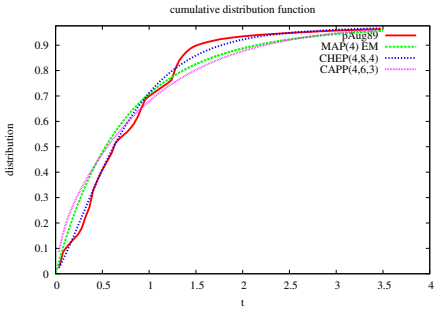
The real traces considered in this empirical study have already been introduced in the previous chapters. In particular, the common benchmark traces *LBL-TCP-3* [130] and *BC-pAug89* [108] from the Internet Traffic Archive [148] (cf. Chapter 3) and the trace *TUDo* [100] (cf. Chapter 5) are taken into account. Several different stochastic processes are fitted to these traces and different quantities like the distribution function, (joint) moments and the autocorrelation coefficients are compared. Additionally, queueing results are generated from a simple server with an exponential distribution as service process and a queue capacity of 10. To obtain some reference values for the model the system was simulated for each of the traces and different utilization levels in a trace driven simulation. After that the model is simulated for the same amount of time with the different fitted stochastic processes as arrival process. The focus of this study is a comparison between MAPs and CHEPs/CAPPs. ARMA and ARTA processes are omitted for the reasons mentioned above and in Chapter 3. Most of the MAPs are taken from [100] where MAPs of different order (from $n = 2$ to $n = 6$) are fitted to the three traces to compare different MAP fitting algorithms. In particular, the MAPs are fitted with an EM algorithm [40] and two approaches that fit the MAPs in two steps by expanding a PH distribution into a MAP, either by considering the empirical joint moments of the trace [42] or the empirical autocorrelation coefficients [100] (see also Section 2.3.6). The PH distribution for the first step is fitted using `Momfit` or `GFIT`.

8.2.1. Results for the Trace *BC-pAug89*

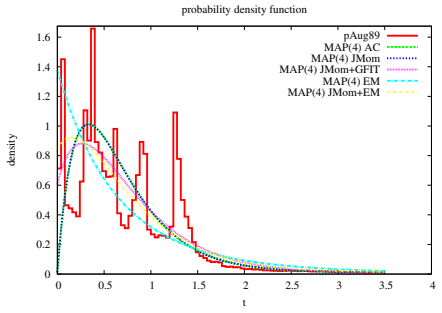
We start with the comparison of the fitted stochastic processes for the trace *BC-pAug89*. Figure 8.5 shows the results for the fitted processes of order 4 concerning the distribution. In Figure 8.6 the results regarding the correlation are shown. The curves resulting from MAP fitting using autocorrelation coefficients are labeled with `AC`. Curves for MAPs from joint moment fitting and expectation maximization are labeled with `JM` and `EM`, respectively. Usually `Momfit` [42] was used to obtain the distribution for joint moment and autocorrelation MAP fitting and for CAPP fitting. `GFIT` [151] was used to fit the marginal distribution for CHEPs. In some cases `GFIT` was used as well to obtain the distribution for the two-step MAP fitting algorithms, which is denoted in the plots. In addition to the pure EM algorithm that starts with a random MAP to improve the likelihood we used the MAPs resulting from AC and JM fitting as initial solutions for the EM algorithm to reduce the number of iterations until convergence was reached. These MAPs are labeled with `JM + EM` and `AC + EM`. Figures 8.5 and 8.6 contain two plots for each quantity under consideration. The first one contains the different MAPs, for the second one the best MAP was plotted together with a $CHEP(4, 8, 4)$ and a $CAPP(4, 6, 3)$.



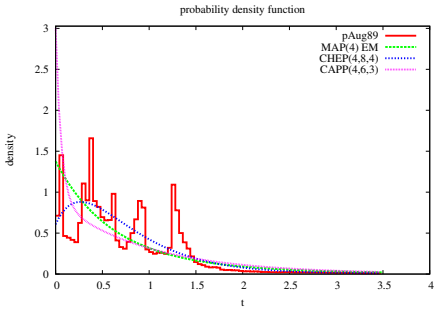
(a) cumulative distribution function (1)



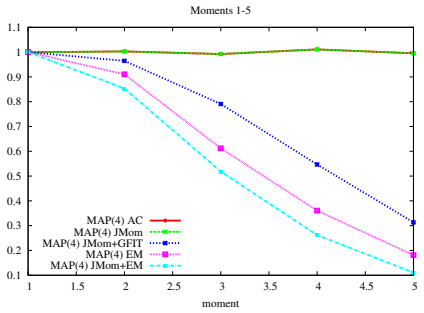
(b) cumulative distribution function (2)



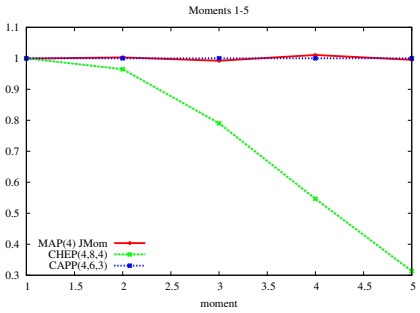
(c) probability density function (1)



(d) probability density function (2)

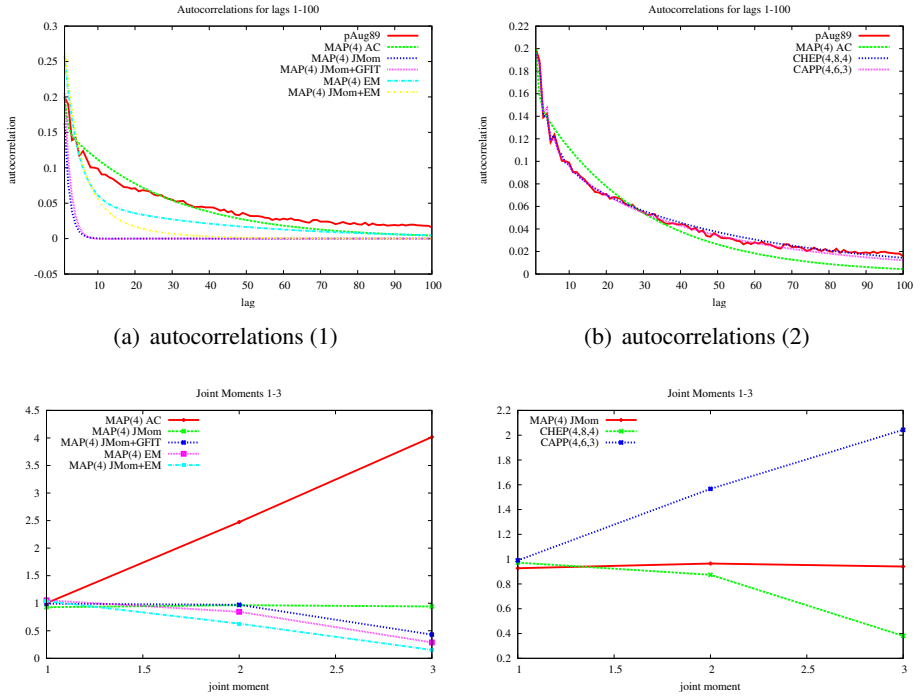


(e) moments (1)



(f) moments (2)

Figure 8.5.: Distribution related fitting results for the trace *BC-pAug89* and MAPs/PH distributions of order 4



(a) autocorrelations (1)

(b) autocorrelations (2)

(c) joint moments (1)

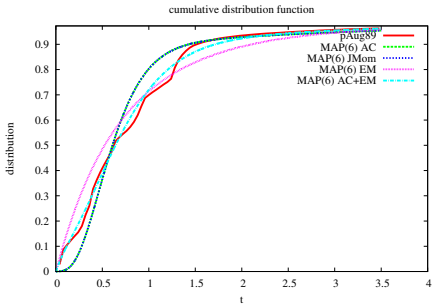
(d) joint moments (2)

Figure 8.6.: Dependence related fitting results for the trace *BC-pAug89* and MAPs/PH distributions of order 4

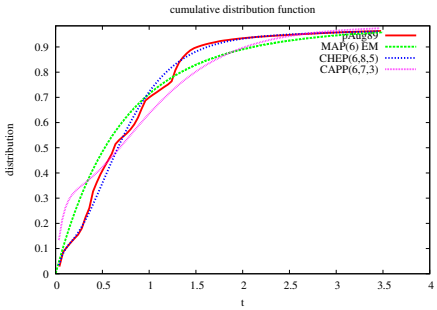
From Figures 8.5(a) and 8.5(b) it becomes visible that all the stochastic processes provide an adequate fitting of the distribution of the trace, but the best results are obtained from fitting approaches that use an EM algorithm, i.e. `MAP(4) EM`, `MAP(4) JMom+GFIT` and the $CHEP(4, 8, 4)$. On the other hand, one can see from Figures 8.5(e) and 8.5(f), that show the first moments of the processes relative to the empirical moments of the trace, that the EM algorithms underestimate the higher moments, while the fitting algorithms that explicitly fit according to the moments provide a much better approximation.

The MAPs resulting from joint moment fitting failed to capture the autocorrelations as one can see from Figure 8.6, while both autocorrelation MAP fitting and the MAP EM algorithm resulted in a much better approximation of the lag-k autocorrelations, although the latter tends to underestimate the autocorrelation. The best results regarding the autocorrelation are obtained from CHEP and CAPP fitting, which provide a very close approximation of the first 100 lags. In contrast only the MAPs resulting from joint moment fitting capture the joint moments of the trace, while the other MAP fitting algorithms and the CHEP/CAPP fitting approaches, which do not fit according to the joint moments, either under- or overestimate the joint moments of the trace.

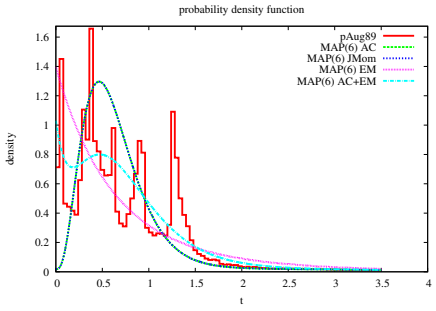
Similar observations can be made for MAPs of order 6 and CHEPs/CAPPs with a PH distribution of order 6, respectively. The results for a $MAP(6)$, a $CHEP(6, 8, 5)$ and a $CAPP(6, 7, 3)$ are shown in Figures 8.7 and 8.8.



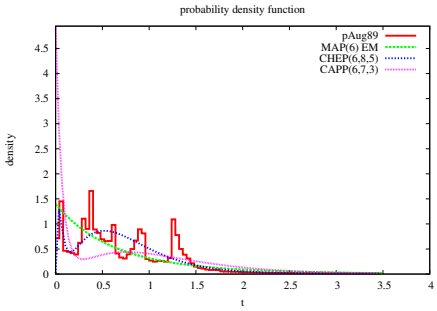
(a) cumulative distribution function (1)



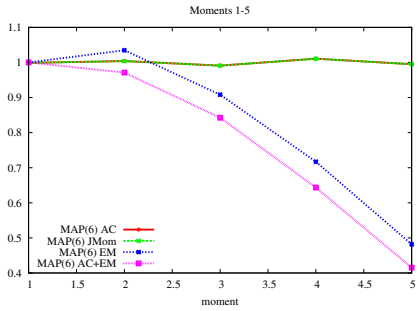
(b) cumulative distribution function (2)



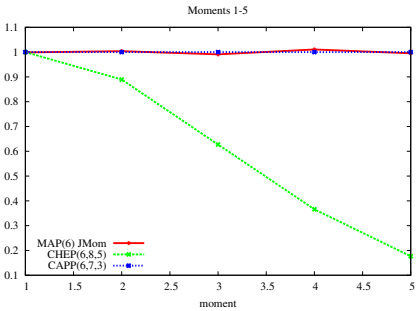
(c) probability density function (1)



(d) probability density function (2)

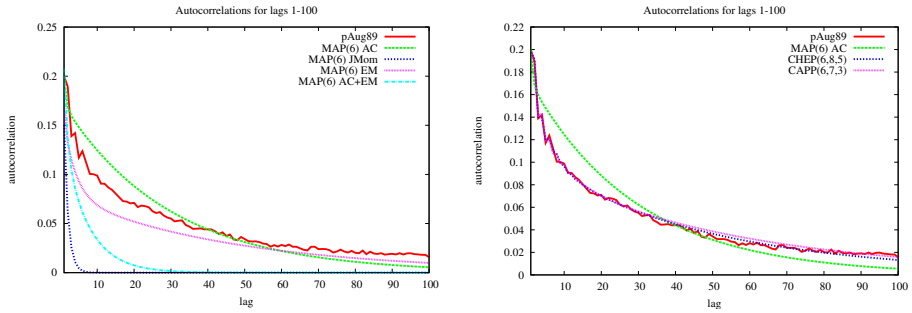


(e) moments (1)

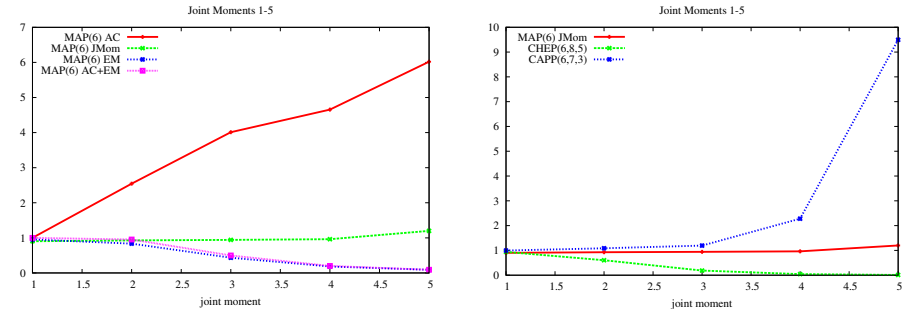


(f) moments (2)

Figure 8.7.: Distribution related fitting results for the trace *BC-pAug89* and MAPs/PH distributions of order 6



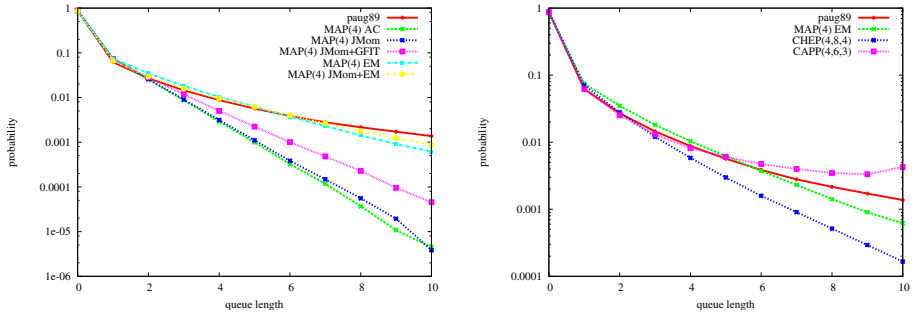
(a) autocorrelations (1) (b) autocorrelations (2)



(c) joint moments (1) (d) joint moments (2)

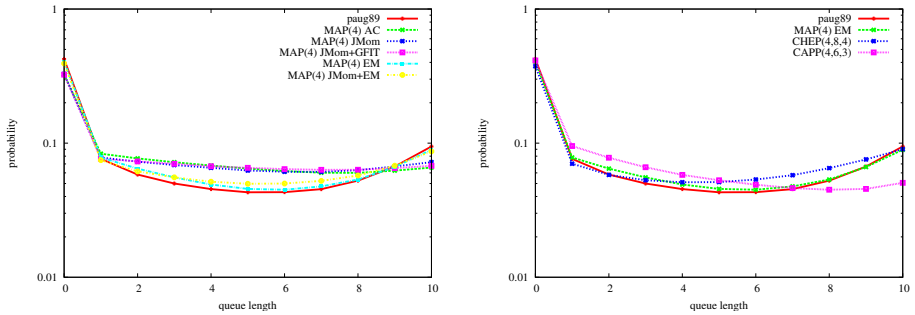
Figure 8.8.: Dependence related fitting results for the trace *BC-pAug89* and MAPs/PH distributions of order 6

Figures 8.9 and 8.10 show the queue length distributions for the original trace and all the fitted stochastic processes of order 4 and 6, respectively. Additionally the mean queue length values are listed in Table 8.1. The figures and the table contain the results for different utilization levels of the server between $\varrho = 0.33$ and $\varrho = 1$. The utilization of $\varrho = 1$ results in a overload situation for the server while for all other utilization levels the server is not overloaded. As one can see from the two figures the



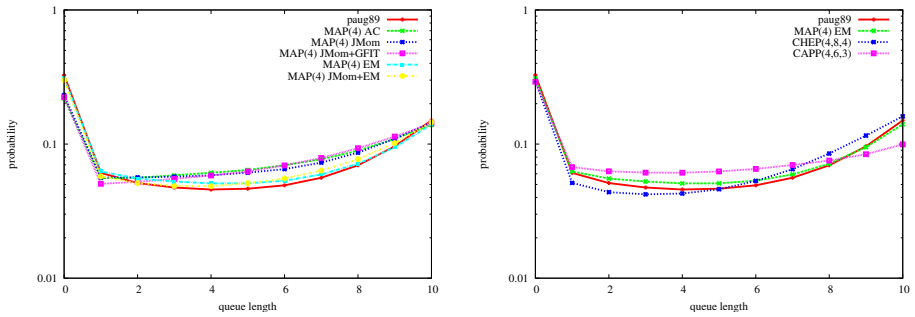
(a) utilization $\varrho = 0.33$ (1)

(b) utilization $\varrho = 0.33$ (2)



(c) utilization $\varrho = 0.83$ (1)

(d) utilization $\varrho = 0.83$ (2)



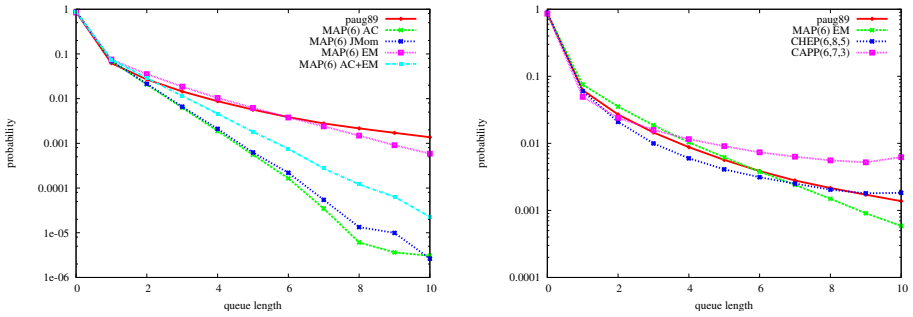
(e) utilization $\varrho = 1$ (1)

(f) utilization $\varrho = 1$ (2)

Figure 8.9.: Queueing results for different utilization levels ϱ using the trace *BC-pAug89* and MAPs/PH distributions of order 4

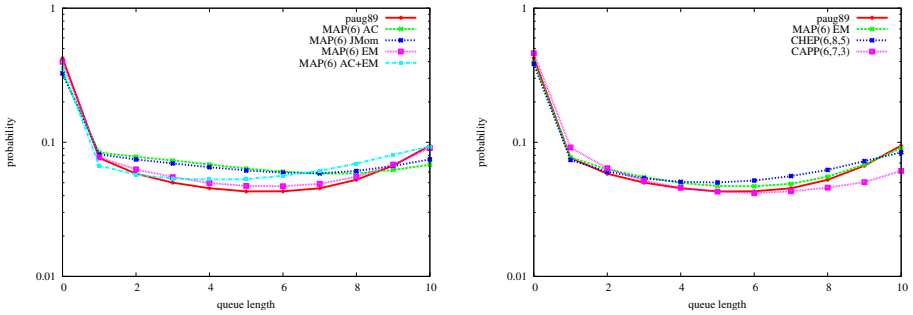
queue length distributions from the MAPs resulting from either the pure EM or the EM algorithm combined with one of the other approaches provide the best approximation of the queue length distribution from the trace. For smaller utilization levels all models

underestimate the larger values of the queue length, albeit the values are very small for the original trace. The queue length distributions for CHEPs and CAPPs are slightly worse than the distribution for the `MAP EM`, although for order 6 they are comparable.



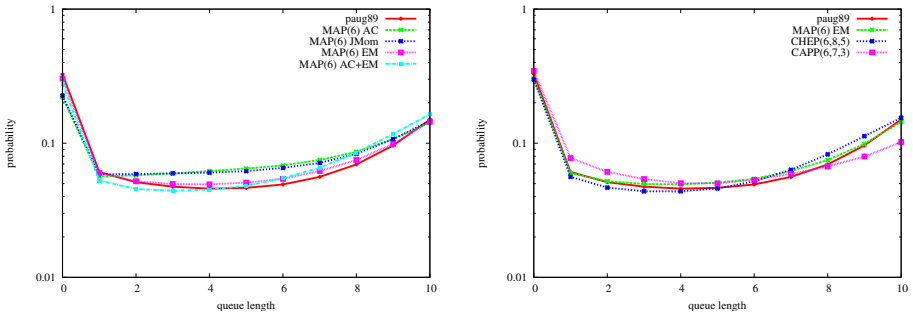
(a) utilization $\varrho = 0.33$ (1)

(b) utilization $\varrho = 0.33$ (2)



(c) utilization $\varrho = 0.83$ (1)

(d) utilization $\varrho = 0.83$ (2)



(e) utilization $\varrho = 1$ (1)

(f) utilization $\varrho = 1$ (2)

Figure 8.10.: Queueing results for different utilization levels ϱ using the trace *BC-pAug89* and MAPs/PH distributions of order 6

Finally, Figure 8.11 shows the results for some CHEPs and MAPs with a higher order for the PH interarrival time distribution. As mentioned before, EM algorithms for MAP fitting have a slow convergence and fitting MAPs with 10 or more states to a trace with a million (or even more) observations is not really feasible using an EM algorithm. Therefore, we only used the autocorrelation fitting approach from [100] for MAP fitting and compared the results to fitted CHEPs/CAPPs. Both fitting algorithms

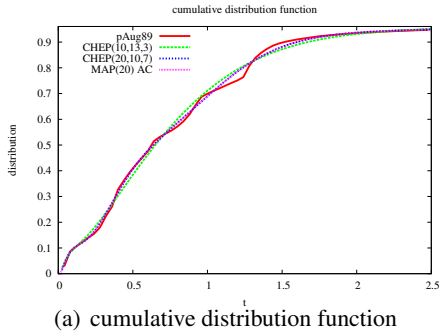
Model	$\varrho = 0.33$	$\varrho = 0.83$	$\varrho = 1$
paug89	0.312013	3.28225	4.33278
MAP(4) AC	0.170661	3.5404	4.95518
MAP(4) JMom	0.174841	3.626	4.90884
MAP(4) JMom+GFIT	0.208016	3.62717	5.02734
MAP(4) EM	0.33521	3.32061	4.35165
MAP(4) JMom+EM	0.317381	3.42371	4.52055
CHEP(4,8,4)	0.224763	3.63461	4.77194
CAPP(4,6,3)	0.371003	2.83832	4.16759
MAP(6) AC	0.14119	3.53162	4.94128
MAP(6) JMom	0.145738	3.6089	4.89891
MAP(6) EM	0.339633	3.39146	4.45859
MAP(6) AC+EM	0.203032	3.79714	4.86803
CHEP(6,8,5)	0.263359	3.50285	4.66143
CAPP(6,7,3)	0.47862	2.75662	3.81987

Table 8.1.: Mean queue length values for the trace *BC-pAug89* and MAPs/PH distributions of order 4 and 6

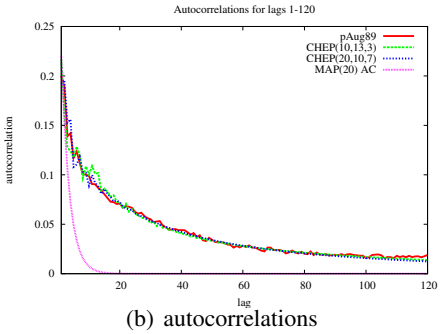
implement a similar approach, i.e. they take an APH distribution as input (for the examples from Figure 8.11 `GFIT` was used) and try to introduce autocorrelation by constructing a matrix $\boldsymbol{D_1}$ for MAPs or an ARMA base process for CHEPs/CAPPs.
As one can see from Figure 8.11(a) the two Hyper-Erlang distributions with 10 and 20 states, respectively, both provided a good approximation of the cumulative distribution function. Note, that the $MAP(20)$ and the $CHEP(20, 10, 7)$ use the same interarrival time distribution. For the autocorrelation MAP fitting algorithm it as difficult to fit a larger number of autocorrelations from the trace, while for the $CHEP(10, 13, 3)$ and the $CHEP(20, 10, 7)$ it was possible to consider the first 100 lags for fitting as one can see from Figure 8.11(b). This is also reflected by the queueing results shown in Figures 8.11(c) and 8.11(d). As one can see the MAP underestimates the tail of the queue length for the lower utilization of 0.5, while the CHEPs provided a very good approximation for the queue length distribution of the trace for both utilization levels.

8.2.2. Results for the Trace *LBL-TCP-3*

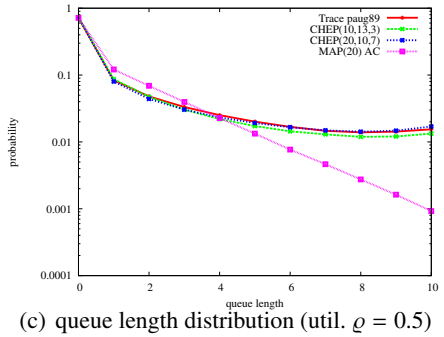
The results for the trace *LBL-TCP-3* confirm the observations from the trace *BC-pAug89*, though the trace was easier to fit and in general the results from all processes provide a better approximation than for *BC-pAug89*. Figures 8.12 and 8.13 summarize the fitting results for MAPs and CHEPs/CAPPs of order 3 and 4, respectively. Regarding the MAPs, again JM fitting and to a lesser degree EM fitting underestimate the autocorrelations, while AC fitting over- and EM fitting underestimate the joint moments. The CHEPs (i.e. a $CHEP(3, 6, 7)$ and a $CHEP(4, 6, 4)$) and CAPPs (i.e. a $CAPP(3, 7, 7)$ and a $CAPP(4, 12, 6)$) are able to capture the autocorrelation coefficients almost exactly, but cannot capture the joint moments, which were not used for the fitting. Concerning the distribution function all models provide a good approximation.



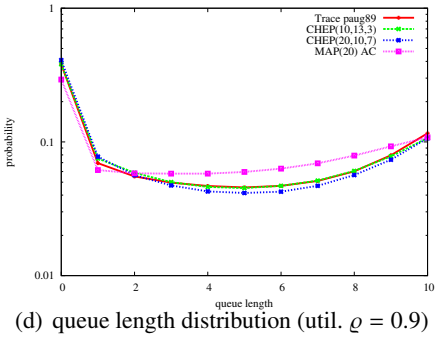
(a) cumulative distribution function



(b) autocorrelations



(c) queue length distribution (util. $\varrho = 0.5$)



(d) queue length distribution (util. $\varrho = 0.9$)

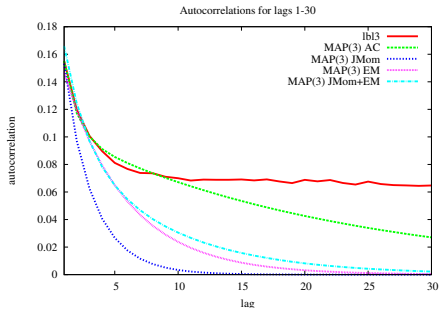
Figure 8.11.: Results for the trace *BC-pAug89* and MAPs/PH distributions of order $10 - 20$

The queueing results for the trace and the fitted models of order 3 and 4 are shown in Figures 8.14 and 8.15 and Table 8.2. Again the MAPs fitted with the EM algorithm resulted in the closest approximation and the results for CHEPs and CAPPs are only slightly worse.

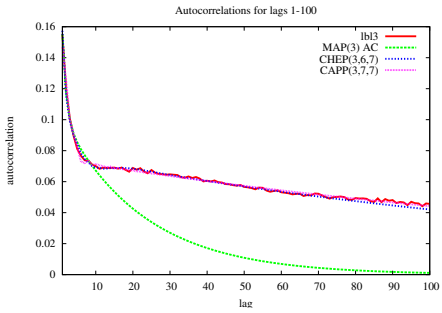
Results for larger models, in particular a $CHEP(10, 9, 3)$ and a $CHEP(20, 9, 3)$, are shown in Figure 8.16. For comparison we used the autocorrelation MAP fitting approach from [100] again, which starts with the same Hyper-Erlang distribution used for fitting the $CHEP(10, 9, 3)$ to construct a $MAP(10)$. Again, we can see that Hyper-Erlang distribution with 10 or 20 states can capture the cumulative distribution function of the trace almost exactly (cf. Figure 8.16(a)). Regarding the autocorrelations the MAP fitting algorithm had problems to capture a larger number of lags for MAPs of a larger order, while the fitting algorithm for CHEPs had no problems to return models that capture the first 100 lags (cf. Figure 8.16(b)). Consequently, the two CHEPs provide a good approximation for the queueing behavior of the trace as one can see from Figures 8.16(c) and 8.16(d), while the MAP underestimates the tail of the queue length distribution for smaller utilization levels.

8.2.3. Results for the Trace *TUDo*

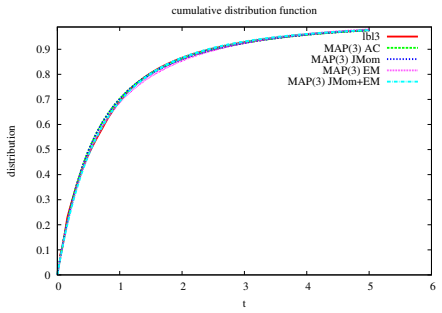
The last trace we used for our comparison was observed at a proxy server at TU Dortmund. It contains various bursts with very small interarrival times followed by a larger



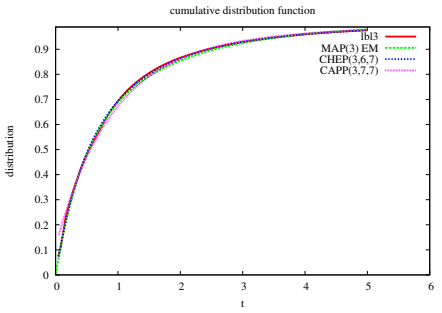
(a) autocorrelations (1)



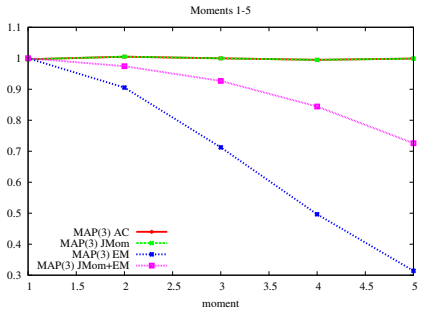
(b) autocorrelations (2)



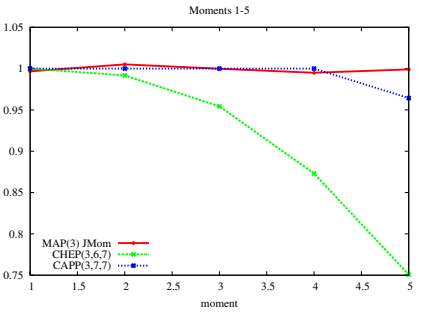
(c) cumulative distribution function (1)



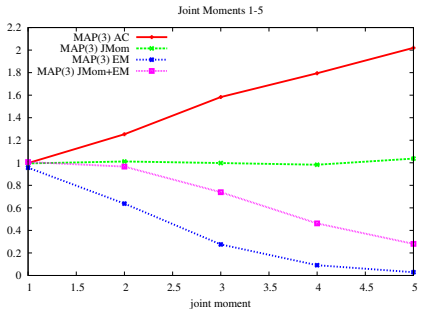
(d) cumulative distribution function (2)



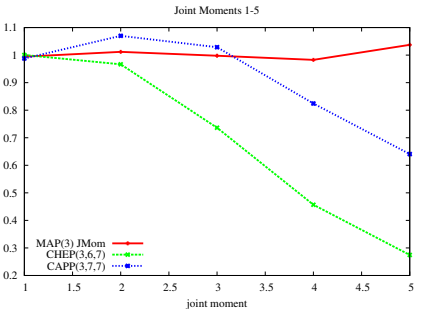
(e) moments (1)



(f) moments (2)

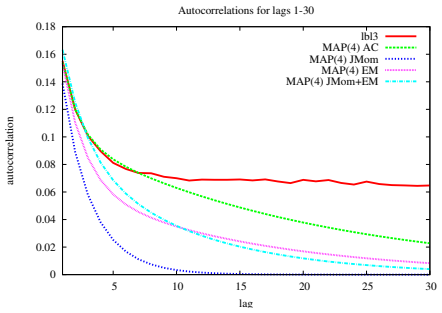


(g) joint moments (1)

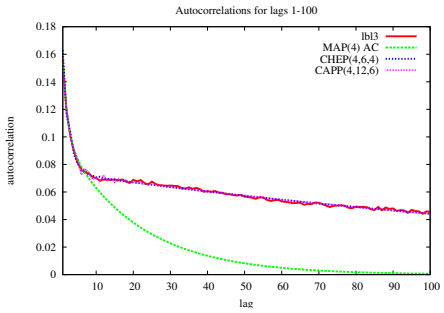


(h) joint moments (2)

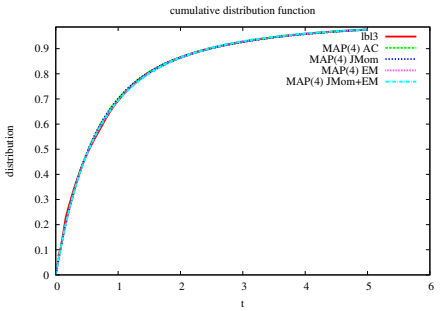
Figure 8.12.: Fitting results for the trace *LBL-TCP-3* and MAPs/PH distributions of order 3



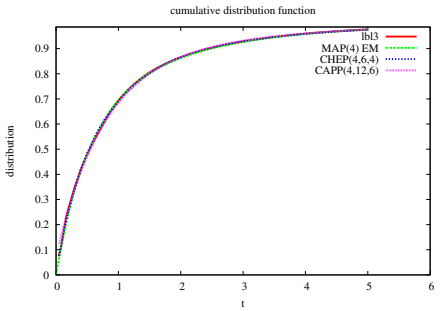
(a) autocorrelations (1)



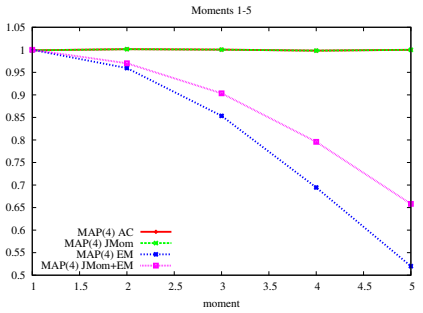
(b) autocorrelations (2)



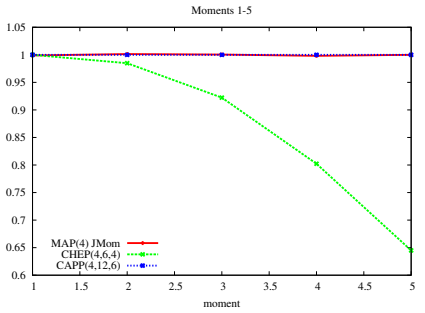
(c) cumulative distribution function (1)



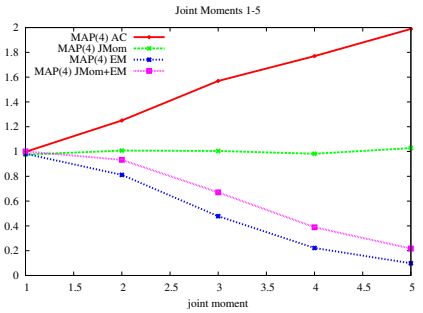
(d) cumulative distribution function (2)



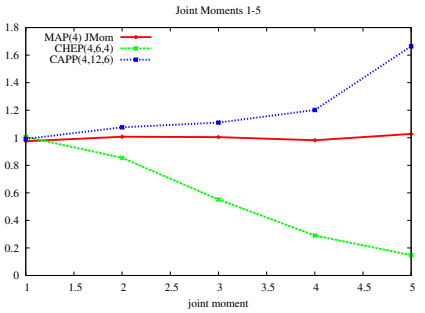
(e) moments (1)



(f) moments (2)

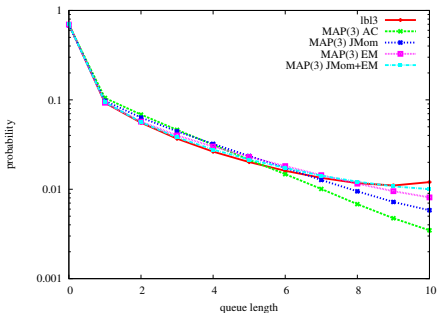


(g) joint moments (1)

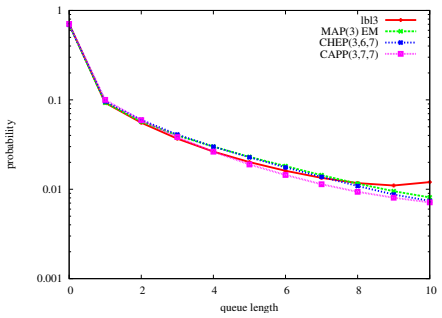


(h) joint moments (2)

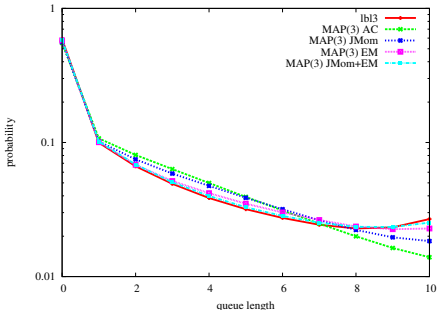
Figure 8.13.: Fitting results for the trace *LBL-TCP-3* and MAPs/PH distributions of order 4



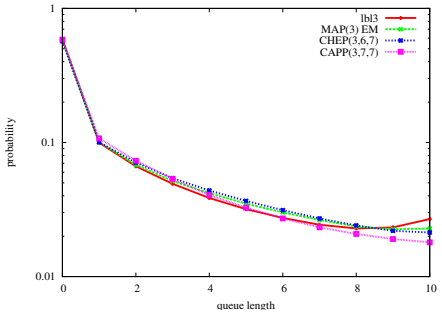
(a) utilization $\varrho = 0.5$ (1)



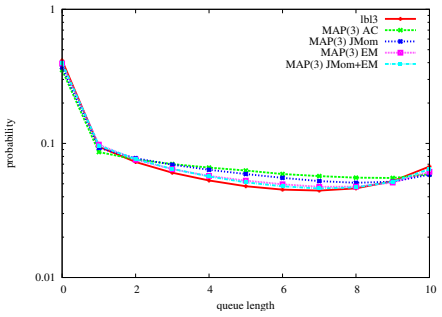
(b) utilization $\varrho = 0.5$ (2)



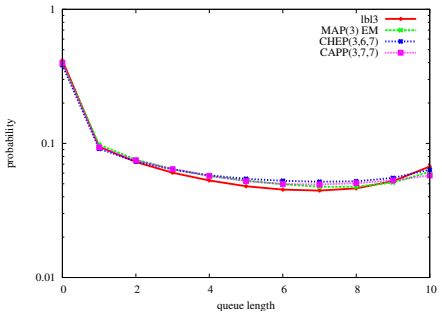
(c) utilization $\varrho = 0.625$ (1)



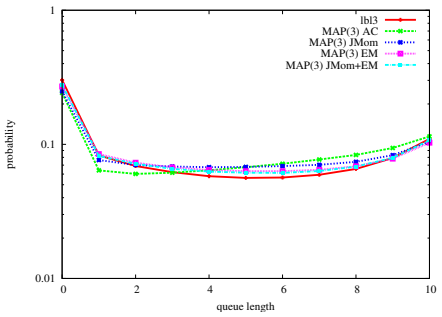
(d) utilization $\varrho = 0.625$ (2)



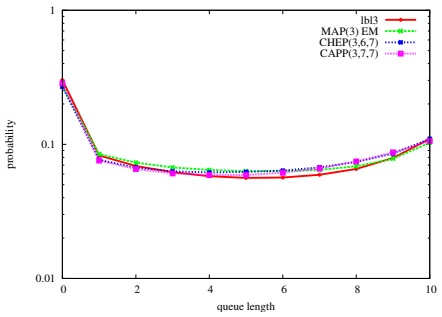
(e) utilization $\varrho = 0.83$ (1)



(f) utilization $\varrho = 0.83$ (2)

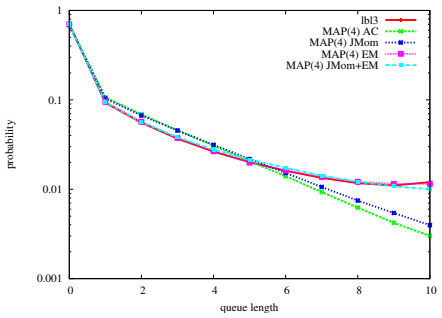


(g) utilization $\varrho = 1$ (1)

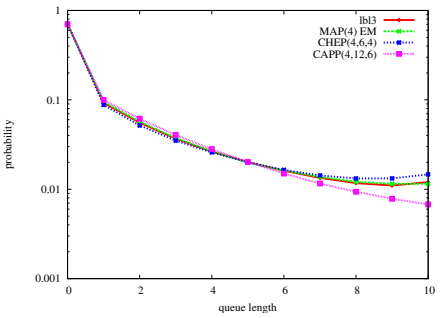


(h) utilization $\varrho = 1$ (2)

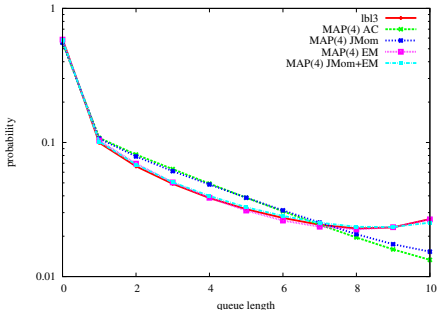
Figure 8.14.: Queueing results for different utilization levels ϱ using the trace *LBL-TCP-3* and MAPs/PH distributions of order 3



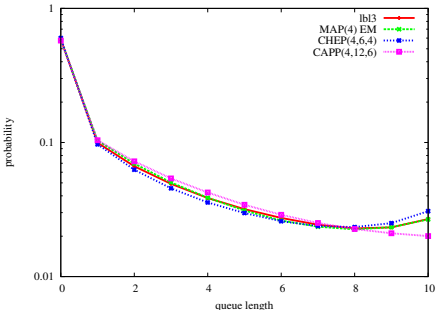
(a) utilization $\varrho = 0.5$ (1)



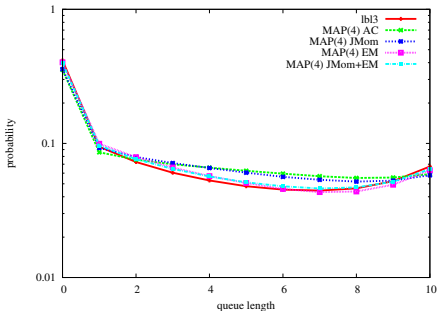
(b) utilization $\varrho = 0.5$ (2)



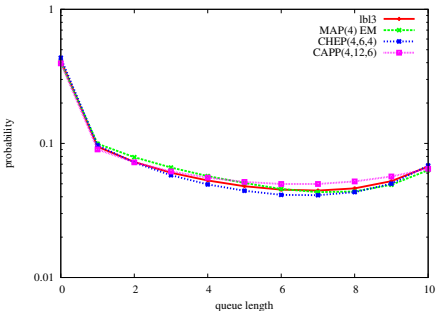
(c) utilization $\varrho = 0.625$ (1)



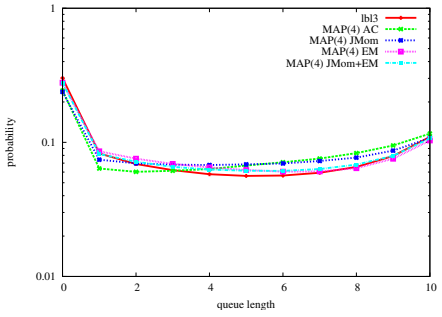
(d) utilization $\varrho = 0.625$ (2)



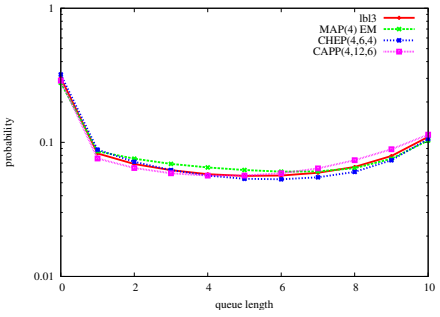
(e) utilization $\varrho = 0.83$ (1)



(f) utilization $\varrho = 0.83$ (2)



(g) utilization $\varrho = 1$ (1)



(h) utilization $\varrho = 1$ (2)

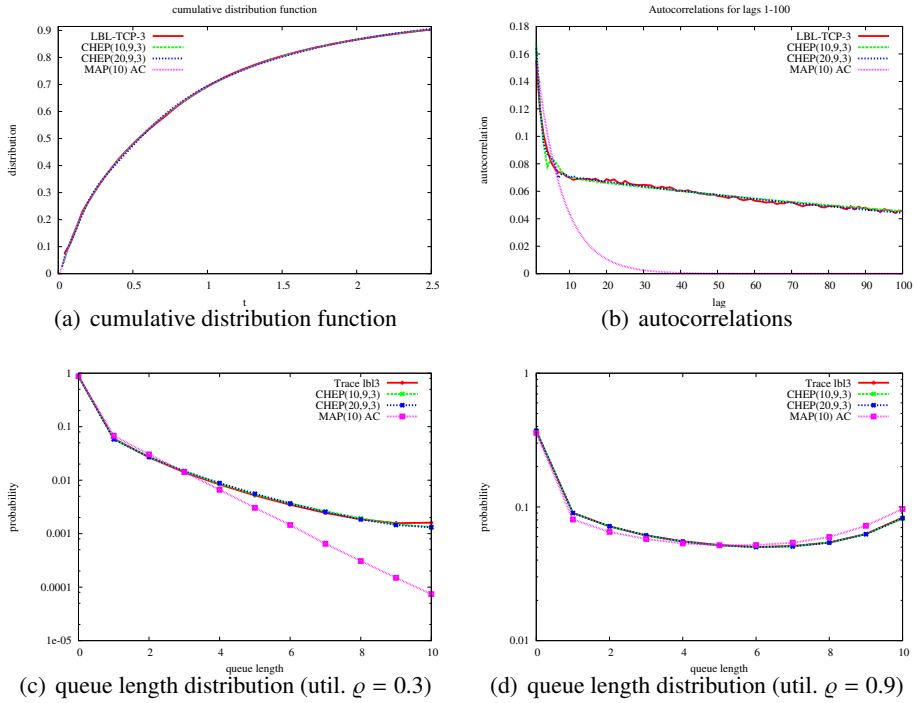
Figure 8.15.: Queueing results for different utilization levels ϱ using the trace *LBL-TCP-3* and MAPs/PH distributions of order 4

Model	$\varrho = 0.5$	$\varrho = 0.625$	$\varrho = 0.83$	$\varrho = 1$
lbl3	1.02379	1.68916	2.96906	4.00852
MAP(3) AC	0.904172	1.66165	3.31171	4.58981
MAP(3) JMom	0.996469	1.72472	3.16331	4.32507
MAP(3) EM	1.03094	1.72014	3.0191	4.11686
MAP(3) JMom+EM	1.03718	1.71215	3.02021	4.12081
CHEP(3,6,7)	1.01296	1.74445	3.16402	4.27034
CAPP(3,7,7)	0.918539	1.58335	3.03115	4.19297
MAP(4) AC	0.873541	1.63978	3.31329	4.59248
MAP(4) JMom	0.921584	1.67595	3.21201	4.43175
MAP(4) EM	1.03573	1.68273	2.93923	4.02889
MAP(4) JMom+EM	1.03718	1.71215	3.02021	4.12081
CHEP(4,6,4)	1.06995	1.69242	2.85101	3.81794
CAPP(4,12,6)	0.941934	1.67115	3.12165	4.22018

Table 8.2.: Mean queue length values for the trace *LBL-TCP-3* and MAPs/PH distributions of order 3 and 4

break until the next burst and therefore has high autocorrelations. As one can see from Figures 8.17 and 8.18 the trace was difficult to fit for all algorithms. The autocorrelations were underestimated by all MAP fitting algorithms except for AC fitting that uses an PH distribution fitted by `GFIT`. For MAPs of order 4 joint moment fitting overestimated the autocorrelations. The CHEPs and CAPPs were able to capture the autocorrelation structure, though for the Hyper-Erlang distribution of order 2 a transformation was necessary to bring the distribution into series canonical form resulting in a $CAPP(2, 4, 5)$. Additionally, a $CHEP(4, 3, 4)$, a $CAPP(2, 6, 3)$ and a $CAPP(4, 8, 3)$ were fitted. Regarding the distribution the best results are provided by the fitting algorithms that maximize the likelihood. Most algorithms underestimated the joint moments of the trace, only the MAPs resulting from joint moment fitting and the $MAP(4)$ fitted according to the autocorrelation provided an adequate fitting of the joint moments.
The difficulties in fitting the trace become also visible from the queueing results shown in Figures 8.19 and 8.20 and Table 8.3. For the higher utilization levels the CHEP and the MAPs fitted using an EM algorithm provided an adequate approximation of the queueing behavior.

Better results for the trace *TUDo* could be obtained for larger processes that use PH distributions of order 5 and 10, respectively. The results are shown in Figure 8.21. It contains the plots of the cumulative distribution function, the autocorrelation coefficients and the queue length distribution for the original trace, a $CHEP(5, 6, 6)$, a $CAPP(5, 6, 7)$, a $CHEP(10, 15, 6)$ and two MAPs of order 5 and 10, respectively. The PH distribution for the $CAPP(5, 6, 7)$ was fitted using the moment matching approach from [42]. All other PH distribution are Hyper-Erlang distribution fitted with `GFIT`. The MAPs have been fitted with the autocorrelation fitting approach from [100]. As one can see from Figure 8.21(a) the Hyper-Erlang distribution provided a better approximation of the empirical distribution than the APH fitted according to the mo-



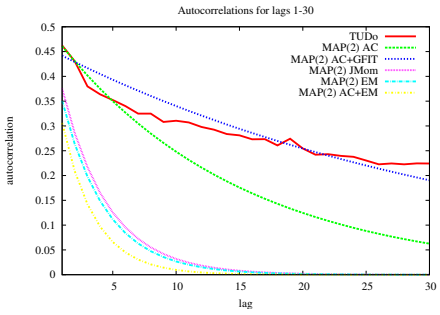
(a) cumulative distribution function (b) autocorrelations

(c) queue length distribution (util. $\varrho = 0.3$) (d) queue length distribution (util. $\varrho = 0.9$)

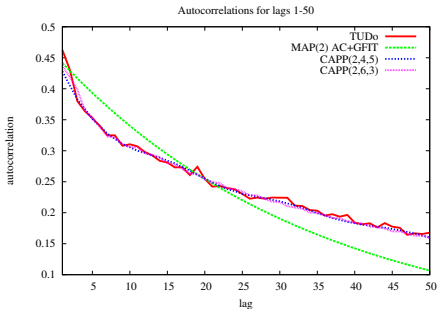
Figure 8.16.: Results for the trace *LBL-TCP-3* and MAPs/PH distributions of order $10 - 20$

ments, but in all cases a good approximation of the autocorrelation was possible, even though the trace exhibits much larger autocorrelations than the previous traces (cf. Figure 8.21(b)). Regarding the queue length distribution shown in Figure 8.21(c) the largest $CHEP(10, 15, 6)$ was able to capture the behavior of the trace best.

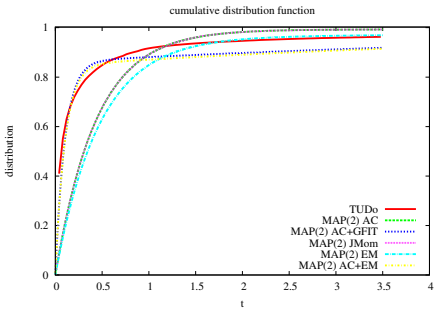
To summarize, the comparison of the different fitting methods and stochastic processes gives a mixed picture. Obviously, using a method that fits a process according to one quantity, like the joint moments or the autocorrelation, gives good results according to this quantity but usually results in a bad fitting according to other quantities that are not used for fitting. Thus, no approach is superior to all others according to all quantities. Considering the queueing model the results indicate that for smaller MAPs up to order 5 or 6 usually the EM algorithms give the best results followed by CHEP fitting, where the EM approach is only used for fitting the distribution. However, MAP fitting using an EM algorithm is by far the most time consuming method and can take up to several hours depending on the length of the trace and the order of the MAP. Furthermore, it should be mentioned that the fitting quality and the effort of the EM algorithm depends on the initial MAP and might be poor for badly chosen initial MAPs. The other MAP fitting approaches are much faster and independent of the trace length, i.e. joint moment fitting only takes a few seconds and autocorrelation fitting few minutes. Concerning time consumption CHEP and CAPP fitting is comparable to autocorrelation fitting for MAPs and usually takes a few minutes only depending on



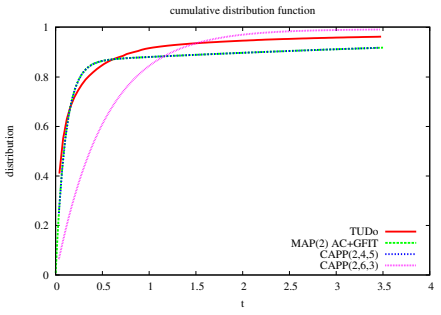
(a) autocorrelations (1)



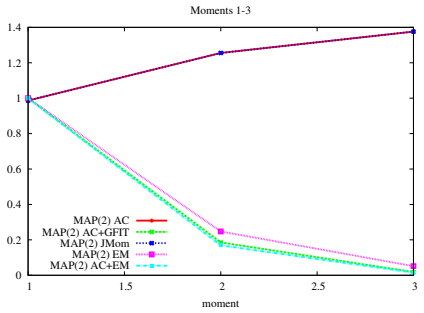
(b) autocorrelations (2)



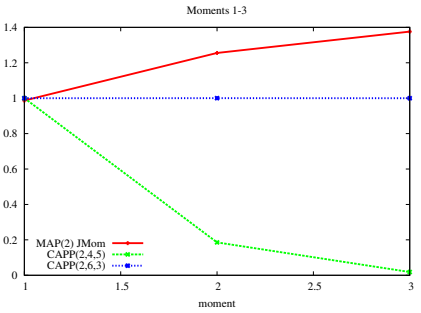
(c) cumulative distribution function (1)



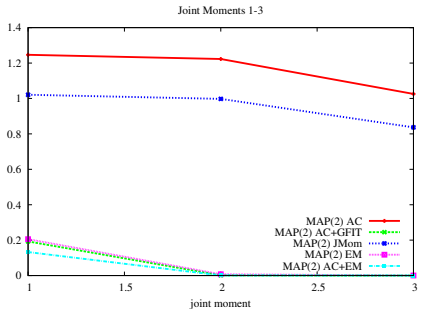
(d) cumulative distribution function (2)



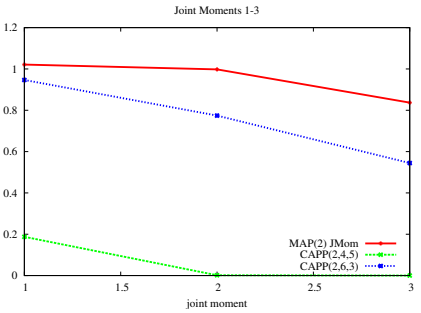
(e) moments (1)



(f) moments (2)

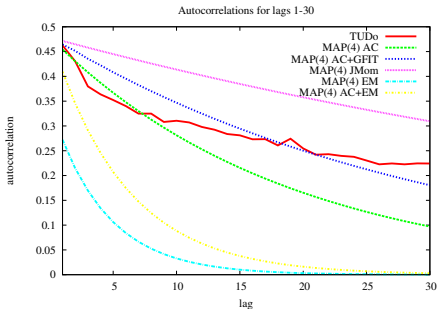


(g) joint moments (1)

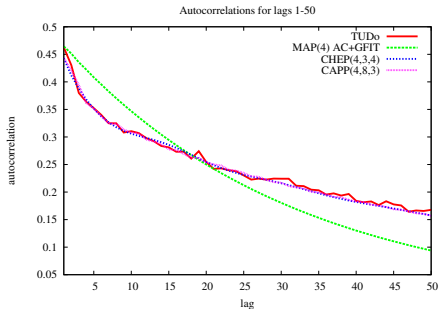


(h) joint moments (2)

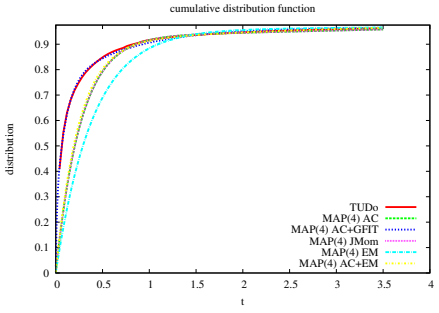
Figure 8.17.: Fitting results for the trace *TUDo* and MAPs/PH distributions of order 2



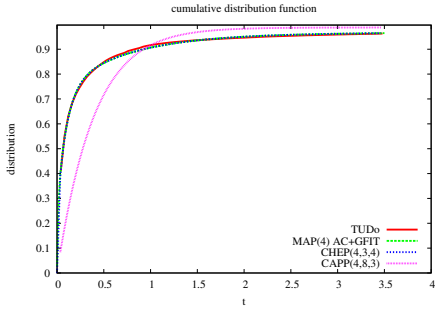
(a) autocorrelations (1)



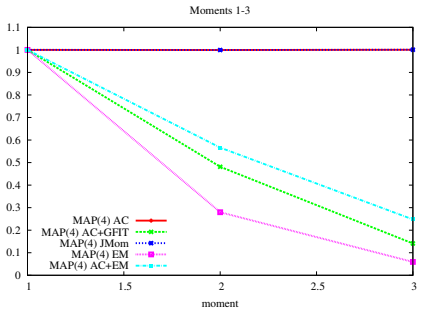
(b) autocorrelations (2)



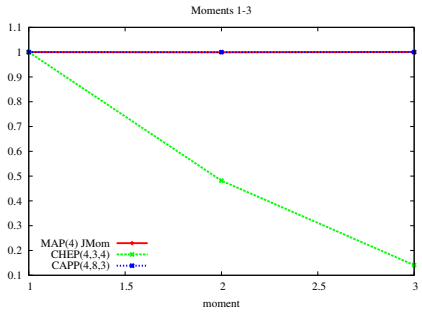
(c) cumulative distribution function (1)



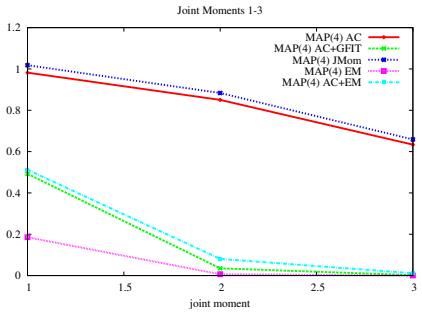
(d) cumulative distribution function (2)



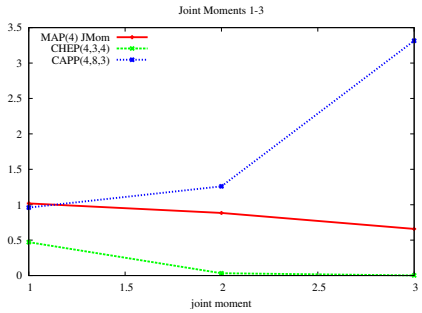
(e) moments (1)



(f) moments (2)

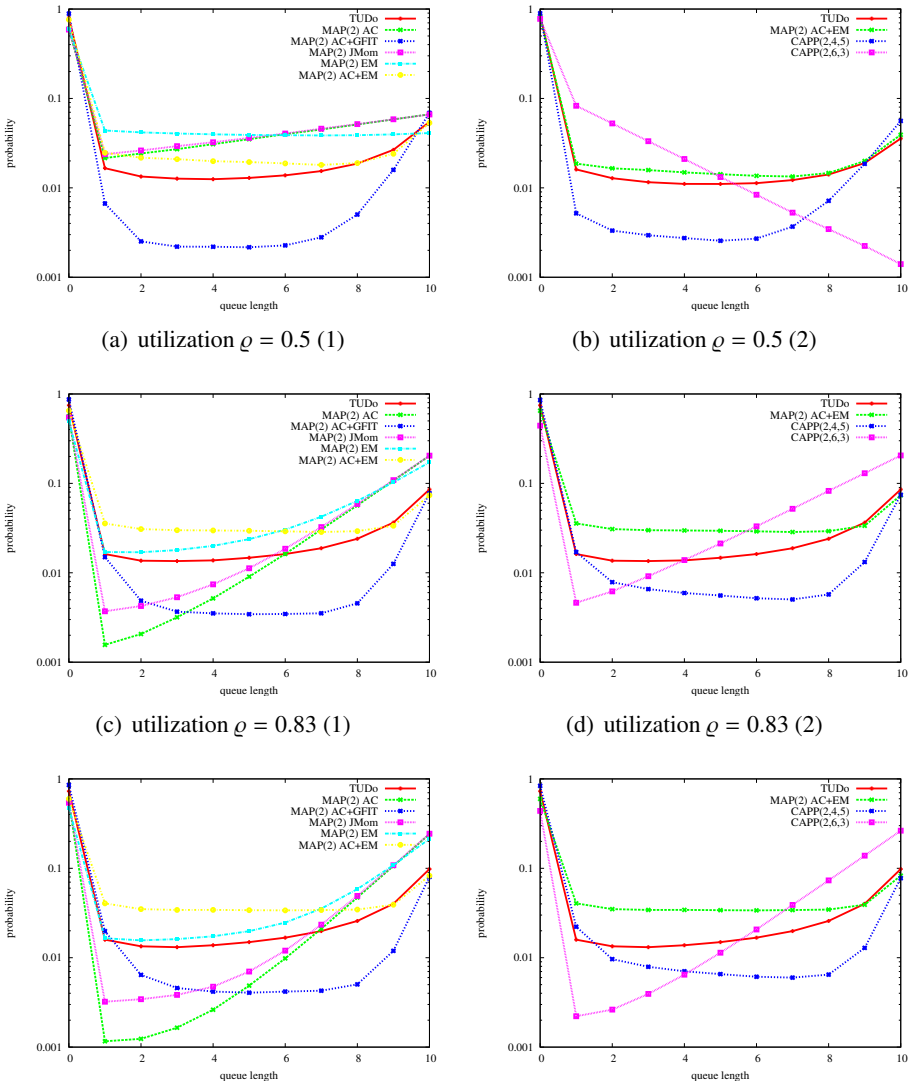


(g) joint moments (1)



(h) joint moments (2)

Figure 8.18.: Fitting results for the trace *TUDo* and MAPs/PH distributions of order 4



(a) utilization $\varrho = 0.5$ (1)

(b) utilization $\varrho = 0.5$ (2)

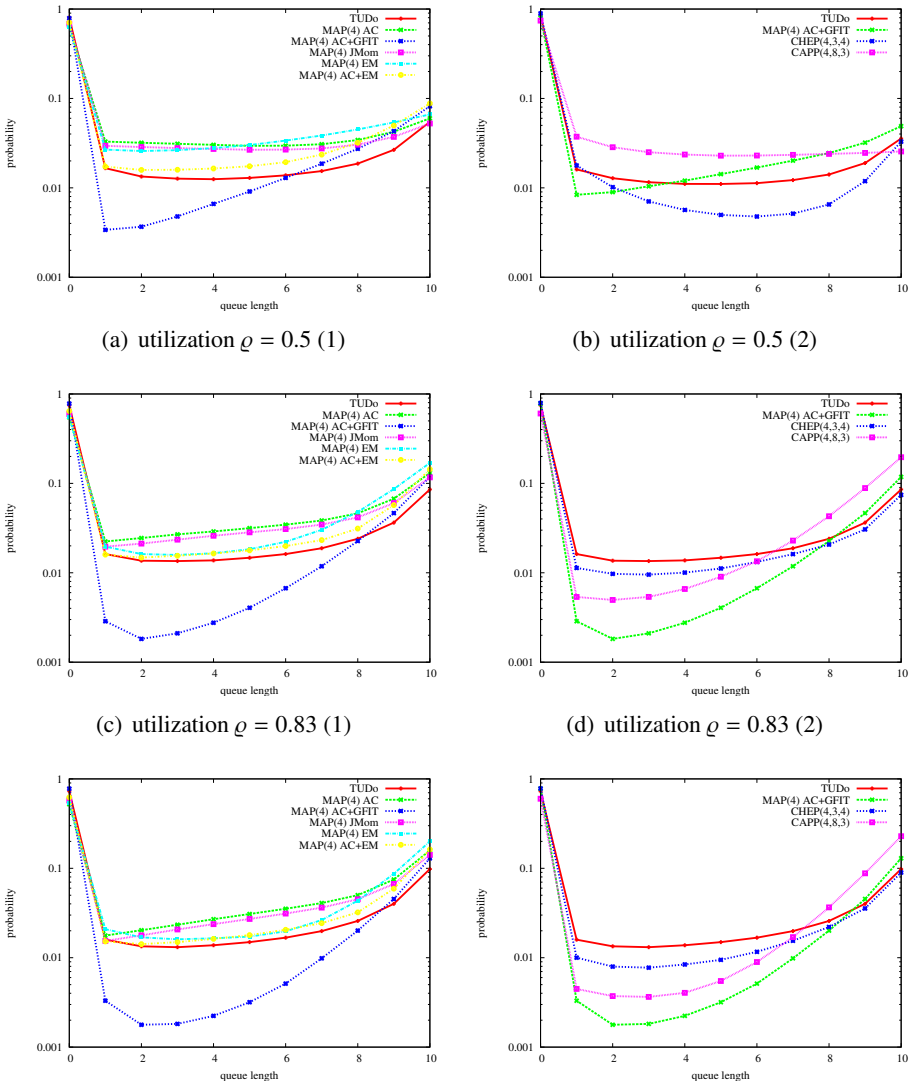
(c) utilization $\varrho = 0.83$ (1)

(d) utilization $\varrho = 0.83$ (2)

(e) utilization $\varrho = 1$ (1)

(f) utilization $\varrho = 1$ (2)

Figure 8.19.: Queueing results for different utilization levels ϱ using the trace *TUDo* and MAPs/PH distributions of order 2



(a) utilization $\varrho = 0.5$ (1)

(b) utilization $\varrho = 0.5$ (2)

(c) utilization $\varrho = 0.83$ (1)

(d) utilization $\varrho = 0.83$ (2)

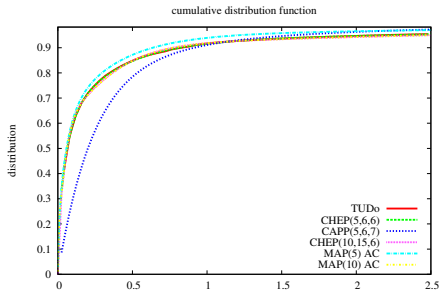
(e) utilization $\varrho = 1$ (1)

(f) utilization $\varrho = 1$ (2)

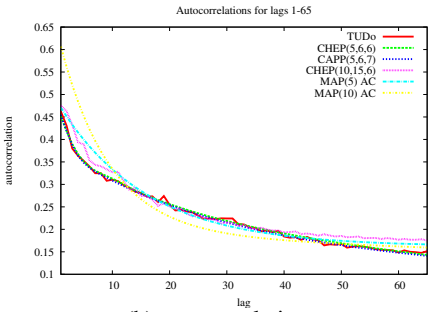
Figure 8.20.: Queueing results for different utilization levels ϱ using the trace *TUDo* and MAPs/PH distributions of order 4

Model	$\varrho = 0.5$	$\varrho = 0.83$	$\varrho = 1$
TUDo	1.32218	1.81573	1.99915
MAP(2) AC	2.60212	3.84889	3.99674
MAP(2) AC+GFIT	0.944084	1.03785	1.08804
MAP(2) JMom	2.63983	3.94447	4.11604
MAP(2) EM	2.1826	3.93407	4.24511
MAP(2) AC+EM	1.44638	2.09979	2.41668
CAPP(2,4,5)	0.946543	1.07986	1.14774
CAPP(2,6,3)	2.24211	4.64881	4.96712
MAP(4) AC	2.10636	3.18556	3.54694
MAP(4) AC+GFIT	1.73081	1.94851	1.99942
MAP(4) JMom	1.8686	2.84723	3.16111
MAP(4) EM	2.39955	3.45668	3.68806
MAP(4) AC+EM	2.11448	2.73165	2.95057
CHEP(4,3,4)	0.940312	1.5315	1.70222
CAPP(4,8,3)	2.48226	3.44748	3.61095

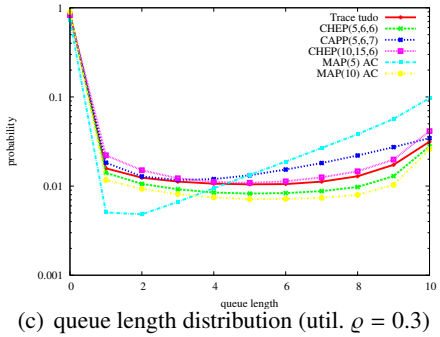
Table 8.3.: Mean queue length values for the trace *TUDo* and MAPs/PH distributions of order 2 and 4



(a) cumulative distribution function



(b) autocorrelations



(c) queue length distribution (util. $\varrho = 0.3$)

Figure 8.21.: Results for the trace *TUDo* and MAPs/PH distributions of order 5 – 10

the number of autocorrelation coefficients and the number of ARMA base processes considered.

From the examples it becomes also visible, that the CHEPs and CAPPs offer a larger flexibility for capturing the autocorrelations than MAP fitting algorithms. Note, that in contrast to MAP fitting the approach for constructing the base process of a CAPP is independent of the number of states of the APH distribution. Equation 6.9 only depends on the number of elementary series and once the base process autocorrelation has been determined the minimization of Equation 4.11 is completely independent of the APH order, but depends on the order of the base process and the number of autocorrelation lags to match. Moreover, for fitting MAPs according to the autocorrelation the given APH distribution has a large influence on the possible entries in matrix $\boldsymbol{D}_1$ and therefore on the autocorrelation structure the MAP can exhibit. For a CAPP the distribution only determines the lower and upper bound for the autocorrelation that is possible but due to the flexibility of the ARMA base process it has little influence on the possible structure of the autocorrelation. Hence, CHEP and CAPP fitting is still possible for larger PH marginal distributions of orders greater than 10 for which MAP fitting is problematic (either because of the runtime for EM algorithms or because of the increased complexity of the optimization for autocorrelation fitting). The examples demonstrated that especially for these larger CHEPs a very good approximation of the characteristics of real network traces is possible in terms of the cumulative distribution function, the autocorrelation structure and the queueing behavior.

Chapter 9

Software Support

As mentioned in Chapter 1 there is only little support for stochastic processes in common simulation software. While there exist various fitting tools for distributions only prototype implementations for fitting the different types of stochastic processes (cf. Chapter 2) are available. Moreover, there is hardly any support for actually using those processes in a simulation model. One reason for this surely is that all of the available fitting tools use their own format for process description, which makes it difficult to integrate the processes into existing simulation models.

These issues have been partially addressed with the development of `ProFiDo` [14], a toolkit that provides a graphical user interface for commandline-based fitting tools and a common XML-based interchange format for stochastic processes. Yet missing is an easy way to integrate those fitted processes into simulation models.

The contribution presented in this chapter is twofold. First, `ProFiDo` is extended to support the processes presented in Chapters 4 - 6. Second, a module for the open-source simulation environment `OMNeT++` [78] is presented that allows for an easy integration of stochastic processes into `OMNeT++` simulation models.

In Section 9.1 a simple commandline-based tool, called `CAPP-Fit`, is introduced that implements the algorithm from Chapter 7 for fitting of extended ARTA, CHEP and CAPP models. Section 9.2 gives a short overview of the fitting toolkit `ProFiDo` and describes the integration of `CAPP-Fit`. While these two sections deal with software support for the fitting of stochastic processes Section 9.3 treats the use of those processes in simulation models by introducing a module for stochastic processes for `OMNeT++`. Finally, in Section 9.4 the complete `ProFiDo` framework for fitting and simulating stochastic processes is summarized.

9.1. CAPP-Fit

`CAPP-Fit` is a simple commandline-based tool that implements the fitting algorithms outlined in Chapter 7. It expects a marginal distribution F_Y, a trace $\mathcal{T}$, the number r of autocorrelation lags to consider for fitting and the base process order (p, q) as input. It is also possible to define an interval for the base process order, such that base processes for all combinations of $p \in [p_{min}, p_{max}]$ and $q \in [q_{min}, q_{max}]$ are fitted and the best base process is selected.

CAPP-Fit can load the marginal distribution from ProFiDo's XML format (see Section 9.2) or import from the tools GFIT or Momfit, which implement the fitting algorithms described in Section 5.3 and Section 6.2, respectively. Depending on the type of marginal distribution CAPP-Fit automatically decides which type of process to fit. For uniform, exponential, Weibull, triangular, normal, lognormal, Johnson, gamma, Erlang and χ^2 distributions an (extended) ARTA model is fitted by an implementation of the Algorithm 7.2. For Hyper-Erlang distributions a CHEP is fitted. For general acyclic PH distributions and for Hyper-Erlang distributions that cannot achieve the desired autocorrelation a CAPP is fitted as described in Algorithm 7.3.

The implementation of CAPP-Fit requires several numerical procedures, which are available from the literature.
For fitting CHEPs and CAPPs the bivariate normal integral has to be computed to obtain the main process autocorrelation for a given base process autocorrelation (cf. Equations 5.6 and 6.9) which is done using the algorithm from [60]. For determining the bounds of the bivariate double integral the inverse of the normal cumulative distribution function is computed using the algorithm from [163]. Fitting (extended) ARTA models requires several procedures that are mostly related to computing the inverse cumulative distribution function of the supported distributions. For normal, lognormal and Johnson distributions this can be done using the algorithm from [163]. For the related distributions gamma, Erlang and χ^2 an algorithm from [20] is available. Additionally, it is necessary to transform the z_t in Equation 4.3 using the normal cumulative distribution function for which a procedure is available in [77]. To compute the double integral in Equation 4.3 a general integration procedure is required, e.g. Romberg integration [8] can be used.
For fitting the ARMA base process to a set of given autocorrelation coefficients as described in Section 4.2 the Nelder-Mead algorithm [117] is used.

9.2. ProFiDo

ProFiDo (Processes Fitting Toolkit Dortmund) [12, 14] is a toolkit written in Java to allow for a consistent use of commandline-based fitting tools for various stochastic processes and distributions presented in Chapter 2. See [12] for an overview of the fitting approaches that are currently supported. ProFiDo provides the framework for graphically modeling workflows that consist of nodes each associated with a fitting tool and that are connected by arcs to specify model and result propagation. Workflows can be specified stepwise by placing input nodes (e.g. traces), job nodes (tools for fitting or result visualization) and output nodes (e.g. the fitted models or plots of their characteristics) on a canvas and by connecting them to determine the flow of results and fitted models from one tool to the other. Since most fitting tools use different input and output formats, an XML-based interchange format has been developed for ProFiDo to enable a consistent data flow between job nodes and converter scripts are responsible for transforming the XML format into the tool's format and vice versa. Workflows can be exported into a shell script to be executed.

This makes ProFiDo a suitable candidate for integration of the commandline-based tool CAPP-Fit introduced in Section 9.1. For this integration ProFiDo has to be modified in two ways. First, ProFiDo's XML interchange format [16] has to be extended

Listing 9.1: XML specification of extended ARTA models

```
<?xml version="1.0"?>
<profido>
  <arta>
    <dist>
      ...
    </dist>
    <arima>
      <arcount> ... </arcount>
      <ar> ... </ar>
      <macount> ... </macount>
      <ma> ... </ma>
      <d> 0.0 </d>
      <variance> ... </variance>
      <mean> 0.0 </mean>
    </arima>
  </arta>
</profido>
```

to be capable to express extended ARTA models, CHEPs and CAPPs. Second, the actual tool has to be integrated into the framework by providing the necessary XML configuration [17] that ProFiDo uses to determine the attributes of the tool and the commandline call.

In the reminder of this section the extensions to the XML interchange format are introduced first. Afterwards the integration of CAPP-Fit is explained and an example workflow is given.

9.2.1. XML Interchange Format

The general outline of the XML specification for extended ARTA models can be seen in Listing 9.1. It contains a description of the distribution within the tags <dist> and </dist> and a description of the base process within the tags <arima> and </arima> that contains the AR and MA coefficients and the variance of the innovations. The mean of the base process and the degree of differencing are 0.0 by definition of the base process. The specification of the extended ARTA process is similar to the ARTA description presented in [16] except that MA coefficients are allowed. [16] also contains the XML descriptions of various distributions that can be used.

Listing 9.2 shows the complete XML specification of an example extended ARTA model with exponential marginal distribution with rate 1.0 and $ARMA(6,3)$ base process with AR coefficients

$$\boldsymbol{\alpha} = (0.840036, 0.824413, -0.712854, 0.00607279, 0.063807, -0.0324686)$$

and MA coefficients

$$\boldsymbol{\beta} = (-1.78786, -0.626879, 1.48718).$$

ProFiDo's XML interchange format already allows for the specification of Hyper-Erlang distributions and ARMA processes [16]. Both descriptions can be combined

Listing 9.2: Example XML specification of extended ARTA model

```
<?xml version="1.0"?>
<profido>
  <arta>
    <dist>
      <expodist>
        <mean> 1.0 </mean>
      </expodist>
    </dist>
    <arima>
      <arcount> 6 </arcount>
      <ar> 0.840036 0.824413 -0.712854 0.00607279 0.063807
          -0.0324686 </ar>
      <macount> 3 </macount>
      <ma> -1.78786 -0.626879 1.48718 </ma>
      <d> 0.0 </d>
      <variance> 0.303949 </variance>
      <mean> 0.0 </mean>
    </arima>
  </arta>
</profido>
```

easily to define a CHEP in XML format as shown in Listing 9.3.

A CHEP description is started by the tag <chep> and ended by </chep>. It contains the definition of a Hyper-Erlang distribution consisting of a vector of initial probabilities, a vector for the number of phases and a vector for the rates of each branch. The values in each vector are separated by blanks. The description of the ARMA base process is identical to the description of the base process of extended ARTA models. An example for the full XML specification of a $CHEP(3,5,3)$ is given in Listing 9.4.

The XML specification consists of the definition of a Hyper-Erlang distribution with 3 states distributed among 2 branches with 1 and 2 states, respectively, and an $ARMA(5,3)$ base process.

The description of CAPPs is similar to the description of CHEPs. Its is contained within the tags <capp> and </capp>. Instead of a Hyper-Erlang distribution an acyclic PH distribution is used. ProFiDo's XML specification [16] already accounts for the description of PH distributions consisting of the number of states, the initial probability vector $\boldsymbol{\pi}$ and the transition rate matrix $\boldsymbol{D}_0$. The ARMA base process is defined in the same way as for CHEPs. The general outline of a CAPP definition is shown in Listing 9.5.

Listing 9.6 shows the full XML description of a $CAPP(5,5,7)$ as example.

The model consists of an acyclic PH distribution $(\boldsymbol{\pi}, \boldsymbol{D}_0)$ with 5 states and

$$\boldsymbol{D}_0 = \begin{bmatrix} -0.327 & 0.002 & 0.029 & 0.011 & 0.014 \\ 0 & -0.348 & 0.030 & 0.012 & 0.016 \\ 0 & 0 & -0.691 & 0.153 & 0.004 \\ 0 & 0 & 0 & -1.084 & 0.640 \\ 0 & 0 & 0 & 0 & -1.961 \end{bmatrix} \quad \text{and}$$

Listing 9.3: XML specification of CHEPs

```
<?xml version="1.0"?>
<profido>
  <chep>
    <dist>
      <hypererlangdist>
        <prob> ... </prob>
        <phases> ... </phases>
        <rates> ... </rates>
      </hypererlangdist>
    </dist>
    <arima>
      <arcount> ... </arcount>
      <ar> ... </ar>
      <macount> ... </macount>
      <ma> ... </ma>
      <d> 0.0 </d>
      <variance> ... </variance>
      <mean> 0.0 </mean>
    </arima>
  </chep>
</profido>
```

Listing 9.4: Example XML specification of a $CHEP(3,5,3)$

```
<?xml version="1.0"?>
<profido>
  <chep>
    <dist>
      <hypererlangdist>
        <prob> 0.091 0.909 </prob>
        <phases> 1 2 </phases>
        <rates> 0.254050 2.838471 </rates>
      </hypererlangdist>
    </dist>
    <arima>
      <arcount> 5 </arcount>
      <ar> 0.891436 0.552937 -0.424392 0.0488508 -0.0780046
         </ar>
      <macount> 3 </macount>
      <ma> -0.258672 -0.36477 -0.209127 </ma>
      <d> 0.0 </d>
      <variance> 0.218322 </variance>
      <mean> 0.0 </mean>
    </arima>
  </chep>
</profido>
```

Listing 9.5: XML specification of CAPPs

```
<?xml version="1.0"?>
<profido>
  <capp>
    <ph>
      <states> ... </states>
      <pi> ... </pi>
      <d0> ... </d0>
    </ph>
    <arima>
      <arcount> ... </arcount>
      <ar> ... </ar>
      <macount> ... </macount>
      <ma> ... </ma>
      <d> 0.0 </d>
      <variance> ... </variance>
      <mean> 0.0 </mean>
    </arima>
  </capp>
</profido>
```

Listing 9.6: Example XML specification of a $CAPP(5, 5, 7)$

```
<?xml version="1.0"?>
<profido>
  <capp>
    <ph>
      <states> 5 </states>
      <pi> 0.00976 0.04418 0.27993 0.01426 0.65197 </pi>
      <d0> -0.327 0.002 0.029 0.011 0.014
        0 -0.348 0.030 0.012 0.016
        0 0 -0.691 0.153 0.004
        0 0 0 -1.084 0.640
        0 0 0 0 -1.961
      </d0>
    </ph>
    <arima>
      <arcount> 5 </arcount>
      <ar> 0.39233 0.464275 0.33436 -0.061696 -0.13593 </ar>
      <macount> 7 </macount>
      <ma> 1.37963 -5.17508 -1.56516 0.179254 2.19242 1.49209
          1.10682 </ma>
      <d> 0.0 </d>
      <variance> 0.0191485 </variance>
      <mean> 0.0 </mean>
    </arima>
  </capp>
</profido>
```

Listing 9.7: XML configuration for CAPP-Fit

```
<program>
  <general>
    <name>CAPP-Fit</name>
    [...]
  </general>
  [...]
  <parameterlist>
    <parametergroup visible="true" type="text">
      <name>Autocorrelations</name>
      <description>Number of autocorrelation lags to match.
        </description>
      <key>-ac=</key>
      <default>30</default>
      <parameter type="static">
        <name>ac</name>
        <description>Number of autocorrelation lags.
          </description>
        <successor></successor>
      </parameter>
      <successor>min. AR order</successor>
    </parametergroup>
    [...]
  </parameterlist>
</program>
```

$$\boldsymbol{\pi} = (0.00976, 0.04418, 0.27993, 0.01426, 0.65197)$$

and an $ARMA(5,7)$ base process with AR coefficients

$$\boldsymbol{\alpha} = (0.39233, 0.464275, 0.33436, -0.061696, -0.13593)$$

and MA coefficients

$$\boldsymbol{\beta} = (1.37963, -5.17508, -1.56516, 0.179254, 2.19242, 1.49209, 1.10682).$$

9.2.2. Integration of CAPP-Fit

The integration of `CAPP-Fit` as a new fitting tool into `ProFiDo` is straightforward and suggests itself. The recommended fitting tools for the marginal distribution, i.e. `GFIT` and `Momfit`, are already supported by `ProFiDo`. Since `CAPP-Fit` natively supports `ProFiDo`'s XML format no converter scripts are necessary to integrate `CAPP-Fit` into the framework. Thus, for incorporating `CAPP-Fit` only the XML description of the tool parameters has to be added into `ProFiDo`'s XML configuration file.

Listing 9.7 shows an excerpt from this XML specification with the description for the parameter that defines the number of lag-k autocorrelation coefficients to consider. The specification consists of the name of the parameter, a description that explains the meaning of the parameter, a key that is used when calling the commandline tool `CAPP-Fit` and a default value. Further parameters like the base process order (p, q),

which are not shown in Listing 9.7, are defined in a similar way. The attribute window that ProFiDo generates from the XML description to enter the tool specific properties is shown in Figure 9.1. Figure 9.2 shows an example workflow in ProFiDo that uses

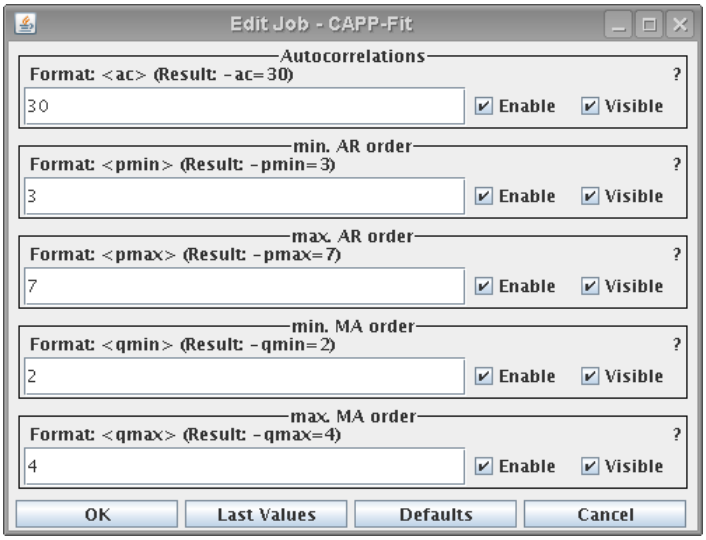


Figure 9.1.: Attribute window of the CAPP-Fit job node in ProFiDo

the new CAPP-Fit job node. The workflow consists of an input node, which loads a trace file, five job nodes, which fit different stochastic processes, and three output nodes to save the fitted models. The job nodes have been renamed for the workflow to clarify which type of process is fitted. The first job node, called MAP_EM fits a MAP to the trace using an EM algorithm. The two job nodes CHEP-Fit and ARTA-Fit both realize calls to the newly integrated tool CAPP-Fit. In the first case G-FIT is used to fit an Hyper-Erlang distribution, which is taken by CAPP-Fit as input to fit a CHEP. In a similar way an exponential distribution is fitted in the second case and CAPP-Fit returns an ARTA model with exponential marginal distribution.

9.3. OMNeT++ Arrival Process Module

OMNeT++ [78] is an open source simulation framework that has been developed and extensively used for the modeling of communication protocols and networks. Since data that has been observed from real systems in this area is known to exhibit autocorrelations [130], OMNeT++ is a well suited candidate for an extension to support stochastic processes. Aside from the simulation of lower-level communication networks OMNeT++ has been used in other areas as well, that could benefit from the possibility to model correlated data. E.g. in [13] OMNeT++ models have been combined with process chains [10] to model Service-Oriented Architectures.

In the following a short overview of the general OMNeT++ model structure is given. Afterwards the concept and implementation of the OMNeT++ Arrival Process Module is presented. A description of an earlier version of the Arrival Process Module for OMNeT++ can be found in [101].

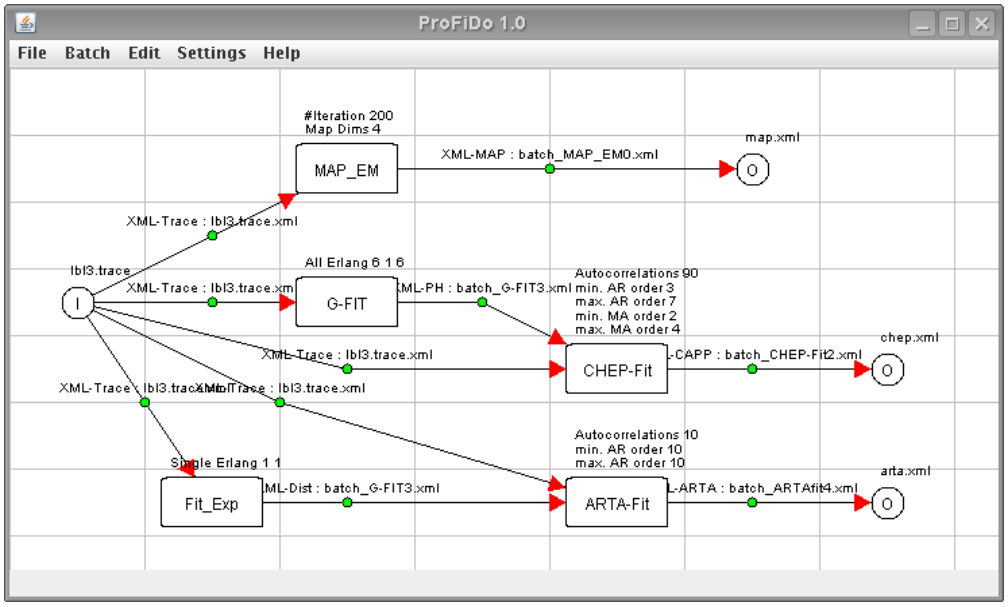


Figure 9.2.: Example workflow using the new `CAPP-Fit` job node

`OMNeT++` models consist of so called modules that can be arranged in a hierarchical way. Modules communicate via gates with message passing. One can distinguish between simple modules, which are the basic building blocks of a model, and compound modules. Simple modules are written in C++ accompanied by a so called NED-description that defines the interface of the module consisting of gates and parameters, which pass configuration data to the simple module. Compound modules are a grouping of simple modules and other compound modules. Compound modules have no behavior of their own, i.e. they are fully described by a NED-description of the submodules and their connections. The general structure of an `OMNeT++` model is shown in Figure 9.3. `OMNeT++` provides a graphical editor for arranging modules into

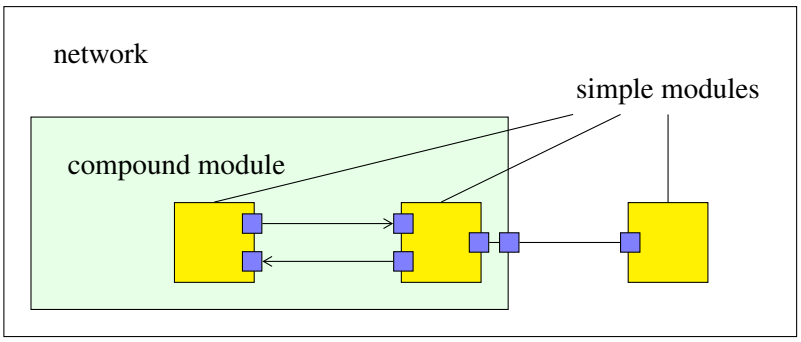


Figure 9.3.: `OMNeT++` model structure (from [78])

one model and the capability for a visualization of the simulation.

The `OMNeT++ Arrival Process Module` developed in this work is a simple mod-

ule that can generate random numbers from Markovian Arrival Processes, (extended) ARTA processes, ARMA processes and CHEP and CAPP models. The model description is parsed from a file in the `ProFiDo` XML interchange format as described in Section 9.2. In this way, the process description derived by `ProFiDo` with some fitting algorithm can be used in an `OMNeT++` simulation model without any additional programming effort.

A simplified class diagram of the `Arrival Process Module` is shown in Figure 9.4.

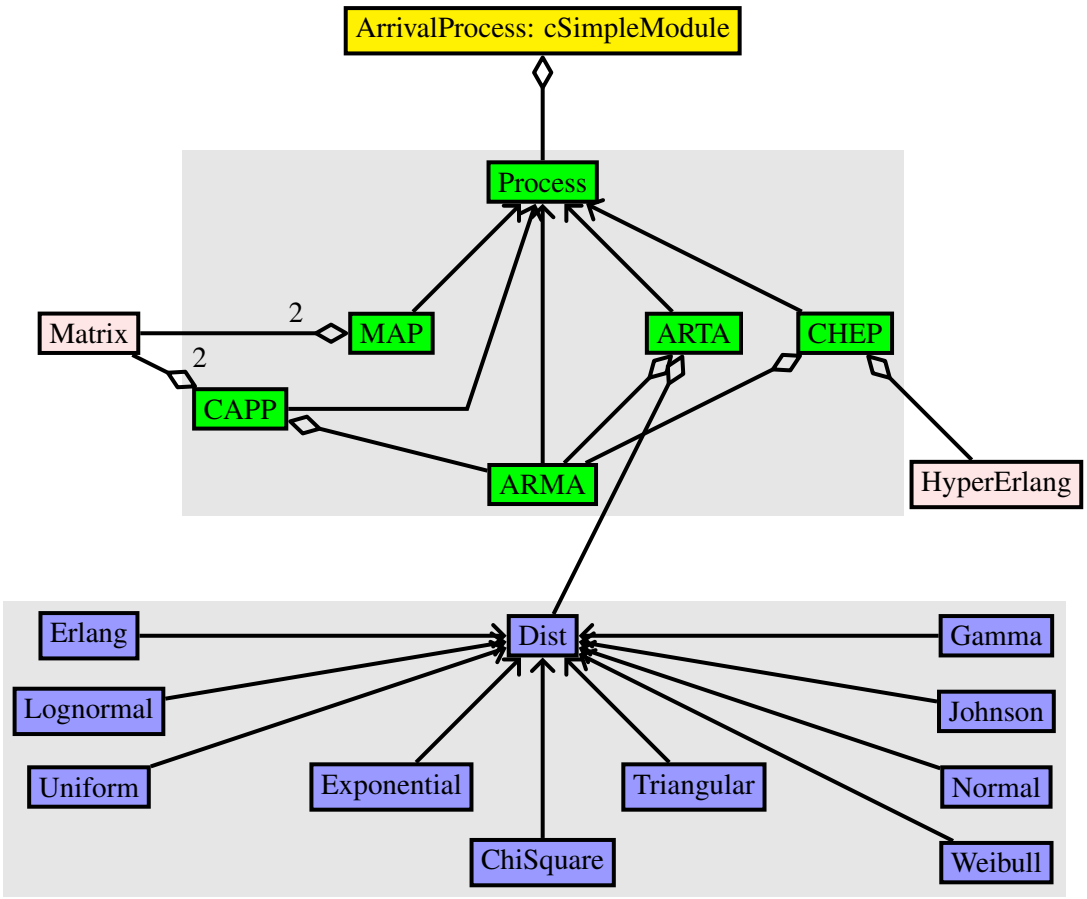


Figure 9.4.: Class hierarchy of the `ArrivalProcess` `OMNeT++` module

The class `ArrivalProcess` contains the C++ implementation of the module. It inherits from `cSimpleModule` as all simple modules in `OMNeT++` do. The corresponding NED-description is shown in Listing 9.8. The module only has one XML type parameter that is used to pass the model description, one string parameter that can be used to specify a function that can be used for a transformation of the time series (cf. Section 9.3.6) and one display parameter that configures the icon of the module shown in the editor and the simulation visualization. The only gate of the module is an output gate. Whenever an arrival according to the stochastic process has occurred a message

Listing 9.8: NED description of the Arrival Process module

```
simple ArrivalProcess
{
 parameters:
  xml model;
  string transform = default("");
  @display("i=block/source");
 gates:
  output out;
}
```

is passed through this gate to another module that should be connected to this gate with one of its input gates and processes the incoming arrival.

The implementation of the class is actually quite simple, since all the code necessary for generating random numbers from the processes is contained in the classes `Process` and its subclasses shown in Figure 9.4. The class `ArrivalProcess` provides methods for loading the process description from an XML file and initializes a `Process`. Additionally it has to provide a method `handleMessage()` for dealing with message events. `Process` is an abstract class that all the actual stochastic processes like MAPs or ARTA processes inherit from. All inheriting classes have to implement a method called `getNextRandomVariate()` for generating the next random number from that process. Hence, `handleMessage()` from the class `ArrivalProcess` schedules the next arrival according to `getNextRandomVariate()`. When the arrival is due, a message is sent to the outgoing gate and the next arrival is scheduled according to the result of `getNextRandomVariate()` by initiating a self-message that arrives at the generation time of the next message. In the following a brief summary of the classes inheriting from `Process`, which perform the random number generation is given.

9.3.1. Class MAP

The class `MAP` is used for simulating Markovian Arrival Processes. Of course it can be used for the simulation of PH distributions as well, if the PH distribution is transformed into a MAP. It uses the class `Matrix` as an utility class to store the two matrices $\boldsymbol{D}_0$ and $\boldsymbol{D}_1$. Additionally, a variable to store the current state is required. A MAP is initialized by drawing the initial state from the distribution defined by $\boldsymbol{\pi}$ that contains the stationary distribution just after an arrival has occurred. Then the class simulates the underlying Markov Chain by drawing an exponentially distributed random number with the rate $-\boldsymbol{D}_0(i,i)$ corresponding to the current state i to determine the next transition time and an uniformly distributed random number to compute the next state according to the probability distribution defined by $\boldsymbol{D}_0(i,j)/(-\boldsymbol{D}_0(i,i))$ and $\boldsymbol{D}_1(i,j)/(-\boldsymbol{D}_0(i,i))$. This is repeated until a transition from $\boldsymbol{D}_1$ occurs. In this case the sum of the transition times is returned and the current state is stored as starting point for the generation of the next random sample. For the simulation it is necessary to draw random numbers from exponential and uniform distribution, which is done using the random number generators of `OMNeT++`.

9.3.2. Class ARMA

The simulation of $ARMA(p,q)$ processes is realized by the class `ARMA`. This class mainly serves as implementation of the base process for ARTA, CHEP and CAPP models and because of the problems and issues mentioned in Chapter 3 that arise when simulating ARMA models it is not recommended to use this process type as input model for a simulation model, although it is possible of course. If using an $ARMA(p,q)$ process as input model for interarrival times this should be done in combination with the specification of a function for transformation of the time series (cf. Listing 9.8 and Section 9.3.6).

The class manages four lists for the p AR coefficients, the q MA coefficients, the last p observations and the last q innovations. The former two lists are fixed after initialization while the latter two have to be updated after each generation of an element of the time series.

For initialization the p previous observations and the q previous innovations have to be determined. For the innovations this is straightforward, because they are iid random numbers, and consequently the $\epsilon_{t-1}, \ldots, \epsilon_{t-q}$ can be drawn from a normal distribution with zero mean and variance σ_ϵ^2. The p previous observations are correlated and have to be initialized by drawing from a multivariate normal distribution. A procedure for this task is for example given in [104]. Let

$$\boldsymbol{\Sigma} = \begin{bmatrix} \gamma_0 & \gamma_1 & \cdots & \gamma_{p-1} \\ \gamma_1 & \gamma_0 & \cdots & \gamma_{p-2} \\ \vdots & \vdots & \ddots & \vdots \\ \gamma_{p-1} & \gamma_{p-2} & \cdots & \gamma_0 \end{bmatrix}$$

be the covariance matrix with the autocovariance values $(\gamma_0, \gamma_1, \ldots, \gamma_{p-1})$ computed from the $ARMA(p,q)$ process and let $\hat{\boldsymbol{\mu}} = \mathbf{0}$ be the mean value for the multivariate normal distribution[5]. We draw p independent random numbers $\boldsymbol{X} = (X_0, X_1, \ldots, X_{p-1})^T$ from a standard normal distribution and compute the lower triangular matrix $\boldsymbol{C}$ with $\boldsymbol{\Sigma} = \boldsymbol{C}\boldsymbol{C}^T$. Then, we can generate the multivariate normal random numbers that are used to initialize the $Z_{t-i}, i = 1, \ldots, p$ by setting $\boldsymbol{Z} = \hat{\boldsymbol{\mu}} + \boldsymbol{C}\boldsymbol{X}$. See [37] and Section 4.2 for a method to compute the autocovariance values of an ARMA process and [67] for a simple method to perform the Cholesky decomposition to obtain matrix $\boldsymbol{C}$.

For simulation the class draws a normally distributed random number as new innovation and computes the weighted sum according to Equation 2.3 to obtain Z_t. Finally, the oldest observation and the oldest innovation are replaced by the newly computed observation and the new innovation, respectively, and $Z_t + \mu$ is returned as next interarrival time.

9.3.3. Class ARTA

The class `ARTA` provides methods for simulating ARTA and extended ARTA models. The implementation is the same for both types of processes, since they only differ in the type of the base process, which is handled by the class `ARMA`.

[5] Recall from Section 2.1 that we defined the Z_t of an $ARMA(p,q)$ process to have zero mean, i.e. the $Z_t = \tilde{Z}_t - \mu$ are deviations from the mean μ of the real process.

The ARTA class has two objects, one of type ARMA and one of type Dist that are required to perform the simulation of the ARTA process. Recall the transformation steps for an ARTA process from Figure 2.3 from which the implementation for simulating the model can directly be deduced. The ARMA class (which either is an $AR(p)$ or an $ARMA(p, q)$ process) is used to generate a normally distributed random number. This random number is transformed twice to have uniform distribution and using the inverse cdf to yield the final ARTA random number. The class Dist is an abstract superclass with the virtual method inverse_cdf(double x) that all the distributions from Figure 9.4 have to implement to compute the inverse cdf for the last transformation step. For simulation of ARTA processes only the ARMA base process has to be initialized as described above, no further initialization step is necessary.

The implementation of the ARTA class requires several numerical methods for which fast approaches exist in the literature (cf. Section 9.1). For the transformation from normally to uniformly distributed random numbers in Figure 2.3 the computation of the normal cdf is necessary, which is done by using the approach from [77]. All other required numerical methods are related to the computation of the inverse cumulative distribution function: The inverse cdf of the normal distribution is computed using the algorithm from [163]. The same holds for the inverse cdf of the lognormal and Johnson distributions, which are both derived from the normal distribution. The algorithm from [20] is used to compute the inverse cdf of gamma, Erlang and χ^2 distributions, which are all related. For the uniform, exponential, Weibull and triangular distributions closed-form expressions exist for the inverse cdf [104].

9.3.4. Class CHEP

The class CHEP implements the simulation of Correlated Hyper-Erlang Processes. It contains two objects, the marginal distribution is realized by an object from the class HyperErlang, which is responsible for sorting the branches according to their mean values, and the base process by an object from the class ARMA. The base process is used to generate a normally distributed random number Z_t. This random number is transformed to have uniform distribution, i.e. $U_t = \Phi(Z_t)$. The correlated U_t are used to select a branch of the Hyper-Erlang distribution according to the distribution $\boldsymbol{\tau}$ that defines the probabilities of the branches. Finally, a random number drawn from the Erlang distribution of the selected branch is returned. Random number generation from an Erlang distribution is provided by OMNeT++'s internal random number generator. The class CHEP does not require an initialization step, only the ARMA base process has to be initialized as described for the class ARMA.

9.3.5. Class CAPP

The class CAPP provides the functionality for simulating Correlated Acyclic Phase-type Processes. It uses two objects of the class Matrix to store the vector $\boldsymbol{\pi}$ and matrix $\boldsymbol{D}_0$ of the APH marginal distribution. The base process is realized by the class ARMA. As preparation the class CAPP computes the elementary series of the APH distribution and sorts them according to their mean values resulting in a list of the rates and the probability for each elementary series. Simulation is similar as for CHEPs. The base process is used to generate a normally distributed random number, which is transformed into

a uniform distribution and used for determining an elementary series. Then the class CAPP has to draw a random number from the corresponding Hypo-Exponential distribution. Since Hypo-Exponential distributions are not directly supported by OMNeT++, the class CAPP draws n exponentially distributed random numbers with rates $\lambda_1, \lambda_2, \ldots, \lambda_n$ and returns the sum of these values for a Hypo-Exponential distribution defined by the rates $\mathbf{\Lambda} = (\lambda_1, \lambda_2, \ldots, \lambda_n)$. Again, only the ARMA base process has to be initialized for simulation.

9.3.6. Postprocessing the Time Series

In some cases it might be desirable or even necessary to perform a postprocessing of the time series that has been generated by a stochastic process. This might for example be the case if the input process has been fitted to a trace that uses a different time scale than the rest of the model. For these models one could adjust the complete process to the desired time scale or transform the generated values of the model in a postprocessing step. Another reason for a transformation of the generated interarrival times are processes that might output invalid values. This is the case for $ARMA(p, q)$ processes or might be the case for ARTA processes with marginal distributions for which $F_Y(x) > 0$ for $x < 0$. For these types of processes a transformation can be used to treat or avoid invalid values, e.g. the linear and non-linear transformations mentioned in Section 2.4 or the approaches used in Chapter 3.

The OMNeT++ ArrivalProcess Module accounts for these requirements by providing the possibility to enter a postprocessing function using OMNeT++'s NED language expressions. NED language expressions have a C-like syntax and may make use of various mathematical functions. A list with possible functions can be found in [126]. E.g. using the postprocessing function `fabs($ARRIVAL)` will always use the absolute values of the generated interarrival times. Note, that $ARRIVAL is a placeholder and the ArrivalProcess Module will replace every instance of $ARRIVAL in the expression with the current value obtained from simulating the stochastic process when the expression is evaluated. Non-linear transformations from Section 2.4 could be realized by the postprocessing function `pow(c_1, $ARRIVAL * (1-c_2)/(c_2+c_3))` for some given constants `c_1`, `c_2` and `c_3`.

9.3.7. Example Models

In the following it will be shown how the ArrivalProcess module can be incorporated into different OMNeT++ models by two application examples. The results obtained from these simulation models support the observation that the negligence of autocorrelation may have serious impact on the simulation results.

Figure 9.5 shows a simple queueing model. The model consists of the OMNeT++ ArrivalProcess module feeding a single server queue with a capacity of 10. We used different configurations of the model by generating interarrival times according to a MAP of order 4, according to an ARTA model with exponential marginal distribution and according to a CHEP with Hyper-Erlang marginal distribution with 6 states.

OMNeT++ models can be parameterized by a textual configuration file, usually called `omnetpp.ini`. This file includes parameters of the model that are subject to frequent changes for different simulation runs, i.e. interarrival time distributions, service times

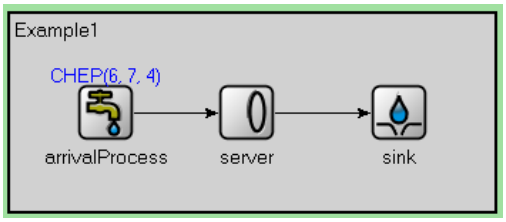


Figure 9.5.: Simple OMNeT++ model with ArrivalProcess module

Listing 9.9: Example configuration file

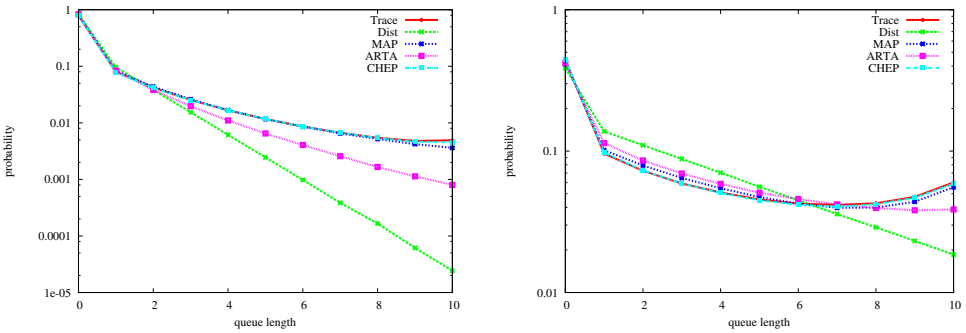
```
[General]
  network = Example1
  **.server.serviceTime = exponential(0.4s)
  **.server.buffer = 10

[Config MAP]
  description = "MAP Arrivals"
  **.arrivalProcess.model = xmldoc("map.xml")

[Config ARTA]
  description = "ARTA Arrivals"
  **.arrivalProcess.model = xmldoc("arta.xml")

[Config CHEP]
  description = "CHEP Arrivals"
  **.arrivalProcess.model = xmldoc("chep.xml")
```

etc. Moreover, OMNeT++ allows for the specification of alternative configurations for one model from which the desired configuration can be chosen when simulating the model. For the ArrivalProcess module the configuration file is used to enter a model file with the process description and optionally to enter a function for postprocessing the generated time series. The OMNeT++ configuration file can be used to define alternative configurations, one for each of the processes, to be used with the same model. Listing 9.9 shows the configuration file for the model from Figure 9.5.

It contains four sections, one for the default configuration and three alternative configurations for different stochastic processes. The first one, named General, is the default and its entries are used for all other three configurations. They determine the network name, the service time and the queue capacity of the server, which is the same in all cases. The remaining three configurations are used to initialize the ArrivalProcess Module with three alternative stochastic processes to generate the arrivals in the model. Each of them consists of a description and a file containing the stochastic process in ProFiDo's XML interchange format, which have been generated by executing the workflow from Figure 9.2. Note, that the stochastic process used by the ArrivalProcess module can easily be exchanged by specifying another XML description using the parameter **.arrivalProcess.model. Using the configuration from Listing 9.9 the model from Figure 9.5 is executed three times to compare the

simulation results of the different processes.

The model is analyzed with several utilization levels for the server between $\varrho = 0.4$ and $\varrho = 0.8$ and the queue length distribution is taken as result measure. The three mentioned stochastic processes have been obtained by fitting MAPs, ARTA and CHEP models to the trace *LBL-TCP-3* [130] from the Internet Traffic Archive [148]. For comparison we simulated the same setup with a slightly modified model that reads the interarrival times of the jobs directly from the trace to obtain reference values for the queue length according to a trace driven simulation. Finally, in another series of experiments we used an exponential distribution (i.e. the same distribution that the ARTA model uses as marginal distribution) fitted to the trace as a traffic generator with independent interarrival times resulting in a Poisson input process. Figure 9.6



(a) Utilization $\varrho = 0.4$ (b) Utilization $\varrho = 0.8$

Figure 9.6.: Queue length distribution for the model from Figure 9.5

shows the queue length distribution for the two different utilization values $\varrho = 0.4$ and $\varrho = 0.8$. From Figure 9.6(a) it is clearly visible that for $\varrho = 0.4$ the probabilities of queue length up to 2 are similar for the trace, the three stochastic processes and the distribution. For larger values towards the tail of the queue length distribution the model with uncorrelated inputs significantly underestimates the probabilities. E.g., for queue length 10 the difference between the values of the trace driven simulation and the model with the Poisson input process is between 2 and 3 orders of magnitude. Among the stochastic processes, the MAP and the CHEP provide much better results than the ARTA process, although it should be noted, that both MAP and CHEP use a larger PH distribution with several states and the ARTA process only has an exponential marginal distribution. For a utilization of $\varrho = 0.8$ the stochastic processes perform better for almost all values of the queue length distribution as shown in Figure 9.6(b). Again, MAP and CHEP give better results than the ARTA process.

As a second example we modified the `NClients` model that is part of the INET Framework to use the `ArrivalProcess` module. Figure 9.7 gives an overview of the network that consists of four client hosts connected to a server via different routers. Figure 9.8 shows the inner view of one of those hosts. The host consists of a module for the TCP protocol implementation, a module for the network layer and several interfaces, which are taken from the implementation of the `StandardHost` of the INET framework. TCP packets are generated according to the events created by the `ArrivalProcess` module. Since the `ArrivalProcess` module only generates

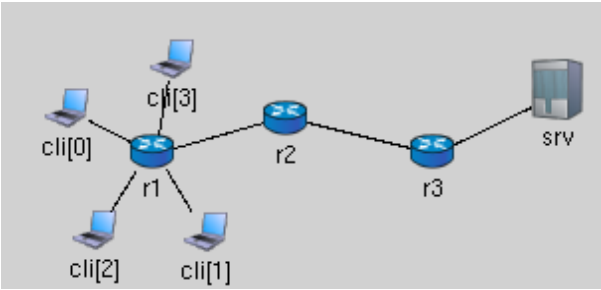


Figure 9.7.: Example model with simple network

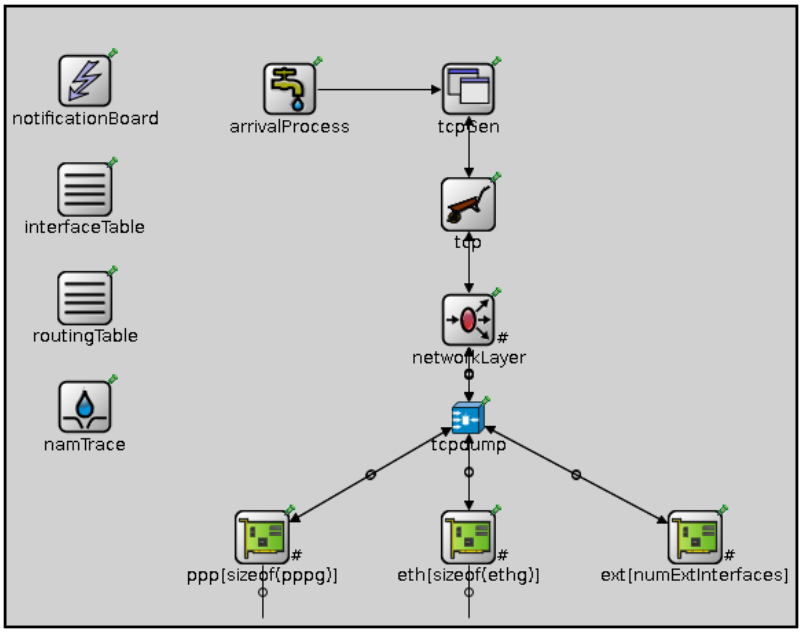
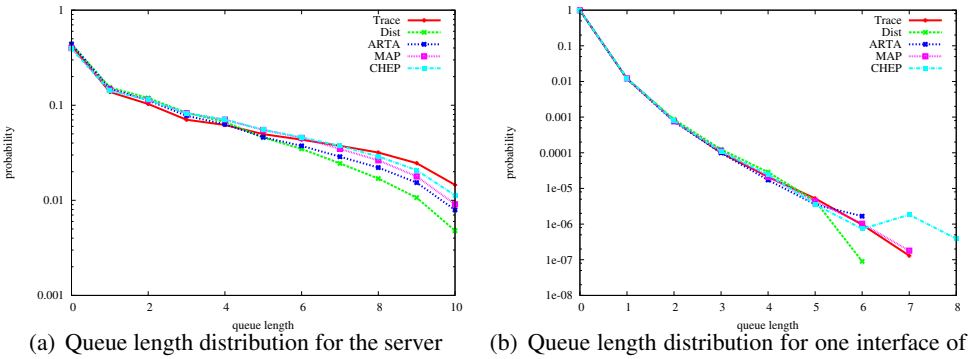


Figure 9.8.: Host from Figure 9.7

correlated events, but is not tailored to a specific protocol implementation, this module is connected to the module `tcpGen`, which serves as an interface between the `ArrivalProcess` module and the modules from the INET framework. `tcpGen` generates a TCP connection and sends packets to the server whenever it is notified by the `ArrivalProcess` module. The server replies with a larger chunk of data to the request of `tcpGen`, i.e. the modules model typical web browsing behavior with a small request to the server and a larger reply by the server. The four `ArrivalProcess` modules are initialized with CHEPs, MAPs and ARTA models with different exponential marginal distributions that all have been fitted to different parts of the trace *BC-pAug89* [130]. For comparison we run the same setup with a slightly altered model where uncorrelated TCP packets are generated according to the exponential distributions that have also been used for the ARTA models. To obtain reference values a trace driven simulation with the original traces was run again. Figure 9.9(a) shows the queue length



(a) Queue length distribution for the server

(b) Queue length distribution for one interface of a router

Figure 9.9.: Queue length distributions for the model from Figure 9.7

distribution of the server's network interface. Again, it is visible that correlated events increase the probability of a larger queue length and that the stochastic processes provide a better approximation of the tail of the queue length distribution than uncorrelated arrivals.

For the first router the load is distributed between several different interfaces connected to the different clients and thus, the effect of the correlated packets is not as significant as for the server. Nevertheless, one can see from Figure 9.9(b) that the maximal queue length increases also for the interfaces of the router.

The two example models demonstrate how stochastic processes can easily be integrated into simulation models using the `ArrivalProcess` module. The second example from Figure 9.7 shows that the module can be incorporated into existing network models with only slight modifications of the model and thus, it can act jointly with the existing frameworks for network modeling.

Moreover, the results from the example models clearly demonstrate the importance of incorporating autocorrelation into input models. In particular, the results of the ARTA processes with exponential marginal distribution compared to the same independent exponential distribution show that adding a few lags of autocorrelation might help to improve the quality of simulation models significantly.

9.4. A Framework for Fitting and Analyzing Stochastic Processes

In the previous sections several additions to the existing toolkit `ProFiDo` [12, 14] have been presented that allow for an easier integration of stochastic processes into simulation models. Figure 9.10 shows the complete framework including the newly added software. `ProFiDo` takes a trace as input and can fit one or several stochastic processes or distributions to the trace. Supported are PH distributions, MAPs, ARMA processes, ARTA processes and the newly added CHEPs and CAPPs. `ProFiDo` includes tools for the visualization of statistical properties like density or distribution functions, lag-k autocorrelation coefficients or (joint) moments. Moreover, `ProFiDo` supports the two

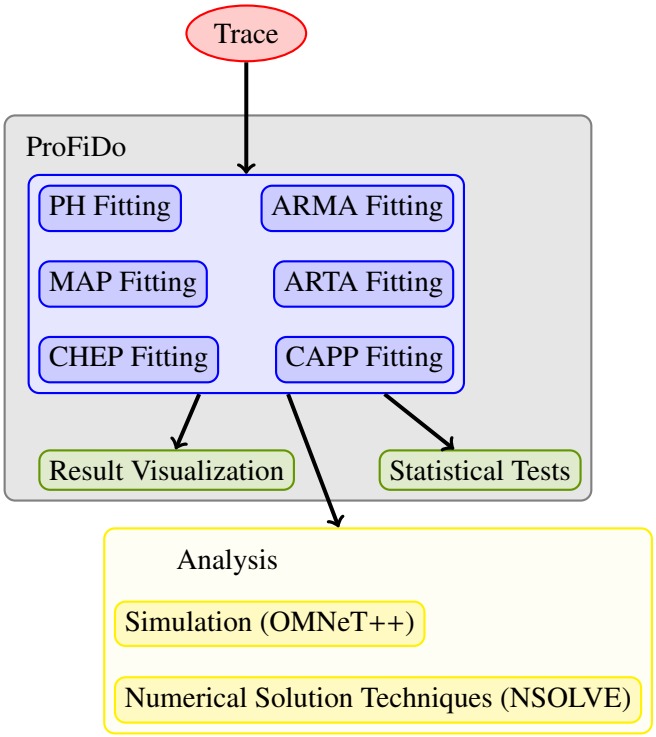


Figure 9.10.: `ProFiDo` framework for fitting and analyzing stochastic processes

sample Kolmogorov-Smirnov and Pearson's Chi-Squared tests to examine, if two samples (e.g. a trace and another trace generated by a distribution fitted to the first trace) originated from the same distribution. Once the processes with the desired properties have been fitted, they can be used in simulation models by importing them with the `OMNeT++` module for Arrival Processes presented above. Additionally, MAPs can be exported to be used with the tool `NSOLVE` [39, 122] that contains numerical solution techniques for large Markov chains. Thus, ProFiDo provides a complete framework for fitting stochastic processes, for assessing the quality of the fitted models and for easily integrating them into simulation models.

Chapter 10

Conclusions

In this work several existing approaches to model correlated traffic data have been assessed for their suitability as simulation input models. To overcome some of the identified weaknesses and disadvantages of these approaches, the ARTA approach, which uses an autoregressive base process and relies on the inversion of the cumulative distribution function, has been extended in several ways. To be able to capture a larger number of empirical autocorrelation coefficients ARTA processes were enabled to use an ARMA base process instead of an AR process. For arbitrary acyclic PH distribution and the subclass of Hyper-Erlang distributions, for which the inverse cumulative distribution function cannot be computed efficiently in general, a novel approach to combine an ARMA base process with PH distributions has been developed, which uses the base process to choose an elementary series of the APH distribution. To increase the possible range of autocorrelation that these novel processes denoted as CHEPs and CAPPs can capture, several transformations of the APH distribution have been proposed. The theoretical work resulted in a fitting tool for CHEPs and CAPPs and a module for the simulation framework `OMNeT++` to easily include stochastic processes into simulation models. Both tools have been integrated into the toolkit `ProFiDo`, which provides a complete framework for fitting, analyzing and simulating stochastic processes. In an extensive empirical study the suitability of CHEPs and CAPPs for capturing the behavior of synthetically generated and real traces has been assessed and the two novel processes have been compared with other existing approaches. It was shown that CHEPs and CAPPs are able to adequately capture both distribution and autocorrelation of a trace. The ability of capturing the behavior of the traces was confirmed by results from several queueing models.

The work presented in this thesis can be extended in several directions.

The first two ideas aim at improving or modifying the proposed fitting algorithms for CHEPs and CAPPs.

Recall, from Chapter 7 that the order of the ARMA base process was not automatically determined, but instead the best base process from a given set of base processes with different orders was selected. A heuristic that automatically determines the best base process order would avoid fitting base processes with different orders that are disregarded later anyway.

For fitting the base process the empirical autocorrelations of the trace have been

used. Since the equations for computing arbitrary joint moments for CHEPs and CAPPs have been given in this work as well, it would be possible to fit a CHEP or CAPP according to joint moments instead of autocorrelation coefficients. On the other hand, the results obtained in Chapter 8 suggest, that this would result in processes that can capture the joint moments almost exactly but fall short of capturing other characteristics not used for fitting.

Additionally, the proposed approaches can be generalized in several ways. In [24] the ARTA approach was generalized to model multivariate random processes with a vector autoregressive base process, called VARTA. The base process is then defined as

$$\boldsymbol{Z_t} = \alpha_1 \boldsymbol{Z_{t-1}} + \alpha_2 \boldsymbol{Z_{t-2}} + \ldots + \alpha_p \boldsymbol{Z_{t-p}} + \boldsymbol{\epsilon_t}$$

where in the k-variate case each $\boldsymbol{Z_t}$ is a random vector with k components, i.e. $\boldsymbol{Z_t} = (Z_{1,t}, Z_{2,t}, \ldots, Z_{k,t})$ and the i-th variate of the VARTA time series with marginal distribution F_{Y_i} is generated by $Y_{i,t} = F_{Y_i}^{-1}(\Phi(Z_{i,t}))$. The same approach can of course be applied for CHEPs and CAPPs resulting in correlated multivariate Phase-type processes.

It is known that the autocorrelation is in some cases not sufficient to describe the dependencies such that additional measures have to be considered. In [21] the VARTA approach is extended to account for these cases and copula-based multivariate input models are proposed. Similar extensions should be possible for CAPP processes as well.

Recall, that for computing the autocorrelation of a CHEP or CAPP only the mean values of the elementary series have to be known. Hence the ARMA base process could be used to generate correlated random numbers from a mixture of arbitrary distributions instead of Erlang branches or Hypo-Exponentially distributed elementary series, as long as all mean values from the distributions are known. Of course, fitting these mixture of (potentially different) distributions to a trace is much more difficult than fitting a single distribution only and split it into series like it was done in this work for Hyper-Erlang distributions and acyclic Phase-type distributions.

Appendix A

Notations

A.1. List of Mathematical Symbols

B	backward shift operator (cf. Section 2.1)
B_i	batch size for stochastic processes with batch arrivals
$\underline{b}_i, \bar{b}_i$	lower and upper bounds for probability of the i-th elementary series of an APH distribution (cf. Chapters 5 and 6)
C_X, C_X^2	(squared) coefficient of variation (cf. Section 1.3)
C_{ij}	covariance (cf. Section 1.3)
$Corr[X_i, X_j]$	correlation between the random variables X_i and X_j (cf. Section 1.3)
$Cov[X_i, X_j]$	covariance between the random variables X_i and X_j (cf. Section 1.3)
$\boldsymbol{D}_0$	transition rate matrix of a PH distribution or a MAP (cf. Section 2.3)
$E[X]$	expectation of the random variable X (cf. Section 1.3)
$f(x)$	probability density function
$F(x)$	cumulative distribution function
F_Y	marginal distribution of a stochastic process Y_t
$\boldsymbol{I}$	identity matrix
$\mathcal{J}$	set of joint moments
$\mathcal{L}(\cdot)$	likelihood function
l	trace length
$\boldsymbol{M} = -\boldsymbol{D}_0^{-1}$	moment matrix of a MAP or PH distribution (cf. Section 2.3)
m	number of paths (elementary series) of an acyclic Phase-type distribution, for Hyper-Erlang and Hyper-Exponential distributions this is equivalent to the number of branches
n	order of a Phase-type distribution or a Markovian Arrival Process
$N(\mu, \sigma^2)$	normal distribution
$\mathcal{N}_r(\boldsymbol{\mu}, \boldsymbol{\Sigma})$	multivariate normal distribution
$p(x)$	probability mass function
$\boldsymbol{P}$	transition probability matrix of a DTMC (cf. Section 1.3)
p	AR order
q	MA order
$\boldsymbol{Q}$	infinitesimal generator matrix of a CTMC (cf. Section 1.3)
$\boldsymbol{S}$	vector with phases for the Erlang branches of a Hyper-Erlang distribution

S_i	number of phases of the i-th elementary series or number of phases of the i-th Erlang branch
$\mathcal{T}$	trace (cf. Section 1.3)
$\boldsymbol{t}$	killing rate vector of a Phase-type distribution (cf. Section 2.3)
t_i	trace element
U	random variable with uniform distribution
$U(a, b)$	uniform distribution on $[a, b]$
U_t	sequence of random variables with uniform distribution on $(0, 1)$
$Var[X]$	variance of random variable X (cf. Section 1.3)
X	random variable
$X(t)$	continuous-time stochastic process
X_t	discrete-time stochastic process, sequence of random variables
Y	random variable
Y_t	ARTA Process, TES Process, CHEP, CAPP
Z_t	ARMA process

α_i	AR coefficient (cf. Section 2.1)
β_i	MA coefficient (cf. Section 2.1)
ϵ_t	innovation of ARMA process (cf. Section 2.1)
$\gamma(k), \gamma_k$	autocovariance at lag k (cf. Section 1.3)
λ, λ_i	transition rates of a Phase-type distribution
$\boldsymbol{\lambda}$	vector with transition rates for the branches of a Hyper-Erlang distribution
$\boldsymbol{\Lambda_i}$	vector with transition rates for the i-th elementary series
μ	mean
μ_i	i-th moment
$\hat{\mu}_i$	estimated i-th moment from a trace
ν_{ij}	joint moment
$\hat{\nu}_{ij}$	estimated joint moment from a trace
$\omega(\cdot)$	relation between base process and main process autocorrelation for CHEPs/CAPPs and ARTA processes
$\Phi(\cdot)$	standard normal cumulative distribution
$\boldsymbol{\pi}$	initial probabilities of a Phase-type distribution and steady-state vector of a DTMC
$\boldsymbol{\pi}'$	steady-state vector of a CTMC
$\boldsymbol{\pi}(i) = \pi_i$	initial probability of state i of a Phase-type distribution
ρ_{ij}	correlation between the random variables X_i and X_j (cf. Section 1.3)
ρ_k	autocorrelation at lag k (cf. Section 1.3)
$\hat{\rho}_k$	estimated autocorrelation of a trace which is the desired autocorrelation of the stochastic process that is fitted to the trace
σ^2	variance
$\hat{\sigma}^2$	estimated variance
σ_ϵ^2	variance of white noise of an $ARMA(p, q)$ process (cf. Section 2.1)
Σ	covariance matrix of multivariate normal distribution
$\boldsymbol{\tau}$	initial probabilities of the branches of a Hyper-Erlang distribution

τ_i	probability of path i from an initial to the absorbing state of a PH distribution, for Hyper-Erlang and Hyper-Exponential distributions this is equivalent to the initial probability of branch i
φ_{ρ_h}	standard bivariate normal density function with correlation ρ_h

Appendix B

Probability Distributions

In the following a brief overview of some common probability distributions used in stochastic modeling is given. The main part of this work deals with Phase-type distributions and related processes, which have been extensively presented in Section 2.3. Throughout this thesis several other distributions not related to Phase-type have been mentioned, either as marginal distribution for stochastic processes or for proving characteristics of stochastic processes. For a better understanding of that part of this work some properties of the distributions used so far will be summarized for reference. More elaborate overviews can for example be found in [89] or [104].

B.1. Uniform Distribution

The uniform distribution $U(a, b)$ is used for variables that can take on values in the range $[a, b]$. The distribution is defined by the density function

$$f(x) = \begin{cases} \frac{1}{b-a} & \text{if } a \leq x \leq b \\ 0 & \text{otherwise} \end{cases}$$

and the distribution function

$$F(x) = \begin{cases} 0 & \text{if } x < a \\ \frac{x-a}{b-a} & \text{if } a \leq x \leq b \\ 1 & \text{if } b < x. \end{cases}$$

The mean is given by $E[X] = \frac{a+b}{2}$ and the variance by $Var[X] = \frac{(b-a)^2}{12}$. The $U(0, 1)$ distribution is of special interest in random number generation, since it is the basis for generating random numbers from other distributions, e.g. by inverting the cumulative distribution function (inversion method) [59].

B.2. Normal Distribution

The normal distribution (sometimes called Gaussian distribution) with parameters μ and σ has probability density function

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}, -\infty < x < \infty. \tag{B.1}$$

The parameters μ and σ are the mean and standard deviation of a random variable X with normal distribution and we write $X \sim N(\mu, \sigma^2)$.

For the standard normal distribution $N(0, 1)$ Equation B.1 becomes

$$f(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}, -\infty < x < \infty.$$

The cumulative distribution function of the normal distribution has no closed form in general and the probabilities are usually obtained from precomputed tables, containing the values of a standard normal distribution, i.e. for $Z \sim N(0, 1)$ it contains the values

$$F_Z(z) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{z} e^{-t^2/2} dt.$$

The values of the distribution function for $X \sim N(\mu, \sigma^2)$ can then be obtained from

$$F_X(x) = F_Z\left(\frac{x-\mu}{\sigma}\right).$$

The normal distribution has the reproductive property, stating that the sum of n normally distributed independent random variables has a normal distribution [155], i.e. let $X_1, X_2, \ldots, X_n$ be independent normally distributed random variables with mean $\mu_1, \mu_2, \ldots, \mu_n$ and standard deviation $\sigma_1^2, \sigma_2^2, \ldots, \sigma_n^2$, respectively, and $c_1, c_2, \ldots, c_n$ real constants. Then the random variable $X = \sum_{i=1}^{n} c_i X_i$ has a normal distribution with mean $\mu = \sum_{i=1}^{n} c_i \mu_i$ and standard deviation $\sigma^2 = \sum_{i=1}^{n} c_1^2 \sigma_i^2$.

B.3. Lognormal Distribution

The lognormal distribution with parameters μ and σ is related to the normal distribution, i.e. $X \sim LN(\mu, \sigma^2) \Leftrightarrow lnX \sim N(\mu, \sigma^2)$. Its density is given by

$$f(x) = \begin{cases} \frac{1}{x\sqrt{2\pi\sigma^2}} e^{\frac{-(lnx-\mu)^2}{2\sigma^2}}, & x > 0 \\ 0, & \text{otherwise.} \end{cases}$$

The lognormal distribution has mean and variance

$$E[X] = e^{\mu+\sigma^2/2} \qquad \text{and} \qquad Var[X] = e^{2\mu+\sigma^2}(e^{\sigma^2} - 1).$$

B.4. Johnson Family of Distributions

The Johnson Translation System [56, 88] describes a family of distributions derived from the normal distribution. A random variable X with Johnson distribution is defined by the cumulative distribution function

$$F(x) = \Phi\left(\gamma + \delta f\left(\frac{x-\xi}{\lambda}\right)\right)$$

with shape parameters γ and δ, location parameter ξ and scale parameter λ. $\Phi(\cdot)$ is the standard normal distribution function and thus, any Johnson random variable is a transformation of a standard normal random variable. The function f can be one of the following transformations:

$$f(y) = \begin{cases} log(y) & \text{lognormal system of distributions } S_L \\ log\left(y + \sqrt{y^2+1}\right) & \text{unbounded system of distributions } S_U \\ log\left(\frac{y}{1-y}\right) & \text{bounded system of distributions } S_B \\ y & \text{normal system of distributions } S_N. \end{cases}$$

Distributions from the Johnson Translation System can match any finite first four moments, i.e. mean, variance, coefficient of skewness and coefficient of kurtosis. Furthermore, the distributions of a family are completely determined by the four parameters $\gamma, \delta, \xi, \lambda$. A comparison of fitting methods for the Johnson Translation System can be found in [144]. A robust method for fitting was presented in [145].

B.5. Triangular Distribution

The triangular distribution is often applied as a rough model when no data is available to select another distribution. It has three parameters, the lower bound a, the upper bound b and the mode c with $a < c < b$ and can be defined in terms of the probability density function

$$f(x) = \begin{cases} \frac{2(x-a)}{(b-a)(c-a)}, & a \le x \le c \\ \frac{2(b-x)}{(b-a)(b-c)}, & c < x \le b \\ 0, & \text{otherwise} \end{cases}$$

and the cumulative distribution function

$$F(x) = \begin{cases} 0, & x < a \\ \frac{(x-a)^2}{(b-a)(c-a)}, & a \le x \le c \\ 1 - \frac{b-x)^2}{(b-a)(b-c)}, & c < x \le b \\ 1, & b < x. \end{cases}$$

Its mean and variance are given by

$$E[X] = \frac{a+b+c}{3} \qquad \text{and} \qquad Var[X] = \frac{a^2+b^2+c^2-ab-ac-bc}{18},$$

respectively.

B.6. Weibull Distribution

A Weibull distribution with parameters[6] λ and α has density function

$$f(x) = \lambda\alpha x^{\alpha-1}e^{-\lambda x^{\alpha}}$$

and distribution function

$$F(x) = 1 - e^{-\lambda x^{\alpha}}$$

with $x \geq 0, \lambda > 0, \alpha > 0$. The mean is given by

$$E[X] = \int_0^{\infty} \lambda x \alpha x^{\alpha-1} e^{-\lambda x^{\alpha}} dx.$$

B.7. Pareto Distribution

The Pareto distribution has density function

$$f(x) = \alpha k^{\alpha} x^{-\alpha-1}, x \geq k, \alpha, k > 0$$

and distribution function

$$F(x) = \begin{cases} 1 - \left(\frac{k}{x}\right)^{\alpha} & \text{if } x \geq k \\ 0 & \text{if } x < k. \end{cases}$$

The Pareto distribution has mean

$$E[X] = \int_k^{\infty} \alpha k^{\alpha} x^{-\alpha} dx = \begin{cases} \frac{k\alpha}{\alpha-1} & \text{if } \alpha > 1 \\ \infty & \text{if } \alpha \leq 2. \end{cases}$$

The Pareto and the Weibull distributions are heavy-tailed distributions, i.e. their complementary cumulative distribution function F^c defined as $F^c(c) = 1 - F(x) = P(X > x)$ decays more slowly than exponentially, or more formally $e^{\gamma x} F^c(x) \to \infty$ as $x \to \infty$ for all $\gamma > 0$.

B.8. Gamma Distribution

The gamma distribution with parameters α and β is a generalization of the Erlang distribution. The scale parameter β corresponds to the rate λ of an Erlang distribution and the shape parameter α corresponds to the number of phases n, though non-integer values are allowed for α. The density function is defined as

$$f(x) = \begin{cases} \frac{\beta^{-\alpha} x^{\alpha-1} e^{-x/\beta}}{\Gamma(\alpha)}, & x > 0 \\ 0, & \text{otherwise} \end{cases}$$

where $\Gamma(\alpha) = \int_0^{\infty} t^{\alpha-1} e^{-t} dt$ is the gamma function. There is no closed form for the cumulative distribution function, except when the gamma distribution is Erlang. The mean is given by $\alpha\beta$ and the variance by $\alpha\beta^2$.

[6] Sometimes a third parameter, the location parameter is added, resulting in the distribution function $F(x) = 1 - e^{-\lambda(x-\theta)^{\alpha}}, x \geq \theta$.

B.9. χ^2 Distribution

Like Erlang the χ^2 distribution with k degrees of freedom is another special case of the gamma distribution, more precisely the χ^2 distribution with parameter k is equivalent to the gamma distribution with parameters $\alpha = k/2$ and $\beta = 2$.

B.10. Multivariate Normal Distribution

The multivariate normal distribution $\mathcal{N}_r(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ is a generalization of the normal distribution for r-dimensional random variables $(X_1, \ldots, X_r)$. It is defined by the vector $\boldsymbol{\mu}$ that contains the mean values of the r random variables and the covariance matrix $\boldsymbol{\Sigma}$ that contains at position (i, j) the covariance between X_i and X_j, i.e. $\boldsymbol{\Sigma}(i, j) = Cov(X_i, X_j)$. Its probability density function is given by

$$f(\boldsymbol{x}) = \frac{1}{(2\pi)^{r/2}|\Sigma|^{1/2}} e^{\left(-1/2(\boldsymbol{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right)}$$

where $|\Sigma|$ is the determinant of Σ and $(\boldsymbol{x} - \boldsymbol{\mu})^T$ is the transposed of $\boldsymbol{x} - \boldsymbol{\mu}$. For the cumulative distribution function no closed-form expression exists and the values have to be approximated numerically.

In this work the two-dimensional case (the standard bivariate normal distribution) with $\mu_1 = \mu_2 = 0$, $\sigma_1 = \sigma_2 = 1$ and correlation ρ received some special attention. For the standard bivariate normal distribution the probability density function can be simplified to

$$\varphi_\rho(x_1, x_2) = \frac{1}{2\pi\sqrt{1-\rho^2}} e^{-\frac{1}{2(1-\rho^2)}(x_1^2 - 2\rho x_1 x_2 + x_2^2)}. \tag{B.2}$$

For an extensive review of the properties of multivariate normal distributions the interested reader is referred to [90] and [153].

Appendix C

Proofs for APH Transformations

This appendix contains additions to some of the proofs and ideas that have only been briefly sketched in Section 6.4.

C.1. Addendum to the Proof of Lemma 6.1

For the proof of Lemma 6.1 it has to be shown that for an elementary series of an APH distribution $\lambda_{j_m} \geq \lambda_{j_{m-1}} \geq \cdots \geq \lambda_{j_1}$ and the two new series

$$\lambda_{j_m} \geq \cdots \geq \lambda_{j_{m-k+1}} \geq \lambda_x \geq \lambda_{j_{m-k}} \geq \cdots \geq \lambda_{j_1}$$

with probability $\tau_j(1 - \lambda_{j_{m-k}}/\lambda_x)$ and

$$\lambda_{j_m} \geq \cdots \geq \lambda_{j_{m-k+1}} \geq \lambda_x \geq \lambda_{j_{m-k-1}} \geq \cdots \geq \lambda_{j_1}$$

with probability $\tau_j\lambda_{j_{m-k}}/\lambda_x$ and $\lambda_x \geq (>)\lambda_{j_{m-k}}$ that have been obtained by a transformation step the following condition holds:

$$\left(\frac{1}{\lambda_{j_m}} + \cdots \frac{1}{\lambda_{j_{m-k+1}}} + \frac{1}{\lambda_x} + \frac{1}{\lambda_{j_{m-k}}} + \cdots + \frac{1}{\lambda_{j_1}}\right)^2 \tau_j \tag{C.1}$$

$$-\left(\left(\frac{1}{\lambda_{j_m}} + \cdots \frac{1}{\lambda_{j_{m-k+1}}} + \frac{1}{\lambda_x} + \frac{1}{\lambda_{j_{m-k}}} + \cdots + \frac{1}{\lambda_{j_1}}\right)^2 \tau_j \frac{\lambda_{j_{m-k}}}{\lambda_x}\right) \tag{C.2}$$

$$+\left(\left(\frac{1}{\lambda_{j_m}} + \cdots \frac{1}{\lambda_{j_{m-k+1}}} + \frac{1}{\lambda_x} + \frac{1}{\lambda_{j_{m-k-1}}} + \cdots + \frac{1}{\lambda_{j_1}}\right)^2 \tau_j \frac{\lambda_{j_{m-k}}}{\lambda_x}\right) \tag{C.3}$$

$$\geq (>) \quad \left(\left(\frac{1}{\lambda_{j_m}} + \cdots \frac{1}{\lambda_{j_{m-k+1}}} + \frac{1}{\lambda_{j_{m-k}}} + \cdots + \frac{1}{\lambda_{j_1}}\right)^2 \tau_j\right). \tag{C.4}$$

Proof. Define

$$f_n(m, m-1, \ldots, 1) = \sum_{l_m + l_{m-1} + \cdots + l_1 = n} \frac{n!}{l_m! l_{m-1}! \ldots l_1!} \left(\frac{1}{\lambda_m^{l_m}} \frac{1}{\lambda_{m-1}^{l_{m-1}}} \cdots \frac{1}{\lambda_1^{l_1}}\right).$$

Of particular interest in the following are

$$f_2(m, m-1, \ldots, 1) = \sum_{l_m+l_{m-1}+\cdots+l_1=2} \frac{2!}{l_m! l_{m-1}! \ldots l_1!} \left(\frac{1}{\lambda_m^{l_m}} \frac{1}{\lambda_{m-1}^{l_{m-1}}} \cdots \frac{1}{\lambda_1^{l_1}} \right)$$

and

$$\begin{aligned} f_1(m, m-1, \ldots, 1) &= \sum_{l_m+l_{m-1}+\cdots+l_1=1} \frac{1!}{l_m! l_{m-1}! \ldots l_1!} \left(\frac{1}{\lambda_m^{l_m}} \frac{1}{\lambda_{m-1}^{l_{m-1}}} \cdots \frac{1}{\lambda_1^{l_1}} \right) \\ &= \sum_{l_m+l_{m-1}+\cdots+l_1=1} \left(\frac{1}{\lambda_m^{l_m}} \frac{1}{\lambda_{m-1}^{l_{m-1}}} \cdots \frac{1}{\lambda_1^{l_1}} \right) \\ &= \left(\frac{1}{\lambda_m^{l_m}} + \frac{1}{\lambda_{m-1}^{l_{m-1}}} + \cdots + \frac{1}{\lambda_1^{l_1}} \right). \end{aligned}$$

Note, that $f_2(m, m-1, \ldots, 1)$ can be split into three sums according to one of the l_i, i.e. one sum for $l_i = 0$, one for $l_i = 1$ and one for $l_i = 2$:

$$\begin{aligned} f_2(m, m-1, \ldots, 1) &= \sum_{l_m+l_{m-1}+\cdots+l_{i+1}+l_i+l_{i-1}+\cdots+l_1=2} \frac{2!}{l_m! l_{m-1}! \ldots l_{i+1}! l_i! l_{i-1}! \ldots l_1!} \\ &\qquad \left(\frac{1}{\lambda_m^{l_m}} \frac{1}{\lambda_{m-1}^{l_{m-1}}} \cdots \frac{1}{\lambda_{i+1}^{l_{i+1}}} \frac{1}{\lambda_i^{l_i}} \frac{1}{\lambda_{i-1}^{l_{i-1}}} \cdots \frac{1}{\lambda_1^{l_1}} \right) \\ &= \sum_{l_m+l_{m-1}+\cdots+l_{i+1}+l_{i-1}+\cdots+l_1=2} \left(\frac{2!}{l_m! l_{m-1}! \ldots l_{i+1}! l_{i-1}! \ldots l_1!} \right. \\ &\qquad \left. \left(\frac{1}{\lambda_m^{l_m}} \frac{1}{\lambda_{m-1}^{l_{m-1}}} \cdots \frac{1}{\lambda_{i+1}^{l_{i+1}}} \frac{1}{\lambda_{i-1}^{l_{i-1}}} \cdots \frac{1}{\lambda_1^{l_1}} \right) \right) \\ &\quad + \frac{2}{\lambda_i} \left(\sum_{l_m+l_{m-1}+\cdots+l_{i+1}+l_{i-1}+\cdots+l_1=1} \frac{1!}{l_m! l_{m-1}! \ldots l_{i+1}! l_{i-1}! \ldots l_1!} \right. \\ &\qquad \left. \left(\frac{1}{\lambda_m^{l_m}} \frac{1}{\lambda_{m-1}^{l_{m-1}}} \cdots \frac{1}{\lambda_{i+1}^{l_{i+1}}} \frac{1}{\lambda_{i-1}^{l_{i-1}}} \cdots \frac{1}{\lambda_1^{l_1}} \right) \right) \\ &\quad + \left(\frac{1}{\lambda_i^2} \right) \\ &= f_2(m, m-1, \ldots, i+1, i-1, \ldots, 1) \\ &\quad + \frac{2}{\lambda_i} f_1(m, m-1, \ldots, i+1, i-1, \ldots, 1) + \frac{1}{\lambda_i^2}. \end{aligned}$$

Obviously, $f_1(m, \ldots, i+1, i, i+1, \ldots, 1) = 1/\lambda_i + f_1(m, \ldots, i+1, i+1, \ldots, 1)$ holds.

By application of the multinomial theorem we may write

$$\begin{aligned} \left(\frac{1}{\lambda_m} + \frac{1}{\lambda_{m-1}} + \cdots + \frac{1}{\lambda_1} \right)^2 &= \sum_{l_m+l_{m-1}+\cdots+l_1=2} \frac{2!}{l_m! l_{m-1}! \ldots l_1!} \left(\frac{1}{\lambda_m^{l_m}} \frac{1}{\lambda_{m-1}^{l_{m-1}}} \cdots \frac{1}{\lambda_1^{l_1}} \right) \\ &= f_2(m, m-1, \ldots, 1). \end{aligned}$$

Hence, we can express the terms C.1, C.2, C.3 and C.4 using the function $f_2(\cdot)$ and split the terms applying the above rules for $f_2(\cdot)$ and $f_1(\cdot)$.

For C.1 we obtain

$$\begin{aligned}
&\left(\frac{1}{\lambda_{j_m}} + \cdots \frac{1}{\lambda_{j_{m-k+1}}} + \frac{1}{\lambda_x} + \frac{1}{\lambda_{j_{m-k}}} + \cdots + \frac{1}{\lambda_{j_1}}\right)^2 \tau_j \\
= \;& \tau_j f_2(j_m, \ldots, j_{m-k+1}, x, j_{m-k}, \ldots, j_1) \\
= \;& \tau_j f_2(j_m, \ldots, j_{m-k+1}, j_{m-k}, \ldots, j_1) \\
&+ \frac{2\tau_j}{\lambda_x} f_1(j_m, \ldots, j_{m-k+1}, j_{m-k}, \ldots, j_1) + \frac{\tau_j}{\lambda_x^2} \\
= \;& \tau_j f_2(j_m, \ldots, j_{m-k+1}, j_{m-k}, \ldots, j_1) \\
&+ \frac{2\tau_j}{\lambda_x} f_1(j_m, \ldots, j_{m-k+1}, j_{m-k-1}, \ldots, j_1) + \frac{2\tau_j}{\lambda_x \lambda_{j_{m-k}}} + \frac{\tau_j}{\lambda_x^2}.
\end{aligned}$$

C.2 can be split into

$$\begin{aligned}
&-\left(\left(\frac{1}{\lambda_{j_m}} + \cdots \frac{1}{\lambda_{j_{m-k+1}}} + \frac{1}{\lambda_x} + \frac{1}{\lambda_{j_{m-k}}} + \cdots + \frac{1}{\lambda_{j_1}}\right)^2 \tau_j \frac{\lambda_{j_{m-k}}}{\lambda_x}\right) \\
= \;& -\frac{\tau_j \lambda_{j_{m-k}}}{\lambda_x} f_2(j_m, \ldots, j_{m-k+1}, x, j_{m-k}, \ldots, j_1) \\
= \;& -\frac{\tau_j \lambda_{j_{m-k}}}{\lambda_x} f_2(j_m, \ldots, j_{m-k+1}, j_{m-k}, \ldots, j_1) \\
&- \frac{2\tau_j \lambda_{j_{m-k}}}{\lambda_x^2} f_1(j_m, \ldots, j_{m-k+1}, j_{m-k}, \ldots, j_1) - \frac{\tau_j \lambda_{j_{m-k}}}{\lambda_x^3} \\
= \;& -\frac{\tau_j \lambda_{j_{m-k}}}{\lambda_x} f_2(j_m, \ldots, j_{m-k+1}, j_{m-k-1}, \ldots, j_1) \\
&- \frac{2\tau_j}{\lambda_x} f_1(j_m, \ldots, j_{m-k+1}, j_{m-k-1}, \ldots, j_1) - \frac{\tau_j}{\lambda_x \lambda_{j_{m-k}}} \\
&- \frac{2\tau_j}{\lambda_x^2} - \frac{2\tau_j \lambda_{j_{m-k}}}{\lambda_x^2} f_1(j_m, \ldots, j_{m-k+1}, j_{m-k-1}, \ldots, j_1) \\
&- \frac{\tau_j \lambda_{j_{m-k}}}{\lambda_x^3}.
\end{aligned}$$

Finally, for C.3 we get

$$\begin{aligned}
&\left(\left(\frac{1}{\lambda_{j_m}} + \cdots \frac{1}{\lambda_{j_{m-k+1}}} + \frac{1}{\lambda_x} + \frac{1}{\lambda_{j_{m-k-1}}} + \cdots + \frac{1}{\lambda_{j_1}}\right)^2 \frac{\tau_j \lambda_{j_{m-k}}}{\lambda_x}\right) \\
= \;& \frac{\tau_j \lambda_{j_{m-k}}}{\lambda_x} f_2(j_m, \ldots, j_{m-k+1}, x, j_{m-k-1}, \ldots, j_1) \\
= \;& \frac{\tau_j \lambda_{j_{m-k}}}{\lambda_x} f_2(j_m, \ldots, j_{m-k+1}, j_{m-k-1}, \ldots, j_1) \\
&+ \frac{2\tau_j \lambda_{j_{m-k}}}{\lambda_x^2} f_1(j_m, \ldots, j_{m-k+1}, j_{m-k-1}, \ldots, j_1) + \frac{\tau_j \lambda_{j_{m-k}}}{\lambda_x^3}.
\end{aligned}$$

After eliminating all identical terms that appear with positive and negative sign in the expressions for C.1, C.2 and C.3 we obtain

$$\tau_j f_2(j_m, \ldots, j_{m-k+1}, j_{m-k}, \ldots, j_1) + \frac{2\tau_j}{\lambda_x \lambda_{j_{m-k}}} - \frac{\tau_j}{\lambda_x \lambda_{j_{m-k}}} + \frac{\tau_j}{\lambda_x^2} - \frac{2\tau_j}{\lambda_x^2}$$

$$
\begin{aligned}
&= \tau_j f_2(j_m, \ldots, j_{m-k+1}, j_{m-k}, \ldots, j_1) + \frac{\tau_j}{\lambda_x \lambda_{j_{m-k}}} - \frac{\tau_j}{\lambda_x^2} \\
&= \tau_j f_2(j_m, \ldots, j_{m-k+1}, j_{m-k}, \ldots, j_1) + \frac{\tau_j}{\lambda_x}\left(\frac{1}{\lambda_{j_{m-k}}} - \frac{1}{\lambda_x}\right) \\
&\geq (>) \quad \tau_j f_2(j_m, \ldots, j_{m-k+1}, j_{m-k}, \ldots, j_1)
\end{aligned}
$$

for $\lambda_x \geq (>)\lambda_{j_{m-k}}$. Observe, that $\tau_j f_2(j_m, \ldots, j_{m-k+1}, j_{m-k}, \ldots, j_1)$ is equal to C.4, which proves the lemma. □

C.2. Effect of the APH Transformations on the Negative Autocorrelation

In Section 6.4 two transformations for APH distributions and the proofs that these transformations increase the maximal positive autocorrelation that can be achieved with a given distribution have been presented. For positive autocorrelations these proofs are straightforward because according to Equation 6.15 for a base process autocorrelation of $\rho = 1$ only combinations of a series with itself contribute to the autocorrelation of the CHEP or CAPP. For a base process autocorrelation of $\rho = -1$ the situation is more difficult, because several cases have to be distinguished for the computation of the CHEP/CAPP autocorrelation as one can see from Equation 6.19. In the following the proof for negative autocorrelations is given for APH distributions where the probabilities of the elementary series have a specific structure. Afterwards some considerations for the general case are presented.

First, note that we may split an elementary series into two or more identical series by adjusting the probabilities of the series.

Lemma C.1. *Let i be an elementary series with mean μ_i and interval $[\underline{b}_i, \bar{b}_i]$. Then, we can split the series into n series $i_1, \ldots, i_n$ each with mean μ_i and the disjoint intervals $[\underline{b}_{i_1}, \bar{b}_{i_1}], \ldots, [\underline{b}_{i_n}, \bar{b}_{i_n}]$, $\underline{b}_i = \underline{b}_{i_1} < \bar{b}_{i_1} = \underline{b}_{i_2} < \bar{b}_{i_2} \ldots \underline{b}_{i_n} < \bar{b}_{i_n} = \bar{b}_i$ without changing the possible autocorrelation of a CAPP as defined by Equation 6.9.*

Proof. A series i may contribute to the double sum in Equation 6.9 in two ways, i.e. with the term for the series (i, i) and with the terms for combinations of (i, j). Assume, that i is split into two series i_1 and i_2. Then,

$$
\begin{aligned}
& \mu_i^2 \int_{\Phi^{-1}(\underline{b}_i)}^{\Phi^{-1}(\bar{b}_i)} \int_{\Phi^{-1}(\underline{b}_i)}^{\Phi^{-1}(\bar{b}_i)} \varphi_{\rho_h}(z_t, z_{t+h}) dz_t dz_{t+h} \\
= \; & \mu_i^2 P(\Phi^{-1}(\underline{b}_i) < Z_t < \Phi^{-1}(\bar{b}_i), \Phi^{-1}(\underline{b}_i) < Z_{t+h} < \Phi^{-1}(\bar{b}_i)) \\
= \; & \mu_i^2 \Big(P(\Phi^{-1}(\underline{b}_{i_1}) < Z_t < \Phi^{-1}(\bar{b}_{i_1}), \Phi^{-1}(\underline{b}_{i_1}) < Z_{t+h} < \Phi^{-1}(\bar{b}_{i_1})) \\
& + P(\Phi^{-1}(\underline{b}_{i_1}) < Z_t < \Phi^{-1}(\bar{b}_{i_1}), \Phi^{-1}(\underline{b}_{i_2}) < Z_{t+h} < \Phi^{-1}(\bar{b}_{i_2})) \\
& + P(\Phi^{-1}(\underline{b}_{i_2}) < Z_t < \Phi^{-1}(\bar{b}_{i_2}), \Phi^{-1}(\underline{b}_{i_1}) < Z_{t+h} < \Phi^{-1}(\bar{b}_{i_1})) \\
& + P(\Phi^{-1}(\underline{b}_{i_2}) < Z_t < \Phi^{-1}(\bar{b}_{i_2}), \Phi^{-1}(\underline{b}_{i_2}) < Z_{t+h} < \Phi^{-1}(\bar{b}_{i_2})) \Big) \\
= \; & \mu_i^2 \int_{\Phi^{-1}(\underline{b}_{i_1})}^{\Phi^{-1}(\bar{b}_{i_1})} \int_{\Phi^{-1}(\underline{b}_{i_1})}^{\Phi^{-1}(\bar{b}_{i_1})} \varphi_{\rho_h}(z_t, z_{t+h}) dz_t dz_{t+h}
\end{aligned}
$$

$$
\begin{aligned}
&+\mu_i^2 \int_{\Phi^{-1}(\underline{b}_{i_1})}^{\Phi^{-1}(\bar{b}_{i_1})} \int_{\Phi^{-1}(\underline{b}_{i_2})}^{\Phi^{-1}(\bar{b}_{i_2})} \varphi_{\rho_h}(z_t, z_{t+h}) dz_t dz_{t+h} \\
&+\mu_i^2 \int_{\Phi^{-1}(\underline{b}_{i_2})}^{\Phi^{-1}(\bar{b}_{i_2})} \int_{\Phi^{-1}(\underline{b}_{i_1})}^{\Phi^{-1}(\bar{b}_{i_1})} \varphi_{\rho_h}(z_t, z_{t+h}) dz_t dz_{t+h} \\
&+\mu_i^2 \int_{\Phi^{-1}(\underline{b}_{i_2})}^{\Phi^{-1}(\bar{b}_{i_2})} \int_{\Phi^{-1}(\underline{b}_{i_2})}^{\Phi^{-1}(\bar{b}_{i_2})} \varphi_{\rho_h}(z_t, z_{t+h}) dz_t dz_{t+h}
\end{aligned}
$$

and

$$
\begin{aligned}
& \mu_i \mu_j \int_{\Phi^{-1}(\underline{b}_i)}^{\Phi^{-1}(\bar{b}_i)} \int_{\Phi^{-1}(\underline{b}_j)}^{\Phi^{-1}(\bar{b}_j)} \varphi_{\rho_h}(z_t, z_{t+h}) dz_t dz_{t+h} \\
= \;& \mu_i \mu_j P(\Phi^{-1}(\underline{b}_i) < Z_t < \Phi^{-1}(\bar{b}_i), \Phi^{-1}(\underline{b}_j) < Z_{t+h} < \Phi^{-1}(\bar{b}_j)) \\
= \;& \mu_i \mu_j \Big(P(\Phi^{-1}(\underline{b}_{i_1}) < Z_t < \Phi^{-1}(\bar{b}_{i_1}), \Phi^{-1}(\underline{b}_j) < Z_{t+h} < \Phi^{-1}(\bar{b}_j)) \\
& + P(\Phi^{-1}(\underline{b}_{i_2}) < Z_t < \Phi^{-1}(\bar{b}_{i_2}), \Phi^{-1}(\underline{b}_j) < Z_{t+h} < \Phi^{-1}(\bar{b}_j)) \Big) \\
= \;& \mu_i \mu_j \int_{\Phi^{-1}(\underline{b}_{i_1})}^{\Phi^{-1}(\bar{b}_{i_1})} \int_{\Phi^{-1}(\underline{b}_j)}^{\Phi^{-1}(\bar{b}_j)} \varphi_{\rho_h}(z_t, z_{t+h}) dz_t dz_{t+h} \\
& + \mu_i \mu_j \int_{\Phi^{-1}(\underline{b}_{i_2})}^{\Phi^{-1}(\bar{b}_{i_2})} \int_{\Phi^{-1}(\underline{b}_j)}^{\Phi^{-1}(\bar{b}_j)} \varphi_{\rho_h}(z_t, z_{t+h}) dz_t dz_{t+h}.
\end{aligned}
$$

Thus, by repeated application a branch may be split into an arbitrary number of branches without modifying the correlation. □

Now, assume that we have an APH distribution $APH(n)$ in series canonical form with an even number of elementary series $1, \ldots, m$. Let $\mu_1, \ldots, \mu_m$ be the mean values and $\tau_1, \ldots, \tau_m$ be the probabilities of the elementary series, respectively. Furthermore, assume that $\tau_i = \tau_{m-i+1}, i = 1, \ldots, m/2$, i.e. the elementary series with the smallest value and the series with the largest value have the same probability, the series with the second smallest value and the series with the second largest value have the same probability and so on. This implies for the probabilities, that

$$
\sum_{i=1}^{m/2} \tau_i = \sum_{j=m/2+1}^{m} \tau_j = 0.5
$$

and for the integration bounds, that

$$
0 = \underline{b}_1 < \bar{b}_1 = \underline{b}_2 < \bar{b}_2 \cdots \underline{b}_{m/2} < \bar{b}_{m/2} = 0.5 = \underline{b}_{m/2+1} < \bar{b}_{m/2+1} \cdots \underline{b}_m < \bar{b}_m = 1. \tag{C.5}
$$

Obviously, from Equation C.5 $\underline{b}_1 + \bar{b}_m = 1$ holds. Since $\tau_1 = \tau_m$, we have that $\underline{b}_1 + \tau_1 + \bar{b}_m - \tau_m = \bar{b}_1 + \underline{b}_m = 1$. Then, $\underline{b}_2 + \bar{b}_{m-1} = 1$, because $\underline{b}_2 = \bar{b}_1$ and $\bar{b}_{m-1} = \underline{b}_m$, and $\bar{b}_2 + \underline{b}_{m-1} = 1$, because $\tau_2 = \tau_{m-1}$. In general, we have that $\underline{b}_i + \bar{b}_{m-i+1} = 1$ and $\bar{b}_i + \underline{b}_{m-i+1} = 1$ for $i = 1, \ldots, m/2$. This implies, that $\underline{b}_i + \underline{b}_{m-i+1} < 1$ and $\bar{b}_i + \bar{b}_{m-i+1} > 1$. Observe, that $\underline{b}_i + \bar{b}_{m-i+1} = 1$ and $\bar{b}_i + \underline{b}_{m-i+1} = 1$ describe the limiting cases for the

different non-zero values for $P_\varphi(i, j)$ in Equation 6.19. Moreover, we have that

$$\begin{aligned} & \underline{b}_i + \bar{b}_{m-i+1} = 1 \\ \Leftrightarrow \quad & \underline{b}_i + \tau_i + \bar{b}_{m-i+1} = 1 + \tau_i \\ \Leftrightarrow \quad & \bar{b}_i + \bar{b}_{m-i+1} = 1 + \tau_i \\ \Leftrightarrow \quad & \bar{b}_i + \bar{b}_{m-i+1} - 1 = \tau_i \end{aligned}$$

and

$$\begin{aligned} & \underline{b}_i + \bar{b}_{m-i+1} = 1 \\ \Leftrightarrow \quad & \underline{b}_i + \bar{b}_{m-i+1} - \tau_{m-i+1} = 1 - \tau_{m-i+1} \\ \Leftrightarrow \quad & \underline{b}_i + \underline{b}_{m-i+1} = 1 - \tau_{m-i+1} \\ \Leftrightarrow \quad & 1 - \underline{b}_i - \underline{b}_{m-i+1} = \tau_{m-i+1}. \end{aligned}$$

$\tau_i = \tau_{m-i+1}$ holds by definition of our special case. Hence, for each pair of two series $(i, m-i+1), i = 1, \ldots, m/2$ we have that

$$\bar{b}_i + \bar{b}_{m-i+1} - 1 = 1 - \underline{b}_i - \underline{b}_{m-1+1} = \tau_i = \tau_{m-i+1},$$

i.e. all the cases for which $P_\varphi(i, j)$ in Equation 6.19 is non-zero take on the same value. Since the intervals $[\underline{b}_i, \bar{b}_i]$ are ordered and disjoint, $\underline{b}_i + \bar{b}_{m-i+1} = 1$ and $\bar{b}_i + \underline{b}_{m-i+1} = 1$ imply that

- $\underline{b}_{i+j} + \underline{b}_{m-i+1} \geq 1, j > 0,$
- $\bar{b}_i + \bar{b}_{m-i+1-j} \leq 1, j > 0,$
- $\bar{b}_{i-j} + \bar{b}_{m-i+1} \leq 1, j > 0$ and
- $\underline{b}_i + \underline{b}_{m-i+1+j} \geq 1, j > 0$ for $i = 1, \ldots, m/2$.

In all these cases $P_\varphi(\cdot, \cdot)$ is zero according to Equation 6.19, because either the sum of lower bounds is greater than one, the sum of upper bounds is less than zero or or the sum is exactly one. In this case we have $\bar{b}_{i_1} + \bar{b}_{i_2} - 1 = 1 - 1 = 0$ or $1 - \underline{b}_{i_1} - \underline{b}_{i_2} = 1 - 1 = 0$.

Thus, for a base process autocorrelation of $\rho = -1$ only the pairs of series with the smallest and the largest mean value, the second smallest and the second largest mean value and so on (or more formally: the pairs $(i, m-i+1), i = 1, \ldots, m/2$) contribute to the autocorrelation according to Equation 6.19 and the expression can be simplified to

$$E[Y_t Y_{t+h}] = 2 \sum_i \left(\mu_i \mu_{m-i+1} P_\varphi(i, m-i+1) \right) \qquad \text{where} \tag{C.6}$$

$$P_\varphi(i, j) = \begin{cases} 0, & \bar{b}_i + \bar{b}_j < 1 \\ 0, & \underline{b}_i + \underline{b}_j \geq 1 \\ \tau_i, & \text{otherwise.} \end{cases}$$

To simplify the following proof, we will combine Transformation 1 and Transformation 2 from Section 6.4 into a single transformation that adds a new state to an APH distribution in series canonical form and modifies the representation such that it is in canonical form again.

Transformation 3. *Let F_{aph} be an acyclic PH distribution in series canonical form with n states, initial probabilities $\boldsymbol{\pi}(1), \ldots, \boldsymbol{\pi}(n)$ and bidiagonal matrix $\boldsymbol{D}_0$ with rates $\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$. Then, the transformation constructs the distribution F'_{aph} by adding a new state $n+1$ with rate $\mu > \lambda_n$ as specified by Transformation 2 and by application of Transformation 1 to bring the representation into series canonical form.*

Obviously, Transformation 3 does not change the distribution, but only the representation of the distribution, since it only applies the two transformations from Section 6.4. For an elementary series with rates $\lambda_n \geq \lambda_{n-1} \geq \ldots \geq \lambda_i$ and probability $\boldsymbol{\pi}(i)$ Transformation 3 performs the following steps:

0)	$\lambda_n \geq \lambda_{n-1} \geq \ldots \geq \lambda_i$	$\boldsymbol{\pi}(i)$	$Transformation\ 2$
1)	$\mu > \lambda_n \geq \lambda_{n-1} \geq \ldots \geq \lambda_i$	$\boldsymbol{\pi}(i)(1-\lambda_n/\mu)$	
	$\mu > \lambda_{n-1} \geq \ldots \geq \lambda_i$	$\boldsymbol{\pi}(i)\lambda_n/\mu$	$Transformation\ 1$
2)	$\mu > \lambda_n \geq \lambda_{n-1} \geq \ldots \geq \lambda_i$	$(\boldsymbol{\pi}(i)\lambda_n/\mu)(1-\lambda_{n-1}/\lambda_n)$	
	$\mu > \lambda_n \geq \lambda_{n-2} \geq \ldots \geq \lambda_i$	$\boldsymbol{\pi}(i)\lambda_{n-1}/\mu$	$Transformation\ 1$
		$\vdots$	
	$\mu > \lambda_n \geq \lambda_{n-1} \geq \ldots \geq \lambda_i$	$(\boldsymbol{\pi}(i)\lambda_n/\mu)(1-\lambda_i/\lambda_{i+1})$	
	$\mu > \lambda_n \geq \lambda_{n-1} \geq \ldots \geq \lambda_{i+1}$	$\boldsymbol{\pi}(i)\lambda_i/\mu$	

In the first step the series is split into two series by adding a new state with rate $\mu > \lambda_n$. The first generated series consists of the rates $\mu > \lambda_n \geq \lambda_{n-1} \geq \ldots \geq \lambda_i$ and already resembles a basic series. Thus, is not modified by the subsequent application of Transformation 1. In the second series a phase with rate λ_n is missing to resemble a basic series and consequently Transformation 1 will treat the rates λ_n and λ_{n-1} in the second step. By repeated application of Transformation 1 one finally gets several series consisting of the rates $\mu > \lambda_n \geq \lambda_{n-1} \geq \ldots \geq \lambda_i$ and one series with rates $\mu > \lambda_n \geq \lambda_{n-1} \geq \ldots \geq \lambda_{i+1}$ and probability $\boldsymbol{\pi}(i)\lambda_i/\mu$. Therefore the sum of the probabilities for all the series with rates $\mu > \ldots \geq \lambda_i$ is $\boldsymbol{\pi}(i)(1-\lambda_i/\mu)$.

Note, that Transformation 3 preserves the order of the elementary series. Consider the two elementary series $(i, i+1)$ with mean values $\mu_i \leq \mu_{i+1}$. Series i consists of phases with rates $\lambda_n \geq \lambda_{n-1} \geq \ldots \geq \lambda_{j+1} \geq \lambda_j$ and series $i+1$ of phases with rates $\lambda_n \geq \lambda_{n-1} \geq \ldots \geq \lambda_{k+1} \geq \lambda_k$. Since the APH distribution is assumed to be in series canonical form series $i+1$ contains at least one phase (i.e. the initial phase) with rate λ_k that is not contained in series i. Therefore $\mu_i + (1/\lambda_k) \leq \mu_{i+1}$ must hold. Transformation 3 splits each of the two series into two new series. Let μ_i^- denote the smaller of the series originating from i, i.e. the one that contains rates $\mu > \lambda_n \geq \ldots \geq \lambda_{j+1}$, and μ_i^+ the larger of the series, i.e. the one that contains rates $\mu > \lambda_n \geq \ldots \geq \lambda_{j+1} \geq \lambda_j$. Then, after the transformation $\mu_i^- < \mu_i^+ \leq \mu_{i+1}^- < \mu_{i+1}^+$ holds, because $\mu_i^+ = \mu_i + (1/\mu) \leq \mu_{i+1} + (1/\mu) - (1/\lambda_k) = \mu_{i+1}^-$.

The above considerations on Transformation 3 will help to prove the following Lemma.

Lemma C.2. *For an APH transformation APH(n) with an even number of elementary series $1, \ldots, m$ with probabilities $\tau_i = \tau_{m-i+1}, i = 1, \ldots, m/2$ Transformation 3 increases the possible range of negative autocorrelation.*

Proof. It will be shown, that for each pair of series $(i, m-i+1)$ the joint moment $E[Y_t Y_{t+h}]$ is decreased and thus, the possible range of negative autocorrelation increased. Since we required $\tau_i = \tau_{m-i+1}, i = 1, \ldots, m/2$ and because of Theorems 6.1

and 6.2 it is sufficient to prove the Lemma for a base process autocorrelation of $\rho = -1$, i.e. we use Equation C.6 for the computation of $E[Y_t Y_{t+h}]$. Before the transformation the summand in Equation C.6 corresponding to $(i, m-i+1)$ is

$$\mu_i \mu_{m-i+1} \tau_i. \tag{C.7}$$

Let λ_j and λ_k be the rates of the first phase of the series i and $m-i+1$, respectively. By application of Transformation 3 a new phase with rate μ is introduced and series i is transformed into two series, one with mean $\mu_i^- = \mu_i + (1/\mu) - (1/\lambda_j)$ and probability $\tau_i \lambda_j/\mu$ and one with mean $\mu_i^+ = \mu_i + (1/\mu)$ and probability $\tau_i(1 - \lambda_j/\mu)$. Similarly, for the series $m-i+1$ we get two series with mean values $\mu_{m-i+1}^- = \mu_{m-i+1} + (1/\mu) - (1/\lambda_k)$ and $\mu_{m-i+1}^+ = \mu_{m-i+1} + (1/\mu)$, respectively, and probabilities $\tau_{m-i+1}\lambda_k/\mu$ and $\tau_{m-i+1}(1 - \lambda_k/\mu)$. Recall, that for a base process autocorrelation of $\rho = -1$ and for probabilities $\tau_i = \tau_{m-i+1}, i = 1, \ldots, m/2$ we only have to consider pairs of series $(i, m-i+1)$ with one smaller and one larger mean value that have the same probability. After the transformation we cannot guarantee, that the probabilities of the generated series are still equal. Hence, depending on the values of the probabilities of the generated series we have to distinguish three cases:

1. $\tau_i \frac{\lambda_j}{\mu} < \tau_{m-i+1}\left(1 - \frac{\lambda_k}{\mu}\right)$: Since $\tau_i = \tau_{m-i+1}$, this implies $\tau_i\left(1 - \frac{\lambda_j}{\mu}\right) > \tau_{m-i+1}\frac{\lambda_k}{\mu}$, i.e. the generated series with the smallest mean μ_i^- has a smaller probability than the generated series with the largest mean μ_{m-i+1}^+ and consequently, the probability corresponding to μ_{m-i+1}^- is smaller than the one corresponding to μ_i^+. By splitting the series with the larger probabilities according to Lemma C.1 we can ensure that we have three pairs of series with matching probabilities and are still able to use Equation C.6 for the computation of $E[Y_t Y_{t+h}]$. In particular, we split the series with mean μ_{m-i+1}^+ according to Lemma C.1 into two series with probabilities $\tau_i \frac{\lambda_j}{\mu}$ and $\tau_{m-i+1}\left(1 - \frac{\lambda_k}{\mu}\right) - \tau_i \frac{\lambda_j}{\mu}$. Similarly, we split the series with mean μ_i^+ into two series with probabilities $\tau_{m-i+1}\frac{\lambda_k}{\mu}$ and $\tau_i\left(1 - \frac{\lambda_j}{\mu}\right) - \tau_{m-i+1}\frac{\lambda_k}{\mu}$ resulting in three pairs of elementary series with matching probabilities. The contribution of these three pairs to Equation C.6 is

$$\begin{aligned}
& \mu_i^- \mu_{m-i+1}^+ \tau_i \frac{\lambda_j}{\mu} \\
& + \mu_i^+ \mu_{m-i+1}^+ \left(\tau_{m-i+1}\left(1 - \frac{\lambda_k}{\mu}\right) - \tau_i \frac{\lambda_j}{\mu}\right) \\
& + \mu_i^+ \mu_{m-i+1}^- \tau_{m-i+1} \frac{\lambda_k}{\mu} \\
= \; & \left(\mu_i + \frac{1}{\mu} - \frac{1}{\lambda_j}\right)\left(\mu_{m-i+1} + \frac{1}{\mu}\right)\tau_i \frac{\lambda_j}{\mu} \\
& + \left(\mu_i + \frac{1}{\mu}\right)\left(\mu_{m-i+1} + \frac{1}{\mu}\right)\left(\tau_i - \tau_i \frac{\lambda_k}{\mu} - \tau_i \frac{\lambda_j}{\mu}\right) \\
& + \left(\mu_i + \frac{1}{\mu}\right)\left(\mu_{m-i+1} + \frac{1}{\mu} - \frac{1}{\lambda_k}\right)\tau_i \frac{\lambda_k}{\mu} \\
= \; & \left(\mu_i \mu_{m-i+1} + \frac{\mu_i}{\mu} + \frac{\mu_{m-i+1}}{\mu} + \frac{1}{\mu\mu} - \frac{\mu_{m-i+1}}{\lambda_j} - \frac{1}{\lambda_j \mu}\right)\tau_i \frac{\lambda_j}{\mu}
\end{aligned}$$

$$
\begin{aligned}
&+\left(\mu_i\mu_{m-i+1}+\frac{\mu_i}{\mu}+\frac{\mu_{m-i+1}}{\mu}+\frac{1}{\mu\mu}\right)\left(\tau_i-\tau_i\frac{\lambda_k}{\mu}-\tau_i\frac{\lambda_j}{\mu}\right)\\
&+\left(\mu_i\mu_{m-i+1}+\frac{\mu_i}{\mu}-\frac{\mu_i}{\lambda_k}+\frac{\mu_{m-i+1}}{\mu}+\frac{1}{\mu\mu}-\frac{1}{\lambda_k\mu}\right)\tau_i\frac{\lambda_k}{\mu}\\
=\;&\mu_i\mu_{m-i+1}\tau_i-\frac{1}{\mu\mu}\tau_i.
\end{aligned}
$$

Note, that the first part is equal to Equation C.7 and the second part is less than zero. Therefore $E[Y_tY_{t+h}]$ is reduced for each pair of branches $(i, m-i+1)$.

2. $\tau_i\frac{\lambda_j}{\mu} = \tau_{m-i+1}\left(1-\frac{\lambda_k}{\mu}\right)$: This implies $\tau_i\left(1-\frac{\lambda_j}{\mu}\right) = \tau_{m-i+1}\frac{\lambda_k}{\mu}$. For this case no splitting of elementary series is necessary, because the series generated by the transformation already have matching probabilities. Since $\tau_i = \tau_{m-i+1}$, we have that $\tau_i\frac{\lambda_j}{\mu} = \tau_i\left(1-\frac{\lambda_k}{\mu}\right)$ implying that $\lambda_j+\lambda_k = \mu$, i.e. this case is only possible if μ is chosen such that it equals the sum of λ_j and λ_k. By using $\lambda_j+\lambda_k=\mu$ the contribution of the two pairs to Equation C.6 can be computed to

$$
\begin{aligned}
&\mu_i^-\mu_{m-i+1}^+\tau_i\frac{\lambda_j}{\mu}\\
&+\mu_i^+\mu_{m-i+1}^-\tau_i\left(1-\frac{\lambda_j}{\mu}\right)\\
=\;&\left(\mu_i+\frac{1}{\mu}-\frac{1}{\lambda_j}\right)\left(\mu_{m-i+1}+\frac{1}{\mu}\right)\tau_i\frac{\lambda_j}{\mu}\\
&+\left(\mu_i+\frac{1}{\mu}\right)\left(\mu_{m-i+1}+\frac{1}{\mu}-\frac{1}{\lambda_k}\right)\tau_i\left(1-\frac{\lambda_j}{\mu}\right)\\
=\;&\left(\mu_i\mu_{m-i+1}+\frac{\mu_{m-i+1}}{\mu}-\frac{\mu_{m-i+1}}{\lambda_j}+\frac{\mu_i}{\mu}+\frac{1}{\mu\mu}-\frac{1}{\lambda_j\mu}\right)\tau_i\frac{\lambda_j}{\mu}\\
&+\left(\mu_i\mu_{m-i+1}+\frac{\mu_{m-i+1}}{\mu}+\frac{\mu_i}{\mu}+\frac{1}{\mu\mu}-\frac{\mu_i}{\lambda_k}-\frac{1}{\mu\lambda_k}\right)\tau_i\left(1-\frac{\lambda_j}{\mu}\right)\\
=\;&\mu_i\mu_{m-i+1}\tau_i-\frac{1}{\mu\mu}\tau_i.
\end{aligned}
$$

3. $\tau_i\frac{\lambda_j}{\mu} > \tau_{m-i+1}\left(1-\frac{\lambda_k}{\mu}\right)$: This case is the symmetric counterpart to the first case. Since $\tau_i = \tau_{m-i+1}$, we have $\tau_i\left(1-\frac{\lambda_j}{\mu}\right) < \tau_{m-i+1}\frac{\lambda_k}{\mu}$, i.e. the generated series with smallest mean μ_i^- has a larger probability than the generated series with largest mean μ_{m-i+1}^+ and consequently, the probability corresponding to μ_{m-i+1}^- is greater than the one corresponding to μ_i^+. Again, we can split the series to obtain pairs with matching probabilities, i.e. we split the series with mean μ_i^- into two series with probabilities $\tau_{m-i+1}\left(1-\frac{\lambda_k}{\mu}\right)$ and $\tau_i\frac{\lambda_j}{\mu}-\tau_{m-i+1}\left(1-\frac{\lambda_k}{\mu}\right)$. Similarly, the series with mean μ_{m-i+1}^- is split into two series with probabilities $\tau_i\left(1-\frac{\lambda_j}{\mu}\right)$ and $\tau_{m-i+1}\frac{\lambda_k}{\mu}-\tau_i\left(1-\frac{\lambda_j}{\mu}\right)$. Then for the contribution of these three pairs to Equation C.6 we get

$$\mu_i^-\mu_{m-i+1}^+\tau_{m-i+1}\left(1-\frac{\lambda_k}{\mu}\right)$$

$$
\begin{aligned}
&+\mu_i^- \mu_{m-i+1}^- \left(\tau_i \frac{\lambda_j}{\mu} - \tau_{m-i+1}\left(1 - \frac{\lambda_k}{\mu}\right)\right) \\
&+\mu_i^+ \mu_{m-i+1}^- \tau_i \left(1 - \frac{\lambda_j}{\mu}\right) \\
= &\left(\mu_i + \frac{1}{\mu} - \frac{1}{\lambda_j}\right)\left(\mu_{m-i+1} + \frac{1}{\mu}\right)\tau_i\left(1 - \frac{\lambda_k}{\mu}\right) \\
&+\left(\mu_i + \frac{1}{\mu} - \frac{1}{\lambda_j}\right)\left(\mu_{m-i+1} + \frac{1}{\mu} - \frac{1}{\lambda_k}\right)\left(\tau_i \frac{\lambda_j}{\mu} - \tau_i\left(1 - \frac{\lambda_k}{\mu}\right)\right) \\
&+\left(\mu_i + \frac{1}{\mu}\right)\left(\mu_{m-i+1} + \frac{1}{\mu} - \frac{1}{\lambda_k}\right)\tau_i\left(1 - \frac{\lambda_j}{\mu}\right) \\
= &\left(\mu_i\mu_{m-i+1} + \frac{\mu_{m-i+1}}{\mu} - \frac{\mu_{m-i+1}}{\lambda_j} + \frac{\mu_i}{\mu} + \frac{1}{\mu\mu} - \frac{1}{\lambda_j\mu}\right)\left(\tau_i - \tau_i\frac{\lambda_k}{\mu}\right) \\
&+\left(\mu_i\mu_{m-i+1} + \frac{\mu_{m-i+1}}{\mu} - \frac{\mu_{m-i+1}}{\lambda_j} + \frac{\mu_i}{\mu} + \frac{1}{\mu\mu} - \frac{1}{\lambda_j\mu} - \frac{\mu_i}{\lambda_k} - \frac{1}{\lambda_k\mu} + \frac{1}{\lambda_j\lambda_k}\right) \\
&\qquad\left(\tau_i\frac{\lambda_j}{\mu} - \tau_i + \tau_i\frac{\lambda_k}{\mu}\right) \\
&+\left(\mu_i\mu_{m-i+1} + \frac{\mu_{m-i+1}}{\mu} + \frac{\mu_i}{\mu} + \frac{1}{\mu\mu} - \frac{\mu_i}{\lambda_k} - \frac{1}{\lambda_k\mu}\right)\left(\tau_i - \tau_i\frac{\lambda_j}{\mu}\right) \\
= &\ \mu_i\mu_{m-i+1}\tau_i - \frac{1}{\mu\mu}\tau_i - \left(\frac{1}{\lambda_j\lambda_k}\right)\tau_i\left(1 - \frac{\lambda_j}{\mu} - \frac{\lambda_k}{\mu}\right).
\end{aligned}
$$

Because $\tau_i\left(1 - \frac{\lambda_j}{\mu}\right) < \tau_i\frac{\lambda_k}{\mu}$ holds for this case, we have that,

$$\tau_i\left(1 - \frac{\lambda_j}{\mu} - \frac{\lambda_k}{\mu}\right) < \tau_i\left(1 - \frac{\lambda_j}{\mu} - \left(1 - \frac{\lambda_j}{\mu}\right)\right) = 0,$$

i.e. the term is always less than $\mu_i\mu_{m-i+1}\tau_i$.

In summary, if $\tau_i = \tau_{m-i+1}, i = 1, \ldots, m/2$ holds for the probabilities of the elementary series of an APH distribution the possible range of negative autocorrelation is increased for all cases that could occur when applying Transformation 3. □

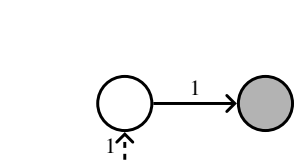
As one could see from the proof of Lemma C.2 already the simple case with matching probabilities of the elementary series before the transformation required the treatment of several different cases. In general, if we allow arbitrary probabilities for the elementary series, the situation gets even more complicated. If the probabilities of the series do not match, one series has to be paired with two or more other series. If for example the probability of the series with the largest mean is larger than the probability of the series with the smallest mean, the remaining probability is consumed by a pairing with the series with the second smallest mean and so on. In general we have to consider all the different cases from Equation 6.19. Moreover, after applying a transformation to the APH distribution, another case from Equation 6.19 might apply for the newly generated series depending on the probabilities and the newly introduced rate μ. When using Lemma C.1 instead to split the series of the APH distribution such that the series have matching probabilities, one encounters similar problems: For

Transformation 3 it was shown that the transformation preserves the order of the elementary series if the original distribution is in series canonical form, which implies that all series have different mean values. If we split one such series, we get two series with the same mean values. The transformation of a series always results in one series with a larger mean value and one series with a smaller mean value than the original series. Hence, if we transform the split series, the resulting series have to be ordered again according to their mean values to fulfill the requirements for CAPPs. This results in various different cases that have to be treated.
Hence, using either Equation 6.19 or Lemma C.1 results in the consideration of different cases, which makes the proof difficult for negative autocorrelations and APH distributions with arbitrary initial probabilities.
However, the examples shown in Appendix D suggest that the transformations increase the range of negative autocorrelation in the general case as well.

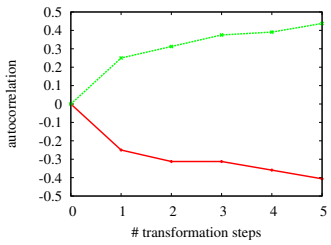
Appendix D

Examples for the APH Transformations

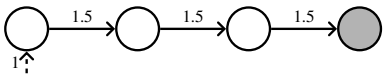
In Section 6.4 two transformations were proposed that can be used to modify the representation of an acyclic PH distribution in cases were the original representation is



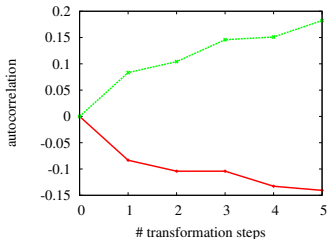
(a) Exponential



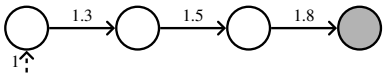
(b) Correlation for exponential



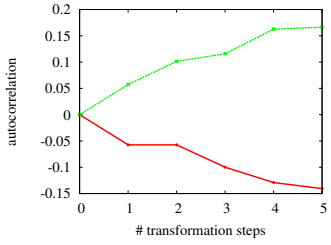
(c) Erlang



(d) Correlation for Erlang

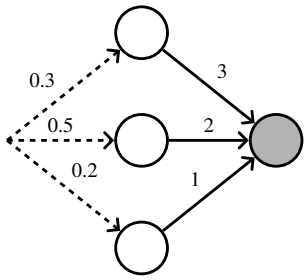


(e) Hypo-Exponential

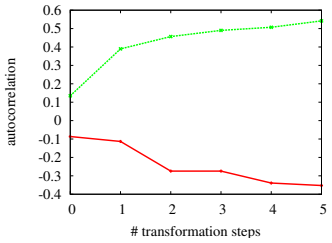


(f) Correlation for Hypo-Exponential

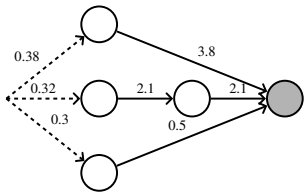
Figure D.1.: Effect on the range of autocorrelation for different APH distributions (1)



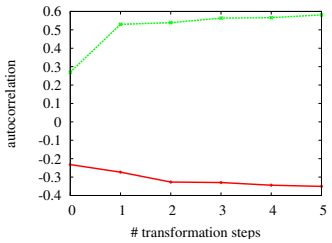
(a) Hyper-Exponential



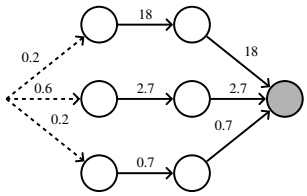
(b) Correlation for Hyper-Exponential



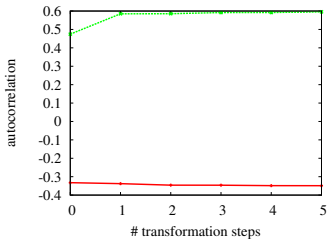
(c) Hyper-Erlang (order 5)



(d) Correlation for Hyper-Erlang (order 5)



(e) Hyper-Erlang (order 6)



(f) Correlation for Hyper-Erlang (order 6)

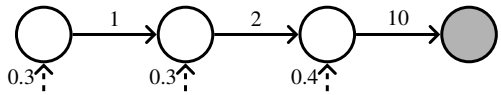
Figure D.2.: Effect on the range of autocorrelation for different APH distributions (2)

not sufficient to model the desired autocorrelation for a CHEP or CAPP. Figures D.1, D.2 and D.3 show the effect of these transformations on different PH distributions.

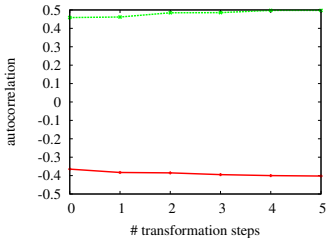
Figure D.1 shows the effect on the range of autocorrelation for three PH distributions, whose original representation cannot express non-zero autocorrelation when used for a CAPP, because the representation only contains a single elementary series. In particular exponential, Erlang and Hypo-Exponential distributions are presented. Figure D.1 shows the original representation on the left and the effect after up to five transformation steps when iterating between Transformation 1 and Transformation 2.

Figure D.2 shows the same results for different Hyper-Erlang distributions and finally, in Figure D.3 the effect for APH distributions in series canonical form is presented.

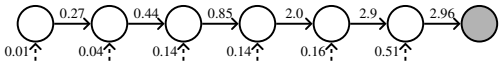
As one can see from the figures the possible range of autocorrelation is increased in every transformation step, although it should be noted, that usually the first steps have the largest impact on the range of autocorrelation, while the effect reduces for subsequent transformation steps, especially if the original representation of the distribution



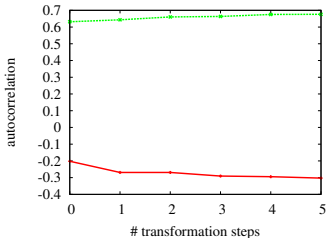
(a) $APH(3)$



(b) Correlation for $APH(3)$



(c) $APH(6)$



(d) Correlation for $APH(6)$

Figure D.3.: Effect on the range of autocorrelation for different APH distributions (3)

was already able to express a relatively high autocorrelation. For positive correlation the increase in the range of autocorrelation was already proven in Section 6.4, for negative correlation this was only shown for a special case in Section C.2. However, the examples presented here suggest, that the transformation also increases the possible range of negative autocorrelation.

Bibliography

[1] H. Akaike. A New Look at the Statistical Model Identification. *IEEE Transactions on Automatic Control*, 19(6):716–723, 1974.

[2] S. Asmussen and M. Bladt. Renewal Theory and Queueing Algorithms for Matrix-Exponential Distributions. In S. R. Chakravarthy and A. S. Alfa, editors, *Matrix-Analytic Methods in Stochastic Models*, volume 183 of *Lecture Notes in Pure and Applied Mathematics*, pages 313–341, New York, 1997.

[3] S. Asmussen and M. Bladt. Point Processes with Finite-dimensional Conditional Probabilities. *Stochastic Processes and their Applications*, 82(1):127–142, 1999.

[4] S. Asmussen and G. Koole. Marked Point Processes as Limits of Markovian Arrival Streams. *Journal of Applied Probability*, 30:365–372, 1993.

[5] S. Asmussen, O. Nerman, and M. Olsson. Fitting Phase-Type Distributions via the EM Algorithm. *Scandinavian Journal of Statistics*, 23(4):419–441, 1996.

[6] S. Asmussen and C. A. O'Cinneide. Matrix-Exponential Distributions - Distributions with a Rational Laplace Transform. In S. Kotz and C. Read, editors, *Encyclopedia of Statistical Sciences*, pages 435–440, New York, 1997. John Wiley & Sons.

[7] J. Banks. *Getting Started With AutoMod*. Brooks Automation, Inc., 2nd edition, 2004. ISBN 0-9729100-3-4.

[8] F. L. Bauer. Algorithm 60: Romberg Integration. *Communications of the ACM*, 4:255–, 1961.

[9] F. Bause. Doubly Stochastic and Circulant Structured Markovian Arrival Processes. Technical Report 824, Dep. of Computer Science, TU Dortmund, 2009.

[10] F. Bause, H. Beilner, and J. Kriege. ProC/B: Eine Modellierungsumgebung zur prozessketten-orientierten Beschreibung und Analyse logistischer Netze. In P. Buchholz and U. Clausen, editors, *Große Netze der Logistik - Die Ergebnisse*

des Sonderforschungsbereichs 559, chapter 2, pages 19–57. Springer Verlag, 2009.

[11] F. Bause, P. Buchholz, and J. Kriege. A Comparison of Markovian Arrival and ARMA/ARTA Processes for the Modeling of Correlated Input Processes. In M. D. Rossetti, R. R. Hill, B. Johansson, A. Dunkin, and R. G. Ingalls, editors, *Proceedings of the Winter Simulation Conference (WSC) 2009*, pages 634–645. Institute of Electrical and Electronics Engineers, Inc., 2009.

[12] F. Bause, P. Buchholz, and J. Kriege. ProFiDo - The Processes Fitting Toolkit Dortmund. In *Proceedings of the 7th International Conference on Quantitative Evaluation of SysTems (QEST 2010)*, pages 87–96. IEEE Computer Society, 2010.

[13] F. Bause, P. Buchholz, J. Kriege, and S. Vastag. A Framework for Simulation Models of Service-Oriented Architectures. In S. Kounev, I. Gorton, and K. Sachs, editors, *Performance Evaluation: Metrics, Models and Benchmarks. Proceedings of SPEC International Performance Evaluation Workshop (SIPEW 2008)*, volume 5119 of *LNCS*, pages 208–227, Darmstadt, 2008. Springer-Verlag.

[14] F. Bause, P. Gerloff, and J. Kriege. ProFiDo - A Toolkit for Fitting Input Models. In B. Müller-Clostermann, K. Echtle, and E. P. Rathgeb, editors, *Proceedings of the 15th International GI/ITG Conference on Measurement, Modelling and Evaluation of Computing Systems and Dependability and Fault Tolerance (MMB & DFT 2010)*, volume 5987 of *LNCS*, pages 311–314. Springer, 2010.

[15] F. Bause and G. Horvath. Fitting Markovian Arrival Processes by Incorporating Correlation into Phase Type Renewal Processes. In *Proceedings of the 7th International Conference on Quantitative Evaluation of SysTems (QEST 2010)*, pages 97–106. IEEE Computer Society, 2010.

[16] F. Bause and J. Kriege. *ProFiDo XML Interchange Format Specification*. Informatik IV, TU Dortmund, 2010.

[17] F. Bause and J. Kriege. *ProFiDo XML Configuration Format Specification*. Informatik IV, TU Dortmund, 2011.

[18] J. Beran, R. Sherman, M. S. Taqqu, and W. Willinger. Long-Range Dependence in Variable-Bit-Rate Video Traffic. *IEEE Transactions on Communications*, 43:1566–1579, 1995.

[19] D. S. Bernstein. *Matrix Mathematics: Theory, Facts, and Formulas with Application to Linear Systems Theory*. Princeton University Press, 2005.

[20] D. J. Best and D. E. Roberts. Algorithm AS 91: The Percentage Points of the χ^2 Distribution. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 24(3):385–388, 1975.

[21] B. Biller. Copula-Based Multivariate Input Models for Stochastic Simulation. *Operations Research*, 57(4):878–892, 2009.

[22] B. Biller and S. Ghosh. Dependence Modeling for Stochastic Simulation. In R. G. Ingalls, M. D. Rossetti, J. S. Smith, and B. A. Peters, editors, *Proceedings of the Winter Simulation Conference 2004*, pages 153–161, Piscataway, New Jersey, 2004. Institute of Electrical and Electronics Engineers, Inc.

[23] B. Biller and B. L. Nelson. Answers to the Top Ten Input Modeling Questions. In E. Yücesan, C.-H. Chen, J. L. Snowdon, and J. M. Charnes, editors, *Proceedings of the Winter Simulation Conference 2002*, 2002.

[24] B. Biller and B. L. Nelson. Modeling and Generating Multivariate Time-Series Input Processes using a Vector Autoregressive Technique. *ACM Transactions on Modeling and Computer Simulation*, 13(3):211–237, 2003.

[25] B. Biller and B. L. Nelson. Fitting Time-Series Input Processes for Simulation. *Operations Research*, 53(3):549–559, 2005.

[26] B. Biller and B. L. Nelson. Evaluation of the ARTAFIT Method for Fitting Time-Series Input Processes for Simulation. *INFORMS Journal on Computing*, 20(3):485–498, 2008.

[27] P. Billingsley. *Probability and Measure*. Wiley, New York, 3rd edition, 1995.

[28] J. Bilmes. A Gentle Tutorial on the EM Algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models. Technical Report TR-97-021, University of Berkeley, 1997.

[29] A. Bobbio and A. Cumani. ML Estimation of the Parameters of a PH Distribution in Triangular Canonical Form. In G. Balbo and G. Serazzi, editors, *Computer Performance Evaluation*, pages 33–46. Elsevier, 1992.

[30] L. Bodrog, P. Buchholz, J. Kriege, and M. Telek. Canonical Form Based MAP(2) Fitting. In *Proceedings of the 7th International Conference on Quantitative Evaluation of SysTems (QEST 2010)*, pages 107–116. IEEE Computer Society, 2010.

[31] L. Bodrog, A. Heindl, G. Horvath, and M. Telek. A Markovian Canonical Form of Second-Order Matrix-Exponential Processes. *European Journal of Operational Research*, 190(2):459–477, 2008.

[32] L. Bodrog, A. Horvath, and M. Telek. Moment Characterization of Matrix Exponential and Markovian Arrival Processes. *Annals of Operations Research*, 160:51–68, 2008.

[33] M. Bohge and M. Renwanz. A Realisitic VoIP Traffic Generation and Evaluation Tool for OMNeT++. In *Proceedings of the 1st International Workshop on OMNeT++*, 2008.

[34] G. E. P. Box, J. S. Hunter, and W. G. Hunter. *Statistics for Experimenters: Design, Innovation, and Discovery*. Wiley, 2nd edition, 2005.

[35] G. E. P. Box and G. M. Jenkins. *Time Series Analysis - Forecasting and Control*. Holden-Day, 1970.

[36] L. Breuer. An EM Algorithm for Batch Markovian Arrival Processes and its Comparison to a Simpler Estimation Procedure. *Annals of Operations Research*, 112:123–138, 2002.

[37] P. J. Brockwell and R. A. Davis. *Time Series: Theory and Methods*. Springer, 2nd edition, 1998.

[38] E. Brown, J. Place, and A. van de Liefvoort. Generating Matrix Exponential Random Variables. *Simulation*, 70:224–230, 1998.

[39] P. Buchholz. Structured Analysis Approaches for Large Markov Chains. *Applied Numerical Mathematics*, 31:375–404, December 1999.

[40] P. Buchholz. An EM-Algorithm for MAP Fitting from Real Traffic Data. In P. Kemper and W. H. Sanders, editors, *Computer Performance Evaluation / TOOLS*, volume 2794 of *Lecture Notes in Computer Science*, pages 218–236. Springer, 2003.

[41] P. Buchholz, P. Kemper, and J. Kriege. Multi Class Markovian Arrival Processes and Their Parameter Fitting. *Performance Evaluation*, 67(11):1092–1106, 2010.

[42] P. Buchholz and J. Kriege. A Heuristic Approach for Fitting MAPs to Moments and Joint Moments. In *Proceedings of the 6th International Conference on Quantitative Evaluation of SysTems (QEST 2009)*, pages 53–62, Los Alamitos, CA, USA, 2009. IEEE Computer Society.

[43] P. Buchholz and J. Kriege. Equivalence Transformations for Acyclic Phase Type Distributions. Technical Report 827, Dep. of Computer Science, TU Dortmund, 2009.

[44] P. Buchholz and A. Panchenko. A Two-Step EM Algorithm for MAP Fitting. In C. Aykanat, T. Dayar, and I. Korpeoglu, editors, *Proceedings of the 19th International Symposium on Computer and Information Sciences (ISCIS)*, volume 3280 of *Lecture Notes in Computer Science*, pages 217–227. Springer, 2004.

[45] P. Buchholz and A. Panchenko. An EM Algorithm for Fitting of Real Traffic Traces to PH-Distribution. *Proceedings of the International Conference on Parallel Computing in Electrical Engineering (PARELEC'2004)*, pages 283–288, 2004.

[46] J. P. Burg. A New Analysis Technique for Time Series Data. *Modern Spectrum Analysis*, pages 42–48, 1978.

[47] M. C. Cario and B. L. Nelson. Autoregressive To Anything: Time-Series Input Processes for Simulation. *Operations Research Letters*, 19(2):51–58, 1996.

[48] M. C. Cario and B. L. Nelson. Numerical Methods for Fitting and Simulating Autoregressive-To-Anything Processes. *INFORMS Journal on Computing*, 10(1):72–81, 1998.

[49] G. Casale, E. Z. Zhang, and E. Smirni. KPC-Toolbox: Simple Yet Effective Trace Fitting Using Markovian Arrival Processes. In *Proceedings of the Fifth International Conference on Quantitative Evaluation of Systems (QEST 2008)*, pages 83–92. IEEE Computer Society, 2008.

[50] G. Casale, E. Z. Zhang, and E. Smirni. Trace Data Characterization and Fitting for Markov Modeling. *Performance Evaluation*, 67(2):61–79, 2010.

[51] J. M. Chambers. *Software for Data Analysis: Programming with R*. Springer, New York, 2008. ISBN 978-0-387-75935-7.

[52] E. Cinlar. *Introduction to Stochastic Processes*. Prentice-Hall, New Jersey, 1975.

[53] D. R. Cox. A Use of Complex Probabilities in the Theory of Stochastic Processes. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 51, pages 313–319, 1955.

[54] M. E. Crovella and A. Bestavros. Self-Similarity in World Wide Web Traffic: Evidence and Possible Causes. In *Proceedings of the 1996 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS '96)*, pages 160–169, New York, NY, USA, 1996. ACM.

[55] A. Cumani. On the Canonical Representation of Homogeneous Markov Processes Modeling Failure-Time Distributions. *Micorelectronics and Reliability*, 22(3):583–602, 1982.

[56] D. J. DeBrota, S. D. Roberts, J. J. Swain, R. S. Dittus, J. R. Wilson, and S. Venkatraman. Input Modeling with the Johnson System of Distributions. In *Proceedings of the 20th Winter Simulation Conference (WSC '88)*, pages 165–179, New York, NY, USA, 1988. ACM.

[57] A. Dempster, N. Laird, and D. Rubin. Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977.

[58] J. L. Devore. *Probability and Statistics for Engineering and the Sciences*. Thomson, 7th edition, 2008.

[59] L. Devroye. *Non-Uniform Random Variate Generation*. Springer, New York, 1986.

[60] Z. Drezner and G. O. Wesolowsky. On the Computation of the Bivariate Normal Integral. *Journal of Statistical Computation and Simulation*, 35:101 – 107, 1990.

[61] A. K. Erlang. Solution of some Problems in the Theory of Probabilities of Significance in Automatic Telephone Exchanges. *The Post Office Electrical Engineer's Journal*, 1917.

[62] M. W. Fackrell. *Characterization of Matrix-Exponantial Distributions*. PhD thesis, School of Applied Mathmatics, University of Adelaide, 2003.

[63] Y. Fang. Hyper-Erlang Distribution Model and its Application in Wireless Mobile Networks. *Wireless Networks*, 7(3):211–219, 2001.

[64] A. Feldmann and W. Whitt. Fitting Mixtures of Exponentials to Long-Tail Distributions to Analyze Network Performance Models. *Performance Evaluation*, 31:245–279, 1998.

[65] W. Feller. *An Introduction to Probability Theory and Its Applications*, volume I, II. Wiley, New York, 1968.

[66] W. Fischer and K. Meier-Hellstern. The Markov-Modulated Poisson Process (MMPP) Cookbook. *Performance Evaluation*, 18(2):149–171, 1993.

[67] G. S. Fishman. *Concepts and Methods in Discrete Event Digital Simulation*. John Wiley, New York, 1973.

[68] S. Fitzgerald, J. Place, and A. van de Liefvoort. Generating Correlated Matrix Exponential Random Variables. *Advances in Engineering Software*, 37:75–84, February 2006.

[69] K. Goseva-Popstojanova and K. S. Trivedi. Effects of Failure Correlation on Software in Operation. In *Proceedings of the 2000 Pacific Rim International Symposium on Dependable Computing*, pages 69–76, 2000.

[70] E. J. Hannan and J. Rissanen. Recursive Estimation of Mixed Autoregressive-Moving Average Order. *Biometrika*, 69:81–94, 1982.

[71] B. Haverkort. *Performance of Computer Communication Systems - A Model-Based Approach*. Wiley, 1998.

[72] Q.-M. He. The Versatility of MMAP[K] and the MMAP[K]/G[K]/1 Queue. *Queueing Systems: Theory and Applications*, 38(4):397–418, 2001.

[73] Q.-M. He and M. Neuts. Markov Arrival Processes with Marked Transitions. *Stochastic Processes and their Applications*, 74:37–52, 1998.

[74] A. Heindl, G. Horvath, and K. Gross. Explicit Inverse Characterizations of Acyclic MAPs of Second Order. In A. Horvath and M. Telek, editors, *Formal Methods and Stochastic Models for Performance Evaluation, Proceedings of the Third European Performance Engineering Workshop (EPEW)*, volume 4054 of *Lecture Notes in Computer Science*, pages 108–122. Springer, 2006.

[75] T. R. Henderson, S. Roy, S. Floyd, and G. F. Riley. ns-3 Project Goals. In *Proceedings of the 2006 Workshop on ns-2: The IP Network Simulator*, New York, NY, USA, 2006. ACM.

[76] D. Heyman, A. Tabatabai, and T. V. Lakshman. Statistical Analysis and Simulation Study of Video Teletraffic in ATM Networks. In *IEEE Trans. Circuits and Systems for Video Technology 2*, pages 49–59, 1992.

[77] I. D. Hill. Algorithm AS 66: The Normal Integral. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 22(3):424–427, 1973.

[78] R. Hornig and A. Varga. An Overview of the OMNeT++ Simulation Environment. In *Proceedings of 1st International Conference on Simulation Tools and Techniques for Communications, Networks and Systems (SIMUTools)*, 2008.

[79] A. Horvath, G. Horvath, and M. Telek. A Traffic Based Decomposition of Two-class Queueing Network with Priority Service. *Computer Networks*, 53(8):1235–1248, 2009.

[80] A. Horvath and M. Telek. Markovian Modeling of Real Data Traffic: Heuristic Phase Type and MAP Fitting of Heavy Tailed and Fractal Like Samples. In M. C. Calzarossa and S. Tucci, editors, *Performance 2002*, volume 2459 of *LNCS*, pages 405–434. Springer, 2002.

[81] A. Horvath and M. Telek. PhFit: A General Purpose Phase Type Fitting Tool. In *Performance Tools 2002*, volume 2324 of *LNCS*, pages 82–91. Springer, 2002.

[82] G. Horvath and M. Telek. A Canonical Representation of Order 3 Phase Type Distributions. In K. Wolter, editor, *Formal Methods and Stochastic Models for Performance Evaluation*, volume 4748 of *LNCS*, pages 48–62. Springer, 2007.

[83] G. Horvath and M. Telek. On the Canonical Representation of Phase Type Distributions. *Performance Evaluation*, 66(8):396–409, 2009.

[84] G. Horvath, M. Telek, and P. Buchholz. A MAP Fitting Approach with Independent Approximation of the Inter-Arrival Time Distribution and the Lag-Correlation. In *Proceedings of the Second International Conference on Quantitative Evaluation of Systems (QEST)*, pages 124–133. IEEE CS Press, 2005.

[85] J. R. M. Hosking. Fractional Differencing. *Biometrika*, 68:165–176, 1981.

[86] H. E. Hurst. Long-Term Storage Capacity of Reservoirs. *Transactions of the American Society of Civil Engineers*, 116:770–799, 1951.

[87] D. L. Jagerman, B. Melamed, and W. Willinger. Stochastic Modeling of Traffic Processes. *Frontiers in Queueing: Models, Methods and Problems*, 1996.

[88] N. L. Johnson. Systems of Frequency Curves Generated by Methods of Translation. *Biometrika*, 36:149–176, 1949.

[89] N. L. Johnson and S. Kotz. *Continuous Univariate Distributions Vol. 1*. Distributions in Statistics. Houghton Mifflin, Boston, 1970.

[90] N. L. Johnson and S. Kotz. *Continuous Multivariate Distributions*. Distributions in Statistics. John Wiley & Sons, New York, 1972.

[91] K. V. Jonsson. HttpTools: A Toolkit for Simulation of Web Hosts in OMNeT++. In *Proceedings of the 2nd International Conference on Simulation Tools and Techniques (Simutools 09)*, 2009.

[92] M. I. Jordan and R. A. Jacobs. Hierarchical Mixtures of Experts and the EM Algorithm. *Neural Computation*, 6(2):181–214, 1994.

[93] S. M. Kay. Recursive Maximum Likelihood Estimation of Autoregressive Processes. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 31:56–65, 1983.

[94] W. D. Kelton, R. P. Sadowski, and D. T. Sturrock. *Simulation with Arena*. McGraw-Hill, 3rd edition, 2004. ISBN 0-07-291981-7.

[95] R. E. A. Khayari, R. Sadre, and B. Haverkort. Fitting World-Wide Web Request Traces with the EM-Algorithm. *Performance Evaluation*, 52:175–191, 2003.

[96] L. Kleinrock. *Queueing Systems - Volume 1: Theory*. John Wiley and Sons, 1975.

[97] A. Klemm, C. Lindemann, and M. Lohmann. Modeling IP Traffic Using the Batch Markovian Arrival Process. *Performance Evaluation*, 54(2):149–173, 2003.

[98] D. Krahl. The Extend Simulation Environment. In *Proceedings of the Winter Simulation Conference 2002*, 2002.

[99] D. Krahl. Extend: An Interactive Simulation Tool. In *Proceedings of the 35th Winter Simulation Conference (WSC '03)*, pages 188–196, 2003.

[100] J. Kriege and P. Buchholz. An Empirical Comparison of MAP Fitting Algorithms. In B. Müller-Clostermann, K. Echtle, and E. P. Rathgeb, editors, *Proceedings of the 15th International GI/ITG Conference on Measurement, Modelling and Evaluation of Computing Systems and Dependability and Fault Tolerance (MMB & DFT 2010)*, volume 5987 of *LNCS*, pages 259–273. Springer, 2010.

[101] J. Kriege and P. Buchholz. Simulating Stochastic Processes with OMNeT++. In *Proceedings of the 4th International OMNeT++ Workshop*, 2011.

[102] H. J. Larson and B. O. Shubert. *Probabilistic Models in Engineering Sciences*. John Wiley, 1979.

[103] G. Latouche and V. Ramaswami. *Introduction to Matrix Analytic Methods in Stochastic Modeling*. Society for Industrial and Applied Mathematics, 1999.

[104] A. M. Law and W. D. Kelton. *Simulation Modeling and Analysis*. McGraw-Hill, Boston, 3rd edition, 2000. ISBN 0-07-059292-6.

[105] A. M. Law and M. G. McComas. How The Expertfit Distribution-Fitting Software Can Make Your Simulation Models More Valid. In S. Chick, P. J. Sanchez, D. Ferrin, and D. J. Morris, editors, *Proceedings of the Winter Simulation Conference 2003*, pages 169–174, 2003.

[106] C. L. Lawson and B. J. Hanson. *Solving Least Squares Problems*. Prentice-Hall, 1974.

[107] L. Leemis. Input Modeling. In D. J. Medeiros, E. F. Watson, J. S. Carson, and M. S. Manivannan, editors, *Proceedings of the Winter Simulation Conference 1998*, pages 15–22, Piscataway, New Jersey, 1998. Institute of Electrical and Electronics Engineers, Inc.

[108] W. Leland, M. Taqqu, W. Willinger, and D. Wilson. On the Self-Similar Nature of Ethernet Traffic. *IEEE/ACM Transactions on Networking*, 2(1):1–15, 1994.

[109] M. Livny, B. Melamed, and A. K. Tsiolis. The Impact of Autocorrelation on Queueing Systems. *Management Science*, 39:322–339, 1993.

[110] D. M. Lucantoni, K. S. Meier-Hellstern, and M. F. Neuts. A Single Server Queue with Server Vacations and a Class of Non-Renewal Arrival Processes. *Advances in Applied Probability*, 22:676–705, 1990.

[111] H. Luetkepohl. *Introduction to Multiple Time Series Analysis*. Springer Verlag, New York, 1991.

[112] B. B. Mandelbrot and J. W. V. Ness. Fractional Brownian Motions, Fractional Noises and Applications. *SIAM Review*, 10:422–437, 1968.

[113] G. J. McLachlan and T. Krishnan. *The EM Algorithm and Extensions*. John Wiley and Sons, 1997.

[114] B. Melamed, J. R. Hill, and D. Goldsman. The TES Methodology: Modeling Empirical Stationary Time Series. In *Proceedings of the 24th Winter Simulation Conference (WSC '92)*, pages 135–144, New York, NY, USA, 1992. ACM.

[115] D. C. Montgomery. *Design and Analysis of Experiments*. John Wiley & Sons, 4th edition, 1997.

[116] L. Muscariello, M. Mellia, M. Meo, M. A. Marsan, and R. L. Cigno. Markov Models of Internet Traffic and a new Hierarchical MMPP Model. *Computer Communications*, 28(16):1835–1851, 2005.

[117] J. C. Nash. *Compact Numerical Methods for Computers: Linear Algebra and Function Minimisation*. Adam Hilger, 2nd edition, 1990.

[118] J. A. Nelder and R. Mead. A Simplex Method for Function Minimization. *Computer Journal*, 7:308–313, 1965.

[119] M. F. Neuts. A Versatile Markovian Point Process. *Journal of Applied Probability*, 16:764–779, 1979.

[120] M. F. Neuts. *Matrix-Geometric Solutions in Stochastic Models*. Johns Hopkins University Press, 1981.

[121] M. F. Neuts. *Algorithmic Probability: A Collection of Problems*. Chapman and Hall, 1995.

[122] Structured Markov Matrix Market. `http://www4.cs.tu-dortmund.de/~buchholz/struct-matrix-market.html`.

[123] C. A. O'Cinneide. On Non-Uniqueness of Representations of Phase-Type Distributions. *Stochastic Models*, 5:247–259, 1989.

[124] C. A. O'Cinneide. Characterization of Phase-Type Distributions. *Stochastic Models*, 6:1–57, 1990.

[125] C. A. O'Cinneide. Phase-Type Distributions: Open Problems and a Few Properties. *Stochastic Models*, 15(4):731–758, 1999.

[126] *OMNeT++ - Discrete Event Simulation System Version 4.0 User Manual*.

[127] A. Panchenko. *Modelling of Network Processes by Means of Markovian Arrival Processes*. PhD thesis, Fakultät Informatik, TU Dresden, 2007.

[128] A. Panchenko and P. Buchholz. A Hybrid Algorithm for Parameter Fitting of Markovian Arrival Processes. In *Proceedings of 14th International Conference on Analytical and Stochastic Modelling Techniques and Applications*, pages 7–12. SCS Press, 2007.

[129] A. Panchenko and A. Thümmler. Efficient Phase-Type Fitting with Aggregated Traffic Traces. *Performance Evaluation*, 64(7-8):629–645, 2007.

[130] V. Paxson and S. Floyd. Wide-Area Traffic: The Failure of Poisson Modeling. *IEEE/ACM Transactions on Networking*, 3:226–244, 1995.

[131] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C - The Art of Scientific Computing*. Cambridge University Press, 2nd edition, 1993.

[132] R. Pulungan and H. Hermanns. Effective Minimization of Acyclic Phase-Type Representations. In *Proceedings of the 15th international Conference on Analytical and Stochastic Modeling Techniques and Applications (ASMTA)*, pages 128–143, Berlin, Heidelberg, 2008. Springer-Verlag.

[133] G. Ramamurthy and B. Sengupta. Modeling and Analysis of a Variable Bit Rate Video Multiplexer. In *Proceedings of the Eleventh Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, pages 817–827, 1992.

[134] J. Rathmell and D. T. Sturrock. Arena: The Arena Product Family: Enterprise Modeling Solutions. In J. L. Snowdon and J. M. Charnes, editors, *Proceedings of the 34th Winter Simulation Conference*, pages 165–172. ACM, 2002.

[135] R. A. Redner and H. F. Walker. Mixture Densities, Maximum Likelihood and the EM Algorithm. *SIAM Review*, 26(2):195–239, 1984.

[136] P. Reinecke, M. Telek, and K. Wolter. Reducing the Cost of Generating APH-Distributed Random Numbers. In B. Müller-Clostermann, K. Echtle, and E. P. Rathgeb, editors, *Proceedings of the 15th International GI/ITG Conference on Measurement, Modelling and Evaluation of Computing Systems and Dependability and Fault Tolerance (MMB & DFT 2010)*, volume 5987 of *Lecture Notes in Computer Science*, pages 274–286. Springer, 2010.

[137] A. Riska, V. Diev, and E. Smirni. An EM-based Technique for Approximating Long-tailed Data Sets with PH Distributions. *Performance Evaluation*, 55:147–164, 2004.

[138] A. Riska and E. Riedel. Long-Range Dependence at the Disk Drive Level. In *Proceedings of the 3rd International Conference on Quantitative Evaluaiton of Systems (QEST 2006)*, pages 41–50, 2006.

[139] M. W. Rohrer and I. McGregor. AutoMod: Simulating Reality Using AutoMod. In J. L. Snowdon and J. M. Charnes, editors, *Proceedings of the 34th Winter Simulation Conference*, pages 173–181. ACM, 2002.

[140] R. Y. Rubinstein, G. Samorodnitsky, and M. Shaked. Antithetic Variates, Multivariate Dependence and Simulation of Stochastic Systems. *Management Science*, 31(1):66–77, 1985.

[141] S. M. Rudolfer and P. C. Watson. Evaluation of Orthant Probabilities for Singular Bi- and Trivariate Normal Distributions. *Journal of Statistical Computation and Simulation*, 48(3):219–231, 1993.

[142] T. Rydén. An EM Algorithm for Estimationin Markov Modulated Poisson Processes. *Computationals Statistics and Data Analysis*, 21:431–447, 1996.

[143] W. J. Stewart. *Introduction to the Numerical Solution of Markov Chains*. Princeton University Press, 1994.

[144] R. H. Storer, J. J. Swain, S. Venkatraman, and J. R. Wilson. Comparison Methods for Fitting Data Using Johnson Translation Distributions. In *Proceedings of the 20th Winter Simulation Conference (WSC '88)*, pages 476–481, New York, NY, USA, 1988. ACM.

[145] J. J. Swain, S. Venkatraman, and J. R. Wilson. Least-Squares Estimation of Distribution Functions in Johnson's Translation System. *Journal of Statistical Computation and Simulation*, 29(4):271–297, 1988.

[146] S. Tartarelli, M. Pagano, and M. Devetsikiotis. Efficient Estimation of the Cell Loss Probability in a Two-Buffer PGPS Scheduler. In *IEEE International Conference on ICC Communications (3)*, pages 1320–1324, 2000.

[147] M. Telek and G. Horvath. A Minimal Representation of Markov Arrival Processes and a Moments Matching Method. *Performance Evaluation*, 64(9-12):1153–1168, 2007.

[148] The Internet Traffic Archive. `http://ita.ee.lbl.gov/`.

[149] The R Project for Statistical Computing. `http://www.r-project.org/`.

[150] A. Thümmler, P. Buchholz, and M. Telek. A Novel Approach for Fitting Probability Distributions to Real Trace Data with the EM Algorithm. In *Proceedings of the International Conference on Dependable Systems and Networks (DSN)*, pages 712–721. IEEE Computer Society, 2005.

[151] A. Thümmler, P. Buchholz, and M. Telek. A Novel Approach for Phase-Type Fitting with the EM Algorithm. *IEEE Transactions on Dependable and Secure Computing*, 3(3):245–258, 2006.

[152] D. P. Tihansky. Properties of the Bivariate Normal Cumulative Distribution. *Journal of the American Statistical Association*, 67(340):903–905, 1972.

[153] Y. L. Tong. *The Multivariate Normal Distribution*. Springer Series in Statistics. Springer-Verlag, New York, 1990.

[154] N. Tran and D. A. Reed. Automatic ARIMA Time Series Modeling for Adaptive I/O Prefetching. *IEEE Transactions on Parallel and Distributed Systems*, 15:362–377, 2004.

[155] K. S. Trivedi. *Probability and Statistics with Reliability, Queuing and Computer Science Applications*. John Wiley and Sons, 2nd edition, 2002.

[156] E. G. Tsionas. Likelihood and Posterior Shapes in Johnson's S_B System. *Sankhya: The Indian Journal of Statistics*, 63(1):3–9, 2001.

[157] A. van de Liefvoort. The Moment Problem for Continuous Distributions. Technical Report WP-CM-1990-02, University of Missouri, Kansas City, 1990.

[158] S. Vincent. Input Data Analysis. In J. Banks, editor, *The Handbook of Simulation*, pages 55–91. John Wiley and Sons, New York, 1998.

[159] G. Walker. On Periodicity in Series of Related Terms. In *Proceedings of the Royal Society of London, A131*, pages 518–532, 1931.

[160] P. P. Ware, T. W. Page, Jr., and B. L. Nelson. Automatic Modeling of File System Workloads using Two-Level Arrival Processes. *ACM Transactions on Modeling and Computer Simulation*, 8(3):305–330, 1998.

[161] E. Weingärtner, H. Vom Lehn, and K. Wehrle. A Performance Comparison of Recent Network Simulators. In *Proceedings of the 2009 IEEE International Conference on Communications (ICC 2009)*, pages 1287–1291, Piscataway, NJ, USA, 2009. IEEE Press.

[162] R. E. Wheeler. Quantile Estimators of Johnson Curve Parameters. *Biometrika*, 67(3), 1980.

[163] M. J. Wichura. Algorithm AS 241: The Percentage Points of the Normal Distribution. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 37(3):477–484, 1988.

[164] C. Williamson. Synthetic Traffic Generation Techniques For ATM Network Simulations. *Simulation*, 72(5):305–312, 1999.

[165] F. Xue, J. Liu, Y. Shu, L. Zhang, and O. W. W. Yang. Traffic Modeling Based on FARIMA Models. In *Proceedings of the 1999 IEEE Canadian Conference on Electrical and Computer Engineering*, pages 162–167, 1999.

[166] G. U. Yule. On a Method of Investigating Periodicities in Disturbed Series, with Special Reference to Wolfer's Sunspot Numbers. *Royal Society of London Philosophical Transactions Series A*, 226:267–298, 1927.

List of Figures

List of Tables

List of Algorithms

Listings