Computational Statistics
https://doi.org/10.1007/s00180-024-01491-4

ORIGINAL PAPER

A simple algorithm for computing the probabilities of count models based on pure birth processes

Mongkol Hunkrajok[1] · **Wanrudee Skulpakdee**[2]

Received: 18 March 2023 / Accepted: 15 March 2024

Abstract

Recently, non-monotonic rate sequences of pure birth processes have been the focus of much attention in the analysis of count data due to their ability to provide a combination of over-, under-, and equidispersed distributions without the need to reuse covariates (traditional methods). They also permit the modeling of excess counts, a frequent issue arising when using count models based on monotonic rate sequences such as the Poisson, gamma, Weibull, Conway-Maxwell-Poisson (CMP), Faddy (1997), etc. Matrix-exponential approaches have always been used for computing the probabilities for count models based on pure birth processes, although none have been proposed for them as a specific algorithm. It is intractable to calculate these pure birth probabilities numerically in an analytic form because severe numerical cancellations may occur. However, we circumvent this difficulty by exploiting a Taylor series expansion, and then a new analytic form is derived. We developed a simple algorithm for efficiently implementing the new formula and conducted numerical experiments to study the efficiency and accuracy of the developed algorithm. The results indicate that this new approach is faster and more accurate than the matrix-exponential methods.

Keywords Poisson distribution · Conway-Maxwell-Poisson distribution · Matrix exponential · Uniformization · `expm` · Complete homogeneous symmetric polynomials

✉ Wanrudee Skulpakdee
wanrudee.sku@nida.ac.th

Mongkol Hunkrajok
hunkrajokmongkol@gmail.com

[1] Independent Researcher, Bangkok, Thailand

[2] Graduate School of Applied Statistics, National Institute of Development Administration, Bangkok, Thailand

Published online: 10 April 2024

1 Introduction

One of the most commonly used regression models for count data is the Poisson regression model. The Poisson distribution assumes that the interarrival times are independent and identically distributed exponential random variables. It leads to the restrictive assumption of equidispersion, where the variance is equal to the mean. In particular, the Poisson model has only a single parameter λ, and covariates can be incorporated into this model by setting λ as a function of the linear predictor $\beta_0 + \beta_1 x_{j1} + ... + \beta_r x_{jr}$, where x_{jk}, $k = 1, ..., r$, is the jth observation of the kth covariate, and β_l, $l = 0, ..., r$, is the lth unknown parameter to be estimated. In real applications, this regression model is usually inadequate because the conditional variance of count data frequently exceeds the conditional mean (overdispersion). The reverse can also occur (underdispersion). The combination of equidispersed (Poisson) distributions is always overdispersed. Therefore, it is clear that it is important to avoid having to fit a Poisson regression model to either underdispersed or equidispersed count data.

Many researchers seeking a more flexible regression model for count data that can accommodate over-, under-, and equidispersion have arrived at several alternative regression models. These regression models are, for example, the gamma count regression model (Winkelmann 1995), the Weibull count regression model (McShane et al. 2008), and the Conway-Maxwell-Poisson (CMP) count regression model (Sellers and Shmueli 2010). These three well-known count regression models have greater flexibility than the Poisson regression model at the cost of an additional (dispersion) parameter α. Whenever $\alpha \neq 1$, the equality restriction between the conditional variance and the conditional mean is removed. However, the gamma, Weibull, and CMP regression models with a constant dispersion parameter α cannot be fitted to a count dataset intermingled with overdispersed and underdispersed distributions. Many statisticians and mathematicians have acknowledged and addressed this issue in the same way.

The widespread technique for solving the above problem uses covariates twice, once with mean (or λ) and once with variance (or α) (e.g., Guikema and Goffelt (2008), Sellers and Shmueli (2009), Saez-Castillo and Conde-Sanchez (2013), Barreto-Souza and Simas (2016), Chanialidis et al. (2018), Forthmann et al. (2020), Morales-Otero and Núñez-Antón (2021), Bourguignon and de Medeiros (2022), Philipson and Huang (2023)). The above three count models incorporating this technique may not be parsimonious in typical applications. The number of parameters to be estimated is often doubled but not providing a substantial increase in log-likelihood. Skulpakdee and Hunkrajok (2022) offered a careful analysis of these three count models. It reveals that these models require the above technique because the rates of interarrival times are equal. Therefore, they presented a new class of parametric models for count data based on pure birth processes with non-monotonic (unequal) rates named unusual-event (UE) models. If a UE count dataset is created as a mixture of over-, under-, and equidispersed distributions, then these UE regression models provide a parsimonious model without reusing the covariates.

Table 1 shows the results of fitting two groups of count regression models to two datasets from Kharrat et al. (2019) and Peluso et al. (2019): the fertility data and the ideal fertility data. The fertility data contain information about a sample size of 1,243 women over 44 years old in 1985 and 10 explanatory variables (see Winkelmann (1995), McShane et al. (2008), Chanialidis et al. (2018), and Kharrat et al. (2019) for more information). The variance and mean numbers of children in this sample are 2.328 and 2.384, respectively. There is slightly underdispersed with the variance-mean ratio equalling $2.328/2.384 = 0.977$. The ideal fertility data is a larger sample of 5,906 young Mexican women between 15 and 17 years old with 50 explanatory variables (see Peluso et al. (2019) for more information). There is a more underdispersion than in the first dataset, with a sample variance of 2.01, which is 0.86 times the mean. These two datasets also exhibit a similar feature of an unusual excess of two counts (see Figure 5 in Winkelmann (1995) and Figure 4 in Peluso et al. (2019)).

The first group consists of the Faddy (1997), Weibull, CMP, and generalized additive discrete Weibull (DW) (Peluso et al. 2019) models with varying dispersion using covariates twice, and the single-unusual-event (SUE) and UE models with varying rates using covariates once belong to the second. The significantly higher likelihoods and the much smaller numbers of parameters in Table 1 for the SUE and UE models provide strong evidence that varying rates mainly cause conditional over-, under-, and equidispersed distributions in the datasets. It indicates that the fertility and ideal fertility data have an excess of two counts due to non-monotonic rate sequences. The rate sequences of the SUE models are $\lambda, \lambda, 0.52\lambda, \lambda, \lambda, \ldots$ and $\lambda, \lambda, 0.31\lambda, \lambda, \lambda, \ldots$ for the fertility and ideal fertility data, respectively. The rate between the second and third events is considerably less than other rates. In other words, the third event is unusual. These count models based on non-monotonic rate sequences can easily capture the women's behavior in these two datasets.

In addition, in the last column in Table 1, the results of the Poisson, CMP, and DW regression models are taken directly from Table 2 of Peluso et al. (2019).

Table 1 Log-likelihood, AIC (smaller is better), and BIC (smaller is better) values of several count regression models for the fertility and ideal fertility data

	Fertility data				Ideal fertility data			
Model	#	-Log-L	AIC	BIC	#	-Log-L	AIC	BIC
Poisson	11	2101.8	4225.6	4282.0	51	9838.8	19779.6	20120.5
Group 1								
Faddy (1997)	23	2052.2	4150.3	4268.2	–	–	–	–
Weibull	22	2051.6	4147.2	4259.9	–	–	–	–
CMP	22	2049.8	4143.6	4256.4	112	9475.6	19175.2	19923.8
DW	–	–	–	–	114	9561.7	19351.5	20113.4
Group 2								
SUE	12	2048.8	4121.5	4183.1	52	8982.4	18068.7	18416.3
UE	13	2040.1	4106.2	4172.9	57	8472.0	17057.9	17438.9

The symbol # stands for the number of parameters in the models

In the second column in Table 1, the results of the Poisson, Weibull, CMP, and Faddy (1997) models are computed using the **stats** (R Core Team 2022), **Countr** (Kharrat and Boshnakov 2022), **COMPoissonReg** (Sellers et al. 2019), and **CountsEPPM** (Smith and Faddy 2018) R packages, respectively. We do not report the results of fitting the gamma count model since the **Countr** package could not achieve convergence. Note that we only use the algorithm described in Sect. 2.3 to calculate the likelihood function for SUE and UE regression models reported in Table 1.

Recently, Skulpakdee and Hunkrajok (2022a) proposed a new INteger-valued Autoregressive Conditional Heteroskedastic (INARCH) model with varying rates and applied it to the time series of strike counts first analyzed by Jung et al. (2005) and re-analyzed by Weiß (2018), Huang and Zhu (2021), and Weiß et al. (2022). Again, the much significantly higher likelihood in Table 2 of Skulpakdee and Hunkrajok (2022a) for this new INARCH model provide strong evidence that varying rates mainly caused over-, under-, and equidispersion are apparent from this fitted model. The several results above help to provide a comprehensive picture of the current understanding of count data.

Count models based on pure birth processes are powerful and important because they can represent a variety of shapes, particularly for empirical data with excess counts. However, there is an important limitation to their applicability. That is, we do not have a fast, accurate, and easily-programmed algorithm to compute the probabilities of these flexible models.

Let $X(t)$ be a discrete random variable representing the total number of events that occur by time t. $\{X(t); t \geq 0\}$ is a pure birth process with $X(0) = 0$ and birth rates λ_n for $n = 0, 1, 2, \ldots$. A problem that arises in pure birth processes is that of computing the probabilities $P_n(t) = P\{X(t) = n \mid X(0) = 0\}$ that satisfy the Chapman-Kolmogorov forward differential equations (Cox and Miller 1965), namely

$$\begin{aligned} P_0'(t) &= -\lambda_0 P_0(t), \\ P_n'(t) &= -\lambda_n P_n(t) + \lambda_{n-1} P_{n-1}(t), \quad n > 0, \end{aligned} \tag{1}$$

with boundary conditions $P_0(0) = 1$ and $P_n(0) = 0,\ n > 0$. The solution of (1) can be written as a matrix-exponential function (Cox and Miller 1965)

$$P_n(t) = (1\ 0\ \ldots\ 0) e^{\boldsymbol{Q}t} (0\ 0\ \ldots\ 1)^T, \tag{2}$$

where $\boldsymbol{Q}$ is the matrix of birth rates

$$\boldsymbol{Q} = \begin{bmatrix} -\lambda_0 & \lambda_0 & 0 & \cdots & 0 \\ 0 & -\lambda_1 & \lambda_1 & \cdots & 0 \\ 0 & 0 & -\lambda_2 & \ddots & 0 \\ \vdots & \vdots & \vdots & \ddots & \lambda_{n-1} \\ 0 & 0 & 0 & \cdots & -\lambda_n \end{bmatrix},$$

and an analytic function (Bartlett 1978, Renshaw 2011, and Crawford et al. 2018)

$$P_n(t) = \left(\prod_{i=0}^{n-1} \lambda_i\right) \sum_{j=0}^{n} \left(\frac{e^{-\lambda_j t}}{\prod_{\substack{k=0 \\ k \neq j}}^{n} (\lambda_k - \lambda_j)} \right) \quad \text{for } n > 0. \tag{3}$$

There has been a long history of studying pure birth count models. Faddy (1994) conjectured that over- and underdispersion hold for increasing and decreasing rate sequences, respectively. Ball (1995) proved this conjecture. Faddy (1997) described a class of pure birth count models with $\lambda_n = \lambda(b+n)^c$ where $c < 0$, $0 < c \leqslant 1$, and $c = 0$ result in under-, over-, and equidispersion, respectively. This rate sequence with $b = 1$ also appears on page 136 of Conway and Maxwell (1962). He commented on the inability to compute the probabilities using Eq. (3). He also pointed out that there are possibilities for applying ordinary differential equation procedures to calculate the probabilities from Eq. (1). Podlich et al. (2004) proposed semi-parameter extended Poisson process models for count data and again stated that the analytic equation (3) is unsuitable for numerical computation. They suggested Eq. (2) for small to moderate counts because numerical techniques exist for computing matrix exponentials (Moler and Van Loan 2003). For large counts, the probabilities are computed by saddlepoint approximations (Daniels 1982; Smyth and Podlich 2002).

Smyth and Podlich (2002) stated that matrix-exponential approaches generally return all probability values, although only the last probability value is required. They may not preserve tiny probability values to good relative accuracy because the larger probability values usually dominate error bounds. Thus, these methods do not guarantee the accuracy of all probabilities in all cases. Our experiments in Sect. 3.2 confirm this statement (see Table 3).

Much research has been conducted on extending the basic Poisson (pure birth) count model (e.g., Faddy and Smith (2008, 2011, 2012)). To the best of our knowledge, Eq. (2) has been used extensively in computing the probabilities of pure birth count models. MATLAB (2017) provides a routine `expm` for computing matrix exponentials. R (R Core Team 2022) has a matrix exponential routine `expm` in the **expm** (Goulet et al. 2022) R package used for calculating Eq. (2) in the **CountsEPPM** (Smith and Faddy 2016, 2018) and **BinaryEPPM** (Smith and Faddy 2019a, 2019) R packages.

Equation (3) is computationally intractable (Crawford et al. 2018, p. 13) when some λ_i for $i = 0, ..., n$ are nearly equal, that is, a significant rate sequence for maximum likelihood fitting of parametric pure birth count models. Moreover, Eq. (3) can occasionally break down since the computation of $P_n(t)$ requires division by $\prod_{k=0, k\neq j}^{n}(\lambda_k - \lambda_j)$, which can be zero. Thus, it provides an impractical way to calculate the log-likelihood function for this distribution. This form is not well-behaved in the presence of errors because of the threat of sudden loss of accuracy through cancellation. It is well known from principles of statistical inference that $0 \leqslant P_n(t) \leqslant 1$. The product of λ_i for $i = 0, ..., n$ may be large, so the sum of the positive and negative terms must be close to zero (i.e., catastrophic cancellation). This effect is potentially quite serious and can be disastrous.

We give an example of a relatively severe case of this cancellation. Let $fl(P_n(t))$ be a floating point number obtained by computing $P_n(t)$ in floating point arithmetic.

Given the relative roundoff error in computing $P_2(t)$, this example shows how cancellation leads to inaccurate results. A computer actually calculates

$$fl(P_2(t)) = \lambda_0\lambda_1 \sum_{j=0}^{2} \left(\frac{e^{-\lambda_j t}}{\prod_{\substack{k=0 \\ k\neq j}}^{2} (\lambda_k - \lambda_j)} \right) \left(1 + \Delta_0 + a_1(1-a_2)\Delta_1 + a_2\Delta_2\right), \quad (4)$$

where

$$a_1 = \left(1 - \frac{(\lambda_2 - \lambda_1)}{(\lambda_2 - \lambda_0)} e^{(\lambda_1-\lambda_0)t}\right)^{-1},$$

$$a_2 = \left(1 - \frac{(\lambda_1 - \lambda_2)}{(\lambda_1 - \lambda_0)} e^{(\lambda_2-\lambda_0)t} - \frac{(\lambda_0 - \lambda_2)}{(\lambda_0 - \lambda_1)} e^{(\lambda_2-\lambda_1)t}\right)^{-1},$$

and $|\Delta_l| \ll 1$ for $l = 0, 1, 2$. The relative roundoff error cannot be bounded because a_1 and a_2 have a denominator that can be close to zero. Thus, the obvious way of computing the sum in (3) may result in a large relative error. The final calculated answer may be very inaccurate when multiplied by $\prod_{i=0}^{n-1} \lambda_i$.

Let us see what happens when we set $a_2 = 1$ (i.e. $a_1 = 1/0$), so that $\lambda_2 - \lambda_1 = (\lambda_1 - \lambda_0)/(e^{(\lambda_1-\lambda_0)} - 1)$. Then

$$a_1(1-a_2) = \frac{e^{-\left(\frac{(\lambda_1-\lambda_0)e^{(\lambda_1-\lambda_0)}}{1-e^{(\lambda_1-\lambda_0)}}\right)}}{1 - e^{(\lambda_1-\lambda_0)}}, \quad (5)$$

where $\lim_{\lambda_1-\lambda_0\to 0^+} a_1(1-a_2) = -\infty$, $\lim_{\lambda_1-\lambda_0\to 0^-} a_1(1-a_2) = \infty$, $\lim_{\lambda_1-\lambda_0\to\infty} a_1(1-a_2) = -1$, and $\lim_{\lambda_1-\lambda_0\to-\infty} a_1(1-a_2) = 1$. Any desired accuracy of (4) (limited only by roundoff errors) can be obtained by increasing enough $|\lambda_1 - \lambda_0|$. P_{eq4} ($\lambda_1 - \lambda_0 = 1$) of the first line in Table 2 agrees with the true solution to sixteen decimal places ($\eta = 2.18 \times 10^{-16}$) using IEEE double-precision machine arithmetic, although $a_1 = -3.65 \times 10^{15}$, which is huge. In contrast, the relative error in P_{eq4} with $\lambda_1 - \lambda_0 = 2^{-50}$ is large (see the third line in Table 2).

Since Δ_1 and Δ_2 may be equal in magnitude but opposite in sign, the differences among λ_0, λ_1 and λ_2 may satisfy Eq. (4) for $|a_1(1-a_2)| + |a_2| = 100$, which we can expect to compute $P_2(1)$ with no less than fourteen correct digits (see the fifth and sixth lines in Table 2). $\lambda_1 - \lambda_0$ and $\lambda_2 - \lambda_1$ of the fifth and sixth rate sequences equal 0.011, but $\lambda_2 - \lambda_1$ and $\lambda_1 - \lambda_0$ of them are significantly different, where $\lambda_2 - \lambda_1 = 10.474$ and $\lambda_1 - \lambda_0 = 1.8017$. The last two lines fail because λ_1 and λ_2 are too close to λ_0, resulting in very large negative probability values. Cancellation occurs when computing a_1, a_2, and $a_1(1-a_2)$, which we can compute more accurately with the help of a `mpfr` function in the **Rmpfr** (Maechler et al. 2024) R package.

This article aims to show that we can use a relatively straightforward Taylor series expansion to put the above problem into an infinite non-alternating series

Table 2 Some rate sequences for computing $P_2(1)$ with Eq. (4) using IEEE double precision (P_{eq4}) and 333-bit (100-digit) precision (P_2) arithmetic

λ_0	λ_1	λ_2	P_{eq4}	P_2	η	a_1	a_2	$a_1(1-a_2)$
2	3	$\lambda_1 + \frac{1}{e-1}$	0.1813	0.1813	2.18e−16	− 3.65e15	≈ 1	−2.83
2	1	$\lambda_1 - \frac{1}{e^{-1}-1}$	0.1643	0.1643	7.64e−17	3.65e15	≈ 1	2.83
2	$2+2^{-50}$	$\lambda_1 + \frac{2^{-50}}{e^{2^{-50}}-1}$	0.3241	0.1991	6.28e−1	− 2.54e30	≈ 1	− 3.06e15
2	$2-2^{-49}$	$\lambda_1 - \frac{2^{-49}}{e^{-2^{-49}}-1}$	0.1366	0.1991	3.14e−1	− 6.34e29	≈ 1	1.53e15
2	2.0110	12.4850	0.0467	0.0467	2.15e−14	≈ -100	2.96e−6	≈ -100
2	3.8017	3.8127	0.1699	0.1699	1.23e−14	1.04	49.57	−50.42
2	3	$2+2^{-51}$	0.7500	0.2987	1.51	1.63e−16	-6.12e15	≈ 1
2	$2+2^{-40}$	$2-2^{-40}$	− 7e7	0.2707	2.48e8	≈ -1	1.21e24	1.21e24
2	$2+2^{-50}$	$2-2^{-50}$	− 7e13	0.2707	2.60e14	≈ -1	1.27e30	1.27e30

The first four and last two rate sequences produce enormous a_1 and a_2 values, so Eq. (4) does not guarantee an accurate result (large relative errors). The symbol η represents the relative error $|(P_{eq4} - P_2)/P_2|$.

form that allows accurate and efficient computation of the probabilities of count models based on pure birth processes. We can simply sum this series until adding another term does not alter the numbers stored in the computer because there is no severely affected by arithmetic cancellation.

The remainder of this paper is laid out as follows. In Sect. 2, we reformulate (3) in which $\lambda_k - \lambda_j$ does not appear in the denominator since the severe difficulty comes when $\lambda_k - \lambda_j$ is small but not negligible. With a suitable parameter value, the new infinite series ensures that cancellation does not occur in the computation of $P_n(t)$. We provide a simple algorithm that should be useful for any of the pure birth count models currently being utilized to analyze count data. Section 3 presents numerical experiments evaluating the accuracy and efficiency of `expm` and our algorithm with a variety of pure birth count models. Finally, Sect. 4 concludes the paper.

2 Computing $P_n(t)$ without matrix operations

In this section, we introduce a new infinite series formula for pure birth probabilities and present its practical algorithm. An advantage of this new formula is that we can scale its terms by choosing a suitable parameter value to guarantee without checking that no cancellations occur.

2.1 Reformulating the original formula

The terms in the expression (3) alternate in sign, leading to a sudden loss of relative accuracy. Thus, this analytic form is useless for computing the probabilities of pure birth count models. For nearly equal λ_i for $i = 0, ..., n$, the individual terms of the summation in (3) become much larger than their sum. The product in the denominators is

very small, and the summation is close to zero. Thus, all accuracy may be lost. If no cancellations occur during the computation of $P_n(t)$, the result will generally be accurate (Watkins 2010, p. 145). The Taylor series expansion of exponential functions is a simple and effective remedy for this problem. Before using this remedy, for simplicity, the form (3) needs to be rewritten with $\lambda \in \mathbb{R}$ as follows:

$$\begin{aligned} P_n(t) &= \left(\prod_{i=0}^{n-1}\lambda_i\right)e^{-\lambda t}\left(\sum_{j=0}^{n-1}\left(\frac{e^{(\lambda-\lambda_j)t}}{\prod_{\substack{k=0\\k\neq j}}^{n}(\lambda_k-\lambda_j)}\right)+\frac{e^{(\lambda-\lambda_n)t}}{\prod_{k=0}^{n-1}(\lambda_k-\lambda_n)}\right) \\ &= \left(\prod_{i=0}^{n-1}\lambda_i\right)e^{-\lambda t}\left(\sum_{j=0}^{n-1}\left(\frac{e^{(\lambda-\lambda_j)t}}{\prod_{\substack{k=0\\k\neq j}}^{n}(\lambda_k-\lambda_j)}\right)-\sum_{j=0}^{n-1}\left(\frac{e^{(\lambda-\lambda_n)t}}{\prod_{\substack{k=0\\k\neq j}}^{n}(\lambda_k-\lambda_j)}\right)\right) \\ &= \left(\prod_{i=0}^{n-1}\lambda_i\right)e^{-\lambda t}\sum_{j=0}^{n-1}\left(\frac{e^{(\lambda-\lambda_j)t}-e^{(\lambda-\lambda_n)t}}{\prod_{\substack{k=0\\k\neq j}}^{n}(\lambda_k-\lambda_j)}\right). \end{aligned} \tag{6}$$

The expression (6) is not completely satisfactory because the evaluation of this revised formula still involves delicate cancellations between successive terms, which are alternately positive and negative. Hence, the cancellation problem can prevent the accurate computation of the formula. We ensure that cancellation and division by zero do not occur in computing $P_n(t)$ by inserting the Taylor series expansion of $e^{(\lambda-\lambda_j)t}$ into (6) and choosing a suitable λ, such as $\lambda = max\,\lambda_i$ for $i = 0, ..., n$. Then, this equation can be written in the most practical form:

$$P_n(t) = \frac{\prod_{i=0}^{n-1}\lambda_i t^n e^{-\lambda t}}{n!}\left(1+\sum_{d=1}^{\infty}\frac{h_d^{(n)}t^d}{\prod_{k=1}^{d}(n+k)}\right), \tag{7}$$

where

$$h_d^{(n)} = h_d(\lambda-\lambda_0, ..., \lambda-\lambda_n) = \begin{cases} 1 & \text{for } d=0 \\ \sum_{k_d=0}^{n}\sum_{k_{d-1}=0}^{k_d}\cdots\sum_{k_1=0}^{k_2}\prod_{j=k_1}^{k_d}(\lambda-\lambda_j) & \text{for } d>0 \end{cases} \tag{8}$$

is the complete homogeneous symmetric polynomial with $n+1$ variables $\lambda-\lambda_j$ for $j = 0, ..., n$, and λ is a cancellation variable. Since

$$\sum_{k_d=0}^{n}\sum_{k_{d-1}=0}^{k_d}\cdots\sum_{k_1=0}^{k_2}\prod_{j=k_1}^{k_d}(\lambda-\lambda_j) = \sum_{d_0+\ldots+d_n=d}\prod_{j=0}^{n}(\lambda-\lambda_j)^{d_j}, \tag{9}$$

where $d_j \geqslant 0$, $h_d^{(n)}$ can also be written as

$$h_d^{(n)} = \begin{cases} 1 & \text{for } d = 0 \\ \sum\limits_{d_0+\ldots+d_n=d} h_{d_0}(\lambda-\lambda_0)h_{d_1}(\lambda-\lambda_1)\ldots h_{d_n}(\lambda-\lambda_n) & \text{for } d > 0. \end{cases} \tag{10}$$

Eqs. (3) and (7) are not equivalent numerically. The first is prone to cancellation when the differences $\lambda_k - \lambda_j$ are small; that is an essential consideration for fitting pure birth count models. We can use the second safely whenever we choose the λ's value carefully; therefore, it is preferred. For all even degrees $d \geqslant 2$, $h_d^{(n)}$ is always non-negative (Rovenţa and Temereancă 2019, Theorem 2.3), and for all odd degrees $d \geqslant 1$, $h_d^{(n)}$ is not always non-negative since $h_d(-(\lambda-\lambda_0),\ldots,-(\lambda-\lambda_n)) = -h_d(\lambda-\lambda_0,\ldots,\lambda-\lambda_n)$. This coefficient $h_d^{(n)}$ can also be viewed as a multinomial summation $\left((\lambda-\lambda_0)+\ldots+(\lambda-\lambda_n)\right)^d$ whose coefficients are equal to one. The series in Eq. (7) is convergent for every λ_i, t and n because $0 \leqslant P_n(t) \leqslant 1$. In fact, by the ratio test,

$$\left|\frac{h_{d+1}^{(n)}}{h_d^{(n)}}\frac{t}{d+n+1}\right| \leqslant \frac{\sum_{i=0}^n(\lambda-\lambda_i)t}{d+n+1} < \frac{\sum_{i=0}^n(\lambda-\lambda_i)t}{N+n+1} < 1 \tag{11}$$

for any fixed λ_i, t, n, $\lambda \geqslant max\,\lambda_i$, and every d greater than some N. Hence the series $|h_0^{(n)}| + |h_1^{(n)}t/(n+1)| + |h_2^{(n)}t^2/(n+1)(n+2)| + \ldots$ with $\lambda < max\lambda_i$ converges by comparison with $h_0^{(n)} + h_1^{(n)}t/(n+1) + h_2^{(n)}t^2/(n+1)(n+2) + \ldots$ with $\lambda \geqslant max\,\lambda_i$, so that the series $h_0^{(n)} + h_1^{(n)}t/(n+1) + h_2^{(n)}t^2/(n+1)(n+2) + \ldots$ with $\lambda < max\lambda_i$ converges absolutely.

The formula (7) turns out to be the first row of $e^{-\lambda t}e^{\boldsymbol{P}t}$, where $\boldsymbol{P} = \boldsymbol{Q} + \lambda\boldsymbol{I}$, $\lambda = max\,\lambda_i$, and $\boldsymbol{I}$ is the $n+1 \times n+1$ identity matrix. This procedure is known as uniformization, originally introduced by Jensen (1953). If $\lambda \geqslant max\,\lambda_i$, then $\left(\lambda-\lambda_0\right)+\ldots+\left(\lambda-\lambda_n\right) \geqslant 0$ implies $h_d^{(n)} \geqslant 0$ for all d. If $\lambda = \lambda_i = max\lambda_i$ (i.e, $h_d^{(n)} = 0$ for $d \geqslant 1$), the expression (7) reduces to the Poisson distribution. If $\lambda < max\lambda_i$, $\left(\lambda-\lambda_0\right)+\ldots+\left(\lambda-\lambda_n\right) \geqslant 0$ does not necessarily mean $h_d^{(n)} \geqslant 0$ for all d, in contrast with the multinomial expansion in which its sum is always greater than or equal to zero. Therefore, $h_d^{(n)}$ are no longer non-negative but real. Since the formula (7) let us scale $h_d^{(n)}$ via the choice of λ to guarantee without checking that no cancellations occur, λ is called the cancellation variable. In other words, as long as $h_d^{(n)} \geqslant 0$ is satisfied, we are assured that the formula (7) has only non-negative terms. Thus, the cancellation problem in (7) is just a consequence of poor choosing λ.

The formula (7) plays a significant role in computing the probabilities for count models based on pure birth processes because the differences among λ_i are quite small. They are equal to zero if the count model is Poisson. We can put the formula (7) in a form closer to a more traditional probability distribution, i.e., the Poisson distribution $P_n(t) = \lambda^n t^n e^{-\lambda t}/n!$. To get it, all we have to do is to

choose λ in (7) with $\sum_{d=1}^{\infty} h_d^{(n)} t^d / \prod_{k=1}^{d}(n+k) = 0$. Thus, we obtain from (7), i.e., $P_n(t) = \prod_{i=0}^{n-1} \lambda_i t^n e^{-\lambda t}/n!$. It may be challenging to determine this value of λ since, in general, it is difficult to find the roots of an infinite series. It is also easy to see that the infinite series in (7) and the finite series in (3) and (6) are invariant under the permutation of λ_i for $i = 0, ..., n$. It suggests that $\boldsymbol{Q}$'s main-diagonal and first super-diagonal entries in (2) can take on any sequence of λ_i for $i = 0, ..., n-1$ whenever $\boldsymbol{Q}$ has $-\lambda_n$ in position $(n+1, n+1)$. The computation of (7) is done without worry about the potential loss of accuracy due to cancellation when the infinite series has only non-negative terms. Therefore, the accuracy of (7) refers primarily to the error introduced by truncating the non-alternating series. It is one component, but not the only, of the accuracy of the computed probabilities. If λ is close to λ_i for $i = 0, ..., n$, the differences $\lambda - \lambda_i$ are small. That is, this pure birth process is close to a Poisson process. Hence the infinite series converges quite rapidly. When the formula (7) with small $\lambda - \lambda_i$ is properly implemented, the result is one of the most effective we know.

2.2 An upper bound on a relative error for $\lambda = max\, \lambda_i$

Let $P_n^N(t)$ be the Nth partial sum and $R_n^N(t)$ be the remainder of $P_n(t)$. Then we have

$$\begin{aligned} P_n(t) &= P_n^N(t) + R_n^N(t) \\ &= \frac{\prod_{i=0}^{n-1} \lambda_i t^n e^{-\lambda t}}{n!} \left[\left(1 + \sum_{d=1}^{N} \frac{h_d^{(n)} t^d}{\prod_{k=1}^{d}(n+k)} \right) + \sum_{d=N+1}^{\infty} \frac{h_d^{(n)} t^d}{\prod_{k=1}^{d}(n+k)} \right] \end{aligned} \tag{12}$$

for $n = 1, 2, ...$ If λ_i are all equal, and $\lambda = max\, \lambda_i$ for $i = 0, 1, ..., n$, then $P_n^N(t) = \lambda^n t^n e^{-\lambda t}/n!$, and $R_n^N(t) = 0$, i.e., the Poisson distribution. Before we derive the relative error in the approximation $P_n(t)$ by $P_n^N(t)$, it is helpful to note the following simple but important facts. We begin by recalling that if λ_i are not all equal, and $\lambda - \lambda_i \geqslant 0$ for $i = 0, 1, ..., n$ (i.e., $\lambda \geqslant max\, \lambda_i$), then

$$\frac{h_{d+2}^{(n)}}{h_{d+1}^{(n)}} \leqslant \frac{h_{d+1}^{(n)}}{h_d^{(n)}} \tag{13}$$

for $d \geqslant 0$, and $n \geqslant 1$ (see the proof of Lemma 10 in Banks and Martin (2013)); that is, the ratio $h_{d+1}^{(n)}/h_d^{(n)}$ does not increase with increasing d. It is easy to see that the upper bound of (13) is

$$\frac{h_1^{(n)}}{h_0^{(n)}} = \sum_{i=0}^{n} \left(\lambda - \lambda_i \right), \tag{14}$$

where $h_0^{(n)} = 1$, and $h_1^{(n)} = \sum_{i=0}^{n} \left(\lambda - \lambda_i \right)$.

The fact (13) tells us immediately that the ratio $h_{d+1}^{(n)} t/(h_d^{(n)}(d+n+1))$ of $P_n(t)$ decreases with increasing d; thus, $P_n(t)$ converges by the ratio test. The series needs

to be truncated in practical computations for $P_n(t)$ because it is impossible to carry out an infinite number of additions (Cheney and Kincaid 2004). The truncated series $P_n^N(t)$ should approximate $P_n(t)$. Here, we wish to know whether an upper bound exists on the relative error when we truncate $P_n(t)$ with $N+1$ summands. Since the infinite series $P_n(t)$ is convergent, its sum is the limit of the sequence of the partial sum $P_n^N(t)$, that is, $P_n(t) = \lim_{N\to\infty} P_n^N(t)$. It means that by adding sufficiently many terms of the convergent series $P_n(t)$, we can obtain the sum of the series as close as we please. In applications, the approximation $P_n^N(t)$ of $P_n(t)$ is simply obtained by summing the series until the desired accuracy is obtained by increasing enough d. This is explained by the fact that the series $R_n^N(t)$ converges faster than a geometric series (Minka et al. 2003).

The relative roundoff error in computing $P_n^N(t)$ needs to be estimated to analyze the effects of the truncation. The approximation $P_n^N(t) \approx fl\left(P_n^N(t)\right) = P_n^N(t)(1+\epsilon)$ obtained by ignoring the relative error (i.e., $\epsilon \ll 1$) is good if λ satisfies $h_d^{(n)} > 0$ for $d = 0, 1, ..., N$ because no cancellations occur during the computation of $P_n^N(t)$. Thus,

$$\begin{aligned}\frac{R_n^N(t)}{P_n^N(t)} &= \frac{\prod_{i=0}^{n-1}\lambda_i t^n e^{-\lambda t}}{P_n^N(t)\ n!}\left(\frac{h_{N+1}^{(n)} t^{N+1}}{\prod_{k=1}^{N+1}(n+k)} + \frac{h_{N+2}^{(n)} t^{N+2}}{\prod_{k=1}^{N+2}(n+k)} + ...\right)\\ &< \frac{\prod_{i=0}^{n-1}\lambda_i e^{-\lambda t} h_N^{(n)} t^{n+N}}{P_n^N(t)\ (n+N)!}\left(\frac{h_{N+1}^{(n)}}{h_N^{(n)}}\frac{t}{n+N+1} + \left(\frac{h_{N+1}^{(n)}}{h_N^{(n)}}\frac{t}{n+N+1}\right)^2 + ...\right).\end{aligned} \tag{15}$$

Therefore, for some N, the relative error can be bounded by

$$\frac{R_n^N(t)}{P_n^N(t)} < \frac{u}{1-r}, \tag{16}$$

where

$$u = \frac{\prod_{i=0}^{n-1}\lambda_i e^{-\lambda t} h_{N+1}^{(n)} t^{n+N+1}}{P_n^N(t)\ (n+N+1)!} = \frac{1}{\left(1 + \sum_{d=1}^{N}\frac{h_d^{(n)} t^d}{\prod_{k=1}^{d}(n+k)}\right)}\frac{h_{N+1}^{(n)} t^{N+1}}{\prod_{k=1}^{N+1}(n+k)}, \tag{17}$$

and

$$r = \frac{h_{N+1}^{(n)}}{h_N^{(n)}}\frac{t}{n+N+1} < 1. \tag{18}$$

From this simple inequality (16), we see that if r is close to zero and $u \ll 1$, $P_n^N(t)$ is an extremely accurate approximation to $P_n(t)$.

Algorithm 1 Given $(\lambda_0, ..., \lambda_n) \in \mathbb{R}^{n+1}$, an iteration limit d_{max}, and a tolerance ϵ, this algorithm computes the probability $P_n(t)$ at time $t = 1$. C_{old} is stored C_n of the previous iteration, and s is the dth partial sum of $P_n(1)$. F is a flag that indicates whether or not the specified tolerance ϵ was achieved by d_{max} iterations.

```
1: if n = 0 then
2:     p ⇐ e^{-λ_0}
3: else
4:     λ ⇐ max λ_i, α ⇐ −λ, γ ⇐ λ_0, β ⇐ 1 × 10^300
5:     for i = 2, ..., n do
6:         φ ⇐ γ · λ_{i−1}/i
7:         if φ > β then
8:             α ⇐ α + ln γ, γ ⇐ λ_{i−1}/i
9:         else
10:            γ ⇐ φ
11:        end if
12:    end for
13:    α ⇐ α + ln γ
14:    s ⇐ 0, F ⇐ 1, d ⇐ n, d_max ⇐ d_max + n
15:    C_i ⇐ 1, λ_i ⇐ λ − λ_i for i = 0, ..., n
16:    while F = 1 and d ⩽ d_max do
17:        d ⇐ d + 1, s ⇐ s + C_n, C_old ⇐ C_n
18:        C_0 ⇐ λ_0 · C_0/d
19:        for i = 1, ..., n do
20:            C_i ⇐ λ_i · C_i/d + C_{i−1}
21:        end for
22:        r ⇐ C_n/C_old
23:        if r < 1 then
24:            u ⇐ C_n/s
25:            if u/(1 − r) < ϵ then
26:                F ⇐ 0
27:            end if
28:        end if
29:    end while
30:    p ⇐ e^{α+ln s}
31: end if
```

2.3 An iterative algorithm to compute $P_n(1)$ for $\lambda = max\,\lambda_i$

In implementing our technique, the terms $h_d^{(n)} / \prod_{k=1}^{d}(n+k)$ of $P_n(1)$ in Eq. (7) may seem a bit daunting initially, but they can be coded recursively without difficulty. We do not construct the complete homogeneous symmetric polynomial $h_d^{(n)}$ and divide it with $\prod_{k=1}^{d}(n+k)$ but instead form the linear recurrence relation

$$C_{d,j} = \begin{cases} \dfrac{(\lambda - \lambda_0)C_{d-1,0}}{n+d} & \text{for } d > 0 \text{ and } j = 0 \\ \dfrac{(\lambda - \lambda_j)C_{d-1,j}}{n+d} + C_{d,j-1} & \text{for } d > 0 \text{ and } j = 1, ..., n, \end{cases} \tag{19}$$

with initial conditions $C_{0,j} = 1$ for $j = 0, ..., n$. If $j = n$, then $C_{d,n} = h_d^{(n)} / \prod_{k=1}^{d}(n+k)$. The term $C_{d,j}$ is computed as a two-term recurrence. There is no need to use $C_{d,j-2}, ..., C_{d,0}$ and $C_{d-1,j-1}, ..., C_{d-1,0}$ for computing $C_{d,j}$ because they were already

computed in the previous steps. It is a consequence of the complete homogeneous symmetric polynomial $h_d^{(n)}$. Therefore, programming a computer to calculate Eq. (19) would be easy. It is usual for a computer program to store $C_{d,j}$ over $C_{d-1,j}$ because $C_{d-1,j}$ is only used in calculating $C_{d,j}$. Thus, we can use a vector of length $n+1$ that contains $C_{0,j}=1$ initially and $C_{N,j}$ at the end stores intermediate results (i.e., $C_{d,j}$ for $d<N$) during the computation.

The recursion formula (19) translated into a code segment lies in lines 18-21, which costs three flops per update array entry. The total number of flops in the segment is $3n+2$, which does not include the cost of calculating $\lambda-\lambda_j$ and $n+d$. Thus, the computation of $C_{d,n}$ is an $O(n)$ operation.

The terms of the infinite series in Eq. (7) are a function of $\lambda-\lambda_j$ for $j=0,...,n$, so the sum of the series, where $\lambda=max\,\lambda_i$, is a measure of closeness to the Poisson distribution: the smaller the sum of the series (i.e., $\sum_{d=1}^{\infty}(h_d^{(n)}/\prod_{k=1}^{d}(n+k))>0$), the closer the formula (7) is Poisson. When $h_1^{(n)}=\sum_{i=0}^{n}\left(\lambda-\lambda_i\right)$ increases, the sum of the infinite series becomes larger and larger. Practically speaking, the series with a sufficiently large $h_1^{(n)}$ is too big to be represented, that is, a harmful overflow.

Let us show there is no chance of overflow in the series of (7) with $t=1$ if $\lambda_{max}-\lambda_{min}$ is small enough, where $\lambda_{max}=max\,\lambda_i$ and $\lambda_{min}=min\,\lambda_i$ for $i=0,...,n$. Since $\lambda_{max}-\lambda_i\leqslant\lambda_{max}-\lambda_{min}$, it follows that

$$
\begin{aligned}
P_n(1)&=\frac{\prod_{i=0}^{n-1}\lambda_i e^{-\lambda_{max}}}{n!}\left(1+\frac{h_1^{(n)}}{(n+1)}+\frac{h_2^{(n)}}{(n+1)(n+2)}+...\right)\\
&\leqslant\frac{\prod_{i=0}^{n-1}\lambda_i e^{-\lambda_{max}}}{n!}\left(1+\binom{n}{1}\frac{(\lambda_{max}-\lambda_{min})}{(n+1)}+\binom{n+1}{2}\frac{(\lambda_{max}-\lambda_{min})^2}{(n+1)(n+2)}+...\right)\\
&<\frac{\prod_{i=0}^{n-1}\lambda_i e^{-\lambda_{max}}}{n!}\left(1+\frac{(\lambda_{max}-\lambda_{min})}{1!}+\frac{(\lambda_{max}-\lambda_{min})^2}{2!}+...\right)\\
&=\frac{\prod_{i=0}^{n-1}\lambda_i e^{-\lambda_{max}}}{n!}e^{\lambda_{max}-\lambda_{min}}
\end{aligned}
\tag{20}
$$

for $n=1,2,....$ We note that the number of distinct terms in $h_d^{(n)}$ is $\binom{d+n}{d}$, the same as for a multinomial expansion $\left(\lambda-\lambda_0+\lambda-\lambda_1+...+\lambda-\lambda_n\right)^n$, and it becomes $\binom{d+n-1}{d}$ because one of $\lambda-\lambda_i$ for $i=0....,n$ is equal to zero ($\lambda_{max}-\lambda_{max}=0$). We can now find the point at which the series does not suffer from overflow. If we use the formula (7), overflow will not occur whenever $\lambda_{max}-\lambda_{min}\leqslant 709$ ($e^{\lambda_{max}-\lambda_{min}}<2^{1024}\approx 1.8\times 10^{308}$) for IEEE double precision arithmetic. This value is generally large enough for calculating the likelihood function of regression count models based on pure birth processes when we fit the models to count data. For example, the maximum values of $\lambda_{max}-\lambda_{min}$ of conditional Poisson, SUE, and UE distributions for the fertility data reported in Table 1 are 0, 2.80, and 3.14, respectively. One might reasonably fear that $\lambda_{max}-\lambda_{min}\leqslant 709$ is not large enough to compute the likelihood function. We can use `expm`, but we should use it

when and only when $\lambda_{max} - \lambda_{min} > 709$ because Algorithm 1 is faster and more accurate than `expm`. None of them are completely satisfactory. We will compare these two algorithms experimentally in the next section.

The following simple procedure can also avoid the overflow problem: Let β be a huge positive number. For IEEE double precision numbers, $\beta < 2^{1024} \approx 1.8 \times 10^{308}$. It is easy to see that $C_{d,j}$ does not decrease with increasing j ($C_{d,j+1} \geqslant C_{d,j}$). Thus, $C_{d,n} = max\, C_{d,j}$ for $j = 0, ..., n$. If $1 + \sum_{d=1}^{l}(h_d^{(n)} / \prod_{k=1}^{d}(n+k)) > \beta$ for $l \leqslant N$, then $(1 + \sum_{d=1}^{l-1}(h_d^{(n)} / \prod_{k=1}^{d}(n+k)))$ and $C_{l,j}$ for $j = 0, ..., n$ are divided by $C_{l,n}$. After taking a natural logarithm of $C_{l,n}$, it is stored in a variable named α. This scaling procedure eliminates any practical possibility of overflow in $C_{d,j}$ for $d \geqslant l$ because now $C_{l,j} \leqslant 1$ for $j = 0, ..., n-1$ and $C_{l,n} = 1$. This procedure can extend the range of $\lambda_{max} - \lambda_{min}$, but it may cause an underflow in $C_{d,j}$ when $\lambda_{max} - \lambda_{min}$ is large. The computed probabilities suffer a loss of accuracy due to the underflow. Thus, we do not use this procedure and keep our algorithm as simple as possible.

The danger of overflows exists not only in the calculation of the infinite series but in the calculation of $\prod_{i=0}^{n-1} \lambda_i / n!$. We can also eliminate this danger by taking a natural logarithm of $\prod_{i=0}^{k-1} \lambda_i / k!$ when $\prod_{i=0}^{k} \lambda_i / (k+1)! > \beta$, where $k \leqslant n-1$, and adding it to α (line 8). Our algorithm for computing $P_n(t)$, where the scalar parameter t is now set to one, is summarized above. The algorithm is intended for use with $t = 1$ because we only focus on the probabilities for count models based on pure birth processes.

3 Numerical experiments

We give a variety of numerical experiments to illustrate the accuracy and efficiency of our algorithm. All computations are carried out in the R (R Core Team 2022) and MATLAB (2017) programming languages using a laptop 1.80GHz Intel Core i7-8565U with 16GB 1600MHz DDR3L RAM. R and MATLAB provide a routine `expm` for computing the exponential of a matrix, and we compare our algorithm with them. As mentioned previously, the function `expm` in R is a function in the **expm** (Goulet et al. 2022) R package used for calculating Eq. (2) in the **CountsEPPM** (Smith and Faddy 2018) and **BinaryEPPM** (Smith and Faddy 2019a) R packages. We use R's `expm` function with its default parameters unless otherwise stated. The default method is "Higham08.b", which implements the scaling and squaring algorithm of Higham (2008) with balancing. The function `expm` in MATLAB is a built-in function and implements the scaling and squaring algorithm of Higham (2005) and Higham (2009). In all of our tests, the functions `expm` in R and MATLAB produced similar results, confirming that they are implementing the same technique. We note that the tolerance and iteration limit values of Algorithm 1 were set to $\epsilon = 1 \times 10^{-16}$ and $d_{max} = 20,000$, respectively, for IEEE double precision arithmetic.

3.1 Several pure birth count models

To evaluate the performance of the two exact algorithms, Algorithm 1 and `expm`, we generate 505,000 random rate sequences, $\{\lambda_0, ..., \lambda_n\} \in \mathbb{R}^{n+1}$ for $n = 4, 8, ..., 400$, a total of 100 points, for each of the four pure birth count models outlined below. To see the effect of $\lambda_{max} - \lambda_{min}$ on the accuracy and efficiency of the algorithms, we set $\lambda_{max} - \lambda_{min} \in \{0, 2, ..., 200\}$ (101 points) and $\lambda_{max} - \lambda_{min} \in \{1, 10^{0.0285}, 10^{2(0.0285)}, 10^{3(0.0285)}, ..., 10^{100(0.0285)} = 707.95\}$ (101 points on a base-10 log scale) to compare the accuracy and efficiency, respectively. Thus, each of the 10,100 pairs $(\lambda_{max} - \lambda_{min}, n)$ contains 50 random rate sequences. We recorded the highest relative error defined in the next section and the average computing time for 50 probabilities $P_n(1)$.

1. A sequence of rates

$$\lambda_i = \lambda(b + i)^c \tag{21}$$

for $i = 0, 1, 2, ..., n$, $\lambda > 0$, $b > 0$, and $c \leqslant 1$ exhibits the Faddy (1997) distribution, a three-parameter count model. Faddy (1997) describes that $c = 0$, $c > 0$, and $c < 0$ correspond to constant, increasing, and decreasing sequences of λ_i's, respectively. For each sequence, we draw c piecewise uniformly from $[-10, 0)$ and $[0, 1]$ with both 0.5 probabilities and b uniformly from 0 to 20. We then solve for λ using the equation:

$$\lambda = \left| \frac{\lambda_{max} - \lambda_{min}}{(b + n)^c - b^c} \right|. \tag{22}$$

2. A sequence of rates

$$\lambda_i = \lambda(b - i)^c \tag{23}$$

for $i = 0, 1, 2, ..., n$, $b = n, n + 1, ...$, $\lambda > 0$, and $c > 0$ exhibits the Faddy and Smith (2008) distribution, a three-parameter count model. Faddy and Smith (2008) and Smith and Faddy (2019) discuss overdispersion and underdispersion relative to the binomial distribution of this count model. They depend on convex ($c > 1$) and concave ($0 < c < 1$) decreasing λ_i sequences, respectively. For each sequence, we draw c piecewise uniformly from (0, 1] and (1, 11] with both 0.5 probabilities and b uniformly from the integers n to $n + 20$. We then solve for λ using the equation:

$$\lambda = \frac{\lambda_{max} - \lambda_{min}}{b^c - (b - n)^c}. \tag{24}$$

3. A sequence of rates

$$\lambda_i = \begin{cases} \lambda & \text{for } i \neq n \\ c\lambda & \text{for } i = n \end{cases} \tag{25}$$

for $i = 0, 1, 2, ..., n$, $\lambda > 0$, and $c > 0$ exhibits the SUE distribution, a two-parameter count model. The $(n + 1)$th event is unusual. The over-, under-, and equidispersion relative to the Poisson distribution for the SUE process depend on both c and λ (Skulpakdee and Hunkrajok 2022). For $c = 1$, this SUE process becomes the Poisson process. If $c \neq 1$ and $n > 0$, then the rate sequence is non-monotonic. For each sequence, we draw c piecewise uniformly from (0, 1] and (1, 11] with both 0.5 probabilities and then solve for λ using the equation:

$$\lambda = \left| \frac{\lambda_{max} - \lambda_{min}}{c - 1} \right|. \tag{26}$$

4. A sequence of rates

$$\lambda_i = \begin{cases} \lambda & \text{for } i < n - 2 \\ a\lambda & \text{for } i = n - 2 \\ b\lambda & \text{for } i = n - 1 \\ c\lambda & \text{for } i = n \end{cases} \tag{27}$$

for $i = 0, 1, 2, ..., n$, $\lambda > 0$, $a > 0$, $b > 0$, and $c > 0$ exhibits the UE distribution, a four-parameter count model. The $(n - 1)$th, nth, and $(n + 1)$th events are unusual. Similar to the SUE process, the four parameters a, b, c, and λ specify whether the distribution is over-, under-, or equidispersed. We draw a, b, and c piecewise uniformly for each sequence from (0, 1] and (1, 11] with both 0.5 probabilities and then solve for λ using the equation:

$$\lambda = \frac{\lambda_{max} - \lambda_{min}}{max(1, a, b, c) - min(1, a, b, c)}. \tag{28}$$

3.2 Accuracy comparisons

Although Algorithm 1 and `expm` can provide exact probability values of pure birth count models, it is desirable to verify their software implementations. In this section, we define the relative error such that

$$E_{rel} = \frac{\left| P_{alg1} - P_{expm} \right|}{min\left\{ P_{alg1}, P_{expm} \right\}}, \tag{29}$$

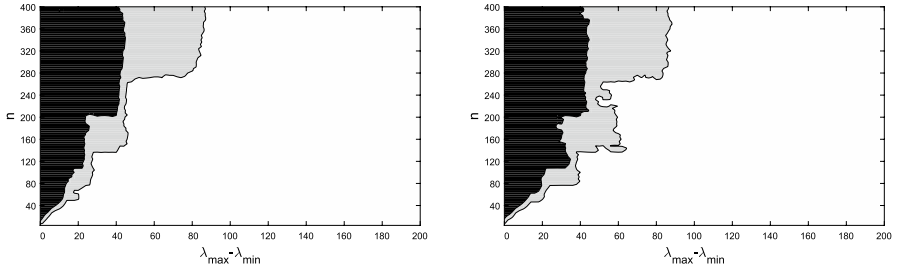
where P_{alg1} and P_{expm} are the probabilities computed by Algorithms 1 and `expm`, respectively. There are two reasons we define this relative error, and the first is that the two algorithms might fail to produce the exact answer of $P_n(1)$. To see this, refer to Table 3 and the introduction section of Smyth and Podlich (2002). The second reason is that either computing P_{alg1} or P_{expm} at high-digit precision, such as 100 digits, is time-consuming when using MATLAB's Symbolic Math Toolbox. For example, `expm` computes the probabilities for 505,000 rate sequences in many days. We

have implemented Algorithm 1 in MATLAB and C++ via the **Rcpp** (Eddelbuettel et al. 2023) package to accelerate computations and applied them to the Faddy (1997), Faddy and Smith (2008), SUE, and UE models using IEEE double-precision arithmetic. The left and right parts of Fig. 1 show the relative error in R and MATLAB, respectively. Comparing the left with the right, we see that they are very similar, indicating that the errors produced by our algorithm are insensitive to different programming platforms.

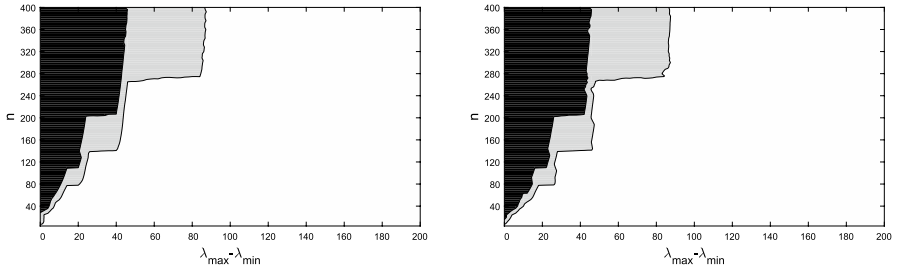
To verify the correctness of Algorithm 1 and `expm` using IEEE double-precision arithmetic, we plot regions of the highest relative error of 50 sets of λ_i's simulated for each $(\lambda_{max} - \lambda_{min}, n)$. These values in the black, gray, and white areas of Fig. 1 are more than 10^{-5}, between 10^{-5} and 10^{-10}, and less than 10^{-10}, respectively. The value 10^{-5} may indicate that the probabilities calculated by Algorithm 1 and `expm` agree in their five decimal places. The figure is an approximate representation of the region to be checked; it is intended only to provide a general idea of which area should be to checked. The two algorithms compute the probability $P_n(1)$ with the same rate sequence in the black region, but the results agree to less than five decimal places. Does none of the algorithms provide high relative accuracy for this area, or just one? We must redo the calculation with much higher precision to answer this question.

For each count model, we select five rate sequences with $n = 23$ from the black area of Fig. 1 and compute the probabilities $P_{23}(1)$ using 100 digits of working precision. The results of Algorithm 1 and `expm` are shown to 15 significant figures in the last columns of Table 3. We can trust all the digits in this column because both results agree to 15 digits. The two algorithms are implemented differently. If we increase the 16-digit precision to 100, then the two algorithms confirm the solutions of lower precision. Thus we have good scientific reasons to claim that the probabilities given by Algorithm 1 are correct to more than 12 digits (see the first column of Table 3) – even if evaluated in the realm of IEEE double-precision arithmetic. In contrast, Matlab's `expm` function gives less than or equal to four correct digits (see the second column of Table 3). The first row of each count model in Table 3 provides an example in which an incorrect probability (no correct digits) is obtained from MATLAB's `expm`. As mentioned in Sect. 1, matrix-exponential approaches may not preserve tiny probabilities to good relative accuracy (Smyth and Podlich 2002, p. 18). Table 3 provides strong evidence in this regard. There is, therefore, a clear reason to favor Algorithm 1 over `expm` in terms of accuracy in IEEE double-precision arithmetic.

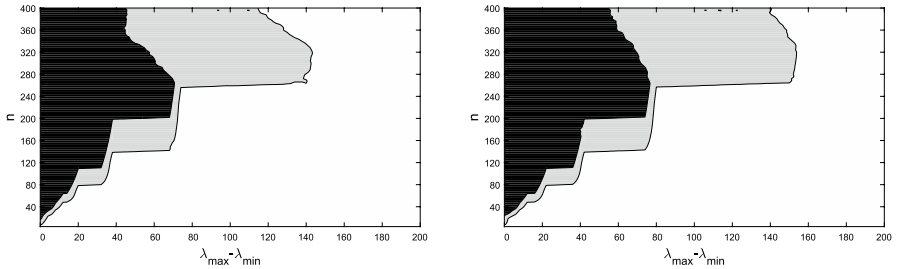
Figure 1 clearly shows that the smaller the $\lambda_{max} - \lambda_{min}$, the larger the relative error, especially while increasing n. We observe that there are jumps in $\lambda_{max} - \lambda_{min}$ in the black regions, and the SUE and UE models' jumps are much larger than those in the Faddy (1997) and Faddy and Smith (2008) models. For example, while n passes 200, the difference between the size of the jumps of the UE and Faddy and Smith (2008) models is roughly 12. The UE model has non-monotonic rate sequences, which may explain this particular behavior. There are also jumps in the gray region, but the integer values of n where $\lambda_{max} - \lambda_{min}$ jumps are greater than those in the black area.



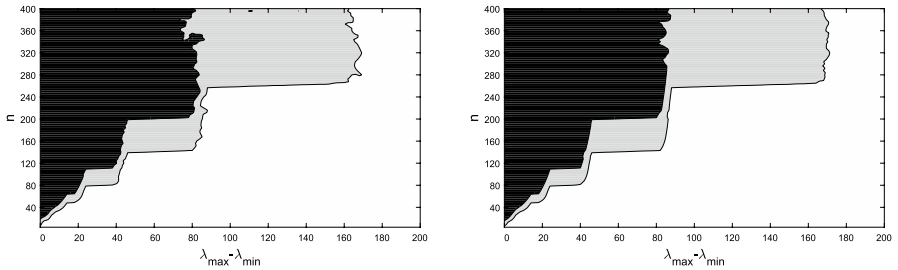
(a) Faddy (1997) model



(b) Faddy and Smith (2008) model



(c) SUE model



(d) UE model

(e) Faddy (1997) model

(f) Faddy and Smith (2008) model

(g) SUE model

(h) UE model

Fig. 1 Three regions of the highest relative error of 50 probabilities computed in R (left) and MATLAB (Right) for each $(\lambda_{max} - \lambda_{min}, n)$. The black area represents these values exceeding 10^{-5}, whereas the white represents them under 10^{-10}. The difference $\lambda_{max} - \lambda_{min}$ indicates how close the distribution is to Poisson ($\lambda_{max} - \lambda_{min}$=0)

Table 3 Algorithm 1 (ALG 1) and MATLAB's `expm` applied to the probabilities $P_{23}(1)$ of various count models with several parameter values for five rate sequences in the black area of Fig. 1

IEEE double-precision				100-digit	
ALG 1	T	`expm`	T	ALG 1 and `expm`	T
$b, c, \lambda \in \{(0.06, -1.04, 0.05), (0.08, -0.19, 0.94),$					
Faddy (1997) model: $(0.16, 0.49, 0.23), (3.52, -0.63, 3.07), (0.81, 0.65, 0.14)\}$					
8.3806 1391 0722 **37**e-74	5.93	**2381.3 3740 4894 75**e-74	29.2	8.3806 1391 0722 17e-74	229
7.3999 6367 6181 1**3**e-28	2.54	7.**4003 5536 0487 64**e-28	6.89	7.3999 6367 6181 11e-28	179
4.3917 6376 8153 60e-28	0.23	4.3**837 6936 4440 89**e-28	2.41	4.3917 6376 8153 60e-28	176
3.2537 5623 8134 77e-28	0.27	3.25**56 8357 5513 71**e-28	2.32	3.2537 5623 8134 77e-28	219
1.0257 9184 5642 7**4**e-28	1.59	1.025**0 9834 1360 40**e-28	8.63	1.0257 9184 5642 73e-28	188
Faddy and Smith (2008) model: $b, c, \lambda \in \{(43, 10.92, 1.45e\text{-}18), (23, 0.04, 0.88),$ $(23, 0.01, 0.97), (23, 0.11, 0.71), (23, 0.49, 0.22)\}$					
6.5874 2724 8419 **38**e-58	0.27	**7.2462 2409 8769 57**e-58	1.77	6.5874 2724 8419 19e-58	178
6.4060 4472 2331 7**8**e-24	0.07	6.**3815 1217 7455 23**e-24	0.43	6.4060 4472 2331 72e-24	179
1.2440 6563 2335 31e-23	0.10	1.2**394 3602 8036 62**e-23	0.40	1.2440 6563 2335 31e-23	190
1.7885 2507 5076 64e-24	0.11	1.78**15 9591 3182 41**e-24	0.39	1.7885 2507 5076 64e-24	208
1.4187 1706 8502 95e-27	0.08	1.418**0 0960 0282 91**e-27	0.38	1.4187 1706 8502 95e-27	182
SUE model: $c, \lambda \in \{(9.7, 0.11), (2.74, 0.57), (3.39e\text{-}2, 1.04), (0.01, 1.01), (2.61, 0.62)\}$					
2.9837 2029 061**7 00**e-45	0.26	**3.3090 2229 9354 29**e-45	1.21	2.9837 2029 0616 97e-45	185
5.0987 7184 4749 **84**e-29	0.07	5.**1105 6780 0106 36**e-29	0.35	5.0987 7184 4749 91e-29	206
3.5167 8783 6656 44e-23	0.07	3.5**046 8837 9691 46**e-23	0.36	3.5167 8783 6656 44e-23	207
1.8480 2843 9164 3**9**e-23	0.06	1.84**13 2086 9501 68**e-23	0.34	1.8480 2843 9164 38e-23	185
3.3537 1758 5619 2**3**e-28	0.05	3.353**1 7031 7715 33**e-28	0.32	3.3537 1758 5619 20e-28	185
UE model: $a, b, c, \lambda \in \{(7.04, 0.52, 0.03, 0.14), (0.98, 0.82, 0.03, 1.03),$ $(0.72, 1.34, 0.38, 1.04), (0.96, 0.58, 0.04, 1.04), (1.56, 0.71, 0.08, 0.68)\}$					
2.7532 9569 1093 05e-42	0.26	**3.0288 0502 6287 90**e-42	1.02	2.7532 9569 1093 05e-42	227
2.3048 6470 3426 2**2**e-23	0.03	2.**2966 5520 6075 22**e-23	0.27	2.3048 6470 3426 23e-23	193
3.3327 7129 6687 79e-23	0.04	3.3**216 9123 6716 55**e-23	0.31	3.3327 7129 6687 79e-23	215
1.9972 6057 3921 26e-23	0.04	1.99**01 3796 0913 49**e-23	0.26	1.9972 6057 3921 26e-23	186
3.1079 8093 1168 6**7**e-27	0.04	3.107**2 4237 4184 51**e-27	0.30	3.1079 8093 1168 68e-27	220

The incorrect digits are set in boldface and underlined. The letter "T" represents running times for the two algorithms in milliseconds (ms), and the last column indicates the time of `expm`. The tolerance and iteration limit values of Algorithm 1 were set to $\epsilon = 1 \times 10^{-100}$ and

$d_{max} = 20,000$ for 100-digit precision

3.3 Efficiency comparisons

The run times for Algorithm 1 and `expm` vary over the different values of n and $\lambda_{max} - \lambda_{min}$ in our numerical experiments. Rather than provide the raw values of run times, we give the run time ratio of `expm` versus Algorithm 1. Thus, we let t_{ratio} denote the time of `expm` divided by the time of Algorithm 1. Since the t_{ratio}'s surfaces have many spikes, they are removed by a Gaussian filter.

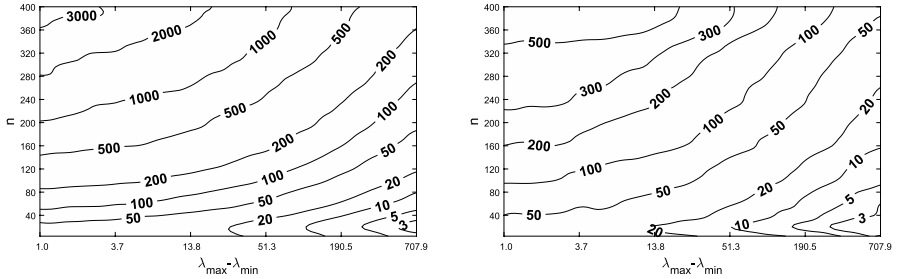
Smooth contour plots showing a two-dimensional graphical representation of the relationship between t_{ratio} and the two independent variables, $\lambda_{max} - \lambda_{min}$ and n, were constructed for the numerical experiments. These smooth plots are a precious visual aid for the human eye, as shown in Fig. 2. Figure 2 shows the contour plots of the Faddy (1997), Faddy and Smith (2008), SUE, and UE count models for comparing Algorithm 1 and `expm` in R (left) and MATLAB (right). The contour plots of Fig. 2a–d (or Fig. 2e–h) are quite similar, indicating that Algorithm 1 and `expm` are insensitive to the different rate sequences (monotonic or non-monotonic).

Comparing the left part of Fig. 2 with the right, we observe that the ratio in R (left) grows more rapidly than MATLAB (right) when n is increased. The algorithms R's `expm` and MATLAB's `expm` are all based on matrix operations (scaling and squaring), so the memory and computation requirements increase dramatically with increasing n, in contrast to Algorithm 1. In other words, as n increases, the birth rate matrix gets bigger, so `expm` consumes more time. Since the ratio grows rapidly as a function of n, `expm` is not competitive with Algorithm 1 unless $\lambda_{max} - \lambda_{min}$ is large. The largest n considered here is of modest size. Extrapolating from these results, we expect `expm` will consume much more time on a large n, although $\lambda_{max} - \lambda_{min}$ is very small. Thus, Algorithm 1 is recommended for fitting pure birth count models because the conditional distributions of count data are typically not too far from Poisson (i.e., a small $\lambda_{max} - \lambda_{min}$). For example, the maximum values of $\lambda_{max} - \lambda_{min}$ of conditional Poisson, SUE, and UE distributions for the ideal fertility data reported in Table 1 are 0, 2.77, and 15.79, respectively.

4 Conclusions

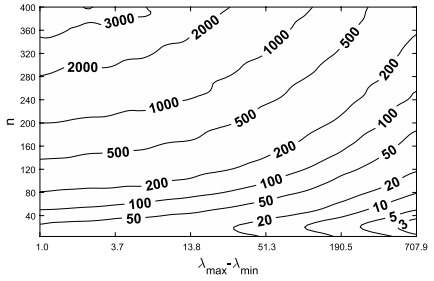
This paper presents a new algorithm for computing the probabilities for count models based on pure birth processes without matrix operations. It is accomplished by reformulating the original analytic form as a new infinite series and choosing a suitable parameter value λ to guarantee without checking that no cancellations occur. The key to the reformulation is the use of a Taylor series expansion to get around the fact that there is no simple way to compute the analytic form with no cancellations. In implementing the formula (7), we are aware of the possibility of cancellation. The obvious choice for λ is $max\ \lambda_i$ for $i = 0, ..., n$ since $\lambda - \lambda_i \geqslant 0$ imply that the series is non-alternating, and the non-negative terms can be coded recursively without difficulty. By restricting the possible values of $\lambda_{max} - \lambda_{min} \leqslant 709$, we do not encounter problems with overflows in the calculated series. This restriction simplifies the algorithm, which is usually sufficient for computing the likelihood function when fitting count models based on pure birth processes.

Our focus is not on modeling, although count models are provided to illustrate the accuracy and efficiency of `expm` and our algorithm. The accuracy of `expm` is lost when $\lambda_{max} - \lambda_{min}$ is small (see Fig. 1). The only way to guarantee accurately computed probabilities in all cases by `expm` is to use extended precision, such as 100 digits (see Table 3). Algorithm 1, however, is very accurate even if these probabilities are computed in the realm of IEEE double-precision arithmetic. Algorithm 1 also runs faster than `expm` for $\lambda_{max} - \lambda_{min} \leqslant 709$, the most significant domain of

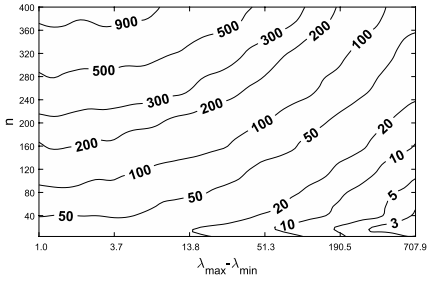


(a) Faddy (1997) model

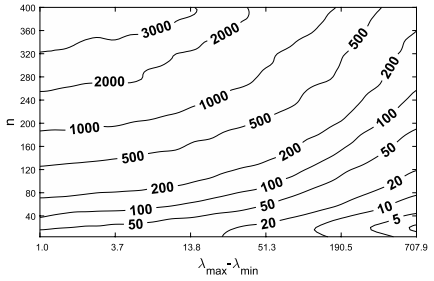
(e) Faddy (1997) model



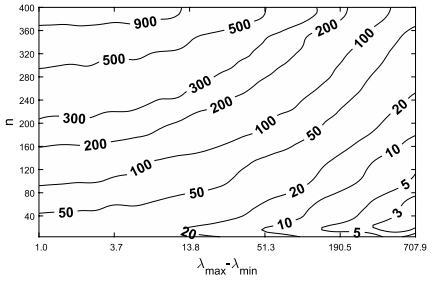
(b) Faddy and Smith (2008) model



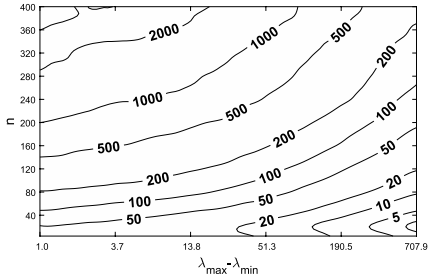
(f) Faddy and Smith (2008) model



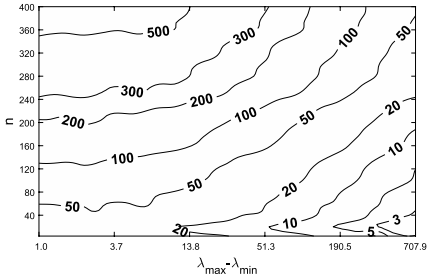
(c) SUE model



(g) SUE model



(d) UE model



(h) UE model

Fig. 2 Semi-log smooth contours of the run time ratio t_{ratio} of R's `expm` versus Algorithm 1 (left) and MATLAB's versus Algorithm 1 (right). The curves with $t_{ratio} = 10$ indicate that Algorithm 1 is ten times faster than `expm`

count data (see Fig. 2). As a consequence of this work, practitioners now have faster, more accurate, and easily programmatic algorithms to compute the probabilities of count models based on pure birth processes, which previously relied on matrix-exponential approaches.

Acknowledgements The authors thank the associate editor and anonymous reviewers for their valuable comments and suggestions.

References

Ball F (1995) A note on variation in birth processes. Math Sci 20:50–55
Banks WD, Martin G (2013) Optimal primitive sets with restricted primes. Integers 13:A69
Barreto-Souza W, Simas AB (2016) General mixed Poisson regression models with varying dispersion. Stat Comput 26(6):1263–1280
Bartlett MS (1978) An introduction to stochastic processes. Cambridge University Press, Cambridge
Bourguignon M, de Medeiros RMR (2022) A simple and useful regression model for fitting count data. TEST 31(3):790–827
Chanialidis C, Evers L, Neocleous T et al (2018) Efficient Bayesian inference for COM-Poisson regression models. Stat Comput 28:595–608
Cheney EW, Kincaid DR (2004) Numerical mathematics and computing, 5th edn. Brooks/Cole Publishing Company, Belmont
Conway RW, Maxwell WL (1962) A queuing model with state dependent service rates. J Ind Eng 12(2):132–136
Cox DR, Miller H (1965) The Theory of Stochastic Processes. Chapman and Hall, London
Crawford FW, Ho LST, Suchard MA (2018) Computational methods for birth-death processes. WIREs Comput Stat 10(2):1–22
Daniels HE (1982) The saddlepoint approximation for a general birth process. J Appl Probab 19(1):20–28
Eddelbuettel D, Francois R, Allaire J, et al (2023) Rcpp: seamless R and C++ integration. https://CRAN.R-project.org/package=Rcpp, R package version 1.0.10
Faddy MJ (1994) On variation in Poisson processes. Math Sci 19(1):47–51
Faddy MJ (1997) Extended Poisson process modelling and analysis of count data. Biom J 39(4):431–440
Faddy MJ, Smith DM (2008) Extended Poisson process modelling of dilution series data. J Roy Stat Soc: Ser C (Appl Stat) 57(4):461–471
Faddy MJ, Smith DM (2011) Analysis of count data with covariate dependence in both mean and variance. J Appl Stat 38(12):2683–2694
Faddy MJ, Smith DM (2012) Extended Poisson process modelling and analysis of grouped binary data. Biom J 54(3):426–435
Forthmann B, Gühne D, Doebler P (2020) Revisiting dispersion in count data item response theory models: The Conway-Maxwell-Poisson counts model. Br J Math Stat Psychol 73:32–50
Goulet V, Dutang C, Maechler M, et al (2022) expm: matrix exponential, log, 'etc'. https://CRAN.R-project.org/package=expm, R package version 0.999-7
Guikema SD, Goffelt JP (2008) A flexible count data regression model for risk analysis. Risk Anal Int J 28(1):213–223
Higham NJ (2005) The scaling and squaring method for the matrix exponential revisited. SIAM J Matrix Anal Appl 26(4):1179–1193
Higham NJ (2008) Functions of matrices: theory and computation. Society for Industrial and Applied Mathematics, Philadelphia
Higham NJ (2009) The scaling and squaring method for the matrix exponential revisited. SIAM Rev 51(4):747–764
Huang J, Zhu F (2021) A new first-order integer-valued autoregressive model with Bell innovations. Entropy 23(6):713
Jensen A (1953) Markoff chains as an aid in the study of Markoff processes. Scand Actuar J 1953(sup1):87–91
Jung RC, Ronning G, Tremayne AR (2005) Estimation in conditional first order autoregression with discrete support. Stat Pap 46(2):195–224

Kharrat T, Boshnakov GN (2022) Countr: flexible univariate count models based on renewal processes. https://CRAN.R-project.org/package=Countr, R package version 3.5.6
Kharrat T, Boshnakov GN, McHale I et al (2019) Flexible regression models for count data based on renewal processes. J Stat Softw 90(13):1–35
Maechler M, Maechler MM, MPFR S, et al (2024) Package 'Rmpfr'. https://CRAN.R-project.org/package=Rmpfr, R package version 0.9-5
MATLAB (2017) version 9.2.0.538062 (R2017a). The MathWorks, Inc., Natick, Massachusetts, United States
McShane B, Adrian M, Bradlow ET et al (2008) Count models based on Weibull interarrival times. J Bus Econ Stat 26(3):369–378
Minka T, Shmueli G, Kadane J, et al (2003) Computing with the COM-poisson distribution. Technical report CMU Statistics Department 776
Moler C, Van Loan C (2003) Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later. SIAM Rev 45(3):3–49
Morales-Otero M, Núñez-Antón V (2021) Comparing Bayesian spatial conditional overdispersion and the Besag-York-Mollié models: application to infant mortality rates. Mathematics 9(3):282
Peluso A, Vinciotti V, Yu K (2019) Discrete Weibull generalized additive model: an application to count fertility data. J Roy Stat Soc Ser C (Appl Stat) 68(3):565–583
Philipson P, Huang A (2023) A fast look-up method for Bayesian mean-parameterised Conway-Maxwell-Poisson regression models. Stat Comput 33(81):1–16
Podlich HM, Faddy MJ, Smyth GK (2004) Semi-parametric extended Poisson process models for count data. Stat Comput 14:311–321
R Core Team (2022) R: a language and environment for statistical computing. R foundation for statistical computing, Vienna, Austria, https://www.R-project.org/
Renshaw E (2011) Stochastic population processes: analysis, approximations, simulations. Oxford University Press, Oxford
Rovenţa I, Temereancă LE (2019) A note on the positivity of the even degree complete homogeneous symmetric polynomials. Mediterr J Math 16(1):1–16
Saez-Castillo AJ, Conde-Sanchez A (2013) A hyper-Poisson regression model for overdispersed and underdispersed count data. J Comput Stat Data Anal 61:148–157
Sellers KF, Shmueli G (2009) A regression model for count data with observation-level dispersion. In: Paper presented at the 24th international workshop on statistical modelling, Cornell University, New York, Accessed 20–24 July 2009
Sellers KF, Shmueli G (2010) A flexible regression model for count data. Ann Appl Stat 4(2):943–961
Sellers KF, Lotze T, Raim AM (2019) COMPoissonReg: Conway-Maxwell Poisson (COM-Poisson) regression. https://CRAN.R-project.org/package=COMPoissonReg, R package version 0.7.0
Skulpakdee W, Hunkrajok M (2022a) 3: a sue-Poisson inarch(1) model. In: Paper presented at the 6th international conference on compute and data analysis, Shanghai, China. Accessed 25-27 February 2022
Skulpakdee W, Hunkrajok M (2022) Unusual-event processes for count data. SORT Stat Oper Res Trans 46(1):39–66
Smith DM, Faddy MJ (2016) Mean and variance modelling of under- and overdispersed count data. J Stat Softw 69(6):1–23
Smith DM, Faddy MJ (2018) CountsEPPM: Mean and variance modeling of count data. https://CRAN.R-project.org/package=CountsEPPM, R package version 3.0
Smith DM, Faddy MJ (2019a) BinaryEPPM: mean and variance modeling of binary data. https://CRAN.R-project.org/package=BinaryEPPM, R package version 2.3
Smith DM, Faddy MJ (2019) Mean and variance modeling of under-dispersed and over-dispersed grouped binary data. J Stat Softw 90(8):1–20
Smyth GK, Podlich HM (2002) An improved saddlepoint approximation based on the negative binomial distribution for the general birth process. Comput Stat 17(1):17–28
Watkins DS (2010) Fundamentals of matrix computations, 3rd edn. Wiley, New Jersey
Weiß CH (2018) An introduction to discrete-valued time series, 1st edn. Wiley, New Jersey
Weiß CH, Zhu F, Hoshiyar A (2022) Softplus INGARCH models. Stat Sin 32(2):099–1120
Winkelmann R (1995) Duration dependence and dispersion in count-data models. J Bus Econ Stat 13(4):467–474