Probability Distributions of Phase Type

Austin Riis-Due, ASA
Ph.D. Candidate, The University of Waterloo
ACTSC 966, Professor Gord Willmot

1 Introduction

Actuaries are faced with the problem of predicting the total liability borne by the insurer for a variety of classical actuarial tasks, including pricing, reserving, and forecasting of surplus. The language of probability theory has helped the actuary greatly in expressing these forecasts. It helps to have a flexible class of distributions to work with to express these quantities, and it is highly desirable that such a class of distributions be both low dimensional and easily computable, so that it might lend itself to the fitting algorithms provided by the tools of statistical inference and modern data science. There are two methods that are of prominence in the modern actuarial toolkit for the modeling of aggregate claims. These include mixtures of distributions of the Erlang type, and distributions of the Phase Type. In this report we will review the distributions of the Phase Type, including discrete, continuous, and inhomogenous Phase Types, and we will use a numerical example to determine whether the mixture of Erlangs or the Phase Type distribution, when properly calibrated, gives a more accurate fit to data. We will also review the methods used to fit them and remark on how efficient it is to fit one of these models to claims data generated artificially by an aggregate model. Finally, we will discuss two important applications of Phase Types, as a model in queueing theory and, in a similar application, as a model for the probability of ruin and associated quantities.

2 The Class of Phase Types

2.1 Definitions

Phase Type distributions are defined by the time to absorption random variable associated with an underlying Markov Chain which has some number of transient states and one absorbing state. More formally, let $\{X_t\}_{t\in\mathbb{R}^+}$ be a Markov process in continuous time, with m transient states, and a single absorbing state, $m+1$. Define

$$\tau = \mathbf{inf}\left\{t \in \mathbb{R}^+ | X_t = m+1\right\} \tag{1}$$

as the smallest time such that the Markov Chain has been absorbed. We can represent the Markov Chain through its transient transition rate matrix $T \in \mathbb{R}^{m \times m}$ and its column vector of exit transitions given by $\mathbf{t} \in \mathbb{R}^{m \times 1}$. Letting $\mathbf{0} \in \mathbb{R}^{m \times 1}$ be a vector of zeros we can represent the overall transition intensity matrix (including exit rates) for the underlying Markov Chain as

$$\mathbf{T} = \left(\begin{array}{c|c} T & \mathbf{t} \\ \hline \mathbf{0} & 0 \end{array}\right)$$

However, we also need to specify the probability vector associated with the starting states. It is frequently assumed that the probability of starting in the absorbing state is zero, i.e. $\mathbb{P}(\tau = 0) = 0$ but in general we may assign positive probability to this case, forming a mass point at zero. Therefore we can define the initial probability vector over the Markov Chain as $\boldsymbol{\pi} \in \Delta^{m+1}$ where Δ^{m+1} is the $m+1$ dimensional simplex, and then assume that $\pi_{m+1} = 0$.

We have a representation for a Phase Type distribution by way of $(\boldsymbol{\pi}, T)$ which is nonunique. Since we may interchange any two states simultaneously in both arguments and obtain a different transition rate matrix and initial probability vector which represent the same random variable τ, there is indeed no unique representation for a given Phase Type distribution. Later when we review the numerical algorithms we will see that this hinders us in the sense that we can't determine representation dependent quantities of the distribution.

Interestingly, this class includes several key subclasses of distributions that are frequently of interest in queueing (and in ACTSC 963!) including exponentials, hyperexponentials, Erlangs, mixtures of Erlangs (only true in the finite case, since the starting state probability vector and transition density must be finite), and Coxian-N's. These are relatively straightforward to show by definition of the Phase Type, simply by considering an appropriate diagram for the representation. For me, it was really insightful to see that the diagram for the associated Markov Chain for the Erlang-n is a row of states, all of which are connected by unidirectional arrows, indicating the process enters a state, spends time there, then moves to the next one with some transition intensity. Finally, the process exits after the n-th state. The resulting random variable is the sum of all the time spent in each state, and the time spent in each state is of course exponential since there is only one way to exit the state. Therefore the Erlang-n is the n-fold convolution of exponentials. Even knowing this fact, it's nice that the logic transfers to this representation of the associated Phase Type distribution.

Moreover, convolutions of Phase Type distributions and mixtures of Phase Type distributions are again Phase Type. Interestingly, a discrete Phase Type mixture of n fold convolutions of Phase Types is again Phase Type, so that if claims frequency and severity are Phase Type, the aggregate loss distribution is again a Phase Type distribution with modified parameters. This is reviewed in more detail in Neuts (1975).

With the introductory aspect aside, we will now proceed with a discussion of some key quantities of these distributions.

2.2 Key Quantities

Using this definition of Phase Type distributions we can now review some principal properties of Phase Types, including their density and cumulative distribution function, moments, and Laplace transform.

We can approach the probability density function using the perspective of a probabilistic argument. First we must note that the continuous time probability transition matrix, i.e. given any starting state, the matrix corresponding to the probability of being in any ending state by time s is $P_s = \exp(Ts)$ where T is the transition rate matrix given above. Now, the probability of a particular path of X_t starting in state i and exiting from state j in an infinitesimal time interval $(s, s+ds)$ is $\mathbb{P}(\tau \in (s, s+ds)|X_0 = i)ds = \pi_i \exp(Ts)_{ij} t_j ds$ and therefore the probability of X_t being within the infinitesimal interval regardless of starting state is the expectation over all possible states

$$\mathbb{P}(\tau \in (s, s+ds))ds = \sum_{i=0}^{m} \pi_i \sum_{j=0}^{m} \exp\left(Ts\right)_{ij} t_j ds = \boldsymbol{\pi} \exp(Ts)\boldsymbol{t} ds$$

which implies that the density is

$$f(s) = \boldsymbol{\pi} \exp\left(Ts\right)\boldsymbol{t} \tag{2}$$

for $s > 0$ which is the quantity we will use to derive the distribution function and Laplace transform.

It is worth noting here that there is a complete correspondence to discrete distributions of Phase Type over Markov Chains $\{X_n\}_{n\in\mathbb{N}}$ and associated stopping time of the form of (1) given by $\sigma = \mathbf{min}\{t \in \mathbb{N}|X_n = m+1\}$ by way of the $m \times m$ probability transition matrix P_s of probabilities of transition over each step and the exit vector of probabilities being defined in parallel to the description above. The resulting probability mass function is then

$$\mathbb{P}(\sigma = s+1) = \boldsymbol{\pi} P_s \boldsymbol{t}$$

where s is a positive integer.

Returning to continuous time, we can obtain the distribution function of the Phase Type by integrating the density. Noting that $\boldsymbol{t} = -T\boldsymbol{e}$ where $\boldsymbol{e}$ is a vector of ones that is length m. Performing the matrix integration on the density gives us

$$\begin{aligned}
\mathbb{P}(\tau \leq s) &= \int_0^s f(u)du \\
&= \int_0^s \boldsymbol{\pi} \exp(Tu)\boldsymbol{t} du \\
&= T^{-1}\boldsymbol{\pi} \exp(Ts)\boldsymbol{t} - T^{-1}\boldsymbol{\pi} I\boldsymbol{t} \\
&= T^{-1}\boldsymbol{\pi} \exp(Ts)(-T\boldsymbol{e}) - T^{-1}\boldsymbol{\pi} I(-T\boldsymbol{e}) \\
&= \boldsymbol{\pi} I\boldsymbol{e} - \boldsymbol{\pi} \exp(Ts)\boldsymbol{e} \\
&= 1 - \boldsymbol{\pi} \exp(Ts)\boldsymbol{e}
\end{aligned}$$

Note that to push the matrices around we needed to use properties of the exponential matrix integral, and we assume the invertibility of the transition matrix.

We can notice something very important about the Phase Type, which is its exponential tail. This leads to issues with fitting to heavy tailed data and means that in this case, to be accurate in the tails we will need a much larger transition rate matrix to give the process the flexibility to fit to the tails.

To obtain quantiles of the distribution, i.e. to determine the value at risk for the distribution we need only solve the above equation numerically for s. This allows for efficient computation of VaR and ES, given a representation for the Phase Type distribution. In fact, these quantities are representation invariant, meaning that regardless of the permutation of the transition rate matrix and initial probability vector. We will leverage this in Section 3 to numerically compute these quantities without knowledge of the underlying Markov Chain's occupation times.

We can also derive the Laplace transform of the density and therefore the moments of the Phase Type distribution as the following.

$$\begin{aligned}
\tilde{f}(z) &= \int_0^\infty \exp(-zu) f(u)du \\
&= \int_0^\infty \exp(-zu)\boldsymbol{\pi} \exp(Tu)\boldsymbol{t} du \\
&= \int_0^\infty \boldsymbol{\pi} \exp((T - Iz)\, u)\boldsymbol{t} du \\
&= 0 - \boldsymbol{\pi}(T - Iz)^{-1} I\boldsymbol{t} \\
&= \boldsymbol{\pi}(Iz - T)^{-1}\boldsymbol{t}
\end{aligned}$$

After taking derivatives this produces the expression

$$\mathbb{E}[\tau^n] = (-1)^n n! \boldsymbol{\pi} T^{-n} \boldsymbol{e} \tag{3}$$

For the n-th moment of the distribution.

These ideas are covered in Asmussen (2000) and Bladt (2005). The thing which makes Phase Types so mathematically appealing is that for any distribution on the positive reals,

there is a sequence of Phase Type distributions which converge weakly to that distribution, and therefore the class of Phase Types is weakly dense in the class of positive real valued probability distributions. For thin tailed distributions this is especially true due to the exponential tail, but in heavy tailed applications including projections of individual insurance claims this may be a difficult assumption to overcome. In the following section we will leverage this property of Phase Types to fit to data in two cases, and compare the fit graphically to a finite mixture of Erlang distributions.

2.3 A Brief Discussion of Finite Mixtures of Erlang Distributions and the Associated E-M Algorithm

Finite mixtures of Erlang distributions are a subclass of Phase Type distributions which are of the form

$$f(x) = \sum_{n=1}^{K} q_n e_{n,\beta}(x) \tag{4}$$

subject to $\sum_{n=1}^{K} q_n = 1$, where $e_{n,\beta}(x)$ is the Erlang-n density, parameterized by the shape n and rate parameter common to all entities in the mixture β. In the limit as K tends to infinity, these types of distributions are also weakly dense in the class of positive real values distributions and have properties that are very useful in the analysis of aggregate insurance claims including tractable distributions over a variety of frequency distributions. We note that only finite mixtures of Erlangs are explicitly contained in the class of Phase Types since the rate matrix and initial probability vector must be strictly finite.

In practice, a finite mixture works well and can be fit to data very easily by a straightforward expectation maximization (E-M) algorithm. Indeed since the shape parameters are fixed, the likelihood need only be tuned to the rate parameter, a single number shared by all distributions. The python algorithm that I use for comparison to Phase Type fits in this project has a good runtime, executing in about one minute for 100 mixture components over 5,000 data points. A short description of the algorithm is as follows.

1. Initialize the weights of the mixtures and the shared scale parameter randomly. Prescribe the number of mixtures.
2. Weights are updated by taking the sum of the density of the corresponding mixture component over all data points and dividing it by the sum of all density components over all data points.
3. Solve the loglikelihood equation for the shared rate parameter β.
4. Return to Step 2. Iterate until convergence within a tolerance.

However, one key issue is how many components to fit, which may be addressed by clustering on further known covariates or through any clustering method applied to univariate data. Penalization methods such as AIC and BIC may be useful although for risk management interests it should be kept in mind that since these methods depend on likelihood metrics they will be dominated by the fit in the body of the distribution and may ignore the fit in the tails.

In any case, the associated E-M algorithm essentially proceeds by prescribing K components, the number of mixtures in our model, and inherently specifying the shape parameters for the components $1, ..., K$. Initialize the weights and shared rate parameter randomly. We update the weights according to the componentwise contribution to the likelihood of that Erlang distribution, and then minimize the negative loglikelihood of the distribution with those weights, and the integer shape parameters specified by the number of components. We optimize this with respect to the shared rate parameter. This proceeds until a specified tolerance is met between the loglikelihoods and we say that the algorithm has converged to the final set of parameters. In Section 3 we will compare this simpler, more tractable subclass of Phase Type distributions with the more general Phase Type described above.

Wenyong Gui (2018) covers a more general algorithm to fit mixtures of Erlangs, the one presented here is much simpler but not as general as their utilization, requiring that we prespecify the number of mixtures.

3 Numerical Study

We first need to discuss how to use the tools from modern statistics to fit Phase Type distributions. There are two common methods, both of which are useful for different reasons. The E-M algorithm follows from arguments in Soren Asmussen (1996), while the MCMC algorithm follows from Bladt (2005). The implementations of these algorithms are my own.

3.1 Fitting a Phase Type Distribution

3.1.1 An E-M Algorithm

The primary issue with fitting Phase Type distributions is that you never observe the underlying Markov Chain, instead only observing the final time spent across all states. This is problematic since we have incomplete data. Suppose we observed the complete data, with $\boldsymbol{j} = (j_1, ..., j_n)$ being the trajectories of the Markov process that come from the observations $(x_1, ..., x_n)$. The associated likelihood is of the form

$$p(\boldsymbol{j}|\boldsymbol{\pi}, T) = \prod_{i=1}^{m} \pi_i^{B_i} \prod_{i=1}^{m} t_i^{N_{im+1}} \exp(-t_i Z_i) \prod_{i=1}^{m} \prod_{j=1 \neq i}^{m} t_{ij}^{N_{ij}} \exp(-t_{ij} Z_i) \tag{5}$$

Where B_i is the number of times the process starts in state i, N_{im+1} is the number of times

the process jumps from state i to state $m+1$, N_{ij} is the number of transient jumps the process makes between states i and j, and Z_i is the total time the process spent in state i.

The maximum likelihood estimators (MLEs) for the above quantities may be computed from the above information and then the E-M algorithm may be applied. However, in the expectation step, the resulting forms for the conditional expectations involve matrix exponentials and integrals of matrix exponentials which are solvable via Runge Kutta methods. Although computationally intensive, this method is more exact than the MCMC method presented below. We will provide a brief description of the algorithm and give two examples of its fit to data.

In Soren Asmussen (1996) it is demonstrated that we may calibrate the distribution by calculating the conditional expectation of the above sample statistics, the "E" step of the algorithm, and then update the associated parameters of the distribution via their maximum likelihood estimates, the "M" step. These estimates follow naturally from explicit calculation of the MLEs, or from appealing to the theory of the exponential family of distributions.

For the $k+1$st step of the algorithm, the MLEs of the updates are

$$\pi_i^{(k+1)} = \frac{B_i^{(k+1)}}{n} \tag{6}$$

$$t_{ij}^{(k+1)} = \frac{N_{ij}^{(k+1)}}{Z_i^{(k+1)}} \tag{7}$$

$$t_i^{(k+1)} = \frac{N_{i0}^{(k+1)}}{Z_i^{(k+1)}} \tag{8}$$

$$t_{ii}^{(k+1)} = -(t_i^{(k+1)} + \sum_{j=1, j\neq i}^{p} t_{ij}^{(k+1)}) \tag{9}$$

Where over all data points $v = 1, ..., n$ we have

$$B_i^{(k+1)} = \sum_{v=1}^{n} \mathbb{E}_{(\boldsymbol{\pi},\boldsymbol{T})^{(k)}}[B_i^{[v]} | Y = y_v] \tag{10}$$

$$Z_i^{(k+1)} = \sum_{v=1}^{n} \mathbb{E}_{(\boldsymbol{\pi},\boldsymbol{T})^{(k)}}[Z_i^{[v]} | Y = y_v] \tag{11}$$

$$N_{ij}^{(k+1)} = \sum_{v=1}^{n} \mathbb{E}_{(\boldsymbol{\pi},\boldsymbol{T})^{(k)}}[N_{ij}^{[v]} | Y = y_v], \text{ for } i, j = 1, 2, ..., m, j \neq i \tag{12}$$

The tough part of this process is the computation of the conditional expectation. Through a short proof using dominated convergence we can show that the following formulas for conditional expectations hold, as in Soren Asmussen (1996).

$$\mathbb{E}_{(\boldsymbol{\pi},\boldsymbol{T})}[B_i^{[v]}|Y=y_v] = \frac{\pi_i b_i(y_v|\boldsymbol{T})}{\boldsymbol{\pi}\boldsymbol{b}(y_v|\boldsymbol{T})} \tag{13}$$

$$\mathbb{E}_{(\boldsymbol{\pi},\boldsymbol{T})}[Z_i^{[v]}|Y=y_v] = \frac{c_i(y_v;i|\boldsymbol{\pi},\boldsymbol{T})}{\boldsymbol{\pi}\boldsymbol{b}(y_v|\boldsymbol{T})} \tag{14}$$

$$\mathbb{E}_{(\boldsymbol{\pi},\boldsymbol{T})}[N_{ij}^{[v]}|Y=y_v] = \frac{t_{ij}c_j(y_v;i|\boldsymbol{\pi},\boldsymbol{T})}{\boldsymbol{\pi}\boldsymbol{b}(y_v|\boldsymbol{T})} \tag{15}$$

$$\mathbb{E}_{(\boldsymbol{\pi},\boldsymbol{T})}[N_{i0}^{[v]}|Y=y_v] = \frac{t_i a_i(y_v|\boldsymbol{\pi},\boldsymbol{T}}{\boldsymbol{\pi}\boldsymbol{b}(y_v|\boldsymbol{T})} \tag{16}$$

The functions a, b, and c here are of the form

$$\boldsymbol{a}(y|\boldsymbol{\pi},\boldsymbol{T}) = \boldsymbol{\pi}\exp(\boldsymbol{T}y), \tag{17}$$

$$\boldsymbol{b}(y|\boldsymbol{T}) = \exp(\boldsymbol{T}y)\boldsymbol{t}, \tag{18}$$

$$\boldsymbol{c}(y;i|\boldsymbol{\pi},\boldsymbol{T}) = \int_0^y \boldsymbol{\pi}\exp(\boldsymbol{T}u)\boldsymbol{e}_i\exp(\boldsymbol{T}(y-u))\,du \tag{19}$$

for $i = 1, ...m$. These equations satisfy the system of ordinary differential equations

$$\boldsymbol{a}'(y|\boldsymbol{\pi},\boldsymbol{T}) = \boldsymbol{a}(y|\boldsymbol{\pi},\boldsymbol{T})\boldsymbol{T} \tag{20}$$

$$\boldsymbol{b}'(y|\boldsymbol{T}) = \boldsymbol{T}\boldsymbol{b}(y|\boldsymbol{T}) \tag{21}$$

$$\boldsymbol{c}'(y;i|\boldsymbol{\pi},\boldsymbol{T}) = \boldsymbol{T}\boldsymbol{c}(y;i|\boldsymbol{\pi},\boldsymbol{T}) + a_i(y|\boldsymbol{\pi},\boldsymbol{T})\boldsymbol{t} \tag{22}$$

for $i = 1, ..., m$ subject to the initial conditions

$$\boldsymbol{a}(0|\boldsymbol{\pi},\boldsymbol{T}) = \boldsymbol{\pi} \tag{23}$$

$$\boldsymbol{b}(0|\boldsymbol{T}) = \boldsymbol{t} \tag{24}$$

$$\boldsymbol{c}(0;i|\boldsymbol{\pi},\boldsymbol{T}) = \boldsymbol{0} \tag{25}$$

for $i = 1, ..., m$ and this $m(m+2)$ dimensional system of ordinary differential equations (ODEs) can be solved numerically via the Runge Kutta method. This is a very important part of the puzzle - to add one additional phase to the model, we add $2m+3$ parameters and that many dimensions to the system of ODEs. Asmussen writes his code in the C programming language, which is more efficient and exact than the python programming language, in which we apply the algorithm. In fitting to actual data, Asmussen used in one case a general structure of $m = 15$, and a structure of $m = 30$, but with a Coxian special case which simplifies the calculations. The remainder of the fits are of order between 2 and 6, producing systems of dimension 8 and 48, respectively. We fit the distribution using $m = 2$ and $m = 3$. With a more involved ODE solver we could potentially fit higher order Phase

Type distributions, but $m = 4$ is already a multi-day run on a 2024 MacBook Pro with an M3 Max processor and 48 GB of RAM.

The algorithm then proceeds as follows:

1. Initiate a random representation for the Phase Type of the order of interest.
2. For each data point, solve the ODE through Runge Kutta methods and obtain the conditional expectations of the sufficient statistics.
3. Sum the sufficient statistics over the data points and update the MLE of the transition intensities and the intial probability vector.
4. Return to Step 2.

When we are interested in the parameters themselves, this algorithm is useful in the sense that it produces exact estimates of the representation, allowing for explicit calculation of the moments of the distribution. When we are interested in representation invariant quantities we can use the following alternative algorithm.

3.1.2 An MCMC Algorithm

This method is limited in the sense that it can only compute quantities which are representation invariant, i.e. when we permute the columns of the transition rate matrix and the initial probability vector these quantities do not change.

The quantities which may be estimated include the density and quantiles, as well as the infinite time ruin probability as we will discuss in Section 4.

The general idea of this method is to create a stationary sequence of distributions whose stationary distribution is Phase Type, given the data. We do so through the parameters of the Phase Type distribution, by noting that the likelihood equation above is of a nice form, proportional to a product of independent gamma random variables for the transition intensities and a Dirichlet random variable for the initial probability vector. This will be the form of the prior. The posterior is the prior with updates from the sufficient statistics,

and so given some data $\{x_i\}_{i=1}^n$ we can draw a representation from the posterior distribution. However, to compute the incremental updates from the posterior we need to sample trajectories from the associated Markov Chain process. We can do so by drawing trajectories from the Markov Chain with the representation and checking if they are absorbed by the time of observation x_i. If so, we toss them out with the bathwater, if not, we draw another similar trajectory and probabilistically keep one and discard one. We then compute the summary statistics and hence the update for our parameters from the trajectory. Once this algorithm burns in, we can sample from the posterior and compute a quantity like the density or infinite time ruin probability to obtain a Monte Carlo estimate of it.

As far as the algorithm, it essentially proceeds as follows:

1. Initialize the prior with some random parameters and draw a sample $(\boldsymbol{\pi}, T)$.

2. For each data point x, apply the Metropolis Hasting algorithm to update the prior parameters.
   (a) Sample a trajectory X from the Markov Chain given by $(\boldsymbol{\pi}, T)$ with exit rate vector $t = -T\boldsymbol{e}$ not absorbed by x.
   (b) Sample a second trajectory Y not absorbed by x and keep this one with probability $\min(1, \frac{t_Y}{t_X})$ where t_X, t_Y are the exit rates associated with the last transient state of X, Y respectively.
   (c) From the trajectory, compute the incremental updates needed for the posterior.
3. Sample a representation $(\boldsymbol{\pi}, T)$ from the posterior and compute a representation invariant functional. Store it for a Monte Carlo estimation of the quantity.
4. Return to Step 2.

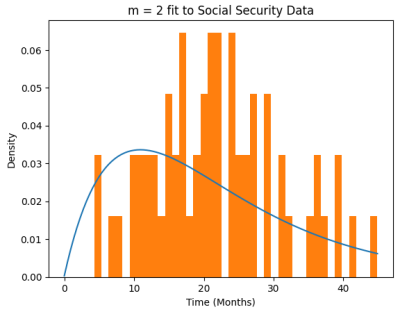
The density estimation for low order Phase Types is given below for the claims data. Higher order Phase Types have more "forces of decrement" pulling them into the exit state at a given time and therefore take longer to find suitable trajectories. Of the two algorithms, the E-M is more computationally efficient but both are plagued by dimensionality. It is much easier to compute finite mixtures of Erlangs for a flexible fit to data.

3.2 Comparison with Finite Mixtures of Erlangs

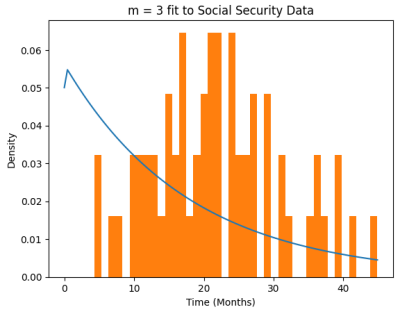
As mentioned above, the fits to data are more difficult when using a Phase Type of the general structure. We can fit special cases more efficiently in higher order but the general case is restricted to order 3 or 4.

We consider two datasets, one is the standard AutoClaims dataset (some large observations in the data have been clipped in the interest of computational time), and the other dataset comes from my mother's law firm, and is the amount of time that a social security claimant has to wait until they are approved in months. The process for a social security claim is a Coxian-3 process, with an initial application stage, a reconsideration stage, a hearnig stage, and an appeal stage. At each step in the process, a claimant may be approved and exit the system. At the end, either the claim is approved or denied, and the claimant may then reenter the system under a new application once their medical or financial situation changes. In any case, the claimant may exit the system after receiving their decision. The data is lump shaped, and so low dimensional Phase Types have issue with fitting to the data, as is exhibited in Soren Asmussen (1996) for the uniform distribution. The AutoClaims data has a natural right tail and therefore even low dimensional Phase Types at least get the shape of the data correct.

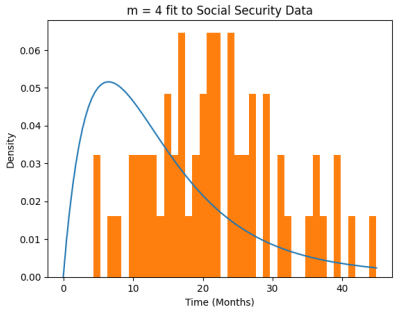
We see that the fit to the lump shaped data is quite poor, and we would need a high dimensional mixture of Erlangs to fit the lump shaped characteristics of the data. Asmussen (2000) discusses in the appendix that for a good fit of a mixture of Erlangs, the corresponding Phase Type that achieves that fit is higher dimensionality. Therefore we would need a very high dimensional Phase Type to fit to this data, and without a more efficient or precise ODE solver this is computationally infeasible.



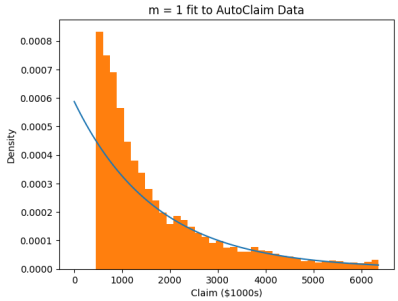
(a) A Phase Type of dimension 2 fit to the Social Security data.



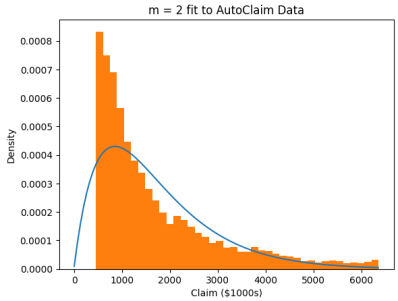
(b) A Phase Type of dimension 3 fit to the Social Security data.



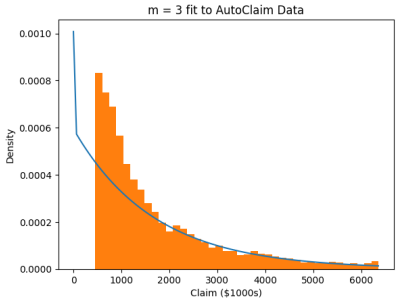
(c) A Phase Type of dimension 4 fit to the Social Security data.



(a) A Phase Type of dimension 1 fit to the AutoClaim data.



(b) A Phase Type of dimension 2 fit to the AutoClaim data.



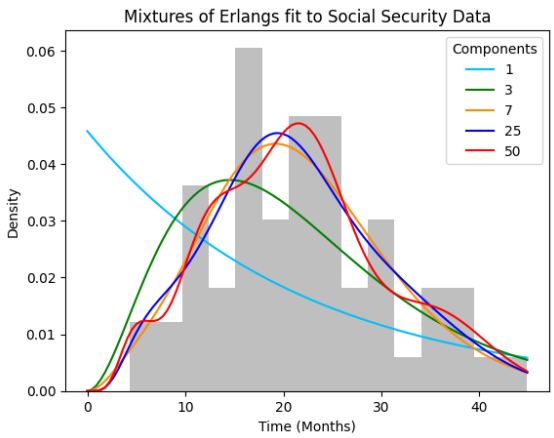
(c) A Phase Type of dimension 3 fit to the AutoClaim data.

4 An Aside: Application to Ruin and Queueing

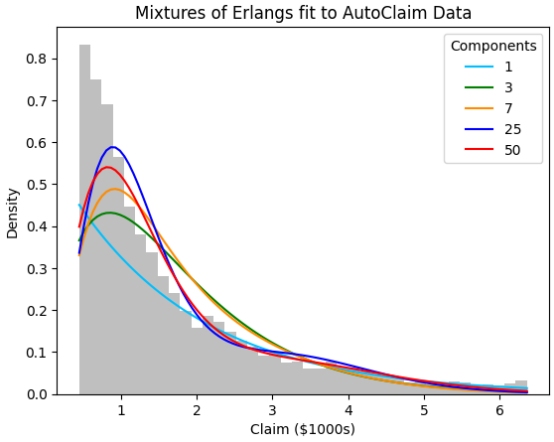
As mentioned above, Phase Type distributions have a deep and intuitive relation to queueing theory and can be used as ruin models.

The idea of using these distributions in a queue is very natural, it simply becomes a model for how long someone spends in a system described by a Markov Chain. As a model for claim sizes for ruin, however, we must expand our intuition and consider each phase as a part of a claim, where it accumulates value over time, the longer it stays in the phase the more value it accumulates. In general the way value accumulates is different in each phase. This leads to the following silly but intuitive idea: suppose a man enters a maze with different rooms from which he must escape, ruled by a wizard who can distort the man's sense of time. The time he takes in each room is random, and we are interested in the total time the man feels he has experienced. This is similar to how insurance claims may accumulate in a Phase Type process where we would like to give interpretation to the phases.

We may apply the Phase Type distribution as the claims severity distribution in the classical Cramer Lundberg model. If we assume we receive a fixed premium c per unit time and we have u initial capital in reserves we can model the surplus as



(a) Mixtures of Erlangs fit to the Social Security data.



(b) Mixtures of Erlangs fit to the AutoClaim data.

$$u_t = u + ct - \sum_{i=0}^{N_t} X_i \tag{26}$$

where N_t is a Poisson count process and X_i are independent and identically distributed Phase Type random variables. Moreover, the counting process is independent of the claim sizes. We then ask about the associated time of ruin $t^*(u) = \inf_{t>0}\{t : u_t \leq 0 | u_0 = u\}$ where the insurer will first go bankrupt given the initial surplus level. Some quantities associated with the time of ruin are the probability that it occurs within finite time, $\psi_V(u) = \mathbb{P}(t^*(u) < V)$ for some time V, the probability that it occurs at all $\psi(u) = \mathbb{P}(t^*(u) < \infty)$ (infinite time ruin), and the distribution function of the excess loss at the time of ruin, also known as the overshoot distribution in queueing. The finite time ruin case is treated in Stanford and Stroinski (1994) and we will review the infinite time ruin case and the associated surplus distribution at the time of ruin.

Let β be the Poisson intensity, B the claim size distribution, which we assume to be Phase Type with representation $(\boldsymbol{\pi}, T), t^*(u)$ the time of ruin at initial surplus level u, $\{S_t\}_{t>0}$ the claim surplus process, and G_+ be the distribution function of $S_{t^*(0)}$, the overshoot at zero surplus.

Asmussen then applies the Pollaczek-Khinchine formula along with qualities of the excess lifetime of a Phase Type and their limiting distributions to obtain that

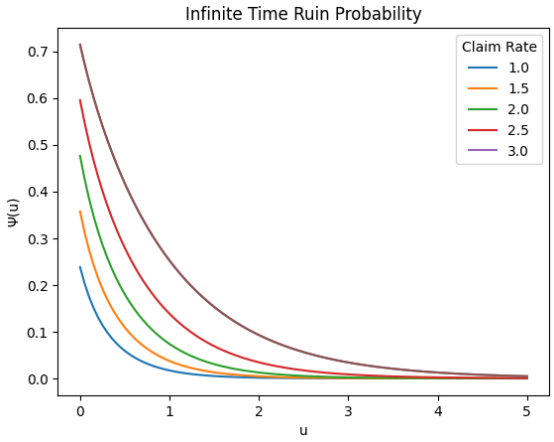
$$\psi(u) = -\beta \boldsymbol{\pi} T^{-1} e^{\left(T - \beta t \boldsymbol{\pi} T^{-1}\right)u} \boldsymbol{e} \tag{27}$$

$$G_+ =^d PH(-\beta \boldsymbol{\pi} T^{-1}, T) \tag{28}$$

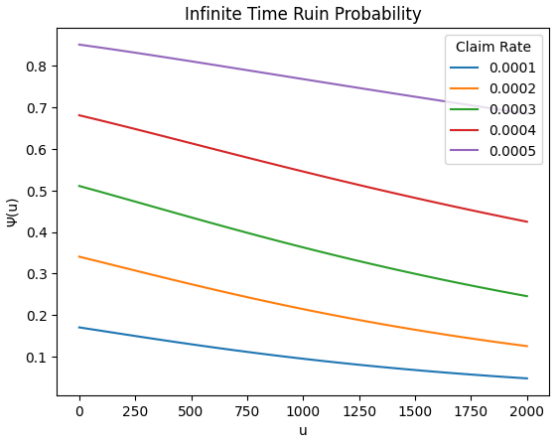
Where $=^d$ denotes equality in distribution, $PH(-\beta \boldsymbol{\pi} T^{-1}, T)$ is a Phase Type representation, and we note that G_+ is defective since there is positive probability that the surplus stays above zero. Note that $\psi(u)$ is a vector since it is state dependent. The nice thing about

the ruin probability is that it is representation invariant so we can calculate it from either numerical algorithm described above.

Using a simple hyperexponential example ($\boldsymbol{\pi} = (0.5, 0.5), T = (-3, -7)_{diag}$) from Asmussen (2000) we obtain the following chart for the total ruin probability for a variety of rate parameters as a function of surplus. We also review the AutoClaims dataset. Note for that graph that the surplus is given in thousands. The rate parameter is constrained so that the surplus process always has positive trend, since otherwise ruin would be certain.



(a) Ruin Probabilities fit to the simple hyperexponential described above.



(b) Ruin probabilities fit to the AutoClaim data. Surplus u given in ($1000s), $m = 2$

We can also assume a more general distribution on the interarrival times for claims, but we come to a nonanalytical representation for the initial probability vector in the associated ruin probabilities. Asmussen (2000) gives a fixed point iteration for the vector and proves its convergence, but the key result is similar - the overshoot distribution is defective Phase Type and we have a representation for the ruin probabilities in terms of the Phase Type quantities.

Esther Frostig (2012) reviews associated quantities including the number of claims and the time to ruin through a Gerber Shiu argument. In general, there has been much work on the quantities of ruin under Phase Type claim sizes, since they are analytically very nice to work with.

5 Conclusion

We have reviewed some basic quantities of Phase Type distributions and how they can be fit to data, and compared them to finite mixtures of Erlangs. While both are dense in the class of positive continuous distributions, the mixture of Erlangs proves easier to fit to data numerically, especially in high dimensions. We also reviewed the associated infinite time ruin problem and how Phase Types may be applied there.

Ultimately, the mixtures of Erlangs are better models when it comes to fitting to data. However, special knowledge of the associated underlying Markov Chain process, as one would

have in queueing, produces a more informed and simplistic Phase Type fitting process. For example, knowing that the structure is a mixture of Coxian-N distributions based on the way that people are allowed to move through the process will lend itself to a more informed and improved fit than attempting to fit a mixture of Erlangs without knowledge of the number of components to use. It may also lead to a more efficient algorithm when many of the parameters are zero since by nature the transitions are not allowed.

In short, when one has the domain knowledge and there is a clear application, as in queueing, the Phase Type is a strong choice, but when the application of the physical interpretation of the Phase Type is not direct, a mixture of Erlangs would be a better selection for a fit. Indeed, in the appendix of Asmussen (2000) the author discusses how in general, for a mixture of Erlangs, the Phase Type which has an equally good fit has much higher dimension. Unless the modeler is confident in the underlying structure of the data being of the appropriate Markov Chain, it is best to use a mixture of Erlangs both in terms of computing resources and fit.

All the code required to reproduce the examples seen here is publicly available on my GitHub account at the link in the references under Riis-Due (2024).

References

Asmussen, S. (2000). *Ruin Probabilities*. World Scientific Advanced Series on Statistical Science and Applied Probability.

Bladt, M. (2005). A review on phase-type distributions and their use in risk theory. *ASTIN Bulletin*, pages Vol. 35, No. 1, 145–161.

Esther Frostig, Susan Pitts, K. P. (2012). The time to ruin and number of claims until ruin for phase-type claims. *Insurance Math Economics*.

Neuts, M. (1975). Probability distributions of phase type. *Liber amicorum Professor emeritus dr. H. Florin*, pages 173–206.

Riis-Due, A. (2024). https://github.com/austinriisdue/actsc966.

Soren Asmussen, Olle Nerman, M. O. (1996). Fitting phase-type distributions via the em algorithm. *Scandanavian Journal of Statistics*, pages Vol. 23, No. 4 419–441.

Stanford, D. and Stroinski, K. (1994). Recursive methods for computing finite-time ruin probabilities for phase-distributed claim sizes. *ASTIN Bulletin*, pages Vol. 24, No. 2, 235–254.

Wenyong Gui, Rongtan Huang, X. S. L. (2018). Fitting the erlang mixture model to data via a gem-cmm algorithm. *Journal of Computational and Applied Mathematics*, pages Vol. 343, No. 4, 189–205.