Annals of Actuarial Science (2025), 1–35
doi:10.1017/S1748499525100134

ACTUARIAL SOFTWARE

matrixdist: an R package for statistical analysis of matrix distributions

Martin Bladt[1], Alaric Mueller[2] and Jorge Yslas[3]

[1]Department of Mathematical Sciences, University of Copenhagen, Copenhagen, Denmark; [2]Faculty of Business and Economics, University of Lausanne, Lausanne, Switzerland; and [3]Institute for Financial and Actuarial Mathematics, University of Liverpool, Liverpool, UK
Corresponding author: Jorge Yslas; Email: jorge.yslas1@gmail.com

(Received 22 October 2024; revised 28 August 2025; accepted 03 September 2025)

Abstract

The `matrixdist` R package provides a comprehensive suite of tools for the statistical analysis of matrix distributions, including phase-type, inhomogeneous phase-type, discrete phase-type, and related multivariate distributions. This paper introduces the package and its key features, including the estimation of these distributions and their extensions through expectation-maximization algorithms, as well as the implementation of regression through the proportional intensities and mixture-of-experts models. Additionally, the paper provides an overview of the theoretical background, discusses the algorithms and methods implemented in the package, and offers practical examples to illustrate the application of `matrixdist` in real-world actuarial problems. The `matrixdist` R package aims to provide researchers and practitioners a wide set of tools for analyzing and modeling complex data using matrix distributions.

Keywords: EM algorithm; matrix distributions; regression

1. Introduction

In recent years, matrix distributions such as phase-type (PH), inhomogeneous phase-type (IPH), discrete phase-type (DPH), and related multivariate distributions have gained prominence in various fields due to their ability to seamlessly model complex or heterogeneous data. Matrix distributions can be employed to capture diverse behaviors and dependencies in real-world scenarios, making them particularly useful in areas such as survival analysis, actuarial science, queuing theory, and finance.

PH distributions (cf. Bladt & Nielsen, 2017) are a natural extension of the exponential distribution and are defined as the time to reach an absorbing state in an otherwise transient time-homogeneous pure-jump Markov process. These distributions possess closed-form formulas in terms of matrices for various functionals, such as density, cumulative distribution function, Laplace transform, moments, and can approximate any other distribution on the positive real line. Although their support is restricted to the positive real line, they can also be used to model real-valued data through suitable transformations. For example, Albrecher *et al.* (2022b) employ PH distributions to model the absolute log-returns of financial assets. Since only the time to absorption is observed and not the actual stochastic process trajectory, models based on PH distributions are hidden Markov models. Hence, they are typically estimated using the expectation-maximization (EM) algorithm (cf. Asmussen *et al.*, 1996). However, their tail behavior remains exponential, which can be a limiting factor in certain applications that require a more accurate tail description. This shortcoming has led to the exploration of classes extending PH

distributions, offering different tail behaviors while retaining PH-like properties. One such class is comprised of IPH distributions (Albrecher & Bladt, 2019), which can be defined equivalently as the law of a transformed PH-distributed random variable or the absorption time of a time-inhomogeneous pure-jump Markov process. The tail behavior of the resulting IPH distribution can be as heavy or light as desired, depending on the chosen transformation. Moreover, for parametric transformations, the estimation procedure has been outlined in Albrecher *et al.* (2022c). Several extensions to the multivariate setting of PH distributions have been introduced in the literature. The most general construction can be found in Kulkarni (1989), called the MPH* class, whose definition consists of considering marginals that depend on a unique Markov jump process but with different rewards at each state. However, explicit expressions for the joint density and distribution function are not available for this class. A more tractable subclass, named the mPH class (see Bladt, 2023), offers greater analytical convenience; we also highlight the work of Cheung *et al.* (2022), which provides closed-form expressions for common risk measures, including value-at-risk (VaR) and tail value-at-risk (TVaR), within the framework of multivariate matrix-exponential affine mixtures, a class that encompasses both the mPH distributions and, in particular, the univariate PH distributions. Finally, with respect to multivariate extensions of the IPH class, Albrecher *et al.* (2022c), Bladt (2023) presented some alternatives that allow estimation.

For modeling count data, the DPH class (cf. Bladt & Nielsen, 2017) provides a tractable and flexible set of distributions, which also allows for estimation via an EM algorithm. Formally, DPH distributions are defined as the time to reach absorption in a discrete Markov chain (MC) with one absorbing state and the remaining states being transient. However, they can also be viewed as an extension of the geometric distribution. The more general class of multivariate DPH distributions was introduced in Campillo-Navarro (2018), employing a similar construction principle as the MPH* class via rewards. More recently, Bladt & Yslas (2023c) presented other more tractable classes of DPH distributions that offer similar versatility in modeling count data.

Matrix distributions have proven to be useful in various applications, motivating the development of regression models tailored to these distributions. Two notable regression models for matrix distributions include the proportional intensities (PI) model and the mixture-of-experts (MoE) specification. The PI model, as proposed by Albrecher *et al.* (2022a), extends the classical Cox proportional hazards model to the IPH setting. This model enables the incorporation of covariate information directly into the intensity matrix of the underlying Markov process. On the other hand, the MoE specification, originally introduced in Bladt & Yslas (2023b), incorporates the covariate information in the vector of initial probabilities and can easily be modified to the multivariate case (cf. Albrecher *et al.*, 2023b).

This paper presents the `matrixdist` package (Bladt & Yslas, 2023a), designed to work with matrix distributions and implemented in R (R Core Team, 2024). The `matrixdist` package leverages the S4 class system in R to ensure a well-structured and organized implementation of the various distribution classes and their associated methods. By utilizing S4 classes, the package offers a user-friendly interface for interacting with PH, IPH, DPH, and related multivariate and regression distributions. Users can create distribution objects with their respective parameters, and the package will automatically manage the underlying data structures and computations. This design promotes code reusability, maintainability, and extensibility, making it easier to add new distribution classes and methods in the future while ensuring a consistent and coherent user experience. The object-oriented design can be particularly efficient when dealing with large numbers of parameters during statistical estimation. Furthermore, most computationally intensive functions are implemented using the C++ language to enhance performance. These functions are then made accessible in the R language via the `Rcpp` package, ensuring both speed and ease-of-use for the `matrixdist` package. The `matrixdist` package is now available in CRAN (https://cran.r-project.org/package=matrixdist) and can be installed through `install.packages("matrixdist")`. The source code is available in the GitHub repository https://github.com/martinbladt/matrixdist_1.0, where bug reports can also be submitted.

https://doi.org/10.1017/S1748499525100134 Published online by Cambridge University Press

It is worth mentioning that the R package `matrixdist` offers a preferable solution compared to existing packages for working with matrix distributions. To provide context, we briefly summarize some alternative software tools for matrix distributions:

- The R package `actuar` (Dutang *et al.*, 2008) contains implementations for the density, cumulative distribution, moments, and moment-generating function of univariate PH distributions.
- The R package `mapfit` (Okamura & Dohi, 2015) offers some estimation methods for PH distributions and for Markovian Arrival Processes (MAP). The latter can loosely be seen as dependent concatenations of PH variables (arrivals).
- The recent `PhaseTypeR` package (Rivas-González *et al.*, 2023) includes implementations for reward transformations and functionals for some of the multivariate extensions of homogeneous PH and DPH distributions. However, no statistical estimation is considered.

The `matrixdist` package stands out by providing a comprehensive and efficient suite of tools that cover univariate, multivariate, continuous, discrete, and regression matrix models, and their statistical estimation. Right-censoring is also supported. The package's versatility, completeness, and high-speed performance make it an ideal choice for researchers and practitioners across all matrix distribution domains.

The remainder of the paper is organized as follows. Section 2 provides the mathematical formulation of univariate DPH, PH, and IPH distributions, their basic properties, and regression extensions. Similar information for multivariate matrix distribution is given in Section 3, where the multivariate extensions of the IPH and DPH classes are introduced. A discussion on estimation methods for the introduced models via EM algorithms is provided in Section 4, and detailed illustrations on the use of the package are provided in Section 5. Finally, Section 6 gives a condensed summary of the methods available in `matrixdist` and shows how further information can be accessed within the package's documentation.

2. Univariate matrix distributions

2.1 Discrete phase-type distributions

Let $(Z_n)_{n\in\mathbb{N}_0}$ be a discrete MC on a finite state space $\{1,\dots,p,p+1\}$ where the first p states are transient, and state $p+1$ is absorbing. Then, the MC has transition matrix $\mathbf{P}$ given by:

$$\mathbf{P}=\begin{pmatrix}\mathbf{S} & \mathbf{s}\\ \mathbf{0} & 1\end{pmatrix},$$

where $\mathbf{S}$ is a $p\times p$ matrix, known as sub-transition matrix, and $\mathbf{s}$ is a column vector of dimension p. Considering that the rows of $\mathbf{P}$ sum to 1, the following relation holds $\mathbf{s}=\mathbf{e}-\mathbf{S}\mathbf{e}$, with $\mathbf{e}=(1,\dots,1)^\top$ the p-dimensional column vector of ones. Furthermore, we assume that $(Z_n)_{n\in\mathbb{N}_0}$ may start in any transient state with a certain probability $\alpha_k=\mathbb{P}(Z_0=k)$, $k=1,\dots,p$, and let $\boldsymbol{\alpha}=(\alpha_1,\dots,\alpha_p)$. Then, the time until absorption $N=\inf\{n\geq 1\mid Z_n=p+1\}$ is said to be DPH-distributed with representation $(\boldsymbol{\alpha},\mathbf{S})$, and we write $N\sim\mathrm{DPH}(\boldsymbol{\alpha},\mathbf{S})$ or $N\sim\mathrm{DPH}_p(\boldsymbol{\alpha},\mathbf{S})$. Such a random variable has probability mass function q and distribution function Q given by

$$q(n)=\boldsymbol{\alpha}\mathbf{S}^{n-1}\mathbf{s}\,,\quad n\geq 1\,,$$

$$Q(n)=1-\boldsymbol{\alpha}\mathbf{S}^{n}\mathbf{e}\,,\quad n\geq 1\,.$$

The absorption time N has κ-factorial moments, $\kappa\in\mathbb{N}$, given by

$$\mathbb{E}(N(N-1)\cdots(N-\kappa+1))=\kappa!\boldsymbol{\alpha}\mathbf{S}^{\kappa-1}\left(\mathbf{I}-\mathbf{S}\right)^{-\kappa}\mathbf{e}\,.$$

https://doi.org/10.1017/S1748499525100134 Published online by Cambridge University Press

In particular, we have that $\mathbb{E}(N) = \boldsymbol{\alpha}\,(\mathbf{I}-\mathbf{S})^{-1}\,\mathbf{e}$. Furthermore, its probability-generating function P_N is given by

$$P_N(t) = \mathbb{E}(t^N) = \boldsymbol{\alpha}\left(t^{-1}\mathbf{I}-\mathbf{S}\right)\mathbf{s}\,, \quad |t| \le 1\,,$$

where $\mathbf{I}$ is the identity matrix of appropriate dimension.

Let $N_1 \sim \mathrm{DPH}_{p_1}(\boldsymbol{\alpha}_1, \mathbf{S}_1)$ and $N_2 \sim \mathrm{DPH}_{p_2}(\boldsymbol{\alpha}_2, \mathbf{S}_2)$ be two independent DPH-distributed random variables. Then,

$$N_1 + N_2 \sim \mathrm{DPH}_{p_1+p_2}\left((\boldsymbol{\alpha}_1, \mathbf{0})\,, \begin{pmatrix} \mathbf{S}_1 & \mathbf{s}_1\boldsymbol{\alpha}_2 \\ \mathbf{0} & \mathbf{S}_2 \end{pmatrix}\right),$$

$$\min(N_1, N_2) \sim \mathrm{DPH}_{p_1p_2}(\boldsymbol{\alpha}_1 \otimes \boldsymbol{\alpha}_2, \mathbf{S}_1 \otimes \mathbf{S}_2),$$

and

$$\max(N_1, N_2) \sim \mathrm{DPH}_{p_1p_2+p_1+p_2}\left((\boldsymbol{\alpha}_1 \otimes \boldsymbol{\alpha}_2, \mathbf{0}, \mathbf{0})\,, \begin{pmatrix} \mathbf{S}_1 \otimes \mathbf{S}_2 & \mathbf{S}_1 \otimes \mathbf{s}_2 & \mathbf{s}_1 \otimes \mathbf{S}_2 \\ \mathbf{0} & \mathbf{S}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{S}_2 \end{pmatrix}\right),$$

where $\otimes$ and $\oplus$ denote the Kronecker product and sum, respectively. It should be noted that the above parametric representation of $N_1 + N_2$ is obtained using a sequential addition interpretation, where N_1 is assumed to be absorbed prior to the initiation of N_2.

Furthermore, if $U \sim \mathit{Bernoulli}(\nu)$, $\nu \in [0,1]$, then

$$UN_1 + (1-U)N_2 \sim \mathrm{DPH}_{p_1+p_2}\left((\nu\boldsymbol{\alpha}_1\,, (1-\nu)\boldsymbol{\alpha}_2)\,, \begin{pmatrix} \mathbf{S}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{S}_2 \end{pmatrix}\right).$$

Thus, the DPH class is closed under addition, minima, maxima, and finite mixtures. We refer to Bladt & Nielsen (2017) for a detailed description of DPH distributions.

Regression

Recently, a regression model for frequency modeling based on DPH distributions was introduced in Bladt & Yslas (2023c) as a generalization of MoE models. The main idea is to incorporate the covariate information on the vector of initial probabilities, which play the role of expert weights. The resulting distributional model is flexible, as it possesses the property of denseness on more general regression models satisfying some mild technical conditions. Next, we present the fundamental concepts of this regression model. Define the mapping

$$\boldsymbol{\alpha} : D \subset \mathbb{R}^h \to \Delta^{p-1},$$

where $\Delta^{p-1} = \{(\alpha_1, \ldots, \alpha_p) \in \mathbb{R}^p \mid \sum_k \alpha_k = 1,\ \alpha_k \ge 0\ \forall k\}$ is the standard $(p-1)$ simplex. Then, for any given vector of covariate information $\mathbf{X} = (X_1, \ldots, X_h) \in \mathbb{R}^h$, the initial probabilities of the underlying MC of a DPH distribution are taken to be $\mathbb{P}(Z_0 = k) = \alpha_k(\mathbf{x}) := (\boldsymbol{\alpha}(\mathbf{x}))_k$, $k = 1, \ldots, p$. Thus, we say that $N \mid \mathbf{X} \sim \mathrm{DPH}(\boldsymbol{\alpha}(\mathbf{X}), \mathbf{S})$ follows the DPH-MoE specification. For $D = \mathbb{R}^h$, we have a convenient parametrization of initial probabilities, namely the softmax parametrization given by

$$\alpha_k(\mathbf{X}; \boldsymbol{\gamma}) = \frac{\exp\left(\mathbf{X}\boldsymbol{\gamma}_k\right)}{\sum_{j=1}^{p}\exp\left(\mathbf{X}\boldsymbol{\gamma}_j\right)} \quad k = 1, \ldots, p,$$

where $\boldsymbol{\gamma}_k \in \bar{\mathbb{R}}^h$ and $\boldsymbol{\gamma} = (\boldsymbol{\gamma}_1^\top, \ldots, \boldsymbol{\gamma}_p^\top)^\top \in \bar{\mathbb{R}}^{p\times h}$. More details about this model as well as precise denseness considerations can be found in Bladt & Yslas (2023c). We also refer to Bladt & Yslas (2023b) for an extension of this regression model to continuous PH distributions.

https://doi.org/10.1017/S1748499525100134 Published online by Cambridge University Press

2.2 Continuous phase-type distributions

Let $(J_t)_{t\geq 0}$ be a time-homogeneous Markov jump process on a finite state space $\{1,\ldots,p,p+1\}$, where states $1,\ldots,p$ are transient, and state $p+1$ is absorbing. Then, the intensity matrix $\mathbf{\Lambda}$ of $(J_t)_{t\geq 0}$ is of the form

$$\mathbf{\Lambda} = \begin{pmatrix} \mathbf{S} & \mathbf{s} \\ \mathbf{0} & 0 \end{pmatrix},$$

where $\mathbf{S}$ is a $p\times p$ matrix, called a sub-intensity matrix, and $\mathbf{s}$ is a p-dimensional column vector. Since every row of $\mathbf{\Lambda}$ sums to zero, it follows that $\mathbf{s} = -\mathbf{S}\,\mathbf{e}$. Assume that the process starts somewhere in the transient space with probability $\alpha_k = \mathbb{P}(J_0 = k)$, $k = 1,\ldots,p$, and let $\boldsymbol{\alpha} = (\alpha_1,\ldots,\alpha_p)$. Here, it is assumed that the probability of starting in the absorbing state $p+1$ is zero, that is, $\mathbb{P}(J_0 = p+1) = 0$. Then, we say that the time until absorption Y has PH distribution with representation $(\boldsymbol{\alpha},\mathbf{S})$, and we write $Y\sim \mathrm{PH}(\boldsymbol{\alpha},\mathbf{S})$ or $Y\sim \mathrm{PH}_p(\boldsymbol{\alpha},\mathbf{S})$. The density f and distribution function F of $Y\sim \mathrm{PH}(\boldsymbol{\alpha},\mathbf{S})$ are given by

$$f(y) = \boldsymbol{\alpha}\exp\left(\mathbf{S}y\right)\mathbf{s}\,,\quad y>0\,,$$

$$F(y) = 1-\boldsymbol{\alpha}\exp\left(\mathbf{S}y\right)\mathbf{e}\,,\quad y>0\,,$$

where the exponential of a matrix $\mathbf{A}$ is defined by

$$\exp\left(\mathbf{A}\right) = \sum_{n=0}^{\infty}\frac{\mathbf{A}^n}{n!}\,.$$

The efficient computation of the exponential of a matrix is not a straightforward task in high dimensions, and we refer to Moler & Van Loan (1978) for a survey of different methods. The moments of $Y\sim \mathrm{PH}(\boldsymbol{\alpha},\mathbf{S})$ are given by

$$\mathbb{E}\left(Y^\theta\right) = \Gamma\left(1+\theta\right)\boldsymbol{\alpha}\left(-\mathbf{S}\right)^{-\theta}\mathbf{e}\,,\quad \theta>0\,,$$

with $\Gamma(\,\cdot\,)$ being the standard gamma function. Moreover, its Laplace transform L_Y is given by

$$L_Y(u) = \mathbb{E}\left(\exp\left(-uY\right)\right) = \boldsymbol{\alpha}\left(u\mathbf{I}-\mathbf{S}\right)\mathbf{s}\,,\quad u\geq 0\,.$$

Let $Y_1\sim \mathrm{PH}_{p_1}(\boldsymbol{\alpha}_1,\mathbf{S}_1)$ and $Y_2\sim \mathrm{PH}_{p_2}(\boldsymbol{\alpha}_2,\mathbf{S}_2)$ be independent PH-distributed random variables. Then

$$Y_1+Y_2\sim \mathrm{PH}_{p_1+p_2}\left(\left(\boldsymbol{\alpha}_1,\mathbf{0}\right),\begin{pmatrix}\mathbf{S}_1 & \mathbf{s}_1\boldsymbol{\alpha}_2\\ \mathbf{0} & \mathbf{S}_2\end{pmatrix}\right),$$

$$\min\left(Y_1,Y_2\right)\sim \mathrm{PH}_{p_1p_2}\left(\boldsymbol{\alpha}_1\otimes\boldsymbol{\alpha}_2,\mathbf{S}_1\oplus\mathbf{S}_2\right),$$

and

$$\max\left(Y_1,Y_2\right)\sim \mathrm{PH}_{p_1p_2+p_1+p_2}\left(\left(\boldsymbol{\alpha}_1\otimes\boldsymbol{\alpha}_2,\mathbf{0},\mathbf{0}\right),\begin{pmatrix}\mathbf{S}_1\oplus\mathbf{S}_2 & \mathbf{I}\otimes\mathbf{s}_2 & \mathbf{s}_1\otimes\mathbf{I}\\ \mathbf{0} & \mathbf{S}_1 & \mathbf{0}\\ \mathbf{0} & \mathbf{0} & \mathbf{S}_2\end{pmatrix}\right),$$

This means that the PH class is closed under addition, minima, and maxima. In fact, it is also closed under any order statistic (cf. Bladt & Nielsen, 2017) for the exact and more involved expression). Moreover, the class is closed under finite mixtures. Concretely, if $U\sim \mathit{Bernoulli}(\nu)$, $\nu\in[0,1]$, then

$$UY_1+(1-U)Y_2\sim \mathrm{PH}_{p_1+p_2}\left(\left(\nu\boldsymbol{\alpha}_1,(1-\nu)\boldsymbol{\alpha}_2\right),\begin{pmatrix}\mathbf{S}_1 & \mathbf{0}\\ \mathbf{0} & \mathbf{S}_2\end{pmatrix}\right).$$

Furthermore, when the mixing probabilities are given by a DPH distribution, the resulting distribution is again PH. More specifically, let $N\sim \mathrm{DPH}_q(\boldsymbol{\pi},\mathbf{T})$ and $Y_1, Y_2\ldots$ be iid, independent

https://doi.org/10.1017/S1748499525100134 Published online by Cambridge University Press

of N, with common element $Y \sim \mathrm{PH}_p(\boldsymbol{\alpha}, \mathbf{S})$. Then

$$\sum_{n=1}^{N} Y_n \sim \mathrm{PH}_{qp}(\boldsymbol{\pi} \otimes \boldsymbol{\alpha}, \mathbf{I} \otimes \mathbf{S} + \mathbf{T} \otimes \mathbf{s}\boldsymbol{\alpha}).$$

We refer to Bladt & Nielsen (2017) for a comprehensive reading on PH distributions.

By decomposing the matrix $\mathbf{S}$ into its Jordan normal form, we observe that the right tail of a PH distribution exhibits asymptotically exponential behavior. This limitation motivates the introduction of PH extensions that can accommodate diverse tail behaviors. One such extension is the class of IPH distributions, which is presented in the following section.

2.3 Inhomogeneous phase-type distributions

In Albrecher & Bladt (2019), the class of IPH distribution was introduced by allowing the Markov jump process to be time-inhomogeneous in the construction principle of PH distributions. In this way, $(J_t)_{t\geq 0}$ has an intensity matrix of the form:

$$\boldsymbol{\Lambda}(t) = \begin{pmatrix} \mathbf{S}(t) & \mathbf{s}(t) \\ \mathbf{0} & 0 \end{pmatrix},$$

where $\mathbf{s}(t) = -\mathbf{S}(t)\,\mathbf{e}$. Here, it is assumed that the vector of initial probabilities is $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_p)$. Then, we say that the time until absorption Y has IPH distribution with representation $(\boldsymbol{\alpha}, \mathbf{S}(t))$, and we write $Y \sim \mathrm{IPH}(\boldsymbol{\alpha}, \mathbf{S}(t))$. However, this setting is too general for applications since its functionals are given in terms of product integrals. Thus, we consider the subclass of IPH distribution when $\mathbf{S}(t)$ is of the form $\mathbf{S}(t) = \lambda(t)\mathbf{S}$, where λ is some known nonnegative real function, and $\mathbf{S}$ is a sub-intensity matrix. In this case, we write $Y \sim \mathrm{IPH}(\boldsymbol{\alpha}, \mathbf{S}, \lambda)$. Note that with $\lambda \equiv 1$, one gets the conventional PH distributions. This subclass is particularly suitable for applications due to the following property. If $Y \sim \mathrm{IPH}(\boldsymbol{\alpha}, \mathbf{S}, \lambda)$, then there exists a function g, named the inhomogeneity function, such that

$$Y \sim g(\tau),$$

where $\tau \sim \mathrm{PH}(\boldsymbol{\alpha}, \mathbf{S})$. The relationship between g and λ is given by the following expression:

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

The density f and distribution function F of $Y \sim \mathrm{IPH}(\boldsymbol{\alpha}, \mathbf{S}, \lambda)$ can again be expressed using the exponential of matrices and are given by

$$f(y) = \lambda(y)\,\boldsymbol{\alpha} \exp\left(\int_0^y \lambda(t)dt\,\mathbf{S}\right)\mathbf{s}, \quad y > 0,$$

$$F(y) = 1 - \boldsymbol{\alpha} \exp\left(\int_0^y \lambda(t)dt\,\mathbf{S}\right)\mathbf{e}, \quad y > 0.$$

The class of IPH distributions is no longer confined to exponential tails, despite being defined in terms of matrix exponentials. Instead, the function λ determines their exact asymptotic behavior (see Albrecher *et al.*, 2022a for details).

It turns out that a number of IPH distributions can be expressed as classical distributions with matrix-valued parameters properly defined using functional calculus. Table 1 contains the IPH transformations as implemented in the `matrixdist` package (see Albrecher & Bladt, 2019; Albrecher *et al.*, 2022a, c for further information and rationale on the different parametrizations).

https://doi.org/10.1017/S1748499525100134 Published online by Cambridge University Press

Table 1. Transformations

	$\lambda(t)$	$g(y)$	Parameters domain
Matrix-Pareto	$(t+\beta)^{-1}$	$\beta\left(\exp(y)-1\right)$	$\beta>0$
Matrix-Weibull	$\beta t^{\beta-1}$	$y^{1/\beta}$	$\beta>0$
Matrix-Lognormal	$\gamma(\log(t+1))^{\gamma-1}/(t+1)$	$\exp(y^{1/\gamma})-1$	$\gamma>1$
Matrix-Loglogistic	$\theta t^{\theta-1}/(t^{\theta}+\gamma^{\theta})$	$\gamma(\exp(y)-1)^{1/\theta}$	$\gamma,\theta>0$
Matrix-Gompertz	$\exp(\beta t)$	$\log(\beta y+1)/\beta$	$\beta>0$
Matrix-GEV	–	$\mu+\sigma(y^{-\xi}-1)/\xi$	$\mu\in\mathbb{R},\sigma>0,\xi\in\mathbb{R}$

Regression

At least two regression models based on IPH distributions exist in the literature. The first model, known as the PI model, was introduced in Albrecher *et al.* (2022a). The authors, inspired by the proportional hazards model (cf. Cox, 1972), formulated a way of regressing on the intensity function λ for different covariate information. More specifically, the main idea of the PI model is that the inhomogeneous intensity $\lambda(\,\cdot\,;\boldsymbol{\theta})$, which is assumed to be fully determined by a vector of parameters $\boldsymbol{\theta}$, is proportionally affected for different covariate information $\mathbf{X}\in\mathbb{R}^h$ according to a positive real-valued and measurable function $m(\,\cdot\,)$. In other words, the model assumes that

$$\lambda(t;\mathbf{X},\boldsymbol{\theta},\boldsymbol{\gamma})=\lambda(t;\boldsymbol{\theta})m(\mathbf{X}\boldsymbol{\gamma}),\quad t\geq 0,$$

where $\boldsymbol{\gamma}$ is a h-dimensional column vector containing regression coefficients. With this assumption, $Y\mid\mathbf{X}\sim\mathrm{IPH}(\boldsymbol{\alpha},\mathbf{S},\lambda(\,\cdot\,;\mathbf{X},\boldsymbol{\theta},\boldsymbol{\gamma}))$ with corresponding density and cumulative functions:

$$f(y)=m(\mathbf{X}\boldsymbol{\gamma})\lambda(y;\boldsymbol{\theta})\boldsymbol{\alpha}\exp\left(m(\mathbf{X}\boldsymbol{\gamma})\int_0^y\lambda(t;\boldsymbol{\theta})\,dt\mathbf{S}\right)\mathbf{s}\,,$$

and

$$F(y)=1-\boldsymbol{\alpha}\exp\left(m(\mathbf{X}\boldsymbol{\gamma})\int_0^y\lambda(t;\boldsymbol{\theta})\,dt\mathbf{S}\right)\mathbf{e}\,.$$

Typically, the measurable function $m()$ is taken to be $m(x)=\exp(x)$, which is the default implementation in `matrixdist`. In this framework, when $p=1$, one recovers the proportional hazard model. However, for $p>1$, the implied hazard functions between different subgroups can deviate from proportionality in the distribution body but are asymptotically proportional in the tail. For more details on the PI model, we refer the reader to Albrecher *et al.* (2022) and Bladt (2022).

The second regression model, called the phase-type mixture-of-experts (PH-MoE) model, was introduced in Bladt & Yslas (2023b) as a generalization of MoE models to the PH distributions framework and follows the same rationale as the DPH-MoE specification described above (cf. Bladt & Yslas, 2023b for further details).

3. Multivariate matrix distributions

In this section, we present the mathematical foundations of the different classes of multivariate matrix distributions implemented in `matrixdist`.

https://doi.org/10.1017/S1748499525100134 Published online by Cambridge University Press

3.1 Multivariate discrete phase-type distributions

MDPH class*

Let $N \sim \text{DPH}_p(\boldsymbol{\alpha}, \mathbf{S})$ be a DPH-distributed random variable with underlying MC $(Z_n)_{n\in\mathbb{N}_0}$. Let $\mathbf{r}_j = (r_j(1), \ldots, r_j(p))^\top$ be column vectors of dimension p taking values in $\mathbb{N}_0^p$, $j = 1, \ldots, d$, and let $\mathbf{R} = (\mathbf{r}_1, \ldots, \mathbf{r}_d)$ be a $p \times d$ matrix, called the reward matrix. We define

$$N^{(j)} = \sum_{n=0}^{N} r_j(Z_n)\,,$$

for all $j = 1, \ldots, d$. Then, we say that the random vector $\mathbf{N} = (N^{(1)}, \ldots, N^{(d)})$ has a multivariate DPH distribution of the MDPH* type, and we write $\mathbf{N} \sim \text{MDPH}^\star(\boldsymbol{\alpha}, \mathbf{S}, \mathbf{R})$. This class of distributions was introduced in Campillo-Navarro (2018), and the construction principle can be seen as an extension of the so-called transformation via rewards for univariate DPH distributions. In this contribution, the author showed that elements in this class possess explicit expressions for joint probability-generating function, joint moment-generating function, and joint moments. Moreover, the class is dense in the class of distributions with support in $\mathbb{N}^d$, and margins are DPH-distributed. However, general closed-form expressions for the joint density and distribution functions are not available, and the estimation of this general class is still an open question, limiting their use in practice. Nonetheless, there are subclasses of MDPH* distributions with explicit expressions of these two functionals that preserve the denseness property of the MDPH* class. The `matrixdist` package provides implementations for two of these subclasses, which we describe next.

fMDPH class

Let $\mathbf{N} = (N^{(1)}, N^{(2)}) \sim \text{MDPH}^\star(\boldsymbol{\alpha}, \mathbf{S}, \mathbf{R})$ with

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

Here, $\mathbf{S}_{11}$ and $\mathbf{S}_{22}$ are sub-transition matrices of dimensions p_1 and p_2 ($p_1 + p_2 = p$), respectively, $\mathbf{S}_{12}$ is a $p_1 \times p_2$ matrix satisfying $\mathbf{S}_{11}\mathbf{e} + \mathbf{S}_{12}\mathbf{e} = \mathbf{e}$, and $\boldsymbol{\pi}$ is a p_1-dimensional vector of initial probabilities. Then, one can show that the joint density of this random vector is given by

$$f_{\mathbf{N}}(n^{(1)}, n^{(2)}) = \boldsymbol{\pi}\mathbf{S}_{11}^{n^{(1)}-1}\mathbf{S}_{12}\mathbf{S}_{22}^{n^{(2)}-1}\mathbf{s}_2\,,$$

where $\mathbf{s}_2 = \mathbf{e} - \mathbf{S}_{22}\mathbf{e}$. In this case, we say that $\mathbf{N}$ is bivariate DPH-distributed of the feed-forward type, and we write $\mathbf{N} \sim \text{fMDPH}(\boldsymbol{\pi}, \mathbf{S}_{11}, \mathbf{S}_{12}, \mathbf{S}_{22})$. In particular, the marginals are parametrized by $N^{(1)} \sim \text{DPH}(\boldsymbol{\pi}, \mathbf{S}_{11})$ and $N^{(2)} \sim \text{DPH}(\boldsymbol{\pi}\ (\mathbf{I} - \mathbf{S}_{11})^{-1}\ \mathbf{S}_{12}, \mathbf{S}_{22})$. For more details on this class of bivariate DPH distributions, we refer the reader to Bladt & Yslas (2023c). Although the above construction principle can be extended to higher dimensions, the implementation of the methods associated with this class becomes more challenging. Fortunately, another subclass of MDPH* distributions introduced in Bladt & Yslas (2023c) can easily be tracked and implemented in higher dimensions, as we describe next.

mDPH class

Let $(Z_n^{(j)})_{n\in\mathbb{N}_0}$, $j = 1, \ldots, d$, be MCs on a common state space with p transient states. All chains are assumed to start in the same state and then evolve independently until respective absorptions. Formally,

$$Z_0^{(j)} = Z_0^{(l)}\,, \quad (Z_n^{(j)})_{n\in\mathbb{N}_0} \perp\!\!\!\perp_{Z_0^{(1)}} (Z_n^{(l)})_{n\in\mathbb{N}_0,\, l\neq j}\,, \quad \forall j, l \in \{1, \ldots, d\}\,.$$

https://doi.org/10.1017/S1748499525100134 Published online by Cambridge University Press

If $N^{(j)}, j = 1, \ldots, d$, are the univariate DPH-distributed absorption times of $(Z_n^{(j)})_{n\in\mathbb{N}_0}$, we say that the vector $\mathbf{N} = (N^{(1)}, \ldots, N^{(d)})$ has a multivariate DPH distribution of the mDPH type, and we write

$$\mathbf{N} \sim \text{mDPH}(\boldsymbol{\alpha}, \widetilde{S}),$$

where $\boldsymbol{\alpha}$ is the common vector of initial probabilities and $\widetilde{S} = \{\mathbf{S}_1, \ldots, \mathbf{S}_d\}$, with $\mathbf{S}_j$ the sub-transition matrix associated with $(Z_n^{(j)})_{n\in\mathbb{N}_0}$, $j = 1, \ldots, d$. Explicit expressions for several functionals of interest of this class can easily be obtained by conditioning on the starting value of the MCs. For instance, the joint density function of $\mathbf{N} \sim \text{mDPH}(\boldsymbol{\alpha}, \widetilde{S})$ is given by

$$f_{\mathbf{N}}(\mathbf{n}) = \sum_{k=1}^{p} \alpha_k \prod_{j=1}^{d} \mathbf{e}_k^\top \mathbf{S}_j^{n^{(j)}-1} \mathbf{s}_j, \quad \mathbf{n} \in \mathbb{N}^d.$$

3.2 Multivariate continuous phase-type distributions

We now present the MPH*, fMPH, and mPH classes of distributions, which follow similar construction principles as their discrete counterparts.

Let $\tau \sim \text{PH}_p(\boldsymbol{\alpha}, \mathbf{S})$ be a PH-distributed random variable, $\mathbf{r}_j = (r_j(1), \ldots, r_j(p))^\top$ be nonnegative column vectors of dimension p, $j = 1, \ldots, d$, and $\mathbf{R} = (\mathbf{r}_1, \ldots, \mathbf{r}_d)$ be a $p \times d$ reward matrix. Vectors $\mathbf{r}_j$ represent rates at which a reward is accumulated when $(J_t)_{t\geq 0}$ is in a specific state. Thus, we denote by

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

the total reward earned prior to absorption by each component j, $j = 1, \ldots, d$. Then, we say that the random vector $\mathbf{Y} = (Y^{(1)}, \ldots, Y^{(d)})$ has a multivariate PH distribution of the MPH* type, and we write $\mathbf{Y} \sim \text{MPH}^*(\boldsymbol{\alpha}, \mathbf{S}, \mathbf{R})$ (see Kulkarni, 1989; Bladt & Nielsen, 2017; Albrecher *et al.*, 2022c for more details). One attractive characteristic of this class of multivariate distributions is that, given a nonnegative vector $\mathbf{w} = (w_1, \ldots, w_d)$, $< \mathbf{Y}, \mathbf{w} >= \sum_{j=1}^{d} w_j Y^{(j)}$ is PH-distributed, with representation $(\boldsymbol{\alpha}_\mathbf{w}, \mathbf{S}_\mathbf{w})$ say. To obtain the exact parametrization, we split the state space $\{1, \ldots, p\} = E_+ \cup E_0$, such that $k \in E_+$ if $(\mathbf{R}\mathbf{w}^\top)_k > 0$ and $k \in E_0$ if $(\mathbf{R}\mathbf{w}^\top)_k = 0$. Then, by a reordering of the states, we can assume that $\boldsymbol{\alpha}$ and $\mathbf{S}$ can be written as:

$$\boldsymbol{\alpha} = \begin{pmatrix} \boldsymbol{\alpha}^+, & \boldsymbol{\alpha}^0 \end{pmatrix} \quad \text{and} \quad \mathbf{S} = \begin{pmatrix} \mathbf{S}^{++} & \mathbf{S}^{+0} \\ \mathbf{S}^{0+} & \mathbf{S}^{00} \end{pmatrix},$$

where the $+$ terms correspond to states in E_+, and the 0 terms to those states in E_0. For instance, $\mathbf{S}^{0+}$ collects transition intensities by which $(J_t)_{t\geq 0}$ jumps from states in E_0 to states in E_+. After this arrangement, we can express the distribution of the continuous part of $< \mathbf{Y}, \mathbf{w} >$ via a PH representation of the form:

$$\boldsymbol{\alpha}_\mathbf{w} = \boldsymbol{\alpha}^+ + \boldsymbol{\alpha}^0 \left(-\mathbf{S}^{00}\right)^{-1} \mathbf{S}^{0+} \quad \text{and} \quad \mathbf{S}_\mathbf{w} = \boldsymbol{\Delta}((\mathbf{R}\mathbf{w}^\top)_+)^{-1} \left(\mathbf{S}^{++} + \mathbf{S}^{+0} \left(-\mathbf{S}^{00}\right)^{-1} \mathbf{S}^{0+}\right).$$

Here, $\boldsymbol{\Delta}((\mathbf{R}\mathbf{w}^\top)_+)$ is a diagonal matrix with entries the vector $(\mathbf{R}\mathbf{w}^\top)_+$ formed of the appropriate ordered values satisfying $(\mathbf{R}\mathbf{w}^\top)_k > 0$. Note that an atom at zero of size $\boldsymbol{\alpha}^0 \left(\mathbf{I} - \left(-\mathbf{S}^{00}\right)^{-1} \mathbf{S}^{0+}\right) \mathbf{e}$ may appear since it is possible that $(J_t)_{t\geq 0}$ starts in a state in E_0 and gets absorbed before reaching a state in E_+. In particular, the above result implies that if $\mathbf{Y} \sim \text{MPH}^*(\boldsymbol{\alpha}, \mathbf{S}, \mathbf{R})$, then its marginals $Y^{(j)}$ are PH-distributed with parameters easily computed with the aforementioned formulas. Moreover, the so-called transformation via rewards for univariate PH distributions is retrieved when $d = 1$ above.

https://doi.org/10.1017/S1748499525100134 Published online by Cambridge University Press

Although members of the MPH* class possess other desirable characteristics, such as explicit expressions for joint Laplace transform and joint moments (see Section 8.1.1 of Bladt & Nielsen, 2017), general closed-form expressions for both joint density and joint distribution functions do not exist. Similar to its discrete counterpart, the MDPH* class, this complicates their use in practice and, more importantly, their estimation. This motivates the introduction of other subclasses of MPH* distributions for which explicit expressions can be achieved. The `matrixdist` package provides implementations for the fMPH and mPH subclasses, which are derived similarly to their discrete equivalents.

The fMPH class is obtained by considering MPH* parametrizations of the form MPH*($\boldsymbol{\alpha}, \mathbf{S}, \mathbf{R}$) with

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

where $\mathbf{S}_{11}$ and $\mathbf{S}_{22}$ are sub-intensity matrices of dimensions p_1 and p_2 ($p_1 + p_2 = p$), respectively, $\mathbf{S}_{12}$ is a $p_1 \times p_2$ matrix satisfying $\mathbf{S}_{11}\mathbf{e} + \mathbf{S}_{12}\mathbf{e} = \mathbf{0}$, and $\boldsymbol{\pi}$ is a p_1-dimensional vector of initial probabilities. In such a case, explicit expressions for different functionals can be obtained. For example, the joint density of a random vector $\mathbf{Y} = (Y^{(1)}, Y^{(2)})$ with the above parametrization is given by

$$f_{\mathbf{Y}}(y^{(1)}, y^{(2)}) = \boldsymbol{\alpha} \exp\left(\mathbf{S}_{11} y^{(1)}\right) \mathbf{S}_{12} \exp\left(\mathbf{S}_{22} y^{(2)}\right) (-\mathbf{S}_{22})\, \mathbf{e}\,,$$

For more details on this class of bivariate PH distribution and an extension to the IPH framework, we refer to Albrecher *et al.* (2022).

The tractable class of mIPH distributions (with a particular instance, the mPH class) was introduced in Bladt (2023) by considering d time-inhomogeneous Markov pure jump processes on a common state space with p transient states that start in the same state and then evolve independently until respective absorptions. If $Y^{(j)}$, $j = 1, \ldots, d$, are the univariate IPH-distributed absorption times of these Markov jump processes, we say that the vector $\mathbf{Y} = (Y^{(1)}, \ldots, Y^{(d)})$ has a multivariate IPH distribution of the mIPH type, and we write

$$\mathbf{Y} \sim \text{mIPH}(\boldsymbol{\alpha}, \widetilde{S}, \widetilde{L}), \quad \text{where} \quad \widetilde{S} = \{\mathbf{S}_1, \ldots, \mathbf{S}_d\} \quad \text{and} \quad \widetilde{L} = \{\lambda_1, \ldots, \lambda_d\}.$$

This construction yields explicit expressions for important functionals. For instance, let $\mathbf{Y} \sim$ mIPH($\boldsymbol{\alpha}, \widetilde{S}, \widetilde{L}$), then we have

$$f_{\mathbf{Y}}(\mathbf{y}) = \sum_{k=1}^{p} \alpha_k \prod_{j=1}^{d} \mathbf{e}_k^{\top} \exp\left(\mathbf{S}_j g_j^{-1}(y^{(j)})\right) \mathbf{s}_j \lambda_j(y^{(j)}), \quad \mathbf{y} \in \mathbb{R}_+^d\,.$$

Note that this allows for combining different types of IPH marginals, given by $\lambda_i(\cdot)$ and $g_i(\cdot)$, which enables different tail behaviors for the marginals.

3.3 Regression

For some of the multivariate classes introduced earlier, regression in the vector of initial probabilities can be applied by utilizing the same principle as in the univariate MoE approach. This allows us to capture the relationships between the multivariate responses and covariates, while taking advantage of the flexibility and expressiveness of the multivariate matrix distributions. Specifically, the mIPH-MoE model was introduced in Albrecher *et al.* (2023b) and used to estimate the joint remaining lifetimes of spouses. Additionally, the mDPH-MoE and fMDPH-MoE specifications were derived in Bladt & Yslas (2023c) and illustrated for modeling insurance frequencies. The construction principle behind these models is similar to that of the univariate DPH-MoE approach presented in Section 2.1 and is therefore omitted here for conciseness. We refer the interested reader to the aforementioned references for further details.

https://doi.org/10.1017/S1748499525100134 Published online by Cambridge University Press

4. Note on estimation algorithms and computational remarks

Most of the distribution classes presented above can be estimated using various forms of EM algorithms. This framework arises naturally when dealing with PH distributions, as we do not observe the trajectory of the underlying Markov jump processes. Instead, we only observe the time to absorption, with no information about the actual path taken. The PH setting is therefore an incomplete-data problem, a context in which the EM algorithm is known to perform particularly well. The original EM algorithm for PH distribution was introduced in Asmussen *et al.* (1996) and later extended for censored data in Olsson (1996). The corresponding algorithm for DPH distributions can be found in Bladt & Nielsen (2017). More recently, Albrecher *et al.* (2022c) derived an EM algorithm for IPH distributions and illustrated how the implementation of the algorithms for MPH* and fMPH distributions can be carried out smoothly. Regarding the regression models, the estimation method for the PI model is presented in Albrecher *et al.* (2022a), while Bladt & Yslas (2023b) provides the algorithm for the estimation of the PH-MoE specification. This method was then adapted in Albrecher *et al.* (2023a) for the mIPH class and in Bladt & Yslas (2023c) for the DPH framework, which includes extensions to the multivariate classes mDPH and fMDPH.

All estimation procedures implemented in the `matrixdist` package related to PH distributions require an effective method for computing matrix-exponential operations, as the various E-steps involved hinge on these types of quantities. Several theoretical methods exist for these purposes (see for instance Moler & Van Loan (1978)), but an efficient numerical implementation is crucial due to the significant computational burden of the algorithms. The `matrixdist` package offers three primary implementations of matrix exponentials: the first, described in Asmussen *et al.* (1996), involves converting the problem into a system of ordinary differential equations (ODE). The second method utilizes the so-called uniformization, with the exact description available in Albrecher *et al.* (2022c). The third and final method is the Padé approximation, which can be found in Moler & Van Loan (1978). Each method presents its own advantages and disadvantages. More specifically, the method based on ODE performs effectively in most cases, though it can face challenges with datasets containing extremely large data points or high-dimensional matrices. Additionally, this estimation approach might encounter difficulties with specific regression models, for example, the PI model, because it requires data to be ordered. On the other hand, the uniformization approach is typically faster and efficient, with its precision and speed being influenced by the selected truncation error threshold. Lastly, the Padé approximation maintains numerical stability across a broad spectrum of inputs; it tends to be slower than the other two methods and can thus be considered a benchmark. Therefore, the choice of estimation procedure depends on the specific problem/distribution at hand.

The EM algorithm for PH distributions also has a useful structural property: it preserves zeros in the parameter $\boldsymbol{\alpha}$ and S. This means that when an element has been estimated to zero, it remains zero in subsequent iterations. Thus, one can impose and maintain a desired structure during estimation by specifying an appropriate initialization. This can help not only to reduce the number of estimated parameters but also to speed up convergence, as pointed out in Asmussen *et al.* (1996). The `matrixdist` package includes random initialization for the following structures:

- *General PH* (`general`). No restrictions, all elements are allowed to be nonzero.
- *Hyperexponential* (`hyperexponential`). Also known as a mixture of exponentials. Here, the Markov process is allowed to start in any of the transient states but gets absorbed without visiting any other state. This means that the matrix S is nonzero only on the diagonal.
- *Generalized Erlang* (`gerlang`). Also known as sum of exponentials. Here, the Markov process starts in state 1 with probability one, that is, $\boldsymbol{\alpha} = (1, 0, \ldots, 0)$ and then visits each subsequent state $2, \ldots, p$, in that order, and finally reaches the absorbing state.
- *Coxian* (`coxian`). Same structure as the generalized Erlang, but the Markov process is allowed to reach absorption from any of the transient states.

https://doi.org/10.1017/S1748499525100134 Published online by Cambridge University Press

- *Generalized Coxian* (`gcoxian`). Same structure as the Coxian, but the Markov process is allowed to start in any state.

Note that in the above, we have included the names of these structures as they are defined and accessible within the `matrixdist` package.

5. Applications

In this section, we illustrate the use of the `matrixdist` package in three actuarial applications of significance. The first example (Section 5.1) addresses the univariate modeling of loss data exhibiting heavy tails. The second example (Section 5.2) deals with the joint modeling of absolute log-returns through MPH* distributions. The final example (Section 5.3) focuses on a multiline insurance claim dataset, addressing both the marginal and joint modeling of claim frequency and severity across two distinct lines of insurance coverage.

The model selection was done using the incremental approach commonly employed in matrix distributions modeling. Typically, this involves beginning with models of low dimension and sparse parameter structures and then incrementally exploring more complex models by increasing the matrix dimension or relaxing structural constraints. The trade-off of adding complexity is assessed by, for example, comparing loglikelihood values and/or using visual diagnostics such as QQ plots. This process continues until a satisfactory model is identified. It is crucial to test multiple initializations of the parameters, as the EM algorithm is sensitive to starting values and may converge to a local maxima or a saddle point, as noted in Asmussen *et al.* (1996). In the multivariate case, in addition to the margin assessment tools, one can resort to contour plots. Distributional distance tests (such as Kolmogorov–Smirnov) can also add concrete statistical evidence, in both univariate and multivariate settings.

5.1 Automobile bodily injury claims

We consider the `AutoBi` dataset from the `insuranceData` package, which contains information about automobile bodily injury claims. The dataset comprises eight variables: one for the economic loss (which we aim to describe), six covariates, and one variable to identify each claim. We start by providing general descriptive statistics for the variable of interest (`LOSS`).

```
# Load the data
data("AutoBi", package = "insuranceData")
claim <- AutoBi
# Economic losses resulting from an automobile accident
loss <- claim$LOSS
# Summary statistics of losses
summary(loss)

#>     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
#>    0.005    0.640    2.331    5.954    3.995 1067.697

# 10 largest values of losses
tail(sort(loss), n = 10)

#>  [1]   82.000   96.007  114.604  150.000  162.047  188.720  193.000
#>  [8]  222.405  273.604  1067.697
```

https://doi.org/10.1017/S1748499525100134 Published online by Cambridge University Press

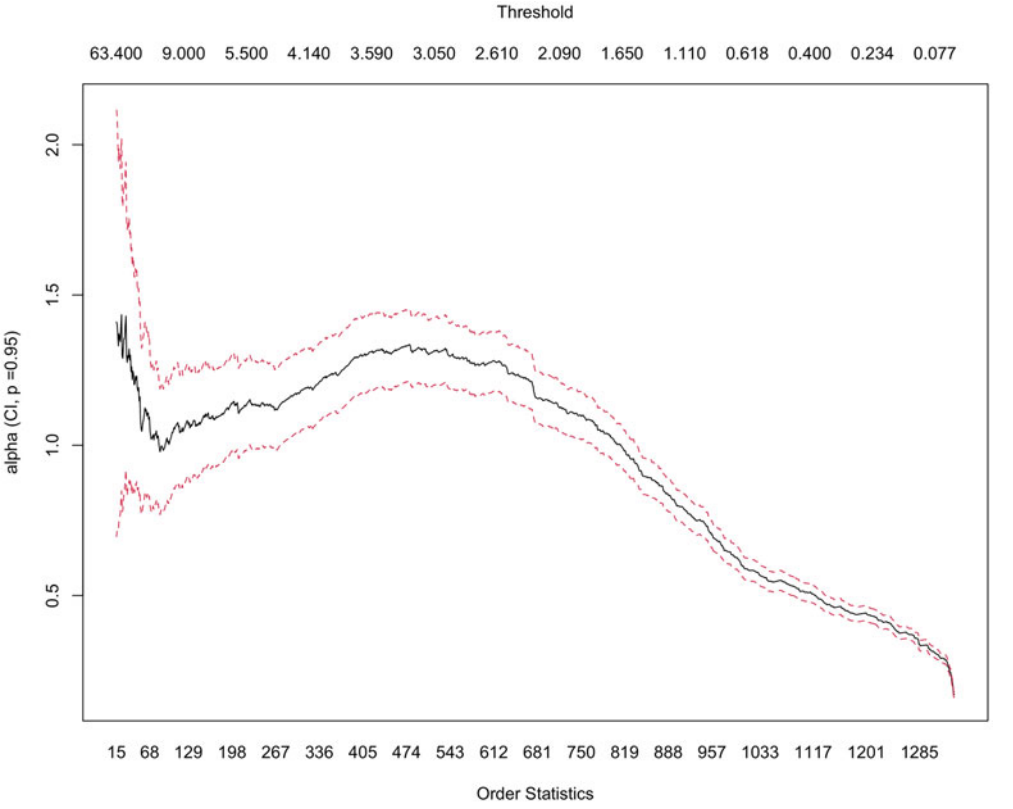

Figure 1. Hill plot for losses in the AutoBi dataset.

We observe that the losses range from 0.005 to 1067.697, with 75% of the data smaller than 3.995. The presence of large values, compared to the body of the distribution, serves as an initial indication of a heavy tail. To further confirm this observation, we create a Hill plot using the `hill()` function from the `evir` package.

```
library(evir)
hill(loss)
```

The plot (Figure 1) exhibits a flat behavior for the largest order statistics, suggesting a Pareto-type tail (or regularly varying). Therefore, we opt for an IPH model to more accurately capture the tail of the losses. More specifically, we use a matrix-Pareto distribution. The fitting process starts by creating an initial IPH distribution, which provides the starting parameters for the EM algorithm. This can be accomplished by employing the `iph()` method in `matrixdist`. For our analysis, we create an IPH object with random parameters of a general structure, a dimension 6, and a Pareto inhomogeneity function.

```
# Initial IPH object
set.seed(22)
x <- iph(gfun = "pareto", structure = "general", dimension = 6)
```

https://doi.org/10.1017/S1748499525100134 Published online by Cambridge University Press

With an initial IPH distribution in place, we can employ the `fit()` method to fit the data. It is important to note that `fit()` requires specifying the number of steps for the EM algorithm. In this instance, we use 1500 steps, as the changes in the loglikelihood become negligible with this number (the same criteria is employed to determine the number of iterations in the subsequent illustration). The method outputs an IPH object containing the fitted parameters. Further steps could be made on the output object if required.

```
# Fitting procedure
z <- fit(x, y = loss, stepsEM = 1500)
```

To visually asses the quality of the fit, we create a QQ plot for the fitted distribution and a histogram of the (log) data against the fitted density:

```
# Probabilities for the for the quantiles in the QQ plot
qq <- seq(0.001, 0.99, length.out = 100)
# Sample quantiles of losses
loss.q <- quantile(loss, probs = qq)
# Quantiles of the fitted IPH distribution
z.q <- quan(z, qq)
# QQ plot
plot(loss.q, z.q,
  main = "QQ plot",
  xlab = "Sample", ylab = "Fitted", pch = 16
)
abline(0, 1, col = "blue")

# Evaluation points for fitted density
q <- log(seq(min(loss), max(loss), by = 0.01))
# Histogram and fitted density
hist(log(loss),
  breaks = 50, freq = F,
  main = "Histogram of the logarithm of losses"
)
lines(q, dens(z, exp(q)) * exp(q),
  col = "blue", lty = 1, lwd = 1.25
)
legend("topright",
  legend = "IPH",
  col = "blue", lty = 1, lwd = 1.25, bty = "n"
)
```

Figures 2 and 3 display these plots, which indicate that the fitted distribution provides a good approximation of the data. In particular, the sample and fitted quantiles align closely with the identity line, especially within the distribution's body. Although there is a slight deviation from the identity line for quantiles in the tail, they remain relatively close to the blue line. The histogram further supports the notion that the fitted IPH distribution is a satisfactory approximation.

https://doi.org/10.1017/S1748499525100134 Published online by Cambridge University Press

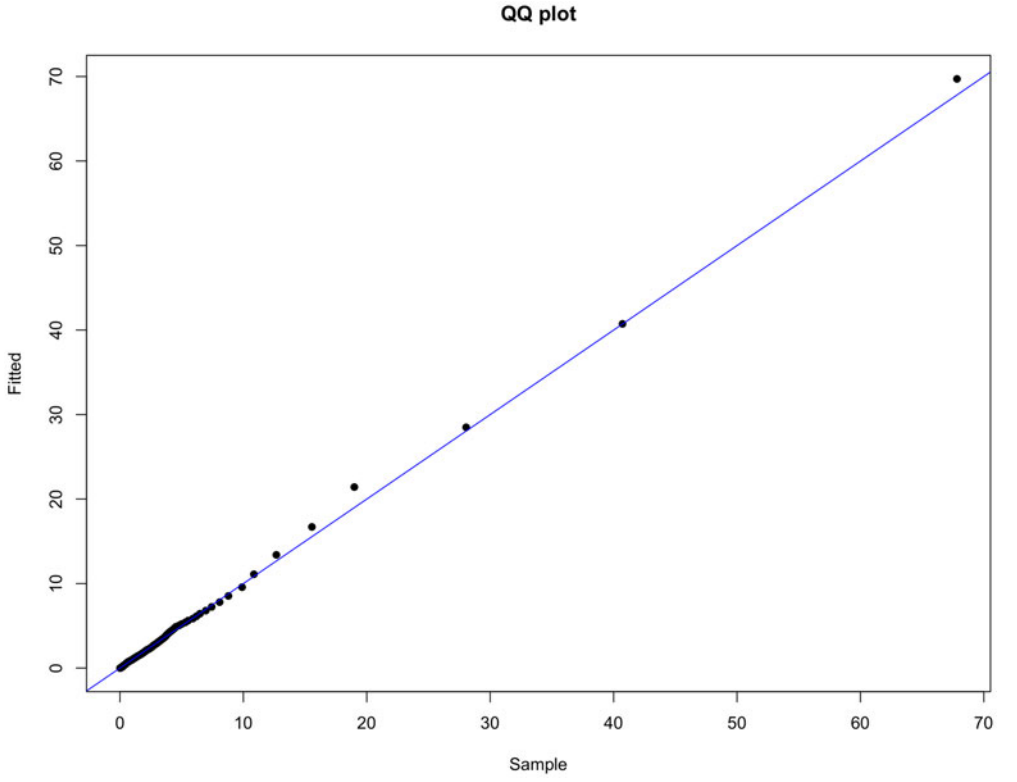

Figure 2. QQ plot of fitted distribution.

PH regression

We now shift our focus to the regression framework. In particular, we employ the PI specification to capture the influence of covariate information on economic losses. The initial step involves removing any incomplete covariate data. Next, we need to split the covariates and the response variable into two different objects. In our case, we investigate the effect of the variables `ATTORNEY`, `CLMSEX`, and `CLMAGE`. The variable `ATTORNEY` indicates whether the claimant was represented by an attorney or not, `CLMSEX` is the claimant's sex, and `CLMAGE` corresponds to their age.

```
# Covariate information, excluding missing entries
cov <- na.omit(claim)
# Separation of the information
X <- cov[, c("ATTORNEY", "CLMSEX", "CLMAGE")]
loss <- cov[, "LOSS"]
```

Now, we can employ the `reg()` method to perform the PI regression. This function works similarly to the `fit()` method, with the main difference being that estimation of the regression parameters is included in each iteration of the EM algorithm (see Albrecher *et al.*, 2022a for details). Along with the data, the `reg()` method requires an initial IPH distribution. In this case, we employ the previously fitted matrix-Pareto distribution without covariates.

https://doi.org/10.1017/S1748499525100134 Published online by Cambridge University Press

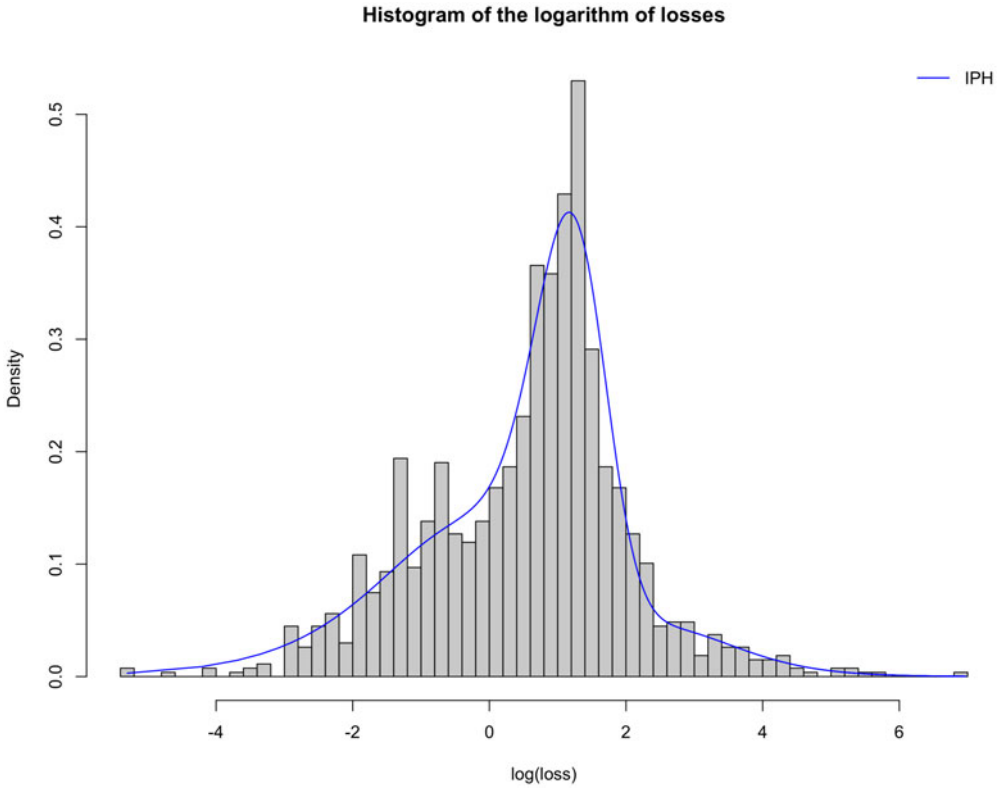

Figure 3. Histogram of the logarithm of losses versus fitted density (blue line).

```
# PI regression
z.reg <- reg(x = z, y = loss, X = X, stepsEM = 250)
```

The resulting object (`z.reg`) contains the parameters of the underlying IPH model, namely `alpha`, `S`, and `beta`, along with the regression coefficients `B`. This information can be easily accessed via the `coef()` function as follows.

```
# Fitted parameters
z.reg.par <- coef(z.reg)
z.reg.par

#> $alpha
#> [1] 2.356094e-04 3.257751e-06 5.920828e-02 3.755407e-04 4.561202e-01
#> [6] 4.840571e-01
#>
#> $S
```

https://doi.org/10.1017/S1748499525100134 Published online by Cambridge University Press

```
#>               [,1]          [,2]        [,3]          [,4]          [,5]
#> [1,] -3.412707e+01  7.592078e-06  1.42774265  1.533919e-01  6.975181e-06
#> [2,]  2.145191e+01 -3.473845e+01  0.38024154  1.288647e+01  3.162116e-05
#> [3,]  3.090083e-01  6.287658e-01 -1.83234510  4.643455e-01  3.670351e-01
#> [4,]  5.666353e-01  1.166288e-04  7.77223071 -2.987094e+01  2.335434e-04
#> [5,]  1.432727e-04  3.578715e+01  0.49172745  2.247409e-04 -3.628114e+01
#> [6,]  4.561651e-06  1.243609e-03  0.08311972  3.771681e-05  4.707313e-02
#>               [,6]
#> [1,]  3.037507e+01
#> [2,]  2.799554e-05
#> [3,]  3.352413e-02
#> [4,]  2.066459e+01
#> [5,]  1.118600e-03
#> [6,] -6.641175e+00
#>
#> $beta
#> [1] 33.10073
#>
#> $B
#>
#>  1.15139949 -0.10888748 -0.01368395
#>
#> $C
#> numeric(0)
```

The above information can be used, for example, to create customized loss distributions for individual claimants based on their distinctive characteristics. Take, for instance, a new claimant: a 60-year-old male, unrepresented by an attorney. The following code leverages our model to estimate his loss distribution.

```
# Covariate information of the new claimant
claimant <- data.frame(ATTORNEY = 2, CLMSEX = 1, CLMAGE = 60)
# Multiplicative effect in the sub-intensity matrix
prop <- exp(sum(z.reg.par$B * claimant))
# IPH distribution for claimant's loss
claimant.iph <- iph(
  alpha = z.reg.par$alpha, S = z.reg.par$S * prop,
  gfun = "pareto", gfun_pars = z.reg.par$beta
)
```

If we wish to visualize this claimant's survival function, we can plot it as follows.

```
# Evaluation points for the survival function
q <- seq(0, 15, 0.01)
# Plot of the survival function
plot(q, cdf(claimant.iph, q, lower.tail = F),
  type = "l", col = "blue", lwd = 1.3,
  ylab = "1-F(y)", xlab = "y", main = "Survival function"
)
```

https://doi.org/10.1017/S1748499525100134 Published online by Cambridge University Press

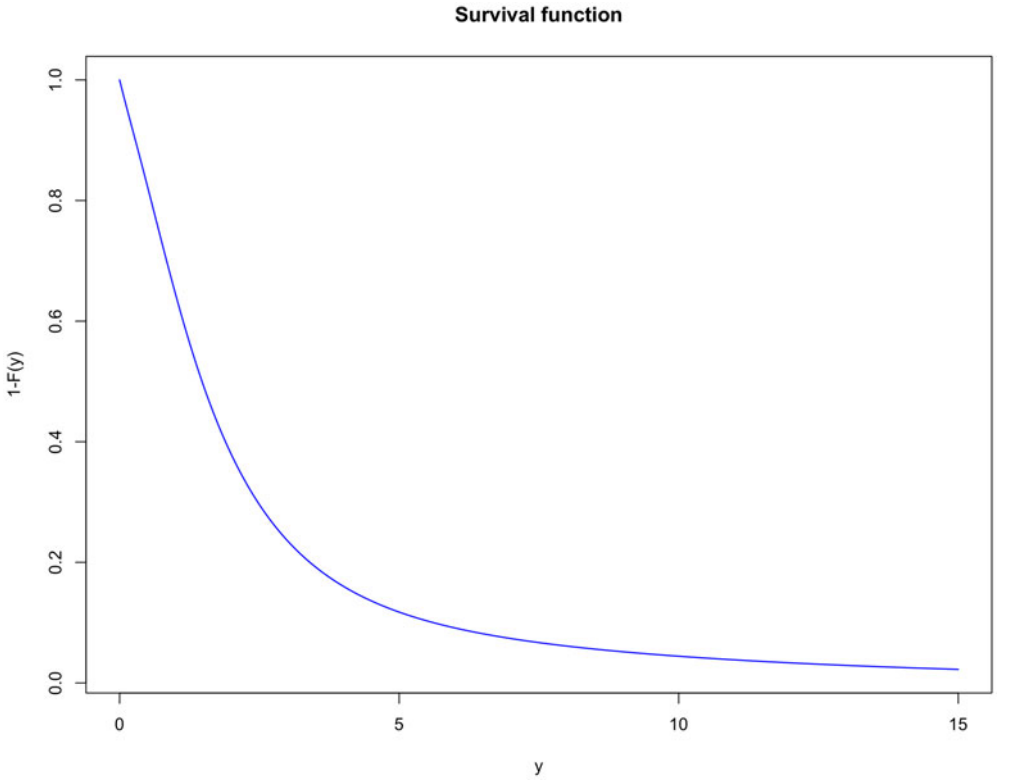

Figure 4. Survival function of losses for the new claimant.

Figure 4 shows the resulting survival function for the new claimant, providing a visual representation of their risk profile. We refer readers to Section 6 and the documentation of `matrixdist` for further available methods for PH and IPH distributions.

5.2 Multivariate modeling of log-returns

To illustrate the capabilities of the implementations for MPH* distributions in `matrixdist`, we use the `rdj` dataset of the `copula` package. More specifically, our focus is to use an MPH* for describing the joint behavior of the absolute log-returns of the three stock prices in this dataset: Intel, Microsoft, and General Electric.

First, we load the data, take the absolute value, and eliminate any data points with atoms at zero.

```
set.seed(543)
# Load the data
data("rdj", package = "copula")
# Absolute log return of stock prices
y <- as.matrix(abs(rdj[, -1]))
# Remove data points with atoms at zero
y <- y[y[, 1] != 0 & y[, 2] != 0 & y[, 3] != 0, ]
```

https://doi.org/10.1017/S1748499525100134 Published online by Cambridge University Press

As in the univariate case, the fitting process requires the definition of an initial `MPHstar` object that is subsequently fed into the `fit()` function. For this class of objects, the function is implemented solely using the uniformization method. As such, a truncation error must be specified. A custom value can be passed via the `uni_epsilon` argument; otherwise, it defaults to 10^{-4}. The method also includes the argument `r`, which can be used to speed up the fitting procedure by applying random subsampling of the data at each iteration. If not specified, no subsampling is applied. For the present analysis, we use an initial `MPHstar` object with randomly generated parameters, a general matrix structure, and dimension 8. We also modify the values of `uni_epsilon` and `r` in `fit()` to illustrate their use.

```
# Initial MPH* distribution
x <- MPHstar(structure = "general", dimension = 8, variables = 3)
# Fitting to data
x.fit <- fit(x = x, y = y, stepsEM = 2000, uni_epsilon = 1e-10, r = 0.85)
```

The parameters of the fitted model can be accessed using `x.fit@pars`, which includes the estimated parameters of the underlying PH model (`alpha` and `S`) as well as the estimated reward matrix (`R`). In particular, the reward matrix can be extracted as follows:

```
# Estimated reward matrix
x.fit@pars$R
```

```
#>              [,1]       [,2]       [,3]
#>  [1,] 0.499025362 0.44723336 0.05374128
#>  [2,] 0.000000000 0.06996296 0.93003704
#>  [3,] 0.592661246 0.01501042 0.39232833
#>  [4,] 0.000000000 1.00000000 0.00000000
#>  [5,] 1.000000000 0.00000000 0.00000000
#>  [6,] 1.000000000 0.00000000 0.00000000
#>  [7,] 0.006405003 0.98536321 0.00823179
#>  [8,] 0.000000000 0.16918651 0.83081349
```

Each row of the reward matrix corresponds to a transient state of the underlying Markov jump process, while each column represents one of the marginal components. The entries indicate the relative contribution of each state to the respective marginals. For example, in row 4, all the weight is placed on the second component, suggesting that this state is almost exclusively associated with that margin.

To assess the quality of the fit, we first examine the marginal behavior of the fitted model. The marginal distributions can be accessed using the `marginal()` function, which allows us to create QQ plots for a visual comparison between the empirical and fitted distributions.

```
# Names for the QQ plot
name <- colnames(y)
# Quantiles' levels
qq <- seq(0.01, 0.99, length.out = 100)
par(mfrow = c(1, 3))
```

https://doi.org/10.1017/S1748499525100134 Published online by Cambridge University Press

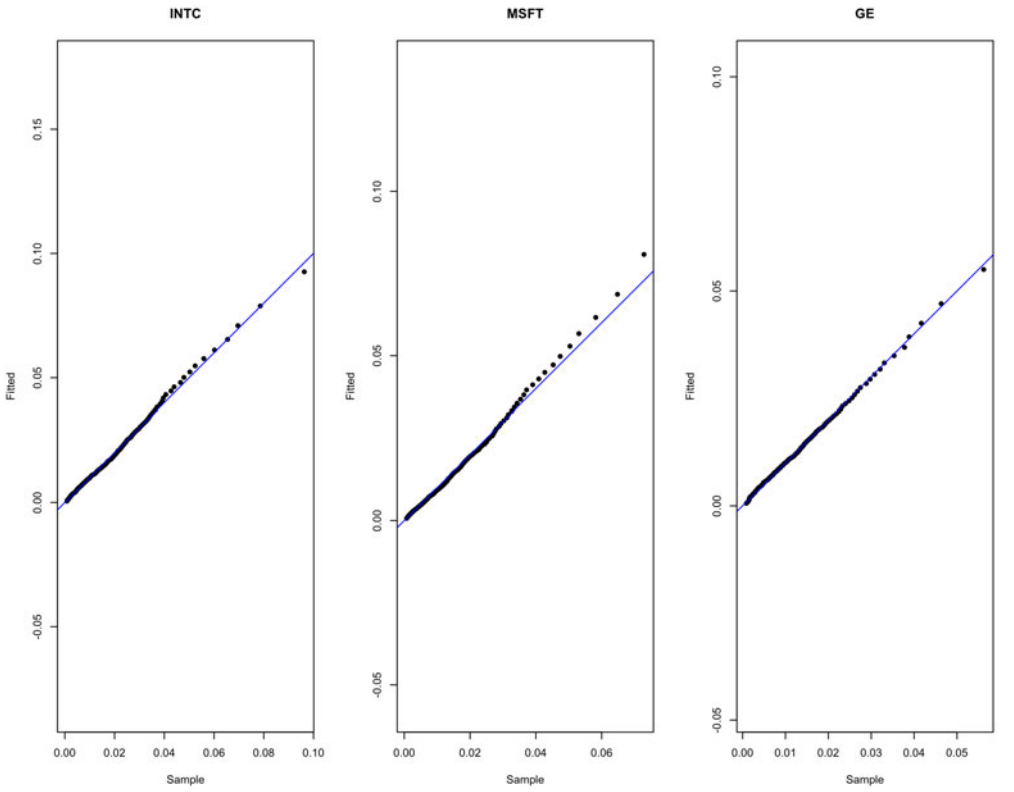

Figure 5. QQ plot of fitted distirbutions for each marginal.

```
for (j in 1:3) {
  # j-th marginal fitted distribution
  x <- marginal(x.fit, j)
  # Marginal observations
  ym <- y[, j]
  # quantiles
  y.q <- quantile(ym, probs = qq)
  x.q <- quan(x, qq)
  # QQ plot
  plot(y.q, x.q, main = name[j], xlab = "Sample",
    ylab = "Fitted", pch = 16, asp = 1
  )
  abline(0, 1, col = "blue")
}
```

Upon inspecting the QQ plots (Figure 5), we can observe that the fitted marginals provide a satisfactory approximation of the data. To assess the joint behavior, we simulate data from the fitted MPH* model and use kernel density estimation to produce contour plots, shown in Figure 6. As MPH* models do not possess a closed-form expression for their joint density, this simulation approach provides a practical alternative for visual inspection. The figure also includes the corresponding contour plot of the original data for comparison, where we observe similar behaviors

https://doi.org/10.1017/S1748499525100134 Published online by Cambridge University Press

in most regions. It is worth mentioning that although joint density evaluation via Laplace inversion is theoretically possible, it is computationally demanding and slow, making the simulation approach the most practical solution.

```
# Simulation from the fitted MPHstar object
set.seed(1)
sim_data <- sim(x.fit, 5000)

# Pairs of margins
margin_pairs <- list(c(1, 2), c(1, 3), c(2, 3))

# Plots for original data
par(mfrow = c(1, 3))
for (pair in margin_pairs) {
  margin1 <- pair[1]
  margin2 <- pair[2]

  # Create 2D density estimates for each pair
  dens.dat <- MASS::kde2d(y[, margin1], y[, margin2], n = 100)

  # Draw the contour plot
  contour(dens.dat,
    xlab = paste("Margin", margin1), ylab = paste("Margin", margin2),
    main = paste("Original data: Margin", margin1, "vs", margin2),
    xlim = c(0, 0.06), ylim = c(0, 0.04), col = "blue"
  )
}

# Plots for simulated data from fitted model
par(mfrow = c(1, 3))
for (pair in margin_pairs) {
  margin1 <- pair[1]
  margin2 <- pair[2]

  # Create 2D density estimates for each pair
  dens.fit <- MASS::kde2d(sim_data[, margin1], sim_data[, margin2], n = 100)

  # Draw the contour plots
  contour(dens.fit,
    xlab = paste("Margin", margin1), ylab = paste("Margin", margin2),
    main = paste("Fitted model: Margin", margin1, "vs", margin2),
    xlim = c(0, 0.06), ylim = c(0, 0.04), col = "blue"
  )
}
```

Note that contour plots behave well on a global scale but can be volatile at the tail regions due to a lack of data. Consequently, one can additionally resort to statistical tests to determine whether the original data has the same distribution as the simulated data. For example, here we use a Cramér two-sample test introduced in Baringhaus & Franz (2004) and as implemented in the `cramer` R package (Franz, 2024), which leads to the null hypothesis that both datasets possess the same distribution cannot be rejected.

https://doi.org/10.1017/S1748499525100134 Published online by Cambridge University Press

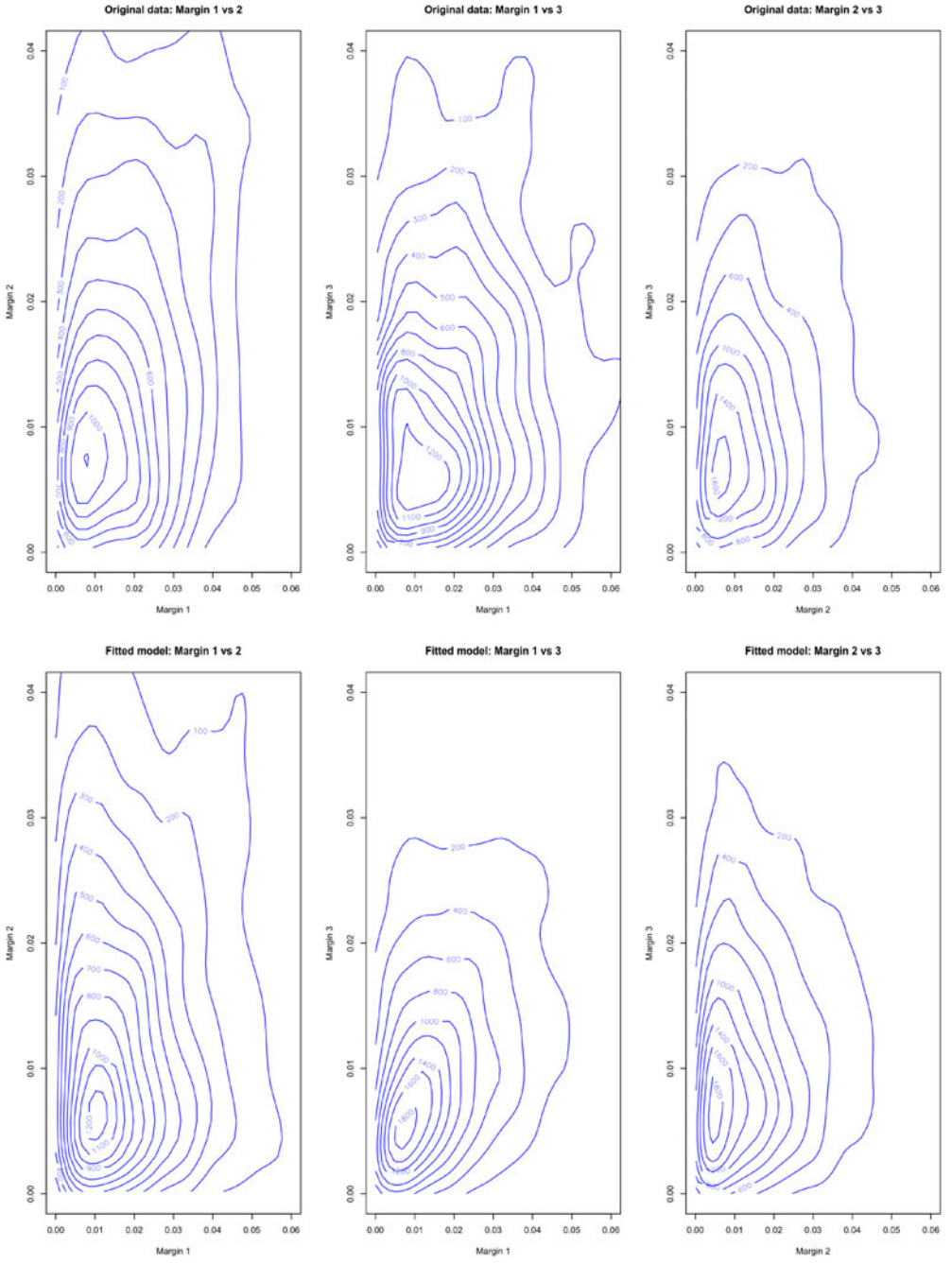

Figure 6. Contour plots of the kernel density estimates for each combination of the margins of the original data (top) and of a sample simulated from the fitted MPH* model (bottom).

https://doi.org/10.1017/S1748499525100134 Published online by Cambridge University Press

```
# Cramér-test for two-sample-problem
library(cramer)
cramer.test(y, sim_data)

#>  3 -dimensional  nonparametric Cramer-Test with kernel phiCramer
#> (on equality of two distributions)
#>
#>  x-sample:  1202  values        y-sample:  5000  values
#>
#> critical value for confidence level  95 % :  0.03169158
#> observed statistic  0.02039719 , so that
#>   hypothesis ("x is distributed as y") is  ACCEPTED .
#> estimated p-value =  0.2407592
#>
#>  [result based on  1000   ordinary  bootstrap-replicates]
```

5.3 Wisconsin property insurance dataset

We now consider the Wisconsin Local Government Property Insurance Fund (LGPIF) dataset (see https://sites.google.com/a/wisc.edu/local-government-property-insurance-fund). This fund was established to provide insurance coverage to government properties not owned by the State of Wisconsin. Governmental entities under this fund include counties, cities, towns, schools, and other miscellaneous entities, and properties covered encompass buildings, vehicles, and equipment. The dataset is divided into a training set covering the years 2006–2010 and a testing set encompassing the year 2011. It includes data on claim frequencies and severities across six different coverage groups: building and content (BC), contractor's equipment (IM), comprehensive new (PN), comprehensive old (PO), collision new (CN), and collision old (CO) coverage.

This dataset, which is publicly available at https://sites.google.com/a/wisc.edu/jed-frees/home, was thoroughly studied in Frees *et al.* (2016), with the corresponding code used for the analysis also accessible at the same URL. Here, we focus on the univariate and multivariate modeling of frequency and severity for two coverages – BC and CO – using matrix models and compare our results to those obtained in the aforementioned study. Note that to reproduce the following analysis, the `setup.R` script available on the above-mentioned site must be run first to access the necessary data objects.

Severity modeling

We start our analysis with the univariate modeling of the average claim size of BC, which we scale with a factor of 10^{-4} to speed up the convergence of the fitting algorithms.

```
sevinBC$yAvgBC <- sevinBC$yAvgBC * 1e-4
```

Regarding explanatory variables, we choose the following as in Frees *et al.* (2016): coverage amount (in log-scale), an indicator for no claims in the previous year, entity type (City, County, Misc, School, Town), and deductible level (in log-scale). We then fit a PH-MoE model of dimension 5 with general Coxian structure of the sub-intensity matrix. For this, we first specify the regression formula and an initial IPH model.

https://doi.org/10.1017/S1748499525100134 Published online by Cambridge University Press

```
# Regression formula
formula <- yAvgBC ~ CoverageBC + lnDeductBC + NoClaimCreditBC + TypeCity
+TypeCounty + TypeMisc + TypeSchool + TypeTown
# Initial IPH object
set.seed(5054)
x <- iph(ph(structure = "gcoxian", dimension = 5),
  gfun = "lognormal", gfun_pars = 1
)
```

These two objects are then passed into the `MoE()` function to obtain the fitted PH-MoE model.

```
# MoE regression
phMoE_BC <- MoE(x = x, formula = formula, data = sevinBC, stepsEM = 1000)
```

Note that we have selected a matrix-lognormal model, as it provided the largest value of the loglikelihood $(-1,836.67)$ compared to other IPH specifications of the same dimension. In fact, it outperforms the Generalized Beta type 2 (GB2) model in Frees *et al.* (2016), which has corresponding loglikelihood of $-1,973.30$. The goodness-of-fit is then checked visually using a QQ plot of the quantiles of normal Cox–Snell residuals, which are compared with normal quantiles (see Figure 7).

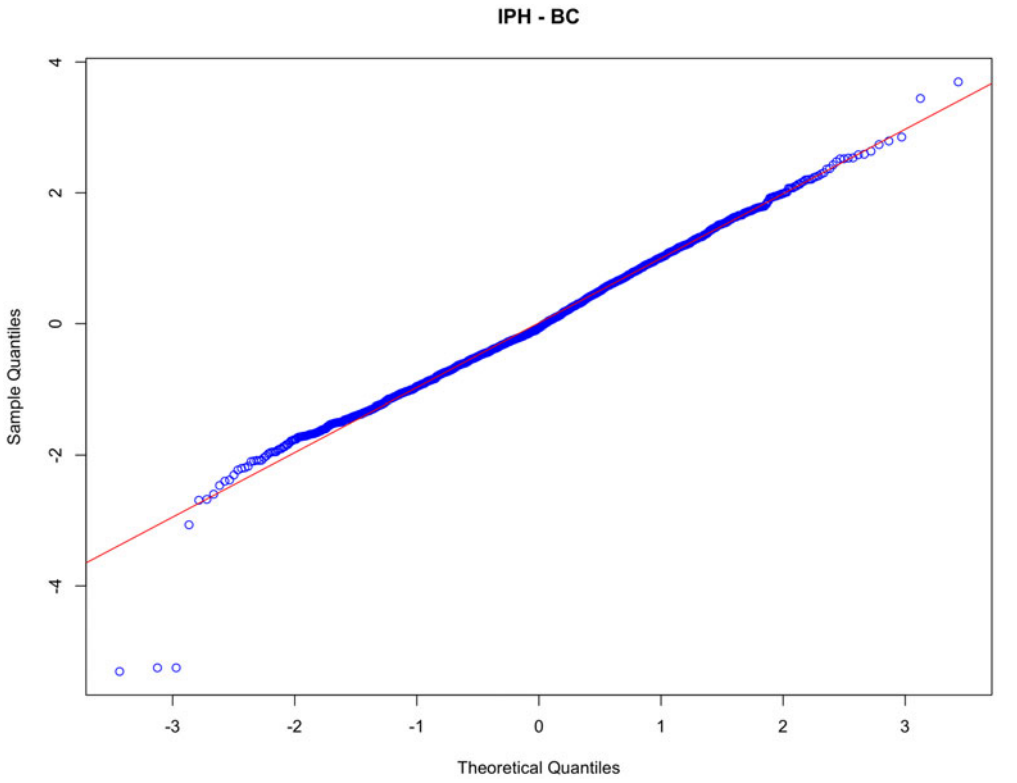

Figure 7. QQ plot for residuals of IPH for BC.

https://doi.org/10.1017/S1748499525100134 Published online by Cambridge University Press

```
# Cox-Snell residuals
cs <- rep(0, length(sevinBC$yAvgBC))
for (k in 1:length(sevinBC$yAvgBC)) {
  cs[k] <- cdf(iph(ph(phMoE_BC$alpha[k, ], phMoE_BC$S),
    gfun = "lognormal", gfun_pars = phMoE_BC$inhom$pars
  ), sevinBC$yAvgBC[k])
}
# QQ-plot of residuals
qqnorm(qnorm(cs), col = "blue", main = "IPH - BC")
qqline(qnorm(cs), col = "red")
```

Moreover, to assess the predictive performance of the fitted model, we use the testing set and compared the mean predictions of the PH-MoE and GB2 models with the observed average severity via the root-mean-square error (RMSE) measure.

```
# Testing set
sevoutBC <- dataout[BCpout, ]
nBC <- length(BCpout)
# Prediction for vector of initial probabilities
alpha_vecs <- stats::predict(phMoE_BC$mm, type = "probs", newdata = sevoutBC)
# Mean values
mean_val <- c()
for (j in 1:nBC) {
  iph_temp <- iph(ph(alpha_vecs[j, ], phMoE_BC$S),
    gfun = "lognormal", gfun_pars = phMoE_BC$inhom$pars
  )
  mean_val[j] <- integrate(function(x) x * dens(iph_temp, x), 0, Inf)$value
}
# RMSE
sqrt(sum((mean_val - sevoutBC$yAvgBC * 1e-04)^2) / nBC)
```

We obtain an RMSE of 7.8305 for the PH-MoE model and 29.8380 for the GB2 one, showcasing the superior predictive performance of the matrix model.

Subsequently, we perform a similar analysis for the CO coverage. In this case, we only consider coverage amount (in log-scale) and the indicator for no claims in the previous year as covariates to align the results with the study in Frees *et al*. (2016). Moreover, we have also selected a matrix-lognormal model of dimension 4 and general Coxian structure of the sub-intensity matrix.

```
# Scaling of average claim sizes
sevinCO$yAvgCO <- sevinCO$yAvgCO * 1e-04
# Regression formula
formula <- yAvgCO ~ CoverageCO + NoClaimCreditCO
# Initial IPH object
set.seed(8919)
x <- iph(ph(structure = "gcoxian", dimension = 4),
  gfun = "lognormal", gfun_pars = 1
)
# MoE regression
phMoE_CO <- MoE(x = x, formula = formula, data = sevinCO, stepsEM = 1000)
```

https://doi.org/10.1017/S1748499525100134 Published online by Cambridge University Press

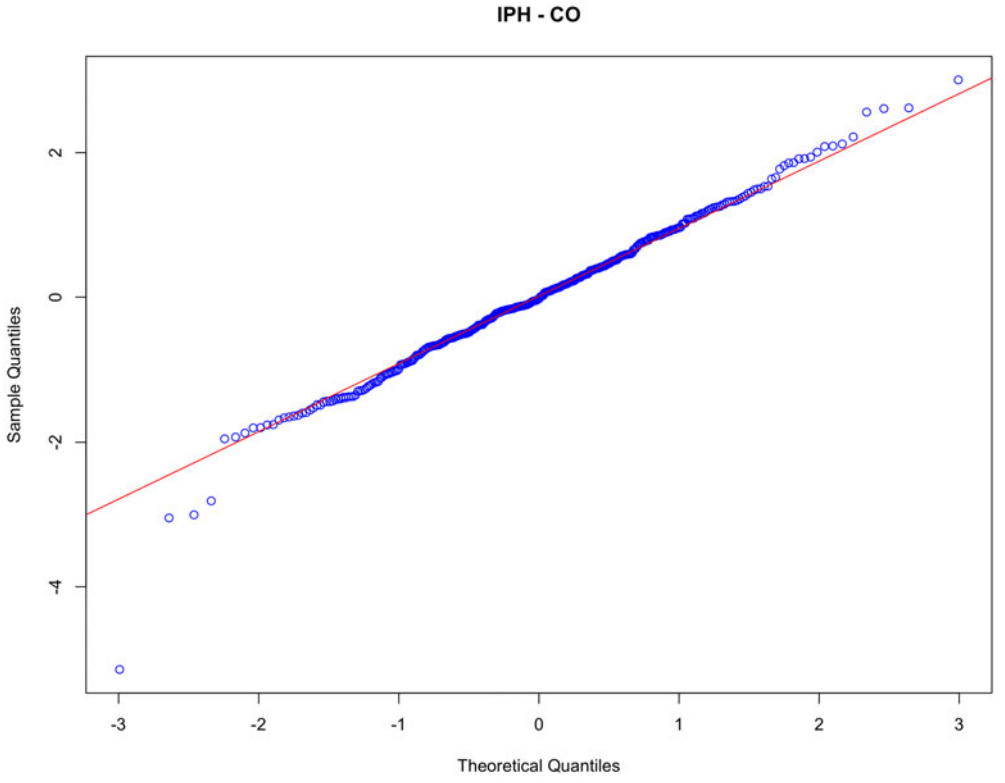

Figure 8. QQ plot for residuals of IPH for CO.

Once again, we have that the matrix model outperforms the GB2 model in terms of loglikelihood with −178.78 for the former and −189.38 for the latter. The QQ plot of the residuals of the PH-MoE model are shown in Figure 8.

```
# Cox-Snell residuals
cs <- rep(0, length(sevinCO$yAvgCO))
for (k in 1:length(sevinCO$yAvgCO)) {
  cs[k] <- cdf(iph(ph(phMoE_CO$alpha[k, ], phMoE_CO$S),
    gfun = inh_name, gfun_pars = phMoE_CO$inhom$pars
  ), sevinCO$yAvgCO[k])
}
# QQ-plot of residuals
qqnorm(qnorm(cs), col = "blue", main = "IPH - CO")
qqline(qnorm(cs), col = "red")
```

Regarding the predictive performance, the PH-MoE specification, with an RMSE of 1.1552, outperforms the GB2 model with an RMSE of 1.2277.

https://doi.org/10.1017/S1748499525100134 Published online by Cambridge University Press

```
# Testing set
sevoutCO <- dataout[COpout, ]
nCO <- length(COpout)
# Prediction for vector of initial probabilities
alpha_vecs <- stats::predict(phMoE_CO$mm, type = "probs", newdata = sevoutCO)
# Mean values
mean_val <- c()
for (j in 1:nCO) {
  iph_temp <- iph(ph(alpha_vecs[j, ], phMoE_CO$S),
    gfun = "lognormal", gfun_pars = phMoE_CO$inhom$pars
  )
  mean_val[j] <- integrate(function(x) x * dens(iph_temp, x), 0, Inf)$value
}
# RMSE
sqrt(sum((mean_val - sevoutCO$yAvgCO * 1e-04)^2) / nCO)
```

We continue the analysis of the severities by presenting a joint modeling of the BC and CO lines using an mIPH model with MoE regression. We start by considering the data with claims in both lines and giving the same scaling of 10^{-4} as before.

```
biv_ind <- which(data$ClaimBC > 0 & data$ClaimCO > 0 &
  data$CoverageCO > 0 & data$CoverageBC > 0)
data_biv <- data[biv_ind, ]

data_biv$yAvgCO <- data_biv$yAvgCO * 1e-04
data_biv$yAvgBC <- data_biv$yAvgBC * 1e-04
data_biv$CoverageBC <- log(data_biv$CoverageBC)
data_biv$CoverageCO <- log(data_biv$CoverageCO)

responses <- cbind(data_biv$yAvgBC, data_biv$yAvgCO)
```

We then specify the regression formula and an initial mIPH model. In this case, we have chosen matrix-lognormal margins of dimension 5 and general Coxian structure of their sub-intensity matrices, primarily based on the previous univariate analysis. The initial mIPH model is created using the `miph()` function, which requires as input an underlying mPH model constructed via `mph()`.

```
# Regression formula
formula_biv <- responses ~ CoverageCO + NoClaimCreditCO + CoverageBC
+lnDeductBC + NoClaimCreditBC + TypeCity + TypeCounty + TypeMisc + TypeSchool
+TypeTown
# Initial mIPH object
set.seed(741)
x1 <- mph(structure = c("gcoxian", "gcoxian"), dimension = 5)
x2 <- miph(x1, gfun = c("lognormal", "lognormal"), gfun_pars = list(c(1), c(1)))
```

https://doi.org/10.1017/S1748499525100134 Published online by Cambridge University Press

Table 2. Loglikelihood and RMSE for severity models

Model	*Loglikelihood*	*RMSE*
PH-MoE (BC)	−1, 836.67	7.8305
GB2 (BC)	−1, 973.30	29.8380
PH-MoE (CO)	−178.78	1.1552
GB2 (CO)	−189.38	1.2277
mPH-MoE (BC + CO)	−332.79	− − −
GB2 + Gaussian copula (BC + CO)	−341.33	− − −

Finally, we fit the regression model using the `MoE()` function.

```
# MoE regression
bivMoE <- MoE(
  x = x2,
  formula = formula_biv,
  y = responses,
  data = data_biv,
  stepsEM = 500
)
```

We note that the fitted model gives a value of the loglikelihood of −332.79, which outperforms the joint model in Frees *et al.* (2016), consisting of GB2 margins bounded together with a Gaussian copula, with a corresponding loglikelihood of −341.33.

To conclude the severity analysis, we present Table 2, which summarizes the performance of all models considered in terms of loglikelihood and RMSE.

Frequency modeling

As in the previous section, we begin our study with the univariate modeling of the frequency of BC. Since the support of DPH distributions is $\mathbb{N}$ and the dataset includes observations equal to zero, we apply a translation to the data to start at one.

```
# Translation
dataBC <- freqinBC
dataBC$FreqBC <- dataBC$FreqBC + 1
```

We now specify the regression formula, which includes the same covariates as the corresponding analysis in Frees *et al.* (2016), and an initial DPH object of dimension 7 with a general Coxian-like structure of the matrix of transition probabilities.

```
# Regression formula
formula <- FreqBC~ CoverageBC + lnDeductBC + NoClaimCreditBC + TypeCity
+TypeCounty + TypeMisc + TypeSchool + TypeTown
# Initial DPH object
set.seed(1212)
x <- dph(structure = "gcoxian", dimension = 7)
```

https://doi.org/10.1017/S1748499525100134 Published online by Cambridge University Press

Regression under the MoE framework can now be performed using the above two objects in the `MoE()` function.

```
# MoE regression
dphMoE_BC <- MoE(x = x, formula = formula, data = dataBC, stepsEM = 500)
```

The fitted model leads to a loglikelihood of −4,777.64, which is considerably higher than the corresponding values of the two models with the largest loglikelihoods in Frees *et al.* (2016); negative binomial (NB) with a value of −5,102.62 and zero-one-inflated negative binomial (zerooneNB) with −4,958.29. To further evaluate the goodness-of-fit, we calculate the chi-square statistic, which yields a value of 22.45 for the DPH-MoE model. This outperforms the NB and zerooneNB specifications with corresponding values of 88.09 and 34.51.

```
# Empirical count
numrow <- max(freqBC) + 1
emp <- rep(0, numrow)
for (i in 1:numrow) {
  emp[i] <- sum(freqBC == (i - 1))
}
# Expected frequency
eDPH <- 0
for (j in 1:nrow(dataBC)) {
  eDPH <- eDPH + dens(dph(dphMoE_BC$alpha[j, ], dphMoE_BC$S), 1:numrow)
}
# Chi-square statistic
sum((c(eDPH[1:19], sum(eDPH[20:numrow])) - c(emp[1:19], sum(emp[20:numrow])))^2
  / c(eDPH[1:19], sum(eDPH[20:numrow])))
```

Regarding the predictive performance, we use the testing set and compare the mean predictions of the DPH-MoE, NB, and zerooneNB models with the observed frequencies using the RMSE.

```
# Testing set
dataBCout <- doutBC
dataBCout$FreqBC <- dataBCout$FreqBC + 1
nBC <- nrow(doutBC)
# Prediction for vector of initial probabilities
alpha_vecs <- stats::predict(dphMoE_BC$mm, type = "probs", newdata = dataBCout)
# Mean values
mean_val <- c()
for (j in 1:nBC) {
  mean_val[j] <- mean(dph(alpha_vecs[j, ], dphMoE_BC$S)) - 1
}
# RMSE
sqrt(sum((mean_val - doutBC$FreqBC)^2) / nBC)
```

We have that the DPH-MoE specification outperforms the other two models with a RMSE of 6.1806. In contrast, the NB model yields a RMSE of 6.8581, while the zerooneNB model shows a value of 6.9722.

https://doi.org/10.1017/S1748499525100134 Published online by Cambridge University Press

We continue the study with the univariate modeling of the frequency for the CO line. In this case, we have selected a DPH-MoE model of dimension 6 with the same sparse structure of the transition matrix.

```
# Translation
dataCO <- freqinCO
dataCO$FreqCO <- dataCO$FreqCO + 1
# Regression formula
formula <- FreqCO~ CoverageCO + NoClaimCreditCO + TypeCity + TypeCounty
+ TypeMisc + TypeSchool + TypeTown
# Initial DPH object
set.seed(8)
x <- dph(structure = "gcoxian", dimension = 6)
# MoE regression
dphMoE_CO <- MoE(x = x, formula = formula, data = dataCO, stepsEM = 1000)
```

We have that the fitted DPH-MoE yields a value of the loglikelihood of −1,022.96, which is larger than the corresponding values of the NB and zerooneNB models with values of −1,071.14 and −1,067.82, respectively. However, note that the value of the chi-square statistics is 10.49, which is marginally bigger than the values of 10.39 and 10.37 coming from the NB and zerooneNB specifications.

```
# Empirical count
numrow <- max(freqCO) + 1
emp <- rep(0, numrow)
for (i in 1:numrow) {
  emp[i] <- sum(freqCO == (i - 1))
}
# Expected frequency
eDPH <- 0
for (j in 1:nrow(dataCO)) {
  eDPH <- eDPH + dens(dph(dphMoE_CO$alpha[j, ], dphMoE_CO$S), 1:numrow)
}
# Chi-square statistic
sum((c(eDPH[1:10], sum(eDPH[11:numrow])) - c(emp[1:10], sum(emp[11:numrow])))^2
  / c(eDPH[1:10], sum(eDPH[11:numrow])))
```

Moreover, we assess the predictive performance of the fitted DPH-MoE model using the testing set and the RMSE measure. We obtain a value of 0.5847 for this measure, which indicates superior performance compared to the NB and zerooneNB specifications, with corresponding values of 0.6615 and 0.6110.

https://doi.org/10.1017/S1748499525100134 Published online by Cambridge University Press

```
dataCOout <- doutCO
dataCOout$FreqCO <- dataCOout$FreqCO + 1
nCO <- nrow(doutCO)
# Prediction for vector of initial probabilities
alpha_vecs <- stats::predict(dphMoE_CO$mm, type = "probs", newdata = dataCOout)
# Mean values
mean_val <- c()
for (j in 1:nCO) {
  mean_val[j] <- mean(dph(alpha_vecs[j, ], dphMoE_CO$S)) - 1
}
# RMSE
sqrt(sum((mean_val - doutCO$FreqCO)^2) / nCO)
```

Finally, we focus on the joint modeling of the frequencies for the BC and CO lines via an mDPH model with MoE regression. We first need to filter the data to consider only the entires with coverage in both lines and apply a translation to one.

```
biv_ind <- which(data$CoverageBC > 0 & data$CoverageCO > 0)
data_biv <- data[biv_ind, ]

data_biv$FreqCO <- data_biv$FreqCO + 1
data_biv$FreqBC <- data_biv$FreqBC + 1
data_biv$CoverageBC <- log(data_biv$CoverageBC)
data_biv$CoverageCO <- log(data_biv$CoverageCO)
responses <- cbind(data_biv$FreqBC, data_biv$FreqCO)
```

We then specify the regression formula and an initial mDPH object. Note that we have selected dimension 7 with a general Coxian-like structure of the transition matrices in accordance with the previous univariate analyses.

```
# Regression formula
formula <- responses ~ CoverageCO + NoClaimCreditCO + CoverageBC + lnDeductBC
+NoClaimCreditBC + TypeCity + TypeCounty + TypeMisc + TypeSchool + TypeTown
# Initial mDPH model
set.seed(12345)
x <- mdph(structure = c("gcoxian", "gcoxian"), dimension = 7)
```

With these two objects at hand, we can now perform regression via the MoE framework using the `MoE()` function.

```
# MoE regression
bivMoE <- MoE(
  x = x,
  formula = formula,
  y = responses,
  data = data_biv,
  stepsEM = 350
)
```

https://doi.org/10.1017/S1748499525100134 Published online by Cambridge University Press

Table 3. Summary of frequency model performance: loglikelihood, chi-square statistic, and RMSE

Model	Loglikelihood	Chi-square	RMSE
DPH-MoE (BC)	−4,777.64	22.45	6.1806
NB (BC)	−5,102.62	88.09	6.8581
ZeroOneNB (BC)	−4,958.29	34.51	6.9722
DPH-MoE (CO)	−1,022.96	10.49	0.5847
NB (CO)	−1,071.14	10.39	0.6615
ZeroOneNB (CO)	−1,067.82	10.37	0.6110
mDPH-MoE (BC + CO)	−2,979.96	—	—
ZeroOneNB + NB + Gaussian copula (BC + CO)	−3,321.73	—	—

Table 4. Classes of matrix distributions available in matrixdist

Class	Constructor function
DPH	dph()
PH	ph()
IPH	iph()
fMDPH	bivdph()
mDPH	mdph()
MPH*	MPHstar()
fMPH	bivph()
mPH	mph()
fMIPH	biviph()
mIPH	miph()

The resulting fitted matrix model yields a loglikelihood of −2,979.96. This outperforms the multivariate model in Frees *et al.* (2016), consisting of a Gaussian copula with zerooneNB margin for BC and NB margin for CO, which has leads to a value of −3,321.73 of the loglikelihood.

We conclude the frequency analysis as in Table 3, which summarizes the performance of the models discussed above based on loglikelihood, chi-square statistics, and RMSE.

6. Summary of methods

We have explored the fundamental properties of some matrix distributions and demonstrated the use of the `matrixdist` package for analyzing real-life data through detailed examples. The package encompasses methods for computing various functionals and estimating most of the classes discussed in this paper. Table 4 presents the families of matrix distributions currently implemented in the package. We refer readers to the package documentation for guidance on using these construction functions. For example, consulting `?ph` will provide instructions on creating PH distributions. Additionally, Table 5 offers a comprehensive list of the implemented methods, along with a mapping to the applicable classes. For examples of the use of these methods, the package help can be consulted. The general structure for accessing the desired examples is `¿method_name,class_name-method‘`. For instance, `¿fit,ph-method‘` directs to the documentation of the `fit()` method for the `ph` class. In future versions of the package, methods

https://doi.org/10.1017/S1748499525100134 Published online by Cambridge University Press

Table 5. Methods for matrix distributions available in matrixdist

Method	Description	Available for
sim()	Simulates iid realizations	dph, ph, iph, bivdph, mdph, MPHstar, bivph, mph, biviph, miph
moment()	Moments	dph, ph, bivdph, mdph, bivph, mph
laplace()	Laplace transform	ph, bivph, mph
mgf()	Moment-generating function	ph, bivph, mph
pgf()	Probability-generating function	dph, bivdph, mdph
+	Parmetrization of the sum of two matrix distributions	dph, ph
minimum()	Parmetrization of the minimum of two matrix distributions	dph, ph, iph
maximum()	Parmetrization of the maximum of two matrix distributions	dph, ph, iph
mixture()	Parmetrization of a mixture of two matrix distributions	dph, ph
Nfold()	N-fold convolution with N DPH-distributed	dph, ph
dens()	Density function	dph, ph, iph, bivdph, mdph, bivph, mph, biviph, miph
cdf()	Cumulative distribution function	dph, ph, iph, mph, miph
haz()	Hazard rate function	ph, iph
quan()	Quantiles	ph, iph
TVR()	Transformation via rewards	dph, ph
fit()	Estimation	dph, ph, iph, bivdph, mdph, MPHstar, bivph, mph, biviph, miph
reg()	PI regression	ph, iph
MoE()	MoE regression	dph, ph, iph, bivdph, mdph, mph, miph
marginal()	Parametrization of marginal distribution	bivdph, mdph, MPHstar, bivph, mph, biviph, miph
linCom()	Parametrization of a linear combination of marginals	MPHstar, bivph
mean()	Mean or vector of means	dph, ph, bivdph, mdph, MPHstar, bivph, mph
var()	Variance or covariance matrix	dph, ph, bivdph, mdph, MPHstar, bivph, mph
cor()	Correlation matrix	bivdph, mdph, MPHstar, bivph, mph

and classes are expected to gradually be included, including general heavy-tailed models as in Albrecher *et al.* (2023a) and Bladt & Yslas (2022). Moreover, dedicated functions for computing risk measures, such as Expected Shortfall, will be provided in future iterations of the package. It is worth mentioning that VaR can be already obtained using the `quan()` method, and moment-based risk measures can be calculated from the raw moments, which are available using the `moment()` method.

Supplementary material. The supplementary material for this article can be found at https://doi.org/10.1017/S1748499525100134.

Data availability statement. Data availability does not apply to this article as no new data were created or analyzed. The `matrixdist` package is available in the Comprehensive R Archive Network (CRAN) and the GitHub repository https://github.com/martinbladt/matrixdist_1.0. The numerical examples are reproducible using the code contained in the manuscript.

https://doi.org/10.1017/S1748499525100134 Published online by Cambridge University Press

Funding statement. The first author was supported by the Carlsberg Foundation, grant CF23-1096.

Competing interests. The authors declare none.

References

Albrecher, **H.**, & **Bladt**, **M.** (2019). Inhomogeneous phase-type distributions and heavy tails. *Journal of Applied Probability*, **56**(4), 1044–1064. doi: 10.1017/jpr.2019.60.

Albrecher, **H.**, **Bladt**, **M.**, **Bladt**, **M.**, & **Yslas**, **J.** (2022a). Mortality modeling and regression with matrix distributions. *Insurance: Mathematics and Economics*, **107**, 68–87. doi: 10.1016/j.insmatheco.2022.08.001.

Albrecher, **H.**, **Bladt**, **M.**, **Bladt**, **M.**, & **Yslas**, **J.** (2023a). Continuous scaled phase-type distributions. *Stochastic Models*, **39**(2), 293–322. doi: 10.1080/15326349.2022.2089683.

Albrecher, **H.**, **Bladt**, **M.**, & **Müller**, **A. J. A.** (2022b). Penalised likelihood methods for phase-type dimension selection. *Statistics & Risk Modeling*, **39**(3-4), 75–92. doi: 10.1515/strm-2021-0026

Albrecher, **H.**, **Bladt**, **M.**, & **Müller**, **A. J. A.** (2023b). Joint lifetime modelling with matrix distributions. *Dependence Modeling*, **11**(1). doi: 10.1515/demo-2022-0153

Albrecher, **H.**, **Bladt**, **M.**, & **Yslas**, **J.** (2022c). Fitting inhomogeneous phase-type distributions to data: The univariate and the multivariate case. *Scandinavian Journal of Statistics*, **49**(1), 44–77. doi: 10.1111/sjos.12505.

Asmussen, **S.**, **Nerman**, **O.**, & **Olsson**, **M.** (1996). Fitting phase-type distributions via the EM algorithm. *Scandinavian Journal of Statistics*, **23**(4), 419–441. https://www.jstor.org/stable/4616418.

Baringhaus, **L.**, & **Franz**, **C.** (2004). On a new multivariate two-sample test. *Journal of Multivariate Analysis*, **88**(1), 190–206. doi: 10.1016/S0047-259X(03)00079-4.

Bladt, **M.** (2022). Phase-type distributions for claim severity regression modeling. *ASTIN Bulletin: The Journal of the IAA*, **52**(2), 417–448. doi: 10.1017/asb.2021.40.

Bladt, **M.** (2023). A tractable class of multivariate phase-type distributions for loss modeling. *North American Actuarial Journal*. doi: 10.1080/10920277.2023.2167833.

Bladt, **M.**, & **Nielsen**, **B. F.** (2017). *Matrix-exponential distributions in applied probability*, vol. **81**. Springer. doi: 10.1007/978-1-4939-7049-0

Bladt, **M.**, & **Yslas**, **J.** (2022). Heavy-tailed phase-type distributions: A unified approach. *Extremes*, **25**(3), 529–565. doi: 10.1007/s10687-022-00436-8.

Bladt, **M.**, & **Yslas**, **J.** (2023a). *Matrixdist: statistics for matrix distributions*. R package version 1.1.9. https://CRAN.R-project.org/package=matrixdist.

Bladt, **M.**, & **Yslas**, **J.** (2023b). Phase-type mixture-of-experts regression for loss severities. *Scandinavian Actuarial Journal*, **4**, 303–329. doi: 10.1080/03461238.2022.2097019.

Bladt, **M.**, & **Yslas**, **J.** (2023c). Robust claim frequency modeling through phase-type mixture-of-experts regression. *Insurance: Mathematics and Economics*, **111**, 1–22. doi: 10.1016/j.insmatheco.2023.02.008.

Campillo-Navarro, **A.** (2018). Order statistics and multivariate discrete phase-type distributions. PhD diss., Ph. D. thesis, DTU Lyngby. https://backend.orbit.dtu.dk/ws/portalfiles/portal/177528739/PHD_2018_492.pdf

Cheung, **E. C. K.**, **Peralta**, **O.** & **Woo**, **J.-K.** (2022). Multivariate matrix-exponential affine mixtures and their applications in risk theory. *Insurance: Mathematics and Economics*, **106**, 364–389. doi: 10.1016/j.insmatheco.2022.07.001. issn: 0167-6687.

Cox, **D. R.** (1972). Regression models and life-tables. *Journal of the Royal Statistical Society: Series B (Methodological)*, **34**(2), 187–202. https://www.jstor.org/stable/2985181

Dutang, **C.**, **Goulet**, **V.**, & **Pigeon**, **M.** (2008). actuar: An R package for actuarial science. *Journal of Statistical Software*, **25**, 1–37. doi: 10.18637/jss.v025.i07.

Frees, **E. W.**, **Lee**, **G.**, & **Yang**, **L.** (2016). Multivariate frequency-severity regression models in insurance. *Risks*, **4**(1), 4. doi: 10.3390/risks4010004.

Franz, **C.** (2024). *Cramer: Multivariate nonparametric Cramer-test for the two-sample-problem*. R package version 0.9-4. https://CRAN.R-project.org/package=cramer.

Kulkarni, **V. G.** (1989). A new class of multivariate phase type distributions. *Operations Research*, **37**(1), 151–158. https://www.jstor.org/stable/171156

Moler, **C.**, & **Van Loan**, **C.** (1978). Nineteen dubious ways to compute the exponential of a matrix. *SIAM Review*, **20**(4), 801–836. doi: 10.1137/S00361445024180.

Okamura, **H.**, & **Dohi**, **T.** (2015). mapfit: An R-based tool for PH/MAP parameter estimation . In Quantitative evaluation of systems: 12th international conference, QEST 2015, Madrid, Spain, September 1-3, 2015, proceedings 12 (pp. 105–112). Springer. doi: 10.1007/978-3-319-22264-6_7

Olsson, **M.** (1996). Estimation of phase-type distributions from censored data. *Scandinavian Journal of Statistics*, **23**(4), 443–460. https://www.jstor.org/stable/4616419

R Core Team (2024). R: A language and environment for statistical computing. ISBN 3-900051-07-0. Vienna, Austria: R Foundation for Statistical Computing. https://www.R-project.org/

Rivas-González, **I.**, **Andersen**, **L. N.**, & **Hobolth**, **A.** (2023). PhaseTypeR: an R package for phase-type distributions in population genetics. *Journal of Open Source Software*, **8**(82), 5054. doi: 10.21105/joss.05054.

Cite this article: Bladt M, Mueller A and Yslas J (2025). matrixdist: an R package for statistical analysis of matrix distributions, *Annals of Actuarial Science*, 1–35. https://doi.org/10.1017/S1748499525100134