FITTING INHOMOGENEOUS PHASE-TYPE DISTRIBUTIONS TO DATA: THE UNIVARIATE AND THE MULTIVARIATE CASE

HANSJÖRG ALBRECHER, MOGENS BLADT, AND JORGE YSLAS

ABSTRACT. The class of inhomogeneous phase-type distributions (IPH) was recently introduced in Albrecher and Bladt (2019) as an extension of the classical phase-type (PH) distributions. Like PH distributions, the class of IPH is dense in the class of distributions on the positive halfline, but leads to more parsimonious models in the presence of heavy tails. In this paper we propose a fitting procedure for this class to given data. We furthermore consider an analogous extension of Kulkarni's multivariate phase-type class (Kulkarni, 1989) to the inhomogeneous framework and study parameter estimation for the resulting new and flexible class of multivariate distributions. As a by-product, we amend a previously suggested fitting procedure for the homogeneous multivariate phase-type case and provide appropriate adaptations for censored data. The performance of the algorithms is illustrated in several numerical examples, both for simulated and real-life insurance data.

1. INTRODUCTION

The development, study and fitting of flexible distributions for random phenomena is an important branch of applied probability and statistics. Some respective approaches are based on a nice blend of theory and practice, among which the class of phase–type (PH) distributions is a prominent example. Originally initiated by Neuts (1975), the realization of a (univariate) phase–type distributed random variable is interpreted as the time until absorption of a time–homogeneous, finite state–space Markov jump process with one absorbing state and the rest being transient. The explicit description through matrix exponentials makes the resulting class of distributions at the same time versatile and analytically tractable (see e.g. Bladt and Nielsen (2017) for a recent survey). The class of phase–type distributions is known to be dense (in the sense of weak convergence) among all distributions on the positive halfline, but for distributions whose shape is very different from combinations of exponential components (which are the building blocks of the probabilistic Markov jump process construction), a suitable phase–type approximation will need a large dimension of the involved matrix (representing the number of phases of the underlying Markov process) and – in addition to computational challenges – may then be seen unnatural. This is particularly the case for heavy–tailed distributions, where the focus in modelling often lies on the tail of the distribution, and the latter is not well captured by the combination of exponential components of the PH construction. After some first amendment involving infinite–dimensional matrices was suggested in Bladt et al. (2015) and Bladt and Rojas-Nandayapa (2017), recently a new way to circumvent this problem was proposed in Albrecher and Bladt (2019). Concretely, when the Markov jump process is allowed to be time–inhomogeneous, one gains a lot of flexibility in terms of the structure of the individual components entering the matrix framework, which can reduce the complexity of appropriate fitting distributions drastically, in particular for distributions with heavy tails. The intensity matrices of the Markov jump process are then a function of time. In the general case, they may not commute at different time epochs, which complicates their statistical estimation due

to a lack of appropriate sufficient statistics. However, there is an important sub–class for which the intensity matrices can be written as a constant matrix scaled by some real function. In this class all matrices commute, and it was shown in Albrecher and Bladt (2019) that along this way one in fact obtains, for instance, Pareto, Weibull and Generalized Extreme Value (GEV) distributions with matrix-valued parameters. These distribution classes are all dense in the class of distributions on the positive halfline and inherit the computational advantages of the PH-type class, but also provide excellent fits for heavy–tailed data already for small dimensions, something that the original PH class could not achieve. In particular, if by some preliminary exploratory analysis one has a good guess for an appropriate scaling function (typically suggested by the empirical tail behavior), the resulting matrix distributions can be very parsimonious yet effective model improvements of the respective base distributions with a genuinely heavy tail. However, while parameter estimation for univariate PH distributions by a standard maximum likelihood procedure based on an EM algorithm has been studied in the seminal paper of Asmussen et al. (1996) (see also the later extension of Olsson (1996) dealing with censored observations and Bladt et al. (2003) for an MCMC approach), parameter estimation for the time–inhomogeneous case has not yet been addressed. PH distributions have played a crucial role in various application areas beyond finance and insurance, such as queueing theory, biology and operational research. The time–inhomogeneous extension is relatively new and we are confident that it may prove equally useful in those areas, in particular due to the flexible tail behaviour of this new class.

Motivated by the flexibility of the approach, in this paper we will also consider an inhomogeneous extension of the multivariate version of the PH distribution. The multivariate phase–type distribution (of MPH* type) was originally introduced by Kulkarni (1989) and is constructed as the joint distribution of certain state-dependent accumulated rewards earned on the same underlying Markov jump process. It has PH-distributed marginals and also enjoys a denseness property in the class of all distributions on the respective positive orthant. Multivariate phase–type distributions have found applications in diverse areas. For instance, Cai and Li (2005a) consider them for determining conditional tail expectations in risk management, Cai and Li (2005b) studied several types of ruin probabilities for a multivariate compound Poisson risk model when the claim size vector follows an MPH* distribution, and Herbertsson (2011) used this class to model default contagion in credit risk. More recently, Bladt et al. (2019) applied MPH* distributions for the calculation of Parisian type ruin probabilities. In terms of fitting of the (time–homogeneous) multivariate MPH* distribution, Ahlström et al. (1999) introduced an algorithm for a bivariate subclass of MPH*, and an EM algorithm for parameter estimation in the general case was proposed in Breuer (2016). However, the latter was not actually implemented and contains an inconsistency in the maximum likelihood estimator (which we amend in this paper).

The inhomogeneous extension of the MPH* to be proposed in this paper will then again serve the purpose of keeping the dimension of the involved matrices low when one faces a non-exponential behavior in the marginals and the joint multivariate behaviour. We would like to point out that an alternative analytically tractable deviation from exponential behavior utilizing Mittag-Leffler distributions in both the univariate and multivariate case can be found in Albrecher et al. (2020a,b,c). A number of commonly used heavy-tailed multivariate distributions are in fact transformed multivariate exponential distributions. For instance, Mardia (1962) was the first to systematically study multivariate Pareto distributions, which he introduced by transforming a Wicksell–Kibble–type multivariate exponential distribution (see Kibble (1941)). He also noticed that estimation methods for the multivariate exponential can then be translated directly towards the estimation of the multivariate Pareto distribution. Arnold (2015) presents some approaches to extend Mardia's analysis to obtain other multivariate distributions with Pareto marginals. Likewise, multivariate versions of the Weibull distribution have been obtained as power transforms of multivariate exponential distributions, see e.g. Lee (1979). The inhomogeneous MPH* extension that we propose in this

paper can to some extent be seen as a generalization and unification of these above models.

The main purpose of this paper is to provide algorithms for the statistical fitting of all these flexible classes of distributions and illustrate and discuss their implementation. We will present a unified maximum–likelihood based approach to fitting phase–type distributions (PH), inhomogeneous phase–type distributions (IPH), multivariate phase–type distributions (of MPH* type) and its newly introduced inhomogeneous extension. These classes contain a large number of mathematically tractable distributions that are sufficiently general to fit any non–negative data set, in the body and for both light or heavy tails. We will also consider extensions of the procedures to adapt for censored data and to the fitting of theoretically known joint distributions.

The structure of the paper is as follows. In Section 2 we provide an overview of the class of IPH distributions and present a new fitting procedure, which we then exemplify on two particular cases, one on a simulated data set and the other on actual data for lifetimes of the Danish population. In Section 3 we shortly recollect some facts about the MPH* class, review existing methods for parameter estimation and provide a substantiation and correction of an algorithm that was previously proposed in the literature. We then extend the algorithm to the case of censored observations, and give more details on an important particular bivariate subclass with explicit density. The section finishes with illustrations of the algorithms for a simulated bivariate sample as well as a phase–type approximation to a known bivariate exponential distribution. In Section 4 we introduce some multivariate extensions to distributions in the IPH class, derive basic properties, provide an EM algorithm for its parameter estimation and again illustrate its use in several examples, including multivariate matrix–Pareto models, multivariate matrix–Weibull models as well as a real data application to a bivariate Danish fire insurance data set. Section 5 concludes.

2. Inhomogeneous phase–type distributions

2.1. **Preliminaries.** Let $\{J_t\}_{t \geq 0}$ denote a time–inhomogeneous Markov jump process on a state space $\{1, \ldots, p, p+1\}$, where states $1, \ldots, p$ are transient and state $p+1$ is absorbing. Then $\{J_t\}_{t \geq 0}$ has an intensity matrix of the form

$$\mathbf{\Lambda}(t) = \begin{pmatrix} \boldsymbol{T}(t) & \boldsymbol{t}(t) \\ \mathbf{0} & 0 \end{pmatrix}, \quad t \geq 0,$$

where $\boldsymbol{T}(t)$ is a $p \times p$ matrix and $\boldsymbol{t}(t)$ is a p–dimensional column vector. Here, for any time $t \geq 0$, $\boldsymbol{t}(t) = -\boldsymbol{T}(t)\boldsymbol{e}$, where $\boldsymbol{e}$ is the p–dimensional column vector of ones. Let $\pi_k = \mathbb{P}(J_0 = k)$, $k = 1, \ldots, p$, $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_p)$ and assume that $\mathbb{P}(J_0 = p+1) = 0$. Then we say that the time until absorption

$$\tau = \inf\{t \geq 0 \mid J_t = p+1\}$$

has an inhomogeneous phase–type distribution with representation $(\boldsymbol{\pi}, \boldsymbol{T}(t))$ and we write $\tau \sim$ IPH$(\boldsymbol{\pi}, \boldsymbol{T}(t))$. If $\boldsymbol{T}(t) = \lambda(t)\boldsymbol{T}$, where $\lambda(t)$ is some known non–negative real function and $\boldsymbol{T}$ is a sub–intensity matrix, then we write $\tau \sim$ IPH$(\boldsymbol{\pi}, \boldsymbol{T}, \lambda)$. Note that for $\lambda(t) \equiv 1$ one returns to the time-homogeneous case, which corresponds to the conventional phase–type distribution with notation PH$(\boldsymbol{\pi}, \boldsymbol{T})$ (a comprehensive account of phase–type distributions can be found in Bladt and Nielsen (2017)). If $X \sim$ IPH$(\boldsymbol{\pi}, \boldsymbol{T}, \lambda)$, then there exists a function g such that

$$(2.1) \qquad\qquad\qquad\qquad\qquad X \sim g(Y),$$

where $Y \sim$ PH$(\boldsymbol{\pi}, \boldsymbol{T})$. Specifically, g is defined by

$$g^{-1}(x) = \int_0^x \lambda(t)dt$$

or, equivalently,

$$\lambda(t) = \frac{d}{dt}g^{-1}(t).$$

The density f_X and distribution function F_X for $X \sim \mathrm{IPH}(\boldsymbol{\pi}, \boldsymbol{T}, \lambda)$ are given by

$$
\begin{aligned}
f_X(x) &= \lambda(x)\,\boldsymbol{\pi}\exp\left(\int_0^x \lambda(t)dt\,\boldsymbol{T}\right)\boldsymbol{t}\,, \\
F_X(x) &= 1 - \boldsymbol{\pi}\exp\left(\int_0^x \lambda(t)dt\,\boldsymbol{T}\right)\boldsymbol{e}\,.
\end{aligned}
$$

For further reading on inhomogeneous phase–type distributions and motivations for their use in modelling we refer to Albrecher and Bladt (2019). For the representation of some IPH distributions, we make use of functional calculus. If h is an analytic function and $\boldsymbol{A}$ is a matrix, we define

$$h(\boldsymbol{A}) = \frac{1}{2\pi i}\oint_\gamma h(z)(z\boldsymbol{I} - \boldsymbol{A})^{-1}dz\,,$$

where γ is a simple path enclosing the eigenvalues of $\boldsymbol{A}$ (cf. (Bladt and Nielsen, 2017, Sec. 3.4) for details). Another standard way to define an analytic matrix-valued function is using the corresponding series expansion, while for non-analytic but sufficiently smooth functions $h(\boldsymbol{A})$ can be defined using the Jordan decomposition of $\boldsymbol{A}$. We refer to Higham (2008) for these and further equivalent ways to define $h(\boldsymbol{A})$.

As illustrated in Albrecher and Bladt (2019), a number of IPH distributions can be expressed as classical distributions with matrix-valued parameter. Important examples include the transformation $g(y) = \beta\,(e^y - 1)$ for $\beta > 0$ in (2.1) leading to a matrix–Pareto distribution with density function and survival function

$$(2.2) \qquad f_X(x) = \boldsymbol{\pi}\left(\frac{x}{\beta} + 1\right)^{\boldsymbol{T}-\boldsymbol{I}}\boldsymbol{t}\,\frac{1}{\beta}\,, \quad \bar{F}_X(x) = 1 - F_X(x) = \boldsymbol{\pi}\left(\frac{x}{\beta} + 1\right)^{\boldsymbol{T}}\boldsymbol{e}\,,$$

respectively, as well as the matrix–Weibull distribution with density and survival function

$$f_X(x) = \boldsymbol{\pi}e^{\boldsymbol{T}x^\beta}\boldsymbol{t}\beta x^{\beta-1}\,, \quad \bar{F}_X(x) = \boldsymbol{\pi}e^{\boldsymbol{T}x^\beta}\boldsymbol{e}\,,$$

obtained from $g(y) = y^{1/\beta}$ ($\beta > 0$), see Albrecher and Bladt (2019) for further details.

2.2. **Parameter estimation.** For the matrix–Pareto distribution (2.2) and $\beta = 1$, the transform is parameter-independent, so that the distribution can be fitted to i.i.d. data $x_1, \dots, x_N$ by fitting a phase–type distribution $\mathrm{PH}(\boldsymbol{\pi}, \boldsymbol{T})$ to the transformed data $\log(1 + x_1), \dots, \log(1 + x_N)$ using an EM algorithm (Asmussen et al., 1996). This was the procedure employed in Albrecher and Bladt (2019) for the numerical illustration there. The general case – where the transform does depend on parameters – is more subtle and shall be dealt with here. The key will be to apply a parameter-dependent transformation in each step of the EM algorithm.

Let $x_1, \dots, x_N$ be an i.i.d. sample of an inhomogeneous phase–type distribution with representation $X \sim \mathrm{IPH}(\boldsymbol{\pi}, \boldsymbol{T}, \lambda(\,\cdot\,;\boldsymbol{\beta}))$, where $\lambda(\,\cdot\,;\boldsymbol{\beta})$ is a parametric non–negative function depending on the vector $\boldsymbol{\beta}$. We then know that $X \stackrel{d}{=} g(Y;\boldsymbol{\beta})$ with $Y \sim \mathrm{PH}(\boldsymbol{\pi}, \boldsymbol{T})$ and g is defined in terms of its inverse function $g^{-1}(x;\boldsymbol{\beta}) = \int_0^x \lambda(t;\boldsymbol{\beta})dt$. In particular $g^{-1}(X;\boldsymbol{\beta}) \stackrel{d}{=} Y \sim \mathrm{PH}(\boldsymbol{\pi}, \boldsymbol{T})$. The EM algorithm for fitting $\mathrm{IPH}(\boldsymbol{\pi}, \boldsymbol{T}, \lambda(\,\cdot\,;\boldsymbol{\beta}))$ then works as follows.

Algorithm 2.2.1 (EM algorithm for transformed phase–type distributions)**.**
0. Initialize with some "arbitrary" $(\boldsymbol{\pi}, \boldsymbol{T}, \boldsymbol{\beta})$.
1. Transform the data into $y_i = g^{-1}(x_i;\boldsymbol{\beta})$, $i = 1, \dots, N$, and apply the E– and M–steps of the conventional EM algorithm of Asmussen et al. (1996) by which we obtain the estimators $(\hat{\boldsymbol{\pi}}, \hat{\boldsymbol{T}})$.

2. Compute

$$\hat{\boldsymbol{\beta}} = \arg\max_{\boldsymbol{\beta}} \sum_{i=1}^{N} \log(f_X(x_i; \hat{\boldsymbol{\pi}}, \hat{\boldsymbol{T}}, \boldsymbol{\beta})) = \arg\max_{\boldsymbol{\beta}} \sum_{i=1}^{N} \log\left(\lambda(x_i; \boldsymbol{\beta})\hat{\boldsymbol{\pi}} \exp\left(\int_0^{x_i} \lambda(t; \boldsymbol{\beta})dt\ \hat{\boldsymbol{T}}\right)\hat{\boldsymbol{t}}\right).$$

3. Assign $(\boldsymbol{\pi}, \boldsymbol{T}, \boldsymbol{\beta}) = (\hat{\boldsymbol{\pi}}, \hat{\boldsymbol{T}}, \hat{\boldsymbol{\beta}})$ *and GOTO 1.*

Then the likelihood function increases for each iteration, and hence converges to a (possibly local) maximum.

Proof. Since the data points x_i are assumed to be i.i.d. realisations from the unknown distribution $\mathrm{IPH}(\boldsymbol{\pi}, \boldsymbol{T}, \lambda)$, there exists a function g such that $y_i = g^{-1}(x_i; \boldsymbol{\beta})$ are i.i.d. realisations of phase–type distributed random variables $\mathrm{PH}(\boldsymbol{\pi}, \boldsymbol{T})$. That function g is assumed to be known up to the value of $\boldsymbol{\beta}$. In turn, $x_i = g(y_i; \boldsymbol{\beta})$, so a data point x_i can be interpreted as the absorption time of the Markov jump process corresponding to $\mathrm{PH}(\boldsymbol{\pi}, \boldsymbol{T})$, which is y_i, but with the scale of the time axis for the y_i–data converted (stretched) into $g(\cdot; \boldsymbol{\beta})$–coordinates instead. The full data likelihood is then given by

$$L(\boldsymbol{\pi}, \boldsymbol{T}, \boldsymbol{\beta}; \boldsymbol{y}) = \prod_{k=1}^{p} \pi_k^{B_k} \prod_{k=1}^{p}\prod_{l\neq k} t_{kl}^{N_{kl}} e^{-t_{kl}Z_k(\boldsymbol{\beta})} \prod_{k=1}^{p} t_k^{N_k} e^{-t_k Z_k(\boldsymbol{\beta})},$$

where B_k is the number of times the Markov process underlying the phase–type distribution initiates in state k, N_{kl} denotes the total number of transitions from state k to l, N_k denotes the number of times an exit to the absorbing state was caused by a jump from state k, and $Z_k(\boldsymbol{\beta})$ is the total time the Markov process has spent in state k. We notice that $Z_k(\boldsymbol{\beta})$ is the only sufficient statistic which depends on the transformation of the time axis for the y–data and hence on $\boldsymbol{\beta}$. Consequently, for any given $\boldsymbol{\beta}$, the E–step is simply the one as in Asmussen et al. (1996), and so is the M–step for $(\boldsymbol{\pi}, \boldsymbol{T})$.

The $\boldsymbol{\beta}$ update in 2. requires a general, usually numerical, maximization of the incomplete data likelihood. Each iteration of the algorithm increases the likelihood. Indeed, let L^I denote the incomplete data likelihood, i.e.,

$$L^I(\boldsymbol{\pi}, \boldsymbol{T}, \boldsymbol{\beta}; \boldsymbol{x}) = \prod_{i=1}^{N} f_X(x_i; \boldsymbol{\pi}, \boldsymbol{T}, \boldsymbol{\beta}) = \prod_{i=1}^{N} \lambda(x_i; \boldsymbol{\beta})\boldsymbol{\pi}\exp\left(g^{-1}(x_i; \boldsymbol{\beta})\ \boldsymbol{T}\right)\boldsymbol{t},$$

and consider parameter values $(\boldsymbol{\pi}_n, \boldsymbol{T}_n, \boldsymbol{\beta}_n)$ after the n-th iteration. In the $(n+1)$-th iteration, we first obtain $(\boldsymbol{\pi}_{n+1}, \boldsymbol{T}_{n+1})$ in 1. so that

$$\prod_{i=1}^{N} \boldsymbol{\pi}_n \exp\left(g^{-1}(x_i; \boldsymbol{\beta}_n)\ \boldsymbol{T}_n\right)\boldsymbol{t}_n \leq \prod_{i=1}^{N} \boldsymbol{\pi}_{n+1} \exp\left(g^{-1}(x_i; \boldsymbol{\beta}_n)\ \boldsymbol{T}_{n+1}\right)\boldsymbol{t}_{n+1},$$

By monotonicity of g and the transformation theorem,

$$L^I(\boldsymbol{\pi}_n, \boldsymbol{T}_n, \boldsymbol{\beta}_n; \boldsymbol{x}) \leq L^I(\boldsymbol{\pi}_{n+1}, \boldsymbol{T}_{n+1}, \boldsymbol{\beta}_n; \boldsymbol{x})$$

and hence, by 2.,

$$L^I(\boldsymbol{\pi}_n, \boldsymbol{T}_n, \boldsymbol{\beta}_n; \boldsymbol{x}) \leq L^I(\boldsymbol{\pi}_{n+1}, \boldsymbol{T}_{n+1}, \boldsymbol{\beta}_n; \boldsymbol{x}) \leq \sup_{\boldsymbol{\beta}} L^I(\boldsymbol{\pi}_{n+1}, \boldsymbol{T}_{n+1}, \boldsymbol{\beta}; \boldsymbol{x}) = L^I(\boldsymbol{\pi}_{n+1}, \boldsymbol{T}_{n+1}, \boldsymbol{\beta}_{n+1}; \boldsymbol{x}).$$

$\square$

Example 2.2.1. (Matrix–Gompertz) Let $X = \log(\beta Y + 1)/\beta$, where $Y \sim \mathrm{PH}(\boldsymbol{\pi}, \boldsymbol{T})$ and $\beta > 0$. Then

(2.3) $$\bar{F}_X(x) = \boldsymbol{\pi} e^{\boldsymbol{T}(e^{\beta x}-1)/\beta}\boldsymbol{e} \quad \text{and} \quad f_X(x) = \boldsymbol{\pi} e^{\boldsymbol{T}(e^{\beta x}-1)/\beta}\boldsymbol{t}e^{\beta x}.$$

We refer to the distribution of X as a *matrix–Gompertz distribution*, since the scale parameter of the usual Gompertz distribution is now replaced by a matrix. Note that the resulting distribution has a lighter tail than a conventional phase–type distribution. The Gompertz distribution is used in a number of applications, most notably it is historically used for the modelling of human lifetimes (Gompertz, 1825). Its matrix version (2.3) provides a natural flexible extension. As an illustration, we fitted a matrix–Gompertz distribution with 3 phases using Algorithm 2.2.1 with 2 500 iterations to the lifetime of the Danish population that died in the year 2 000 at ages 50 to 100 (data obtained from the Human Mortality Database (HMD) and available in the R-package *MortalitySmooth* (Camarda, 2012)). Here and in later examples, the number of iterations in the algorithm is chosen in such a way that the changes in the successive log–likelihoods become negligible. Concerning running times, our implementation makes use of the gradient ascent method for the maximization part of the algorithm, in which the running times highly depend on the step–length and the actually chosen stopping criterion. In the present example we employed a step–length of 10^{-8} and run gradient ascent until the absolute value of the derivative is less than 0.001 leading to a running time of about 35 seconds on a usual PC (with 2.9 GHz Dual–Core Intel Core i5 processor 5287U) for the 2 500 iterations of the EM algorithm. Note that this choice of stopping criterion is to prioritize precision over speed, and an improvement on running times can be attained by using a different maximization procedure. The obtained parameters are as follows:

$$\hat{\boldsymbol{\pi}} = (0.0450,\ 0.1303,\ 0.8246)\,,$$

$$\hat{\boldsymbol{T}} = \begin{pmatrix} -0.1357 & 0.1214 & 0 \\ 0.0130 & -0.0421 & 0.0288 \\ 0.1415 & 0.0184 & -0.1620 \end{pmatrix}\,,$$

$$\hat{\beta} = 0.1019\,.$$

Figure 2.1 shows that the fitted density recovers the structure of the data quite well. Note that conventional phase–type distributions have been used to model the distribution of lifespans (see for instance Asmussen et al. (2019)). However, the number of phases required to capture the tail behavior of the data with the latter is rather large, due to the lighter than exponential tail. In contrast, the matrix–Gompertz distribution provides an excellent fit with comparably fewer parameters (phases).
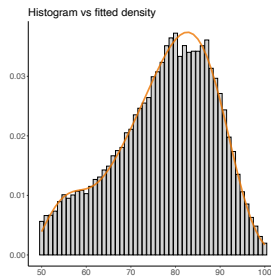

FIGURE 2.1. Histogram of lifetimes of the Danish population that died in the year 2 000 at ages 50 to 100 versus the density of the fitted matrix–Gompertz distribution.

Example 2.2.2. (Matrix–GEV) Algorithm 2.2.1 can also be applied to estimate distributions that are not IPH in a strict sense, but that are defined as a transformation of a PH distribution. This is for instance the case for $g(y) = \mu - \sigma(y^{-\xi} - 1)/\xi$ with $\mu \in \mathbb{R}$, $\sigma > 0$, and $\xi \in \mathbb{R}$. Recall from

Albrecher and Bladt (2019) that

$$
F_X(x) = \begin{cases} \boldsymbol{\pi} \exp\left(\boldsymbol{T}\left(1 + \xi\frac{x-\mu}{\sigma}\right)^{-1/\xi}\right)\boldsymbol{e}, & \xi \neq 0, \\[3mm] \boldsymbol{\pi} \exp\left(\boldsymbol{T}\exp\left(-\frac{x-\mu}{\sigma}\right)\right)\boldsymbol{e}, & \xi = 0, \end{cases}
$$

$$
f_X(x) = \begin{cases} \dfrac{1}{\sigma}\boldsymbol{\pi}\exp\left(\boldsymbol{T}\left(1 + \xi\frac{x-\mu}{\sigma}\right)^{-1/\xi}\right)\boldsymbol{t}\left(1 + \xi\frac{x-\mu}{\sigma}\right)^{-(1+\xi)/\xi}, & \xi \neq 0, \\[3mm] \dfrac{1}{\sigma}\boldsymbol{\pi}\exp\left(\boldsymbol{T}\exp\left(-\frac{x-\mu}{\sigma}\right)\right)\boldsymbol{t}\exp\left(-\frac{x-\mu}{\sigma}\right), & \xi = 0, \end{cases}
$$

from which it becomes clear that this distribution can be interpreted as a matrix version of the generalized extreme value (GEV) distribution, see e.g. Beirlant et al. (2004). As an illustration, we generated an i.i.d. sample of size $5\,000$ from such a distribution of 3 phases with parameters

$$
\boldsymbol{\pi} = (1,\, 0,\, 0)\,,
$$

$$
\boldsymbol{T} = \begin{pmatrix} -1 & 0.5 & 0 \\ 0.2 & -2 & 0.8 \\ 1 & 1 & -5 \end{pmatrix},
$$

$$
\mu = 2\,, \quad \sigma = 0.5\,, \quad \xi = 0.4\,,
$$

which has theoretical moments $\mathbb{E}(X) = 2.2524$ and $SD(X) = 1.4423$. The generated sample has moments $\hat{\mathbb{E}}(X) = 2.2607$ and $\hat{SD}(X) = 1.3307$. We then fitted such a matrix–GEV distribution with the same number of phases using Algorithm 2.2.1 with $1\,500$ steps, obtaining the following parameters:

$$
\hat{\boldsymbol{\pi}} = (0.0772,\, 0.1268,\, 0.7960)\,,
$$

$$
\hat{\boldsymbol{T}} = \begin{pmatrix} -8.9772 & 0.0964 & 0.0001 \\ 0.2891 & -2.8439 & 0.3542 \\ 3.2353 & 0.0137 & -5.7731 \end{pmatrix},
$$

$$
\hat{\mu} = 1.3852\,, \quad \hat{\sigma} = 0.2285\,, \quad \hat{\xi} = 0.4251\,.
$$

We observe that the algorithm estimates pretty well the shape parameter ξ, which determines the heaviness of the tail. Moreover, the fitted distribution has moments $\mathbb{E}(X) = 2.2640$ and $SD(X) = 1.6587$, which resemble the ones of the sample, and Figure 2.2 shows that the algorithm recovers both body and tail of the data. Note also that the log–likelihood of the fitted matrix–GEV is $-4\,104.541$, while the log–likelihood using the original matrix–GEV distribution is $-4\,107.005$. Such a comparison of the log–likelihoods works as additional evidence for the performance of the algorithm. One can observe that the parameters estimated for $\boldsymbol{\pi}$, $\boldsymbol{T}$, μ and σ do not resemble the original parameter values, but this is linked with the well-known identifiability issue for phase–type distributions (namely that other parameter combinations may lead to a very similar density shape). In fact, the algorithm finds the parameters that maximize the likelihood for the given sample, and as the concrete numbers above show, the present parameters even outperform the original model underlying the sample(!), see also the convincing QQ–plot in Figure 2.2. Here, the step–length is 10^{-5} and the gradient ascent is run until the norm of the derivative is less than 0.1 leading to a running time of $2\,609$ seconds for the $1\,500$ iterations of Algorithm 2.2.1.
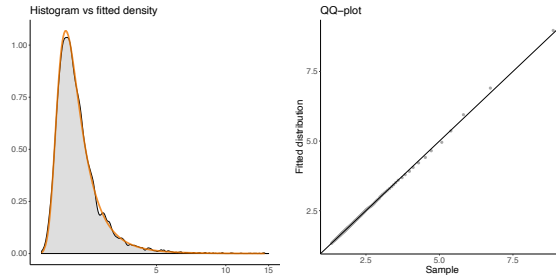
FIGURE 2.2. Histogram of simulated sample versus density of the fitted matrix–GEV distribution in log–scale (left) as well as QQ–plot of simulated sample versus fit (right).

3. Multivariate phase-type distributions

3.1. **Preliminaries.** Let $\tau \sim \mathrm{PH}(\boldsymbol{\pi}, \boldsymbol{T})$ be a (conventional) p–dimensional phase–type distributed random variable with underlying time–homogeneous Markov jump process $\{J_t\}_{t \geq 0}$. Let $\boldsymbol{r}_j = (r_j(1), \ldots, r_j(p))'$ be non–negative p–dimensional column vectors, $j = 1, \ldots, d$, and let

$$\boldsymbol{R} = (\boldsymbol{r}_1, \boldsymbol{r}_2, \ldots, \boldsymbol{r}_d)$$

be a $p \times d$–dimensional *reward matrix*. Now define

$$Y^{(j)} = \int_0^\tau r_j(J_t)\, dt$$

for all $j = 1, \ldots, d$. If we interpret $r_j(k)$ as the rate at which a reward is obtained while J_t is in state k, then $Y^{(j)}$ is the total reward for component j obtained prior to absorption. We then say that the random vector $\boldsymbol{Y} = (Y^{(1)}, \ldots, Y^{(d)})'$ has a multivariate phase–type distribution of the MPH* type (as defined in Kulkarni (1989), see also Bladt and Nielsen (2017)) and we write $\boldsymbol{Y} \sim \mathrm{MPH}^*(\boldsymbol{\pi}, \boldsymbol{T}, \boldsymbol{R})$.

While each member of the MPH* class has an explicit expression for the (joint) Laplace transform and the joint moments of any order (see Section 8.1.1 of Bladt and Nielsen (2017)), there are no general explicit expressions for the density and distribution functions. However, for certain structures and sub–classes explicit expressions for the latter do exist (like Example 8.1.13 of Bladt and Nielsen (2017)).

If $\boldsymbol{Y} = (Y^{(1)}, \ldots, Y^{(d)})' \sim \mathrm{MPH}^*(\boldsymbol{\pi}, \boldsymbol{T}, \boldsymbol{R})$, then each marginal $Y^{(j)}$ has a phase–type distribution, $\mathrm{PH}(\boldsymbol{\pi}_j, \boldsymbol{T}_j)$ say. First we decompose

$$\boldsymbol{r}_j = \begin{pmatrix} \boldsymbol{r}_j^+ \\ \boldsymbol{r}_j^0 \end{pmatrix}, \quad \boldsymbol{\pi} = (\boldsymbol{\pi}^+ \ \boldsymbol{\pi}^0) \quad \text{and} \quad \boldsymbol{T} = \begin{pmatrix} \boldsymbol{T}^{++} & \boldsymbol{T}^{+0} \\ \boldsymbol{T}^{0+} & \boldsymbol{T}^{00} \end{pmatrix},$$

where we have reordered the state space such that the $+$ terms correspond to the states k for which the rewards $r_k^{(j)}$ are strictly positive, and the 0 terms to the states with zero rewards. E.g., $\boldsymbol{T}_{+0}$ corresponds to the intensities by which the underlying Markov jump process $\{J_t\}_{t \geq 0}$ jumps from a state with positive reward to a state with zero reward. Then the phase–type distribution of $Y^{(j)}$ is given by an atom at zero of size $\boldsymbol{\pi}_0(\boldsymbol{I} - (-\boldsymbol{T}_{00})^{-1}\boldsymbol{S}_{0+})\boldsymbol{e}$, where $\boldsymbol{e}$ is the column vector of ones of appropriate dimension, and

$$(3.1) \quad \boldsymbol{\pi}_j = \boldsymbol{\pi}^+ + \boldsymbol{\pi}^0(-\boldsymbol{T}^{00})^{-1}\boldsymbol{T}^{0+} \quad \text{together with} \quad \boldsymbol{T}_j = \boldsymbol{\Delta}(\boldsymbol{r}_j^+)^{-1}\left(\boldsymbol{T}^{++} + \boldsymbol{T}^{+0}(-\boldsymbol{T}_{00})^{-1}\boldsymbol{T}^{0+}\right),$$

where $\Delta(\boldsymbol{a})$ denotes the $d' \times d'$ diagonal matrix with entries $a^{(m)}$, $m = 1, \ldots, d'$, from a d'–dimensional vector $\boldsymbol{a}$. The atom appears in case there is a positive probability of starting in a non–reward–earning state (0) and the underlying Markov process gets absorbed before visiting a reward earning state ($+$). The Markov jump process generating $Y^{(j)}$ starts in the same state as

J_t if the reward is positive (hence $\boldsymbol{\pi}_+$) or it starts in the first state with positive rewards that J_t enters after starting in a zero reward state (hence the term $\boldsymbol{\pi}^0(-\boldsymbol{T}^{00})^{-1}\boldsymbol{T}^{0+}$). Similar arguments apply to the generator $\boldsymbol{T}_j$, where only reward-earning terms will form part of the state space for $Y^{(j)}$. We refer to Bladt and Nielsen (2017) for further details.

Summarizing, each marginal $Y^{(j)}$ has a phase–type distribution, which is based on the original Markov process $\{J_t\}_{t\geq 0}$, but with a possibly smaller state space and with rescaled parameters.

3.2. **Parameter estimation.** We next provide an algorithm for estimating MPH* distributed data. The data consist of a d–dimensional multivariate sample of N i.i.d. observations

$$\boldsymbol{y}_i = (y_i^{(1)}, \ldots, y_i^{(d)})', \quad i = 1, \ldots, N\,.$$

That is, we only observe the times to absorption, $y_i^{(j)}$, of each phase–type distributed marginal. Hence we are clearly in an incomplete data set–up and we shall employ the EM algorithm for fitting $(\boldsymbol{\pi}, \boldsymbol{T}, \boldsymbol{R})$.

The EM algorithm works by replacing unavailable sufficient statistics by their conditional expectations given data under given parameters, and thereby updating the parameters by using known formulas for the maximum likelihood estimator in the complete data domain. Iteration of the procedure then produces a sequence of parameter values which increases the likelihood in each step.

For the present situation, we define the complete data as both the trajectories of the underlying Markov process which generates the phase–type distribution from which the marginals of the multivariate vector are constructed, and the Markov jump processes representing the rewards in all marginal distributions. It is not sufficient with complete knowledge of the marginal trajectories only. Indeed, one can easily construct examples where the underlying processes cannot be reconstructed from the marginals only. The complete knowledge of both marginals and the underlying Markov process which generates the marginals creates another problem in relation to the incomplete data since we do not have observations for the absorption times of the underlying Markov process. We can get around this problem by assuming that the rows of the reward matrix $\boldsymbol{R}$ sum to one, i.e. $\boldsymbol{Re} = \boldsymbol{e}$. This assumption is not restrictive and can be imposed without losing generality due to the great ambiguity of (multivariate) phase–type representations. Hence our data consists of marginals $\boldsymbol{y}^{(j)} = (y_1^{(j)}, \ldots, y_N^{(j)})'$, $j = 1, \ldots, d$, and their sums $\boldsymbol{y}^{(S)} = \sum_{l=1}^d \boldsymbol{y}^{(l)}$.

In the complete data domain, the estimation is straightforward and works as follows. Using the notation introduced in the proof of Algorithm 2.2.1, the complete data MLE for $(\boldsymbol{\pi}, \boldsymbol{T})$ is given by

$$\hat{\pi}_k = \frac{B_k}{N}\,, \quad \hat{t}_{kl} = \frac{N_{kl}}{Z_k}\,, \quad \hat{t}_k = \frac{N_k}{Z_k}\,, \quad \hat{t}_{kk} = -\sum_{l\neq k} \hat{t}_{kl} - \hat{t}_k\,.$$

The rewards of the marginals are then given by

$$\hat{r}_j(k) = \frac{Z_k^{(j)}}{Z_k} = \frac{Z_k^{(j)}}{\sum_{l=1}^d Z_k^{(l)}}\,,$$

where $Z_k^{(j)} = r_j(k)Z_k$ is the over-all amount of time the j'th component has spent in state k.

In the EM algorithm, we now must replace all aforementioned sufficient statistics by their conditional expectations given data. Concerning Z_k, N_{kl}, N_k and B_k, these only depend on the underlying Markov jump process and are computed conditionally on $\boldsymbol{y}^{(S)}$ only. Their formulas are then as stated in the algorithm below (see Asmussen et al. (1996)).

Concerning the conditional expectation of $Z_k^{(j)}$, we must calculate the expected reward (under $(\boldsymbol{\pi}, \boldsymbol{T})$) given all data of marginal j, which amounts to calculating the conditional expected time given data for the corresponding phase–type representation of the j-th marginal, $(\boldsymbol{\pi}_j, \boldsymbol{T}_j)$. These

are readily given by (again using Asmussen et al. (1996))

$$\mathbb{E}\left(Z_k^{(j)} \mid \boldsymbol{Y}^{(j)} = \boldsymbol{y}^{(j)}\right) = \sum_{i=1}^{N} \frac{\int_0^{y_i^{(j)}} \boldsymbol{e}_k' \mathrm{e}^{\boldsymbol{T}_j(y_i^{(j)}-u)} \boldsymbol{t}_j \boldsymbol{\pi}_j \mathrm{e}^{\boldsymbol{T}_j u} \boldsymbol{e}_k du}{\boldsymbol{\pi}_j \mathrm{e}^{\boldsymbol{T}_j y_i^{(j)}} \boldsymbol{t}_j} \, .$$

Then

$$\hat{r}_j(k) = \frac{\mathbb{E}\left(Z_k^{(j)} \mid \boldsymbol{Y}^{(j)} = \boldsymbol{y}^{(j)}\right)}{\mathbb{E}\left(Z_k \mid \boldsymbol{Y}^{(S)} = \boldsymbol{y}^{(S)}\right)}.$$

Iterating the above finally provides a (single) full EM algorithm for the estimation of $(\boldsymbol{\pi}, \boldsymbol{T}, \boldsymbol{R})$. We summarize the results in the following.

Algorithm 3.2.1 (EM algorithm for MPH* distributions)**.**

0. Initialize with some "arbitrary" $(\boldsymbol{\pi}, \boldsymbol{T}, \boldsymbol{R})$ with $\boldsymbol{Re} = \boldsymbol{e}$, and compute $\boldsymbol{\pi}_j$ and $\boldsymbol{T}_j$, $j = 1, \ldots, d$, using (3.1).

1. (E–step) Calculate

$$\mathbb{E}\left(B_k \mid \boldsymbol{Y}^{(S)} = \boldsymbol{y}^{(S)}\right) = \sum_{i=1}^{N} \frac{\pi_k \boldsymbol{e}_k' \mathrm{e}^{\boldsymbol{T} y_i^{(S)}} \boldsymbol{t}}{\boldsymbol{\pi} \mathrm{e}^{\boldsymbol{T} y_i^{(S)}} \boldsymbol{t}}$$

$$\mathbb{E}\left(Z_k \mid \boldsymbol{Y}^{(S)} = \boldsymbol{y}^{(S)}\right) = \sum_{i=1}^{N} \frac{\int_0^{y_i^{(S)}} \boldsymbol{e}_k' \mathrm{e}^{\boldsymbol{T}(y_i^{(S)}-u)} \boldsymbol{t} \boldsymbol{\pi} \mathrm{e}^{\boldsymbol{T} u} \boldsymbol{e}_k du}{\boldsymbol{\pi} \mathrm{e}^{\boldsymbol{T} y_i^{(S)}} \boldsymbol{t}}$$

$$\mathbb{E}\left(N_{kl} \mid \boldsymbol{Y}^{(S)} = \boldsymbol{y}^{(S)}\right) = \sum_{i=1}^{N} t_{kl} \frac{\int_0^{y_i^{(S)}} \boldsymbol{e}_l' \mathrm{e}^{\boldsymbol{T}(y_i^{(S)}-u)} \boldsymbol{t} \boldsymbol{\pi} \mathrm{e}^{\boldsymbol{T} u} \boldsymbol{e}_k du}{\boldsymbol{\pi} \mathrm{e}^{\boldsymbol{T} y_i^{(S)}} \boldsymbol{t}}$$

$$\mathbb{E}\left(N_k \mid \boldsymbol{Y}^{(S)} = \boldsymbol{y}^{(S)}\right) = \sum_{i=1}^{N} t_k \frac{\boldsymbol{\pi} \mathrm{e}^{\boldsymbol{T} y_i^{(S)}} \boldsymbol{e}_k}{\boldsymbol{\pi} \mathrm{e}^{\boldsymbol{T} y_i^{(S)}} \boldsymbol{t}}$$

$$\mathbb{E}\left(Z_k^{(j)} \mid \boldsymbol{Y}^{(j)} = \boldsymbol{y}^{(j)}\right) = \sum_{i=1}^{N} \frac{\int_0^{y_i^{(j)}} \boldsymbol{e}_k' \mathrm{e}^{\boldsymbol{T}_j(y_i^{(j)}-u)} \boldsymbol{t}_j \boldsymbol{\pi}_j \mathrm{e}^{\boldsymbol{T}_j u} \boldsymbol{e}_k du}{\boldsymbol{\pi}_j \mathrm{e}^{\boldsymbol{T}_j y_i^{(j)}} \boldsymbol{t}_j} \, .$$

2. (M–step) Let

$$\hat{\pi}_k = \frac{1}{N} \mathbb{E}\left(B_k \mid \boldsymbol{Y}^{(S)} = \boldsymbol{y}^{(S)}\right), \quad \hat{t}_{kl} = \frac{\mathbb{E}\left(N_{kl} \mid \boldsymbol{Y}^{(S)} = \boldsymbol{y}^{(S)}\right)}{\mathbb{E}\left(Z_k \mid \boldsymbol{Y}^{(S)} = \boldsymbol{y}^{(S)}\right)}, \quad \hat{t}_k = \frac{\mathbb{E}\left(N_k \mid \boldsymbol{Y}^{(S)} = \boldsymbol{y}^{(S)}\right)}{\mathbb{E}\left(Z_k \mid \boldsymbol{Y}^{(S)} = \boldsymbol{y}^{(S)}\right)},$$

$$\hat{t}_{kk} = -\sum_{l \neq k} \hat{t}_{kl} - \hat{t}_k, \quad \hat{\boldsymbol{\pi}} = (\hat{\pi}_1, \ldots, \hat{\pi}_p), \quad \hat{\boldsymbol{T}} = \{\hat{t}_{kl}\}_{k,l=1,\ldots,p} \ \ and \ \ \hat{\boldsymbol{t}} = (\hat{t}_1, \ldots, \hat{t}_p)'.$$

and

$$\hat{r}_j(k) := \frac{\mathbb{E}\left(Z_k^{(j)} \mid \boldsymbol{Y}^{(j)} = \boldsymbol{y}^{(j)}\right)}{\mathbb{E}\left(Z_k \mid \boldsymbol{Y}^{(S)} = \boldsymbol{y}^{(S)}\right)} = \frac{\mathbb{E}\left(Z_k^{(j)} \mid \boldsymbol{Y}^{(j)} = \boldsymbol{y}^{(j)}\right)}{\sum_{l=1}^{d} \mathbb{E}\left(Z_k^{(l)} \mid \boldsymbol{Y}^{(l)} = \boldsymbol{y}^{(l)}\right)} \quad and \quad \hat{\boldsymbol{R}} = \{\hat{r}_j(k)\}_{k=1,\ldots,p, j=1,\ldots,d} \, .$$

3. Assign $\boldsymbol{\pi} := \hat{\boldsymbol{\pi}}, \boldsymbol{T} := \hat{\boldsymbol{T}}, \boldsymbol{t} := \hat{\boldsymbol{t}}, \boldsymbol{R} := \hat{\boldsymbol{R}}$ and compute $\boldsymbol{\pi}_j, \boldsymbol{T}_j, j = 1, \ldots, d$, using (3.1). GOTO 1.

Remark 3.2.1. Algorithm 3.2.1 was originally proposed in Breuer (2016) as two consecutive EM algorithms and its original statement contained a minor error in the M–step update for the reward matrix. To see why Algorithm 3.2.1 can be decomposed into the two consecutive EM algorithms, we argue as follows. Running the EM Algorithm 3.2.1, $(\hat{\boldsymbol{\pi}}, \hat{\boldsymbol{T}})$ will eventually converge (without input from the part involving the reward components). For constant $(\hat{\boldsymbol{\pi}}, \hat{\boldsymbol{T}})$, Algorithm 3.2.1 is indeed equivalent to the second EM algorithm in Breuer (2016). More specifically, the algorithm takes the following form.

First EM.

0. Initialize with some "arbitrary" $(\boldsymbol{\pi}, \boldsymbol{T})$.
1. (E–step) Calculate

$$\mathbb{E}\left(B_k \mid \boldsymbol{Y}^{(S)} = \boldsymbol{y}^{(S)} \right) = \sum_{i=1}^{N} \frac{\pi_k \boldsymbol{e}_k' \mathrm{e}^{\boldsymbol{T} y_i^{(S)}} \boldsymbol{t}}{\boldsymbol{\pi} \mathrm{e}^{\boldsymbol{T} y_i^{(S)}} \boldsymbol{t}}$$

$$\mathbb{E}\left(Z_k \mid \boldsymbol{Y}^{(S)} = \boldsymbol{y}^{(S)} \right) = \sum_{i=1}^{N} \frac{\int_0^{y_i^{(S)}} \boldsymbol{e}_k' \mathrm{e}^{\boldsymbol{T}(y_i^{(S)}-u)} \boldsymbol{t} \boldsymbol{\pi} \mathrm{e}^{\boldsymbol{T} u} \boldsymbol{e}_k du}{\boldsymbol{\pi} \mathrm{e}^{\boldsymbol{T} y_i^{(S)}} \boldsymbol{t}}$$

$$\mathbb{E}\left(N_{kl} \mid \boldsymbol{Y}^{(S)} = \boldsymbol{y}^{(S)} \right) = \sum_{i=1}^{N} t_{kl} \frac{\int_0^{y_i^{(S)}} \boldsymbol{e}_l' \mathrm{e}^{\boldsymbol{T}(y_i^{(S)}-u)} \boldsymbol{t} \boldsymbol{\pi} \mathrm{e}^{\boldsymbol{T} u} \boldsymbol{e}_k du}{\boldsymbol{\pi} \mathrm{e}^{\boldsymbol{T} y_i^{(S)}} \boldsymbol{t}}$$

$$\mathbb{E}\left(N_k \mid \boldsymbol{Y}^{(S)} = \boldsymbol{y}^{(S)} \right) = \sum_{i=1}^{N} t_k \frac{\boldsymbol{\pi} \mathrm{e}^{\boldsymbol{T} y_i^{(S)}} \boldsymbol{e}_k}{\boldsymbol{\pi} \mathrm{e}^{\boldsymbol{T} y_i^{(S)}} \boldsymbol{t}} \,.$$

2. (M–step) Let

$$\hat{\pi}_k = \frac{1}{N} \mathbb{E}\left(B_k \mid \boldsymbol{Y}^{(S)} = \boldsymbol{y}^{(S)} \right), \quad \hat{t}_{kl} = \frac{\mathbb{E}\left(N_{kl} \mid \boldsymbol{Y}^{(S)} = \boldsymbol{y}^{(S)} \right)}{\mathbb{E}\left(Z_k \mid \boldsymbol{Y}^{(S)} = \boldsymbol{y}^{(S)} \right)}, \quad \hat{t}_k = \frac{\mathbb{E}\left(N_k \mid \boldsymbol{Y}^{(S)} = \boldsymbol{y}^{(S)} \right)}{\mathbb{E}\left(Z_k \mid \boldsymbol{Y}^{(S)} = \boldsymbol{y}^{(S)} \right)},$$

$$\hat{t}_{kk} = -\sum_{l \neq k} \hat{t}_{kl} - \hat{t}_k, \quad \hat{\boldsymbol{\pi}} = (\hat{\pi}_1, \ldots, \hat{\pi}_p), \quad \hat{\boldsymbol{T}} = \{\hat{t}_{kl}\}_{k,l=1,\ldots,p} \quad \text{and} \quad \hat{\boldsymbol{t}} = (\hat{t}_1, \ldots, \hat{t}_p)'.$$

3. Assign $\boldsymbol{\pi} := \hat{\boldsymbol{\pi}}$, $\boldsymbol{T} := \hat{\boldsymbol{T}}$, $\boldsymbol{t} := \hat{\boldsymbol{t}}$ and GOTO 1.

Second EM. Use the estimated $(\boldsymbol{\pi}, \boldsymbol{T})$ of the first EM.

0. Initialize with some "arbitrary" $\boldsymbol{R}$ with $\boldsymbol{R}\boldsymbol{e} = \boldsymbol{e}$, and compute $\boldsymbol{\pi}_j$ and $\boldsymbol{T}_j$, $j = 1, \ldots, d$, using (3.1).
1. (E–step) Calculate

$$\mathbb{E}\left(Z_k^{(j)} \mid \boldsymbol{Y}^{(j)} = \boldsymbol{y}^{(j)} \right) = \sum_{i=1}^{N} \frac{\int_0^{y_i^{(j)}} \boldsymbol{e}_k' \mathrm{e}^{\boldsymbol{T}_j(y_i^{(j)}-u)} \boldsymbol{t}_j \boldsymbol{\pi}_j \mathrm{e}^{\boldsymbol{T}_j u} \boldsymbol{e}_k du}{\boldsymbol{\pi}_j \mathrm{e}^{\boldsymbol{T}_j y_i^{(j)}} \boldsymbol{t}_j} \,.$$

2. (M–step) Let

$$\hat{r}_j(k) := \frac{\mathbb{E}\left(Z_k^{(j)} \mid \boldsymbol{Y}^{(j)} = \boldsymbol{y}^{(j)} \right)}{\sum_{l=1}^{d} \mathbb{E}\left(Z_k^{(l)} \mid \boldsymbol{Y}^{(l)} = \boldsymbol{y}^{(l)} \right)} \quad \text{and} \quad \hat{\boldsymbol{R}} = \{\hat{r}_j(k)\}_{k=1,\ldots,p, j=1,\ldots,d} \,.$$

3. Assign $\boldsymbol{R} := \hat{\boldsymbol{R}}$ and compute $\boldsymbol{\pi}_j$ and $\boldsymbol{T}_j$, $j = 1, \ldots, d$, using (3.1). GOTO 1.

Remark 3.2.2. The main computational burden lies in the E–steps, where matrix exponentials and integrals thereof must be evaluated. In Asmussen et al. (1996) this is done by converting the problem into a system of ODEs, which are then solved via a Runge–Kutta method of fourth order (a C implementation, called EMpht, is available online (Olsson, 1998)). While this approach is adequate for fitting univariate phase–type distributions, the Runge–Kutta method fails to work in some cases in the multivariate setting, in particular for the second EM, when an element in the reward matrix approaches zero. The reason is that the sub-intensity matrix of (at least) one of the marginals will adjust to this change by increasing some of the entries of the matrix in each iteration, and thus requiring an increasingly smaller step–size in the Runge–Kutta method to accurately approximate the solution to the system. Our implementation includes an approach for the computation of matrix exponentials based on uniformization, and it is a slight variation of the method in (Neuts, 1995, p.232). We explain briefly the method. By taking $\phi = \max(-t_{kk})_{k=1,\ldots,p}$ and defining $\boldsymbol{P} := \phi^{-1}\left(\phi \boldsymbol{I} + \boldsymbol{T}\right)$, which is in fact a transition matrix, we have that

$$\exp(\boldsymbol{T}y) = \sum_{n=0}^{\infty} \frac{(\phi y)^n}{n!} e^{-\phi y} \boldsymbol{P}^n \,.$$

Then

$$\left| e^{\boldsymbol{T}y} - \sum_{n=0}^{M} \frac{(\phi y)^n}{n!} e^{-\phi y} \boldsymbol{P}^n \right| \le \sum_{n=M+1}^{\infty} \frac{(\phi y)^n}{n!} e^{-\phi y} |\boldsymbol{P}^n| \le \sum_{n=M+1}^{\infty} \frac{(\phi y)^n}{n!} e^{-\phi y} = \mathbb{P}(N_{\phi y} > M) \,,$$

where $N_{\phi y}$ is Poisson distributed with mean ϕy. Hence, we can find M such that the difference of the matrix exponential with a finite sum is less than or equal to a given error $\epsilon > 0$. Of course, larger values of ϕy give bigger values of M, dismissing any computational improvement for large observations. A way to circumvent this problem is to observe that $e^{\boldsymbol{T}y} = (e^{\boldsymbol{T}y/2^m})^{2^m}$, thus we can find m such that $\phi y/2^m < 1$, compute $e^{\boldsymbol{T}y/2^m}$ by a finite sum and then retrieve $e^{\boldsymbol{T}y}$ by squaring.

To compute the integrals involving matrix exponentials, we observe that by defining

$$\boldsymbol{G}(y; \boldsymbol{\pi}, \boldsymbol{T}) := \int_0^y \mathrm{e}^{\boldsymbol{T}(y-u)} \boldsymbol{t}\boldsymbol{\pi} \mathrm{e}^{\boldsymbol{T}u} du \,,$$

we have that (see Van Loan (1978))

$$\exp\left(\left(\begin{array}{cc} \boldsymbol{T} & \boldsymbol{t}\boldsymbol{\pi} \\ \boldsymbol{0} & \boldsymbol{T} \end{array}\right) y\right) = \left(\begin{array}{cc} \mathrm{e}^{\boldsymbol{T}y} & \boldsymbol{G}(y; \boldsymbol{\pi}, \boldsymbol{T}) \\ \boldsymbol{0} & \mathrm{e}^{\boldsymbol{T}y} \end{array}\right) \,.$$

Correspondingly, a simple (and efficient) way to compute $\boldsymbol{G}(y; \boldsymbol{\pi}, \boldsymbol{T})$ is by calculating the matrix exponential of the left hand side.

Approaches to improve the speed of the EM algorithm in the univariate case exist in the literature; for instance, Okamura et al. (2011) proposed a method also based on uniformization.   □

3.3. **Parameter estimation for censored data.** In certain applications, some or all of the data may be censored. We call a data point *right–censored* at v if it takes an unknown value above v, *left–censored* at w if it takes an unknown value below w, and *interval–censored* if it is contained in the interval $(v, w]$, but its exact value is unknown. Left–censoring is a special case of interval–censoring with $v = 0$, while right–censoring can be obtained by fixing v and letting $w \to \infty$.

The EM Algorithm 3.2.1 works much in the same way as for uncensored data, with the only difference that we are no longer observing exact data points $Y^{(j)} = y^{(j)}$, but only $Y^{(j)} \in (v^{(j)}, w^{(j)}]$. This will only change the E–steps, where the conditional expectations can be calculated using the formulas in Olsson (1996). We now explain in detail how to adapt Algorithm 3.2.1 to censored data.

  First EM
    It is possible that a data point consists of a combination of marginals with both censored (not necessarily in the same intervals) and uncensored data (this is relevant in the first EM algorithm

when considering data of the sum of the marginals). Table 3.1 contains all possible combinations one might have in the data and the way of treating them. Note that for $d > 2$, one simply repeats the same rules iteratively.

$Y^{(1)}$	$Y^{(2)}$	$Y^{(S)} = Y^{(1)} + Y^{(2)}$
Uncensored with value $y^{(1)}$	Uncensored with value $y^{(2)}$	Uncensored with value $y^{(1)} + y^{(2)}$
Right–censored at $v^{(1)}$	Uncensored with value $y^{(1)}$	Right–censored at $v^{(1)} + y^{(2)}$
Right–censored at $v^{(1)}$	Right–censored at $v^{(2)}$	Right–censored at $v^{(1)} + v^{(2)}$
Right–censored at $v^{(1)}$	Interval–censored $(v^{(2)}, w^{(2)}]$	Right–censored at $v^{(1)} + v^{(2)}$
Interval–censored $(v^{(1)}, w^{(1)}]$	Uncensored with value $y^{(2)}$	Interval–censored $(v^{(1)} + y^{(2)}, w^{(1)} + y^{(2)}]$
Interval–censored $(v^{(1)}, w^{(1)}]$	Interval–censored $(v^{(2)}, w^{(2)}]$	Interval–censored $(v^{(1)} + v^{(2)}, w^{(1)} + w^{(2)}]$

TABLE 3.1. Rules for censored data.

For completeness, we include here the conditional expectations needed (see also Olsson (1996)).

$$
\mathbb{E}\left(B_k \mid Y^{(S)} \in (v, w]\right) = \frac{\pi_k \boldsymbol{e}'_k \mathrm{e}^{\boldsymbol{T}v}\boldsymbol{e} - \pi_k \boldsymbol{e}'_k \mathrm{e}^{\boldsymbol{T}w}\boldsymbol{e}}{\boldsymbol{\pi}\mathrm{e}^{\boldsymbol{T}v}\boldsymbol{e} - \boldsymbol{\pi}\mathrm{e}^{\boldsymbol{T}w}\boldsymbol{e}},
$$

$$
\mathbb{E}\left(Z_k \mid Y^{(S)} \in (v, w]\right) = \frac{\displaystyle\int_v^w \boldsymbol{\pi}\mathrm{e}^{\boldsymbol{T}u}\boldsymbol{e}_k du - \left(\displaystyle\int_0^w \boldsymbol{e}'_k\mathrm{e}^{\boldsymbol{T}(w-u)}\boldsymbol{e}\boldsymbol{\pi}\mathrm{e}^{\boldsymbol{T}u}\boldsymbol{e}_k du - \displaystyle\int_0^v \boldsymbol{e}'_k\mathrm{e}^{\boldsymbol{T}(v-u)}\boldsymbol{e}\boldsymbol{\pi}\mathrm{e}^{\boldsymbol{T}u}\boldsymbol{e}_k du\right)}{\boldsymbol{\pi}\mathrm{e}^{\boldsymbol{T}v}\boldsymbol{e} - \boldsymbol{\pi}\mathrm{e}^{\boldsymbol{T}w}\boldsymbol{e}},
$$

$$
\mathbb{E}\left(N_{kl} \mid Y^{(S)} \in (v, w]\right) = t_{kl}\frac{\displaystyle\int_v^w \boldsymbol{\pi}\mathrm{e}^{\boldsymbol{T}u}\boldsymbol{e}_k du - \left(\displaystyle\int_0^w \boldsymbol{e}'_l\mathrm{e}^{\boldsymbol{T}(w-u)}\boldsymbol{e}\boldsymbol{\pi}\mathrm{e}^{\boldsymbol{T}u}\boldsymbol{e}_k du - \displaystyle\int_0^v \boldsymbol{e}'_l\mathrm{e}^{\boldsymbol{T}(v-u)}\boldsymbol{e}\boldsymbol{\pi}\mathrm{e}^{\boldsymbol{T}u}\boldsymbol{e}_k du\right)}{\boldsymbol{\pi}\mathrm{e}^{\boldsymbol{T}v}\boldsymbol{e} - \boldsymbol{\pi}\mathrm{e}^{\boldsymbol{T}w}\boldsymbol{e}},
$$

$$
\mathbb{E}\left(N_k \mid Y^{(S)} \in (v, w]\right) = t_k \frac{\displaystyle\int_v^w \boldsymbol{\pi}\mathrm{e}^{\boldsymbol{T}u}\boldsymbol{e}_k du}{\boldsymbol{\pi}\mathrm{e}^{\boldsymbol{T}v}\boldsymbol{e} - \boldsymbol{\pi}\mathrm{e}^{\boldsymbol{T}w}\boldsymbol{e}}.
$$

Second EM

The second EM algorithm works as above, with the only difference that for marginals with censored data the corresponding conditional expectation is calculated as

$$
\mathbb{E}\left(Z_k^{(j)} \mid Y^{(j)} \in (v^{(j)}, w^{(j)}]\right)
$$

$$
= \frac{\displaystyle\int_{v^{(j)}}^{w^{(j)}} \boldsymbol{\pi}_j\mathrm{e}^{\boldsymbol{T}_j u}\boldsymbol{e}_k du - \left(\displaystyle\int_0^{w^{(j)}} \boldsymbol{e}'_k\mathrm{e}^{\boldsymbol{T}_j(w^{(j)}-u)}\boldsymbol{e}\boldsymbol{\pi}_j\mathrm{e}^{\boldsymbol{T}_j u}\boldsymbol{e}_k du - \displaystyle\int_0^{v^{(j)}} \boldsymbol{e}'_k\mathrm{e}^{\boldsymbol{T}_j(v^{(j)}-u)}\boldsymbol{e}\boldsymbol{\pi}_j\mathrm{e}^{\boldsymbol{T}_j u}\boldsymbol{e}_k du\right)}{\boldsymbol{\pi}_j\mathrm{e}^{\boldsymbol{T}_j v^{(j)}}\boldsymbol{e} - \boldsymbol{\pi}_j\mathrm{e}^{\boldsymbol{T}_j w^{(j)}}\boldsymbol{e}}.
$$

3.4. **A bivariate phase–type distribution with explicit density.** For a general MPH* distribution an explicit density is not available. Kulkarni (1989) characterized the density by a system of partial differential equations, and in Breuer (2016) a semi–explicit form is deduced. The following type of bivariate phase–type distributions does lead to an explicit density:
Let $\boldsymbol{Y} = (Y^{(1)}, Y^{(2)})' \sim \mathrm{MPH}^*(\boldsymbol{\pi}, \boldsymbol{T}, \boldsymbol{R})$ with

$$
(3.2) \qquad \boldsymbol{T} = \begin{pmatrix} \boldsymbol{T}_{11} & \boldsymbol{T}_{12} \\ \boldsymbol{0} & \boldsymbol{T}_{22} \end{pmatrix}, \quad \boldsymbol{\pi} = (\boldsymbol{\alpha}, \boldsymbol{0}) \quad \text{and} \quad \boldsymbol{R} = \begin{pmatrix} \boldsymbol{e} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{e} \end{pmatrix},
$$

where $\boldsymbol{T}_{11}$ and $\boldsymbol{T}_{22}$ are sub–intensity matrices of dimensions p_1 and p_2 ($p = p_1 + p_2$), respectively, and $\boldsymbol{T}_{11}\boldsymbol{e} + \boldsymbol{T}_{12}\boldsymbol{e} = \boldsymbol{0}$. Then the joint density of $\boldsymbol{Y}$ is given by

$$(3.3) \qquad f_{\boldsymbol{Y}}\left(y^{(1)}, y^{(2)}\right) = \boldsymbol{\alpha}\mathrm{e}^{\boldsymbol{T}_{11}y^{(1)}}\boldsymbol{T}_{12}\mathrm{e}^{\boldsymbol{T}_{22}y^{(2)}}(-\boldsymbol{T}_{22})\boldsymbol{e},$$

with marginals $Y^{(1)} \sim \mathrm{PH}(\boldsymbol{\alpha}, \boldsymbol{T}_{11})$ and $Y^{(2)} \sim \mathrm{PH}(\boldsymbol{\alpha}(-\boldsymbol{T}_{11})^{-1}\boldsymbol{T}_{12}, \boldsymbol{T}_{22})$. Note that the Baker–type bivariate distributions introduced in Bladt et al. (2019) are a particular case. The latter have some remarkable properties: one can construct a distribution of this type with specific given marginals and a given Pearson correlation coefficient; this class is also dense within the set of bivariate distributions with support in $\mathbb{R}_+^2$ (this follows from the fact that the class of Bernstein copulas can be used to approximate arbitrarily well any copula (see Sancetta and Satchell (2004)) and that the class of phase–type distributions can approximate arbitrarily well any distribution with support on $\mathbb{R}_+$), making the bigger class of bivariate distributions also dense.

3.4.1. *Tail independence.* The existence of an explicit form of the density allows us to compute the upper tail dependence coefficient λ_U. Recall that the latter is defined as

$$\lambda_U = \lim_{q \to 1^-} \mathbb{P}\left(Y^{(1)} > F_{Y^{(1)}}^{-1}(q) \mid Y^{(2)} > F_{Y^{(2)}}^{-1}(q)\right).$$

It is a classical measure of dependence in the tail and of considerable interest in applications in insurance and finance, where the modelling of tail events is crucial. From (3.3) we have

$$\bar{F}_{\boldsymbol{Y}}(y^{(1)}, y^{(2)}) = \mathbb{P}\left(Y^{(1)} > y^{(1)}, Y^{(2)} > y^{(2)}\right) = \boldsymbol{\alpha}\left(-\boldsymbol{T}_{11}\right)^{-1}\mathrm{e}^{\boldsymbol{T}_{11}y^{(1)}}\boldsymbol{T}_{12}\mathrm{e}^{\boldsymbol{T}_{22}y^{(2)}}\boldsymbol{e}.$$

Then, if $-\lambda_j$ is the real part of the eigenvalue of $\boldsymbol{T}_{jj}$ with largest real part and k_j is the dimension of the Jordan block of λ_j for $j = 1, 2$, it is easy to see that

$$\bar{F}_{\boldsymbol{Y}}(y^{(1)}, y^{(2)}) \sim b(y^{(1)})^{k_1-1}\mathrm{e}^{-\lambda_1 y^{(1)}}(y^{(2)})^{k_2-1}\mathrm{e}^{-\lambda_2 y^{(2)}}, \quad \text{as} \quad y^{(1)}, y^{(2)} \to \infty,$$

where b is a positive constant. Hence

$$\lambda_U = \lim_{q \to 1^-} \frac{b(F_{Y^{(1)}}^{-1}(q))^{k_1-1}\mathrm{e}^{-\lambda_1(F_{Y^{(1)}}^{-1}(q))}(F_{Y^{(2)}}^{-1}(q))^{k_2-1}\mathrm{e}^{-\lambda_2(F_{Y^{(2)}}^{-1}(q))}}{c(F_{Y^{(2)}}^{-1}(q))^{k_2-1}\mathrm{e}^{-\lambda_2(F_{Y^{(2)}}^{-1}(q))}} = 0,$$

with c positive constant. In other words, $\boldsymbol{Y}$ is upper-tail-independent.

3.4.2. *Estimation.* The density (3.3) allows for a special form of EM algorithm. Such an algorithm was introduced in Ahlström et al. (1999) and we include it for completeness, subsequent use and comparison purposes.

Algorithm 3.4.1.
   0. Initialize with some "arbitrary" $(\boldsymbol{\alpha}, \boldsymbol{T})$.
   1. (E–step) Calculate

$$\mathbb{E}\left(B_k \mid \boldsymbol{Y} = \boldsymbol{y}\right) = \sum_{i=1}^{N} \frac{\alpha_k \boldsymbol{e}_k' \mathrm{e}^{\boldsymbol{T}_{11}y_i^{(1)}}\boldsymbol{T}_{12}\mathrm{e}^{\boldsymbol{T}_{22}y_i^{(2)}}(-\boldsymbol{T}_{22})\boldsymbol{e}}{f_{\boldsymbol{Y}}(y_i^{(1)}, y_i^{(2)}; \boldsymbol{\alpha}, \boldsymbol{T})}, \quad k = 1, \ldots, p_1$$

$$\mathbb{E}\left(Z_k \mid \boldsymbol{Y} = \boldsymbol{y}\right) = \begin{cases} \displaystyle\sum_{i=1}^{N} \frac{\displaystyle\int_0^{y_i^{(1)}} \boldsymbol{\alpha}\mathrm{e}^{\boldsymbol{T}_{11}u}\boldsymbol{e}_k\boldsymbol{e}_k'\mathrm{e}^{\boldsymbol{T}_{11}(y_i^{(1)}-u)}\boldsymbol{T}_{12}\mathrm{e}^{\boldsymbol{T}_{22}y_i^{(2)}}\left(-\boldsymbol{T}_{22}\right)\boldsymbol{e}\,du}{f_{\boldsymbol{Y}}(y_i^{(1)},y_i^{(2)};\boldsymbol{\alpha},\boldsymbol{T})}, & k = 1,\ldots,p_1 \\[3em] \displaystyle\sum_{i=1}^{N} \frac{\displaystyle\int_0^{y_i^{(2)}} \boldsymbol{\alpha}\mathrm{e}^{\boldsymbol{T}_{11}y_i^{(1)}}\boldsymbol{T}_{12}\mathrm{e}^{\boldsymbol{T}_{22}u}\boldsymbol{e}_{(k-p_1)}\boldsymbol{e}_{(k-p_1)}'\mathrm{e}^{\boldsymbol{T}_{22}(y_i^{(2)}-u)}\left(-\boldsymbol{T}_{22}\right)\boldsymbol{e}\,du}{f_{\boldsymbol{Y}}(y_i^{(1)},y_i^{(2)};\boldsymbol{\alpha},\boldsymbol{T})}, & k = p_1+1,\ldots,p \end{cases}$$

$$\mathbb{E}\left(N_{kl} \mid \boldsymbol{Y} = \boldsymbol{y}\right) =$$

$$\begin{cases} \displaystyle\sum_{i=1}^{N} t_{kl} \frac{\displaystyle\int_0^{y_i^{(1)}} \boldsymbol{\alpha}\mathrm{e}^{\boldsymbol{T}_{11}u}\boldsymbol{e}_k\boldsymbol{e}_l'\mathrm{e}^{\boldsymbol{T}_{11}(y_i^{(1)}-u)}\boldsymbol{T}_{12}\mathrm{e}^{\boldsymbol{T}_{22}y_i^{(2)}}\left(-\boldsymbol{T}_{22}\right)\boldsymbol{e}\,du}{f_{\boldsymbol{Y}}(y_i^{(1)},y_i^{(2)};\boldsymbol{\alpha},\boldsymbol{T})}, & k,l = 1,\ldots,p_1,\ k \neq l \\[3em] \displaystyle\sum_{i=1}^{N} t_{kl} \frac{\boldsymbol{\alpha}\mathrm{e}^{\boldsymbol{T}_{11}y_i^{(1)}}\boldsymbol{e}_k\boldsymbol{e}_{l-p_1}'\mathrm{e}^{\boldsymbol{T}_{22}y_i^{(2)}}\left(-\boldsymbol{T}_{22}\right)\boldsymbol{e}}{f_{\boldsymbol{Y}}(y_i^{(1)},y_i^{(2)};\boldsymbol{\alpha},\boldsymbol{T})}, & k = 1,\ldots,p_1,\ l = p_1+1,\ldots,p \\[3em] \displaystyle\sum_{i=1}^{N} t_{kl} \frac{\displaystyle\int_0^{y_i^{(2)}} \boldsymbol{\alpha}\mathrm{e}^{\boldsymbol{T}_{11}y_i^{(1)}}\boldsymbol{T}_{12}\mathrm{e}^{\boldsymbol{T}_{22}u}\boldsymbol{e}_{k-p_1}\boldsymbol{e}_{l-p_1}'\mathrm{e}^{\boldsymbol{T}_{22}(y_i^{(2)}-u)}\left(-\boldsymbol{T}_{22}\right)\boldsymbol{e}\,du}{f_{\boldsymbol{Y}}(y_i^{(1)},y_i^{(2)};\boldsymbol{\alpha},\boldsymbol{T})}, & k,l = p_1+1,\ldots,p,\ k \neq l \end{cases}$$

$$\mathbb{E}\left(N_k \mid \boldsymbol{Y} = \boldsymbol{y}\right) = \sum_{i=1}^{N} t_k \frac{\boldsymbol{\alpha}\mathrm{e}^{\boldsymbol{T}_{11}y_i^{(1)}}\boldsymbol{T}_{12}\mathrm{e}^{\boldsymbol{T}_{22}y_i^{(2)}}\boldsymbol{e}_{k-p_1}}{f_{\boldsymbol{Y}}(y_i^{(1)},y_i^{(2)};\boldsymbol{\alpha},\boldsymbol{T})}, \quad k = p_1+1,\ldots,p.$$

2. *(M–step) Let*

$$\hat{\alpha}_k = \frac{1}{N}\mathbb{E}\left(B_k \mid \boldsymbol{Y} = \boldsymbol{y}\right), \quad \hat{t}_{kl} = \frac{\mathbb{E}\left(N_{kl} \mid \boldsymbol{Y} = \boldsymbol{y}\right)}{\mathbb{E}\left(Z_k \mid \boldsymbol{Y} = \boldsymbol{y}\right)}, \quad \hat{t}_k = \frac{\mathbb{E}\left(N_k \mid \boldsymbol{Y} = \boldsymbol{y}\right)}{\mathbb{E}\left(Z_k \mid \boldsymbol{Y} = \boldsymbol{y}\right)}, \quad \hat{t}_{kk} = -\sum_{l \neq k}\hat{t}_{kl} - \hat{t}_k,$$

$$\hat{\boldsymbol{\alpha}} = (\hat{\alpha}_1,\ldots,\hat{\alpha}_{p_1}), \quad \hat{\boldsymbol{T}} = \{\hat{t}_{kl}\}_{k,l=1,\ldots,p} \quad and \quad \hat{\boldsymbol{t}} = (\hat{t}_{p_1+1},\ldots,\hat{t}_p)'.$$

3. *Assign* $\boldsymbol{\alpha} := \hat{\boldsymbol{\alpha}}$, $\boldsymbol{T} := \hat{\boldsymbol{T}}$, $\boldsymbol{t} := \hat{\boldsymbol{t}}$ *and GOTO 1.*

We now provide two detailed illustrations. When Algorithm 3.2.1 is employed, given that an explicit form of the joint density is not available, we choose the number of iterations in such a way that the changes in the successive log–likelihoods in the first EM become negligible and the changes in the successive parameter estimates become negligible in the second EM. For Algorithm 3.4.1 we used a criterion similar to the univariate case. For a visual assessment of the quality of the fits in the multivariate setting, we also added some contour plots the interpretation of which, however, is more subjective than the one of the corresponding one-dimensional graphs. Nevertheless, we believe that they do provide some helpful insight concerning the shape of the distributions.

Example 3.4.1 (Simulation study)**.** The objective of the present example is to compare the performance of Algorithm 3.2.1 and Algorithm 3.4.1. We will illustrate that the more general Algorithm 3.2.1 also provides reasonable results when dealing with a sample from a bivariate distribution

with density (3.3), for which the more specific Algorithm 3.4.1 is particularly well-suited. We generated an i.i.d. sample of size $10\,000$ from a MPH* distribution with parameters

$$\boldsymbol{\pi} = (0.15,\, 0.85,\, 0,\, 0)\ , \quad \boldsymbol{T} = \begin{pmatrix} -2 & 0 & 2 & 0 \\ 9 & -11 & 0 & 2 \\ 0 & 0 & -1 & 0.5 \\ 0 & 0 & 0 & -5 \end{pmatrix}, \quad \text{and} \quad \boldsymbol{R} = \begin{pmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{pmatrix},$$

which has theoretical mean $\mathbb{E}(\boldsymbol{Y}) = \big(\mathbb{E}(Y^{(1)}), \mathbb{E}(Y^{(2)})\big)' = (0.5, 0.9609)'$, and correlation coefficient $\rho(Y^{(1)}, Y^{(2)}) = 0.1148$. Moreover, we know that $\lambda_U = 0$. The simulated sample has numerical values $\hat{\mathbb{E}}(\boldsymbol{Y}) = (0.5046, 0.9650)'$, $\hat{\rho} = 0.1235$ and Kendall's tau $\hat{\rho}_\tau = 0.1613$. We now use Algorithms 3.2.1 and 3.4.1 to recover the underlying structure of the data, and we assess the quality of the estimation by comparing densities, QQ plots, numerical properties of the distributions and contour plots.

Using Algorithm 3.2.1 with the same number of phases $p = 4$, random initial values and $3\,500$ steps in each EM algorithm, we obtain the following parameters:

$$\hat{\boldsymbol{\pi}} = (0.8592,\, 0.0002,\, 0.0005,\, 0.1402)\ ,$$

$$\hat{\boldsymbol{T}} = \begin{pmatrix} -9.2654 & 0.3805 & 7.2657 & 1.6192 \\ 0.0039 & -1.1038 & 0.0002 & 0.0118 \\ 0.7429 & 0.1640 & -7.1356 & 4.9516 \\ 0.3295 & 1.6162 & 0.5278 & -2.4740 \end{pmatrix},$$

$$\hat{\boldsymbol{R}} = \begin{pmatrix} 0.5950 & 0.4050 \\ 0 & 1 \\ 0.4254 & 0.5746 \\ 0.8347 & 0.1653 \end{pmatrix}.$$

The fitted distribution has mean $\mathbb{E}(\boldsymbol{Y}) = (0.5205, 0.9491)'$ and $\rho = 0.2559$, which approximates reasonably well the mean of the original distribution and to a lesser degree well the correlation coefficient. We also approximated λ_U and ρ_τ via simulation obtaining $\hat{\lambda}_U = 0.0058$ and $\hat{\rho}_\tau = 0.2843$. Figures 3.1 and 3.2 show that the algorithm is able to recover the structure of the marginals and the sum of the marginals. Moreover, Figure 3.3 shows that the contour plot of the fitted distribution is similar to the one of the sample. Here, the running time of Algorithm 3.2.1 with its $3\,500$ iterations was 924 seconds.
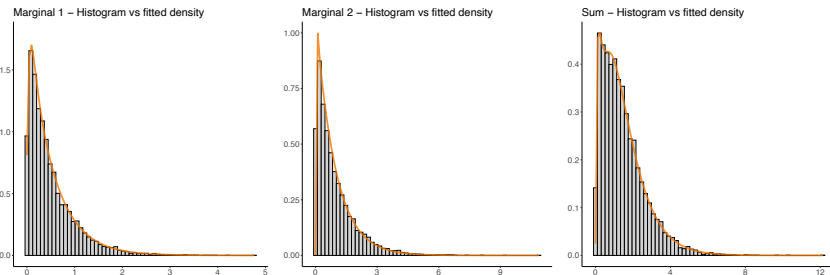
FIGURE 3.1. Histograms of simulated sample versus densities of the MPH* distribution fitted using Algorithm 3.2.1.

Next we use Algorithm 3.4.1 with 4 phases ($p_1 = 2$ and $p_2 = 2$), random initial values and $2\,500$ steps in the EM algorithm (leading to a running time of 623 seconds). The estimated parameters are:

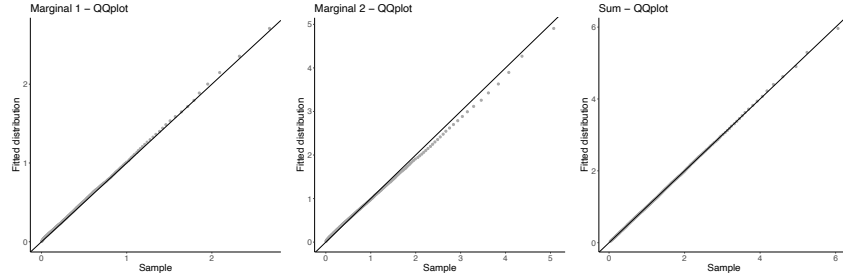$$\hat{\boldsymbol{\pi}} = (0.1218,\, 0.8782,\, 0,\, 0)\ ,$$

FIGURE 3.2. QQ plots of simulated sample versus fitted MPH* distribution using Algorithm 3.2.1.
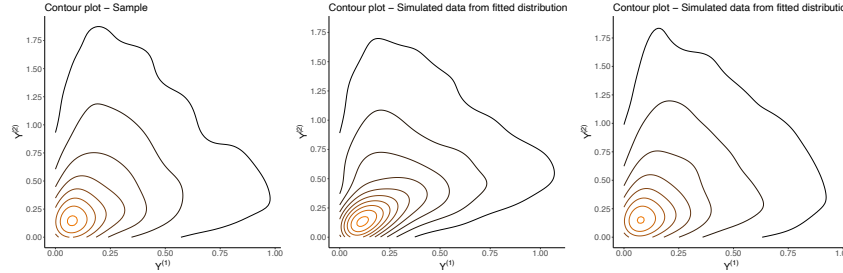

FIGURE 3.3. Contour plot of sample (left), contour plot of a simulated sample from the MPH* distribution fitted with Algorithm 3.2.1 (center) and contour plot of of a simulated sample from the distribution fitted with Algorithm 3.4.1 (right).

$$
\hat{\boldsymbol{T}} = \left(\begin{array}{cccc} -2.2656 & 0.2832 & 1.9823 & 0.0002 \\ 8.2508 & -10.2295 & 0.0257 & 1.9530 \\ 0 & 0 & -1.0115 & 0.6419 \\ 0 & 0 & 0.0992 & -4.7781 \end{array} \right),
$$

$$
\hat{\boldsymbol{R}} = \left(\begin{array}{cc} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{array} \right),
$$

with corresponding mean $\mathbb{E}(\boldsymbol{Y}) = (0.5046, 0.9650)'$ and $\rho = 0.1156$. ρ_τ for this fit can be estimated via simulation, giving $\hat{\rho}_\tau = 0.1448$. A contour plot of the fit is available in Figure 3.3 and we see that the algorithm recovers the original structure of the data even better. Note that again the log–likelihood of the fitted MPH* distribution ($-12\,329.82$) outperforms the log–likelihood using the original MPH* distribution ($-12\,330.96$).

Finally we would like to remark that Algorithm 3.4.1 already starts with a more specific structure on its parameters which resembles the one of the distribution from which the data come from. On the other hand, Algorithm 3.2.1 does not require any prior assumption on the initial structure of its parameters. Thus, a better fit from Algorithm 3.4.1 is expected, since Algorithm 3.2.1 needs to find a distribution in a larger set. In line with the non-identifiability issue, one sees that one can obtain a quite reasonable fit in that larger class that captures some main features of the original distribution, whereas the more specific Algorithm 3.4.1 finds a fit that even exhibits nicely the original correlation pattern. Yet, the flexibility of Algorithm 3.2.1 is a considerable advantage when dealing with data sets without the additional knowledge about the underlying distribution.

Example 3.4.2 (Known distribution – Marshall-Olkin exponential)**.** We now would like to illustrate that Algorithms 3.2.1 and 3.4.1 can be modified to fit a MPH* model to a theoretically

given joint distribution H. The idea is along the lines of Asmussen et al. (1996) and consists of considering sequences of empirical distributions with increasing sample size. We exemplify this by considering a bivariate Marshall–Olkin exponential distribution, whose joint survival function is of the form

$$\overline{F}\left(y^{(1)}, y^{(2)}\right) = \exp\left(-\lambda_1 y^{(1)} - \lambda_2 y^{(2)} - \lambda_{12} \max(y^{(1)}, y^{(2)})\right).$$

We take $\lambda_1 = 1$, $\lambda_2 = 3$ and $\lambda_{12} = 1$, then the distribution has theoretical moments $\mathbb{E}(\boldsymbol{Y}) = (0.5, 0.25)$ and $\rho = 0.2$. It is easy to see that the Marshall–Olkin bivariate exponential is upper-tail-independent, i.e., $\lambda_U = 0$. Moreover, we approximate ρ_τ via simulation, obtaining $\hat{\rho}_\tau = 0.2012$. Then we fit a MPH* distribution using the Algorithm 3.2.1. With 3 phases and random initial values together with 2 500 steps in each EM algorithm (overall running time about 120 seconds), we obtain the parameters

$$\hat{\boldsymbol{\pi}} = (0.8233, 0.1633, 0.0134),$$

$$\hat{\boldsymbol{T}} = \begin{pmatrix} 2.5894 & 1.6637 & 0.7601 \\ 0.0087 & -2.0699 & 0.2102 \\ 0.1032 & 0.3465 & -4.2765 \end{pmatrix},$$

$$\hat{\boldsymbol{R}} = \begin{pmatrix} 0.4412 & 0.5588 \\ 0.9514 & 0.0486 \\ 0.2348 & 0.7652 \end{pmatrix},$$

which has corresponding moments $\mathbb{E}(\boldsymbol{Y}) = (0.4942, 0.2565)'$ and $\rho = 0.3260$. λ_U and ρ_τ for the resulting model can be approximated by simulation to be $\hat{\lambda}_U = 0.0547$ and $\hat{\rho}_\tau = 0.2745$. Together with the densities (Figure 3.4) and QQ plots (Figure 3.5), one sees that this algorithm recovers rather well the structure of the original joint distribution.
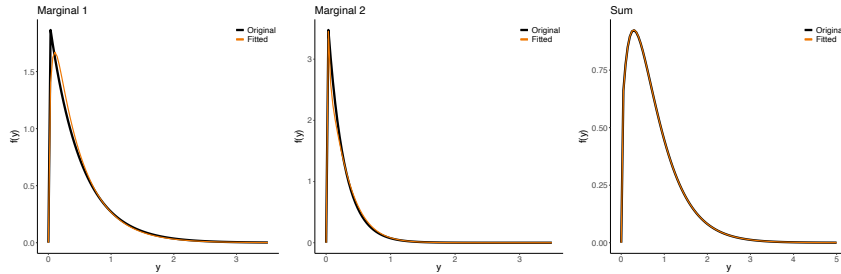
FIGURE 3.4. Densities of the original Marshall–Olkin distribution versus densities of the MPH* distribution fitted using Algorithm 3.2.1.
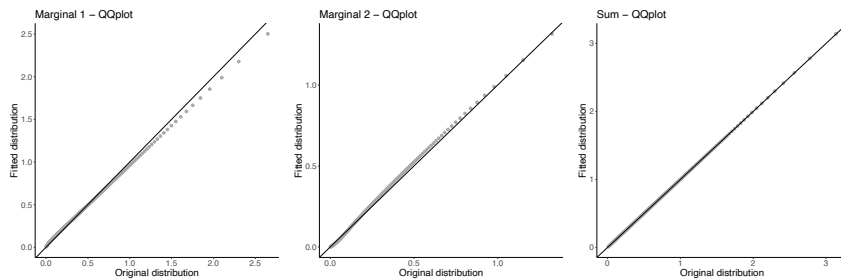
FIGURE 3.5. QQ plots of original Marshall–Olkin distribution versus fitted MPH* distribution using Algorithm 3.2.1.
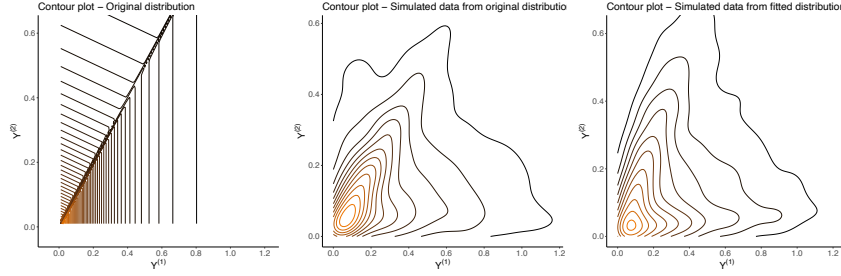
FIGURE 3.6. Contour plot of the original Marshall–Olkin distribution (left) contour plot of simulated sample from Marshall–Olkin distribution (middle) and contour plot of simulated sample from the distribution fitted with Algorithm 3.2.1 (right).

4. MULTIVARIATE INHOMOGENEOUS PHASE-TYPE DISTRIBUTIONS

There are various possibilities for extending inhomogeneous phase-type distributions to more than one dimension. In the following we suggest one particular approach and provide an algorithm for the parameter estimation.

4.1. **Definition and Properties.** Let $\boldsymbol{Y} \sim \mathrm{MPH}^*(\boldsymbol{\pi}, \boldsymbol{T}, \boldsymbol{R})$ and define $\boldsymbol{X} := (g_1(Y^{(1)}), \ldots, g_d(Y^{(d)}))'$, where $g_j : \mathbb{R}_+ \to \mathbb{R}_+$ are increasing and differentiable functions with range $\mathbb{R}_+$ for $j = 1, \ldots, d$, then we say that $\boldsymbol{X}$ has an *inhomogeneous MPH** distribution.

Several of its properties follow directly from the definition:

(1) The marginals $X^{(j)} = g_j(Y^{(j)})$ are IPH distributed, since each $Y^{(j)}$ is phase–type distributed, $j = 1, \ldots, d$.

(2) If g_j is strictly increasing for all $j = 1, \ldots, d$, the copula of $\boldsymbol{X}$ is the same as the copula of $\boldsymbol{Y}$ (see e.g. (McNeil et al., 2015, Prop.7.7)).

(3) For fixed $g_j(\cdot)$, $j = 1, \ldots, d$, this new class is dense in $\mathbb{R}_+^d$ (by the denseness of the MPH* class).

In the sequel we will provide an algorithm for parameter estimation for which an explicit expressions of the bivariate density is needed. We therefore restrict it to the bivariate case.

4.2. **Parameter estimation in the bivariate case.** In the bivariate case, for any $\boldsymbol{Y} \sim \mathrm{MPH}^*(\boldsymbol{\pi}, \boldsymbol{T}, \boldsymbol{R})$ with parameters (3.2), it is easy to see that the density of $\boldsymbol{X} = (g_1(Y^{(1)}), g_2(Y^{(2)}))'$ is given by

$$(4.1) \qquad f_{\boldsymbol{X}}\left(x^{(1)}, x^{(2)}\right) = \boldsymbol{\alpha} \mathrm{e}^{\boldsymbol{T}_{11} g_1^{-1}(x^{(1)})} \boldsymbol{T}_{12} \mathrm{e}^{\boldsymbol{T}_{22} g_2^{-1}(x^{(2)})} (-\boldsymbol{T}_{22}) \boldsymbol{e} \, \frac{1}{g_1'(g_1^{-1}(x^{(1)})) g_2'(g_2^{-1}(x^{(2)}))} \, .$$

If we assume that $g_j(\,\cdot\,; \boldsymbol{\beta}_j)$ is a parametric non–negative function depending on the vector $\boldsymbol{\beta}_j$, $j = 1, 2$, and let $\boldsymbol{\beta} = (\boldsymbol{\beta}_1, \boldsymbol{\beta}_2)$. Then, we can formulate an algorithm analogous to Algorithm 2.2.1:

Algorithm 4.2.1 (EM algorithm for bivariate inhomogeneous MPH* distributions)**.**

0. Initialize with some "arbitrary" $(\boldsymbol{\alpha}, \boldsymbol{T}, \boldsymbol{\beta})$.

1. Transform the data into $y_i^{(j)} := g_j^{-1}(x_i^{(j)}; \boldsymbol{\beta}_j)$, $i = 1, \ldots, N$, $j = 1, 2$, and apply the E– and M–steps of Algorithm 3.4.1 by which we obtain the estimators $(\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{T}})$.

2. Compute

$$\hat{\boldsymbol{\beta}} = \arg\max_{\boldsymbol{\beta}} \sum_{i=1}^{N} \log(f_{\boldsymbol{X}}(x_i^{(1)}, x_i^{(2)}; \hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{T}}, \boldsymbol{\beta}))$$

3. Assign $(\boldsymbol{\alpha}, \boldsymbol{T}, \boldsymbol{\beta}) = (\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{T}}, \hat{\boldsymbol{\beta}})$ and GOTO 1.

We now consider particular multivariate distributions obtained through such a transformation of an MPH* random vector.

4.3. Multivariate matrix–Pareto models.

Let $\boldsymbol{X} = (g_1(Y^{(1)}), \ldots, g_d(Y^{(d)}))'$, where $\boldsymbol{Y} \sim \mathrm{MPH}^* (\boldsymbol{\pi}, \boldsymbol{T}, \boldsymbol{R})$ and

$$(4.2) \qquad g_j(y) = \beta_j(e^y - 1), \quad \beta_j > 0, \ j = 1, \ldots, d.$$

Then we say that $\boldsymbol{X}$ follows a *multivariate matrix–Pareto distribution.* Some special properties of this class of distributions are:

(1) Marginal distributions are matrix–Pareto distributed.
(2) Moments and cross–moments of $\boldsymbol{X}$ can be obtained from the moment generating function of $\boldsymbol{Y}$ (see (Bladt and Nielsen, 2017, Theorem 8.1.2)), provided that they exist.
(3) Products of the type $\prod_{i=1}^{M} \left(\frac{X^{(j_i)}}{\beta_{j_i}} + 1 \right)^{a_i}$ are matrix–Pareto distributed, $a_i > 0$, $j_i \in \{1, \ldots, d\}$, $i = 1, \ldots, M$, $M \le N$, since linear combinations of $\boldsymbol{Y}$ are PH distributed.

In the bivariate case, (4.2) and (4.1), lead to

$$f_{\boldsymbol{X}}(x^{(1)}, x^{(2)}) = \boldsymbol{\alpha} \left(\frac{x^{(1)}}{\beta_1} + 1 \right)^{\boldsymbol{T}_{11} - \boldsymbol{I}} \boldsymbol{T}_{12} \left(\frac{x^{(2)}}{\beta_2} + 1 \right)^{\boldsymbol{T}_{22} - \boldsymbol{I}} (-\boldsymbol{T}_{22}) \boldsymbol{e} \, \frac{1}{\beta_1 \beta_2} \, .$$

and

$$\bar{F}_{\boldsymbol{X}}(x^{(1)}, x^{(2)}) = \boldsymbol{\alpha} \, (-\boldsymbol{T}_{11})^{-1} \left(\frac{x^{(1)}}{\beta_1} + 1 \right)^{\boldsymbol{T}_{11}} \boldsymbol{T}_{12} \left(\frac{x^{(2)}}{\beta_2} + 1 \right)^{\boldsymbol{T}_{22}} \boldsymbol{e}$$

Moreover, in this bivariate case, linear combinations of $X^{(i)}$ are regularly varying, and the respective index is the real part of the eigenvalue with largest real part of the sub–intensity matrices of the marginals, which follows from (Davis and Resnick, 1996, Lem. 2.1) and asymptotic independence of $\boldsymbol{Y}$. The general case is not clear since the condition of asymptotic independence does not hold.

Remark 4.3.1. The Marshall–Olkin Pareto distribution (see Hanagal (1996a)) is a particular case of this class of distributions.

4.3.1. *Parameter estimation.* As in the univariate case, if we assume the simpler transformation $g_j(y) = e^y - 1$, $j = 1, \ldots, d$, then we can use fitting methods of the MPH* class by taking the logarithm of the marginal observations. I.e., we can apply Algorithms 3.2.1 and 3.4.1 to the transformed data $y_i^{(j)} := \log(x_i^{(j)} + 1)$, $i = 1, \ldots, N$, $j = 1, \ldots, d$, to estimate the parameters $(\boldsymbol{\pi}, \boldsymbol{T}, \boldsymbol{R})$. We exemplify the use of this method in two examples.

Example 4.3.1. (Mardia type I) We generated an i.i.d. sample of size $10\,000$ from a (translated) Mardia type I Pareto distribution (see Mardia (1962)) with parameters $\sigma_1 = \sigma_2 = 1$ and $\alpha = 2$. This distribution has theoretical numerical values $\mathbb{E}(\boldsymbol{X}) = (1, 1)'$, $\lambda_U = 0$ and $\rho_\tau = 0.2$. The simulated sample has numerical values $\hat{\mathbb{E}}(\boldsymbol{X}) = (0.9812, 0.9712)'$ and $\hat{\rho}_\tau = 0.2049$.

We fitted a bivariate matrix–Pareto distribution using Algorithm 3.4.1 with $p_1 = p_2 = 2$ (i.e., $p = 4$) and $2\,000$ steps on the transformed data (with a running time of 530 seconds), getting the following parameters:

$$\hat{\boldsymbol{\alpha}} = (0.1666, 0.8334, 0, 0) \, ,$$

$$\hat{\boldsymbol{T}} = \begin{pmatrix} -2.0022 & 0.0238 & 1.9784 & 0 \\ 9.2506 & -11.2967 & 0.1204 & 1.9256 \\ 0 & 0 & -2.0175 & 1.8247 \\ 0 & 0 & 0.0834 & -12.8829 \end{pmatrix} \, ,$$

$$\hat{\boldsymbol{R}} = \begin{pmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{pmatrix}.$$

The tails of the marginals of the fitted distribution are determined by the real part of the eigenvalues with largest real part of the sub–intensity matrices of the marginal distributions, which are $\lambda_1^{(\max)} = -1.9785$ and $\lambda_2^{(\max)} = -2.0035$. These resemble well the ones of the original distribution. The fitted distribution has first moment $\mathbb{E}(\boldsymbol{X}) = (1.0164, 0.9963)'$. Moreover, we estimated ρ_τ via simulation, obtaining $\hat{\rho}_\tau = 0.1582$. The QQ plots are available in Figure 4.1 and contour plots are depicted in Figure 4.2, from where it becomes clear that the algorithm recovers the structure of the data well. Again, the log–likelihood of the fitted bivariate matrix–Pareto $(-15\,550.52)$ exceeds the log–likelihood using the original Mardia distribution $(-16\,146.65)$.
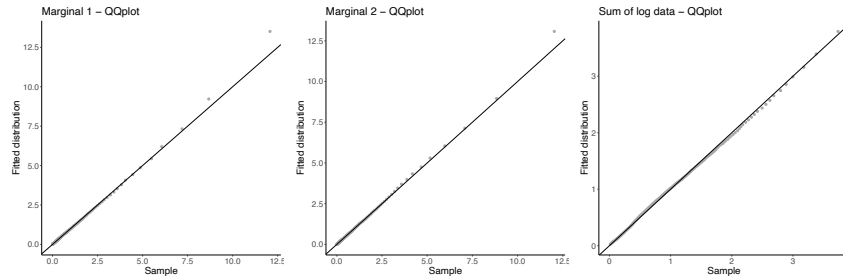
FIGURE 4.1. QQ plots of simulated Mardia Pareto sample versus fitted multivariate matrix–Pareto distribution using Algorithm 3.4.1.
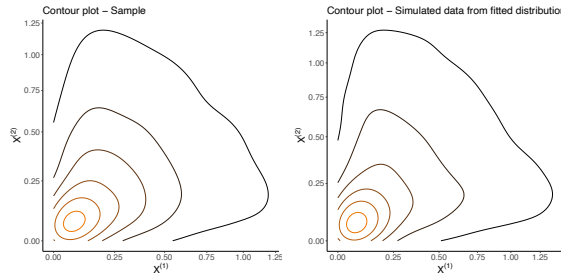
FIGURE 4.2. Contour plot of simulated Mardia Pareto sample (left) and contour plot of a simulated sample from the Matrix–Pareto distribution fitted using Algorithm 3.4.1 (right).

The concrete structure of the intensity and reward matrix underlying Algorithm 3.4.1 restricts its application to tail-independent models. While in the previous example this was justified, in situations with tail-dependent data one should rather look for fits in the general MPH* class by using Algorithm 3.2.1 on the transformed data. The following example illustrates such an approach.

Example 4.3.2. (Gumbel copula with matrix–Pareto marginals) We generated an i.i.d. sample of size $5\,000$ from a three-dimensional random vector with first marginal being a matrix–Pareto with parameters

$$\boldsymbol{\pi}_1 = (0.2,\, 0.8)\,,$$

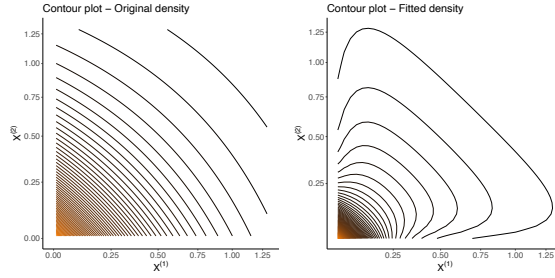$$\boldsymbol{T}_1 = \begin{pmatrix} -2 & 0 \\ 0 & -3 \end{pmatrix},$$

FIGURE 4.3. Contour plot of original Mardia Pareto (left) and contour plot of Matrix–Pareto distribution fitted using Algorithm 3.4.1 (right).

$$\beta_1 = 1 \,,$$

second marginal being a matrix–Pareto with parameters

$$\boldsymbol{\pi}_2 = (0.5, 0.5) \,,$$

$$\boldsymbol{T}_2 = \begin{pmatrix} -2 & 0 \\ 0 & -1.5 \end{pmatrix} \,,$$

$$\beta_2 = 1 \,,$$

and the third marginal being a conventional Pareto with shape parameter 2.5, and a Gumbel copula with parameter $\theta = 4$. The choice of a Gumbel copula instead of a multivariate matrix–Pareto model based on an MPH* construction is to show that the algorithm can be used to model any type of dependence structure. The Gumbel copula is known to have positive tail dependence. This distribution has theoretical numerical values $\mathbb{E}(\boldsymbol{X}) = (0.6, 1.5, 0.6667)'$, $\lambda_U = 2 - 2^{1/4} \approx 0.8108$ and $\rho_\tau = 0.75$, and the real part of the eigenvalues that determine the heaviness of the tails are $\lambda_1^{(\max)} = -2$, $\lambda_2^{(\max)} = -1.5$ and $\lambda_3^{(\max)} = -2.5$. The simulated sample has numerical values $\hat{\mathbb{E}}(\boldsymbol{X}) = (0.6035, 1.4721, 0.6732)'$, $\hat{\lambda}_U(X_1, X_2) = 0.8142$, $\hat{\lambda}_U(X_1, X_3) = 0.7429$, $\hat{\lambda}_U(X_2, X_3) = 0.7857$, $\hat{\rho}_\tau(X_1, X_2) = 0.7513$, $\hat{\rho}_\tau(X_1, X_3) = 0.7526$ and $\hat{\rho}_\tau(X_2, X_3) = 0.7544$. Then, we fitted a multivariate matrix–Pareto distribution using Algorithm 3.2.1 with $p = 6$ and $5\,000$ steps on the transformed data (running time $6\,501$ seconds), obtaining the following parameters:

$$\hat{\boldsymbol{\alpha}} = (0.0062\,, 0.0606\,, 0.4394\,, 0.0600\,, 0.0608\,, 0.3731) \,,$$

$$\hat{\boldsymbol{T}} = \begin{pmatrix} -16.9867 & 1.7532 & 1.3902 & 3.2835 & 1.0736 & 2.1224 \\ 0.2710 & -1.3282 & 0.2835 & 0.0757 & 0.0562 & 0.2987 \\ 0.3714 & 0.0161 & -2.8735 & 0.6658 & 0.8798 & 0.9208 \\ 1.1742 & 0.5387 & 0.2421 & -4.0067 & 0.5156 & 0.5571 \\ 0.0786 & 0.5130 & 0.2916 & 1.1454 & -4.2894 & 1.5148 \\ 2.0244 & 0.2912 & 0.7007 & 0.7777 & 2.0237 & -6.2507 \end{pmatrix} \,,$$

$$\hat{\boldsymbol{R}} = \begin{pmatrix} 0.9414 & 0.0558 & 0.0028 \\ 0.2856 & 0.4594 & 0.2549 \\ 0.2934 & 0.5233 & 0.1833 \\ 0.0906 & 0.5548 & 0.3546 \\ 0.4230 & 0.0750 & 0.5020 \\ 0.0748 & 0.5992 & 0.3260 \end{pmatrix} \,.$$

The real part of the eigenvalues with largest real part of the sub–intensity matrices of the marginal distributions are $\lambda_1^{(\max)} = -2.5370$, $\lambda_2^{(\max)} = -1.5941$ and $\lambda_3^{(\max)} = -2.4167$, which are close to the ones of the original distribution. The fitted distribution has first moment $\mathbb{E}(\boldsymbol{X}) = (0.6006, 1.5344, 0.6850)'$. Moreover, we estimated λ_U and ρ_τ via simulation, obtaining $\hat{\lambda}_U(X_1, X_2) =$

0.70403, $\hat{\lambda}_U(X_1, X_3) = 0.736438$, $\hat{\lambda}_U(X_2, X_3) = 0.672417$, $\hat{\rho}_\tau(X_1, X_2) = 0.6924$, $\hat{\rho}_\tau(X_1, X_3) = 0.7470$ and $\hat{\rho}_\tau(X_2, X_3) = 0.7728$. Comparing all numerical properties, together with the QQ plots (see Figure 4.4) and contour plots (see Figure 4.5), we see that this algorithm also recovers relatively well the structure of the data.
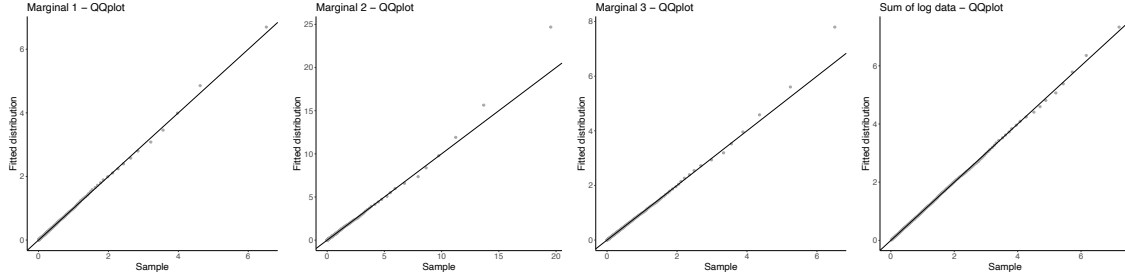

FIGURE 4.4. QQ plots of simulated sample versus fitted multivariate matrix–Pareto distribution using Algorithm 3.2.1.
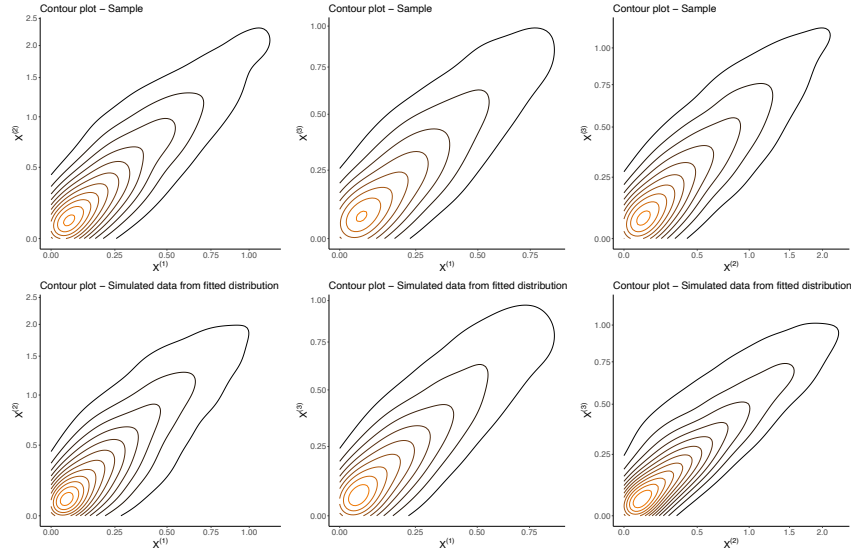
FIGURE 4.5. Contour plots of simulated sample from Gumbel copula with matrix-Pareto marginals (top) and contour plots of a simulated sample from the Matrix–Pareto distribution fitted using Algorithm 3.2.1 (bottom).

We would like to comment on the relative inaccuracy of the obtained estimate for $\lambda_1^{(\max)}$, especially when compared to the convincing estimates for the other marginals. The first marginal here is in fact a mixture of Pareto distributions with shape parameters 2 and 3, and mixing probabilities 0.2 and 0.8, respectively. That is, a major proportion of the data for the first marginal are to be expected to stem from the Pareto distribution with shape parameter 3. Since Algorithm 3.2.1 fits body and tail at the same time, one cannot expect the resulting estimate for the tail to be as good as techniques of classical extreme value theory, which focusses on the tail fit only. To illustrate this point, we also applied the algorithm to data simulated from changed initial probabilities in the first marginal according to $\boldsymbol{\pi}_1 = (0.8, 0.2)$ (and otherwise identical parameters). Now most of the data points will stem from the heavier distribution in the mixture. Indeed, the estimate for $\lambda_1^{(\max)}$ in that case turns out to be -2.0152, while the tail estimate for the other marginals remains almost at the same value (-1.5966 and -2.4154).

Next we present a parameter–dependent example with real data, employing Algorithm 4.2.1.

Example 4.3.3. (Danish fire insurance data) Consider the famous Danish fire insurance claim data set (see e.g. Grün and Miljkovic (2019)). We propose here a bivariate matrix–Pareto distribution as a model for the components building and content with observations in the set $(1, \infty) \times (1, \infty)$. To that end, we first translate the sample to the origin, thus the sample has numerical values $\hat{\mathbb{E}}(\boldsymbol{X}) = (3.2476, 4.6856)'$ and $\hat{\rho}_\tau = 0.2143$. Then, we fit a bivariate matrix–Pareto distribution with $p_1 = p_2 = 2$ using Algorithm 4.2.1 with 2 000 steps (with a step–length of 0.05 and gradient ascent until the norm of the derivative is less than 0.1, the running time is 438 seconds), obtaining the following parameters:

$$\hat{\boldsymbol{\alpha}} = (0,\, 1,\, 0,\, 0)\,,$$

$$\hat{\boldsymbol{T}} = \begin{pmatrix} -2.6333 & 0 & 0.0005 & 2.6328 \\ 1.2788 & -3.8336 & 2.5548 & 0 \\ 0 & 0 & -10.9822 & 0 \\ 0 & 0 & 2.4131 & -2.4732 \end{pmatrix},$$

$$\hat{\boldsymbol{R}} = \begin{pmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{pmatrix},$$

$$\hat{\beta}_1 = 5.0377\,, \quad \hat{\beta}_2 = 13.4538\,.$$

The real part of the eigenvalues that determines the heaviness of the tails are $\lambda_1^{(\max)} = -2.6333$ and $\lambda_2^{(\max)} = -2.4732$, respectively. The fitted distribution has mean $\mathbb{E}(\boldsymbol{X}) = (3.1698, 4.6803)'$. Moreover, estimating ρ_τ via simulation gives $\hat{\rho}_\tau = 0.2190$. From the QQ plots (Figure 4.6) and contour plots (Figure 4.7), we see that the fitted distribution is a reasonable model for the data.
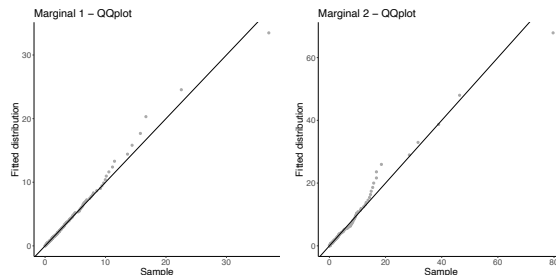

FIGURE 4.6. QQ plots of Danish fire insurance claim size sample versus fitted bivariate matrix–Pareto distribution using Algorithm 4.2.1.

This bivariate data set was recently also studied in (Albrecher et al., 2017, Sec. 4.5.2), where a splicing model with a bivariate mixed Erlang for the body and a bivariate Generalized Pareto distribution (GPD) for the tail was proposed. That approach required a threshold selection for the fitting of the tails, and univariate extreme value analysis led there to values of regular variation of around 2 for the building component and 1.67 for the contents component. Even though our estimates are further away from these values than their bivariate model (1.75 and 1.54, respectively), we would like to emphasize that the fitting of a matrix–Pareto distribution does not require any threshold selection. Furthermore, if we were to use more phases and a general form of the sub–intensity and reward matrices, the fit would quickly improve and for about 6 phases reach the accuracy of the bivariate GPD model, but then the overall number of parameters compared to the size of the present data set may not be considered commensurate, which is why we stick to the above
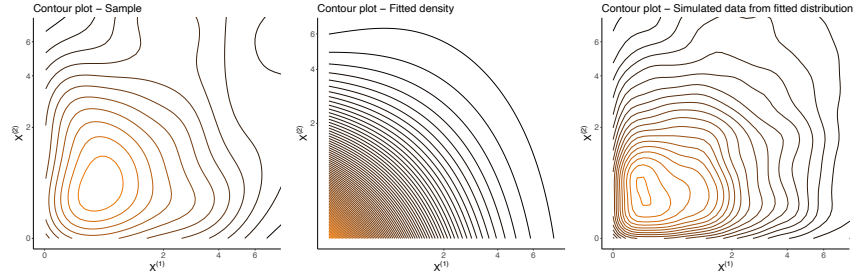
FIGURE 4.7. Contour plot of the Danish fire insurance claim size sample (left), contour plot of the Matrix–Pareto distribution fitted using Algorithm 4.2.1 (middle) and contour plot of the simulated sample from the fitted distribution (right).

choice. Note that our proposed procedure is fully automatic and the respective implementation can easily be applied to any other data set as well.

4.4. **Multivariate Matrix–Weibull models.** Let $\boldsymbol{X} = (g_1(Y^{(1)}), \dots, g_d(Y^{(d)}))'$, where $\boldsymbol{Y} \sim$ MPH$^*(\boldsymbol{\pi}, \boldsymbol{T}, \boldsymbol{R})$ and $g_j(y) = y^{1/\beta_j}$, $\beta_j > 0$, $j = 1, \dots, d$, then we say that $\boldsymbol{X}$ has a *multivariate matrix–Weibull distribution*. Some special properties of this type of distribution are:

(1) Marginal distributions are matrix–Weibull distributed.
(2) For a vector $\boldsymbol{a} = (a^{(1)}, \dots, a^{(d)})$, with $a^{(j)} > 0$, $j = 1, \dots, d$, $\Delta(\boldsymbol{a})\boldsymbol{X}$ is multivariate matrix–Weibull distributed.

For the bivariate case we get

$$f_{\boldsymbol{X}}\left(x^{(1)}, x^{(2)}\right) = \boldsymbol{\alpha} e^{\boldsymbol{T}_{11}(x^{(1)})^{\beta_1}} \boldsymbol{T}_{12} e^{\boldsymbol{T}_{22}(x^{(2)})^{\beta_2}} (-\boldsymbol{T}_{22}) \boldsymbol{e}\, \beta_1 (x^{(1)})^{\beta_1 - 1} \beta_2 (x^{(2)})^{\beta_2 - 1}\,.$$

and

$$\bar{F}_{\boldsymbol{X}}(x^{(1)}, x^{(2)}) = \boldsymbol{\alpha} \left(-\boldsymbol{T}_{11}\right)^{-1} e^{\boldsymbol{T}_{11}(x^{(1)})^{\beta_1}} \boldsymbol{T}_{12} e^{\boldsymbol{T}_{22}(x^{(2)})^{\beta_2}} \boldsymbol{e}$$

Remark 4.4.1. The Marshall–Olkin Weibull distribution (see Hanagal (1996b)) is a particular case of this distribution.

4.4.1. *Parameter estimation.* In contrast to Section 4.3.1, all transformations are parameter–dependent, and the fitting procedures of the previous subsection are not applicable. However, we can apply Algorithm 4.2.1 in the bivariate case.

Example 4.4.1. (Bivariate Matrix–Weibull) We generate an i.i.d. sample of size 5 000 of a bivariate random vector with matrix–Weibull marginals with parameters

$$\boldsymbol{\pi}_1 = (0.8,\ 0.2)\,,$$
$$\boldsymbol{T}_1 = \begin{pmatrix} -1 & 0.5 \\ 0 & -0.5 \end{pmatrix}\,,$$
$$\beta_1 = 0.4$$

for the first marginal and

$$\boldsymbol{\pi}_2 = (0.5,\ 0.25,\ 0.25)\,,$$
$$\boldsymbol{T}_2 = \begin{pmatrix} -1 & 1 & 0 \\ 0 & -0.5 & 0.5 \\ 0 & 0 & -0.1 \end{pmatrix}\,,$$
$$\beta_2 = 0.6$$

for the second marginal, and a Gaussian copula with parameter $\rho = 0.5$. While any copula, or also simply a bivariate matrix–Weibull based on a MPH* construction could be used, we choose the

Gaussian copula here to illustrate that the algorithm is able to work with any type of dependence structure. This distribution has theoretical mean $\mathbb{E}(\boldsymbol{X}) = (18.7997, 86.3711)'$. The sample has numerical values $\hat{\mathbb{E}}(\boldsymbol{X}) = (18.7690, 88.1637)'$ and $\hat{\rho}_\tau = 0.3431$.

We fit a bivariate matrix–Weibull distribution with $p_1 = p_2 = 3$ using Algorithm 4.2.1 with $1\,500$ steps (with a running time of $3\,930$ seconds for a step–length of 10^{-5}), getting the following parameters:

$$\hat{\boldsymbol{\alpha}} = (0.2101,\, 0,\, 0.7899,\, 0,\, 0,\, 0)\,,$$

$$\hat{\boldsymbol{T}} = \begin{pmatrix} -4.2507 & 0.5527 & 1.2916 & 0 & 0 & 2.4064 \\ 0 & -0.3069 & 0 & 0 & 0.3069 & 0 \\ 0.0089 & 0.2575 & -0.6903 & 0.4238 & 0 & 0 \\ 0 & 0 & 0 & -0.0946 & 0 & 0.0946 \\ 0 & 0 & 0 & 0.0360 & -0.0360 & 0 \\ 0 & 0 & 0 & 0.0026 & 0 & -0.2542 \end{pmatrix},$$

$$\hat{\boldsymbol{R}} = \begin{pmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \end{pmatrix},$$

$$\beta_1 = 0.4689\,, \quad \beta_2 = 0.7340\,.$$

One sees that the algorithm estimates the shape parameters of the matrix–Weibull marginals reasonably well. The fitted distribution has mean $\mathbb{E}(\boldsymbol{X}) = (18.6988, 88.1166)'$, and from simulated data we get $\hat{\rho}_\tau = 0.3236$. The QQ and contour plots are given in Figure 4.8 and Figure 4.10, respectively. The log–likelihood of the fitted bivariate matrix–Weibull is $-39\,748$, which is to be compared with the log–likelihood $-39\,687.19$ using the original distribution.
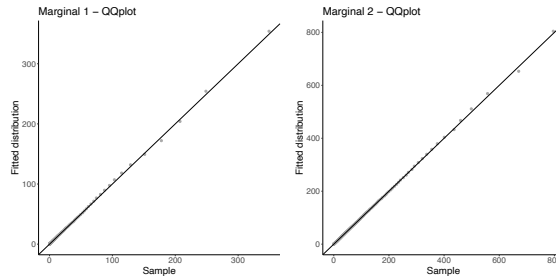

FIGURE 4.8. QQ plots of sample versus fitted bivariate matrix–Weibull distribution using Algorithm 4.2.1.

Remark 4.4.2. In all examples of this section the marginals were assumed to be of the same type (both matrix–Pareto or both matrix–Weibull). We would like to mention that the generality of Algorithm 4.2.1 also allows to fit models with marginals of different types (e.g. one marginal matrix–Pareto and the other matrix–Weibull).

5. Conclusion

In this paper we provided a guide for the statistical fitting of homogeneous and inhomogeneous phase–type distributions to data, both for the univariate and multivariate case. For that purpose, we derived a new EM algorithm for IPH distributions that are obtained through parameter–dependent transformations. In addition, we introduced new classes of multivariate distributions with IPH
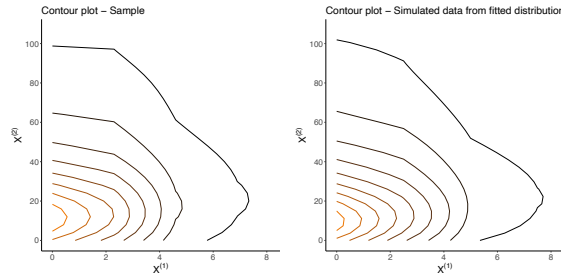
FIGURE 4.9. Contour plot of simulated sample (left) and contour plot of a simulated sample from the bivariate Matrix–Weibull distribution fitted using Algorithm 4.2.1 (right).
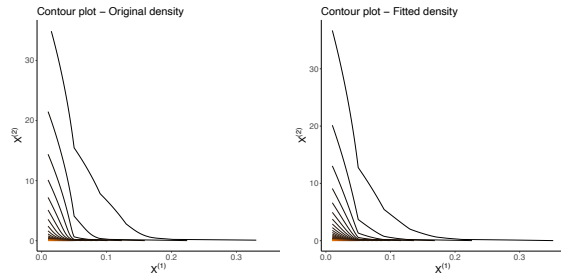

FIGURE 4.10. Contour plot of original bivarite matrix-Weibull (left) and contour plot of bivariate Matrix–Weibull distribution fitted using Algorithm 4.2.1 (right).

marginals and some attractive properties. As a by-product, we amended the estimation method proposed by Breuer (2016) for the homogeneous MPH* case and illustrated its usefulness and flexibility. We furthermore discussed extensions for censored data and the fitting of the phase–type classes to given continuous joint distribution functions. The performance of the proposed algorithms was exemplified in various numerical examples, both on simulated and real data. In order to facilitate the implementation of the proposed algorithms for fitting this general class of distributions to given data, a respective R package is in preparation and will be made available on Cran.

References

Ahlström, L., Olsson, M., and Nerman, O. (1999). A parametric estimation procedure for relapse time distributions. *Lifetime Data Analysis*, 5(2):113–132.

Albrecher, H., Beirlant, J., and Teugels, J. L. (2017). *Reinsurance: Actuarial and Statistical Aspects*. John Wiley & Sons, Chichester.

Albrecher, H. and Bladt, M. (2019). Inhomogeneous phase-type distributions and heavy tails. *Journal of Applied Probability*, 56(4):1044–1064.

Albrecher, H., Bladt, M., and Bladt, M. (2020a). Matrix Mittag–Leffler distributions and modeling heavy-tailed risks. *Extremes.* To appear. doi: 10.1007/s10687-020-00377-0.

Albrecher, H., Bladt, M., and Bladt, M. (2020b). Multivariate fractional phase–type distributions. *arXiv preprint arXiv:2003.11122*.

Albrecher, H., Bladt, M., and Bladt, M. (2020c). Multivariate Matrix Mittag-Leffler distributions. *Ann. Inst. Statist. Math.* In Press, doi: 10.1007/s10463-020-00750-7.

Arnold, B. C. (2015). *Pareto distributions*. Chapman and Hall/CRC.

Asmussen, S., Laub, P. J., and Yang, H. (2019). Phase-type models in life insurance: Fitting and valuation of equity-linked benefits. *Risks*, 7(1):17.

Asmussen, S., Nerman, O., and Olsson, M. (1996). Fitting phase-type distributions via the em algorithm. *Scandinavian Journal of Statistics*, pages 419–441.

Beirlant, J., Goegebeur, Y., Segers, J., and Teugels, J. L. (2004). *Statistics of extremes: theory and applications*. John Wiley & Sons, Chichester.

Bladt, M., Gonzalez, A., and Lauritzen, S. L. (2003). The estimation of phase-type related functionals using Markov Chain Monte Carlo methods. *Scandinavian Actuarial Journal*, 2003(4):280–300.

Bladt, M. and Nielsen, B. F. (2017). *Matrix-Exponential Distributions in Applied Probability*. Springer.

Bladt, M., Nielsen, B. F., and Peralta, O. (2019). Parisian types of ruin probabilities for a class of dependent risk-reserve processes. *Scandinavian Actuarial Journal*, 2019(1):32–61.

Bladt, M., Nielsen, B. F., and Samorodnitsky, G. (2015). Calculation of ruin probabilities for a dense class of heavy tailed distributions. *Scandinavian Actuarial Journal*, 2015(7):573–591.

Bladt, M. and Rojas-Nandayapa, L. (2017). Fitting phase–type scale mixtures to heavy–tailed data and distributions. *Extremes*, pages 1–29.

Breuer, L. (2016). A semi-explicit density function for Kulkarni's bivariate phase-type distribution. *Stochastic Models*, 32(4):632–642.

Cai, J. and Li, H. (2005a). Conditional tail expectations for multivariate phase-type distributions. *Journal of Applied Probability*, 42(3):810–825.

Cai, J. and Li, H. (2005b). Multivariate risk model of phase type. *Insurance: Mathematics and Economics*, 36(2):137–152.

Camarda, C. G. (2012). Mortalitysmooth: An R package for smoothing Poisson counts with P-splines. *Journal of Statistical Software*, 50:1–24.

Davis, R. A. and Resnick, S. I. (1996). Limit theory for bilinear processes with heavy-tailed noise. *The Annals of Applied Probability*, 6(4):1191–1210.

Gompertz, B. (1825). On the nature of the function expressive of the law of human mortality, and on a new mode of determining the value of life contingencies. *Philosophical Transactions of the Royal Society of London*, 115:513583.

Grün, B. and Miljkovic, T. (2019). Extending composite loss models using a general framework of advanced computational tools. *Scandinavian Actuarial Journal*, 2019(8):642–660.

Hanagal, D. D. (1996a). A multivariate Pareto distribution. *Communications in Statistics-Theory and Methods*, 25(7):1471–1488.

Hanagal, D. D. (1996b). A multivariate Weibull distribution. *Economic Quality Control*, 11:193–200.

Herbertsson, A. (2011). Modelling default contagion using multivariate phase-type distributions. *Review of Derivatives Research*, 14(1):1–36.

Higham, N. J. (2008). *Functions of matrices: theory and computation*. SIAM.

Kibble, W. (1941). A two-variate gamma type distribution. *Sankhyā: The Indian Journal of Statistics*, pages 137–150.

Kulkarni, V. G. (1989). A new class of multivariate phase type distributions. *Operations Research*, 37(1):151–158.

Lee, L. (1979). Multivariate distributions having Weibull properties. *Journal of Multivariate Analysis*, 9(2):267–277.

Mardia, K. V. (1962). Multivariate Pareto distributions. *The Annals of Mathematical Statistics*, 33(3):1008–1015.

McNeil, A. J., Frey, R., and Embrechts, P. (2015). *Quantitative risk management: concepts, techniques and tools-revised edition.* Princeton university press.

Neuts, M. (1975). Probability distributions of phase type. In *Liber Amicorum Professor Emeritus H. Florin*, pages 173–206. Department of Mathematics, University of Louvian, Belgium.

Neuts, M. F. (1995). *Algorithmic probability: a collection of problems*, volume 3. CRC Press.

Okamura, H., Dohi, T., and Trivedi, K. S. (2011). A refined em algorithm for ph distributions. *Performance Evaluation*, 68(10):938–954.

Olsson, M. (1996). Estimation of phase-type distributions from censored data. *Scandinavian Journal of Statistics*, pages 443–460.

Olsson, M. (1998). The EMpht programme. Manual. Chalmers University of Technology and götborg university.

Sancetta, A. and Satchell, S. (2004). The Bernstein copula and its applications to modeling and approximations of multivariate distributions. *Econometric theory*, 20(3):535–562.

Van Loan, C. (1978). Computing integrals involving the matrix exponential. *IEEE transactions on automatic control*, 23(3):395–404.

DEPARTMENT OF ACTUARIAL SCIENCE, FACULTY OF BUSINESS AND ECONOMICS, UNIVERSITY OF LAUSANNE AND SWISS FINANCE INSTITUTE, UNIL-DORIGNY, 1015 LAUSANNE
Email address: `hansjoerg.albrecher@unil.ch`

DEPARTMENT OF MATHEMATICS, UNIVERSITY OF COPENHAGEN, UNIVERSITETSPARKEN 5, DK-2100 COPENHAGEN, DENMARK
Email address: `Bladt@math.ku.dk`

DEPARTMENT OF MATHEMATICS, UNIVERSITY OF COPENHAGEN, UNIVERSITETSPARKEN 5, DK-2100 COPENHAGEN, DENMARK
Email address: `jorge@math.ku.dk`