Supplementary File S1: Birth-death process with catastrophic extinction

1 Introduction

This study investigates a continuous-time Markov process dubbed as the linear birth-death process with catastrophic extinction (B-D-C process). The primary motivation is to consider the B-D-C process as an auxiliary model for a more complex stochastic model that simulates the spread of different strains of *Gyrodactylus* parasites over the external surfaces of the host (refer to the main paper). Here, the B-D-C process is used to refine the summary statistics of a modified approximate Bayesian computation (ABC) in calibrating the multidimensional stochastic model based on estimates of the B-D-C model parameters (see the main paper). The simulation of the B-D-C process using a tau-leaping algorithm also provides additional insights on how to accelerate the simulation of the sophisticated stochastic model by proposing a good error threshold based on the trade-off between simulation accuracy and computational speed.

The constant-rate linear B-D-C process is a discrete state-space stochastic process where each host gives birth to new hosts at a constant rate $\lambda > 0$, dies at constant rate $\mu > 0$ and the entire population becomes extinct due to a catastrophic event at a constant $\rho > 0$ (for a formal definition, see section 1.1). Thus, the linear B-D-C process is an extension of the classical linear birth-death process where the process is subjected to catastrophes that result in parasite population extinction [3]. Due to the application of the B-D-C process in the current study (for other modelling purposes), the catastrophe rate is assumed to depend on the parasite population size since host mortality (defined as the catastrophe event) for ectoparasitic infection occurs at a rate proportional to parasite abundance. Although these class of stochastic models are simple in terms of their model framework, the exact transition function and parameter estimation can be challenging to obtain in the setting of discretely observed processes [2, 9].

In this current study, we derive the analytical transition function of the B-D-C process (section 1.2). The derived transition function is further validated analytically using mathematical induction and is numerically validated based on Monte Carlo estimation (sections 1.2–1.4). Additionally, we estimate the B-D-C model parameters by comparing different estimation methods (Maximum likelihood estimation, generalised method of moments and embedded Galton-Watson approach) based on three different *in silico* simulation experiments where parasite population size is large, moderate or low (section 2). The bias, variance, and mean square error of the parameter estimates and

the estimation methods' computational times are compared. Finally, we develop and compare two different hybrid τ-leaping algorithm based on leap-size selection methods proposed by Gillespie [5] and Gillespie and Petzold [6], respectively, to accelerate the simulation of the B-D-C process (section 3). We propose a good error threshold by exploring the trade-off between simulation accuracy and computational speed of the three different *in silico* simulation experiments where parasite numbers are high (Case 1), moderate (Case 2) or low (Case 3). The differences between the two τ-leaping methods are the leap-size selection procedure (which is proportional to the simulation error bound) and their respective leap conditions. All the mathematical theorems in this supplementary file are proposed and proved for the first time in the current study.

1.1 Definition of the Linear B-D-C process

Let $\{X_t, t\geq 0\}$, the number of parasites on host at any time t, be a linear birth and death process with catastrophic extinction (B-D-C process) defined on the state space, $S = \{0, 1, 2, \cdots\}$, determined in accordance with the following scheme:

Event	**Transition**	**Rate**
Birth	$X_t \to X_t + 1$	λX_t
Death	$X_t \to X_t - 1$	μX_t
Catastrophe	$X_t \to 0$	ρX_t

where $\lambda > 0$, $\mu > 0$ and $\rho > 0$ are the birth, death and catastrophe rates respectively. The catastrophe event is defined as the state where the parasite population suddenly hit 0 due to host mortality. The exact transition probabilities of the defined B-D-C process,

$$P_{m,n}(t) = P\{X(t) = n | X(0) = m\}$$

can be obtained from the probability generating function $G_m(z,t)$ as presented by Lemma 1. The probability generating function (PGF) given by Lemma 1 is taken from Karlin and Tavaré [9].

Lemma 1. Given the rates λ, μ and ρ, suppose v_0 and v_1 are the roots of the equation

$$\lambda v + (\mu/v) = \lambda + \mu + \rho; \quad 0 < v_0 < 1 < v_1.$$

Then, the probability generating function of the B-D-C process, given $X(0) = m \geq 1$, is defined as:

$$\begin{aligned} G_m(z,t) &= \sum_{n=0}^{\infty} P_{m,n}(t) z^n = \left[\frac{\nu_0\nu_1(1-\sigma) + z(\nu_1\sigma - \nu_0)}{\nu_1 - \sigma\nu_0 - z(1-\sigma)}\right]^m + C(t) \\ &= \left(\frac{k_1 + k_2 z}{1 - k_3 z}\right)^m + C(t) \quad \text{for} \quad |z| < 1; \end{aligned} \tag{1}$$

where $k_1 = \frac{\nu_0\nu_1(1-\sigma)}{\nu_1-\sigma\nu_0}$, $k_2 = \frac{\nu_1\sigma-\nu_0}{\nu_1-\sigma\nu_0}$, $k_3 = \frac{1-\sigma}{\nu_1-\sigma\nu_0}$, $\sigma = \mathrm{e}^{-\lambda(\nu_1-\nu_0)t}$, $\nu_0 = \frac{(\lambda+\mu+\rho)-\sqrt{(\lambda+\mu+\rho)^2-4\mu\lambda}}{2\lambda}$, $\nu_1 = \frac{(\lambda+\mu+\rho)+\sqrt{(\lambda+\mu+\rho)^2-4\mu\lambda}}{2\lambda}$, and the probability of catastrophic extinction, $C(t)$, is given as

$$C(t) = 1 - \left(\frac{k_1+k_2}{1-k_3}\right)^m.$$

Remark. If the catastrophe rate $\rho = 0$, then the linear B-D-C process $\{X_t, t\geq 0\}$ (with birth rate $\lambda > 0$, death rate $\mu > 0$ and catastrophe rate $\rho = 0$) is the standard linear birth-death process with its probability generating function given as

$$\tilde{G}_m(z,t) = \left[\frac{(1-\sigma)+(\sigma-\gamma)z}{1-\sigma\gamma-z\gamma(1-\sigma)}\right]^m, \quad m \geq 1, \quad |z| < 1$$

where $\sigma = \mathrm{e}^{-(\mu-\lambda)t}$ and $\gamma = \lambda/\mu$. Also, for catastrophe rate $\rho > 0$, the linear B-D-C process conditioned on non-extinction is a birth-death process (where $C(t) = 0$).

1.2 Derivation of the transition function and theoretical moments of B-D-C process from its PGF

We propose and prove the analytical form of the *nth* derivative of the PGF given by equation 1 (w.r.t. z) of the B-D-C process (in accordance to Theorem 1, proposed for the first time in the current study) using mathematical induction. The exact transition probability function and other theoretical moments (mean, variance and the 3rd uncentred moment) are explicitly derived for further modelling purposes.

Theorem 1. *Given the probability generating function $G_m(z,t)$ defined in Lemma 1, the nth derivative w.r.t. z is given as*

$$G_m^{(n)}(z,t) = \sum_{j=1}^{\min(m,n)} \gamma_j^{(n)} \frac{m!}{(m-j)!} k_3^{n-j} (k_2+k_1k_3)^j \left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-j} (1-k_3z)^{-(n+j)}, \quad m,n \geq 1 \tag{2}$$

where $\gamma_j^{(n)}$ is defined recursively by

$$\gamma_j^{(n)} = \gamma_{j-1}^{(n-1)} + (n+j-1)\gamma_j^{(n-1)} \quad \textit{for} \quad j = 2,3,\cdots,\min(m,n-1)$$

and

$$\gamma_1^{(n)} = \min(m,n)\gamma_1^{(n-1)},$$

with

$$\gamma_n^{(n)} = \gamma_{n+1}^{(n+1)} = 1, \quad \forall n \in \mathbb{N}.$$

Proof by mathematical induction.

For $n = 1$, $m \geq 1$

$$\begin{aligned}
\frac{\partial}{\partial z} G_m(z,t) &= \frac{\partial}{\partial z}\left[\left(\frac{k_1 + k_2 z}{1 - k_3 z}\right)^m + C(t)\right] \\
&= m(k_2 + k_1 k_3)\left(\frac{k_1 + k_2 z}{1 - k_3 z}\right)^{m-1} (1 - k_3 z)^{-2}, \quad \gamma_1^{(1)} = 1 \\
&= \gamma_1^{(1)} \frac{m!}{(m-1)!}(k_2 + k_1 k_3)\left(\frac{k_1 + k_2 z}{1 - k_3 z}\right)^{m-1} (1 - k_3 z)^{-2} \\
&= \sum_{j=1}^{1} \gamma_j^{(1)} \frac{m!}{(m-j)!} k_3^{1-j} (k_2 + k_1 k_3)^j \left(\frac{k_1 + k_2 z}{1 - k_3 z}\right)^{m-j} (1 - k_3 z)^{-(1+j)} \\
&= G_m^{(1)}(z,t) \quad \text{as required.}
\end{aligned}$$

Now, suppose equation 2 holds for $n = k$, we show it holds for $n = k + 1$.

Case 1: $k + 1 \leq m$

$$G_m^{(k+1)}(z,t) = \tfrac{\partial}{\partial z} G_m^{(k)}(z,t)$$

$$= \frac{\partial}{\partial z}\left[\sum_{j=1}^{k} \gamma_j^{(k)} \frac{m!}{(m-j)!} k_3^{k-j} (k_2 + k_1 k_3)^j \left(\frac{k_1 + k_2 z}{1 - k_3 z}\right)^{m-j} (1 - k_3 z)^{-(k+j)}\right]$$

$$= \frac{\partial}{\partial z}\left[\gamma_1^{(k)} \frac{m!}{(m-1)!} k_3^{k-1} (k_2 + k_1 k_3)\left(\frac{k_1 + k_2 z}{1 - k_3 z}\right)^{m-1} (1 - k_3 z)^{-(k+1)}\right]$$

$$+ \frac{\partial}{\partial z}\left[\gamma_2^{(k)} \frac{m!}{(m-2)!} k_3^{k-2} (k_2 + k_1 k_3)^2 \left(\frac{k_1 + k_2 z}{1 - k_3 z}\right)^{m-2} (1 - k_3 z)^{-(k+2)}\right] + \cdots +$$

$$\frac{\partial}{\partial z}\left[\gamma_{k-1}^{(k)} \frac{m!}{(m-(k-1))!} k_3^{k-(k-1)} (k_2 + k_1 k_3)^{k-1} \left(\frac{k_1 + k_2 z}{1 - k_3 z}\right)^{m-(k-1)} (1 - k_3 z)^{-(k+k-1)}\right] +$$
$$\frac{\partial}{\partial z}\left[\gamma_k^{(k)} \frac{m!}{(m-k)!} k_3^{k-k} (k_2 + k_1 k_3)^k \left(\frac{k_1 + k_2 z}{1 - k_3 z}\right)^{m-k} (1 - k_3 z)^{-(k+k)}\right]$$

$$= \gamma_1^{(k)} \frac{m!}{(m-1)!} k_3^{k-1}(k_2 + k_1k_3) \left[\left(\frac{k_1 + k_2z}{1 - k_3z} \right)^{m-1} \frac{\partial}{\partial z}(1 - k_3z)^{-(k+1)} + \right.$$

$$\left. (1 - k_3z)^{-(k+1)} \frac{\partial}{\partial z} \left(\frac{k_1 + k_2z}{1 - k_3z} \right)^{m-1} \right] +$$

$$\gamma_2^{(k)} \frac{m!}{(m-2)!} k_3^{k-2}(k_2 + k_1k_3)^2 \left[\left(\frac{k_1 + k_2z}{1 - k_3z} \right)^{m-2} \frac{\partial}{\partial z}(1 - k_3z)^{-(k+2)} + \right.$$

$$\left. (1 - k_3z)^{-(k+2)} \frac{\partial}{\partial z} \left(\frac{k_1 + k_2z}{1 - k_3z} \right)^{m-2} \right] + \cdots +$$

$$\gamma_{k-1}^{(k)} \frac{m!}{(m-(k-1))!} k_3^{k-(k-1)}(k_2 + k_1k_3)^{k-1} \left[\left(\frac{k_1 + k_2z}{1 - k_3z} \right)^{m-(k-1)} \frac{\partial}{\partial z}(1 - k_3z)^{-(k+k-1)} + \right.$$

$$\left. (1 - k_3z)^{-(k+k-1)} \frac{\partial}{\partial z} \left(\frac{k_1 + k_2z}{1 - k_3z} \right)^{m-(k-1)} \right] +$$

$$\gamma_k^{(k)} \frac{m!}{(m-k)!} k_3^{k-k}(k_2 + k_1k_3)^k \left[\left(\frac{k_1 + k_2z}{1 - k_3z} \right)^{m-k} \frac{\partial}{\partial z}(1 - k_3z)^{-(k+k)} + \right.$$

$$\left. (1 - k_3z)^{-(k+k)} \frac{\partial}{\partial z} \left(\frac{k_1 + k_2z}{1 - k_3z} \right)^{m-k} \right]$$

$$= \gamma_1^{(k)} \frac{m!}{(m-1)!} k_3^{k-1}(k_2 + k_1k_3) \left[\left(\frac{k_1 + k_2z}{1 - k_3z} \right)^{m-1} k_3(k+1)(1 - k_3z)^{-(k+2)} + \right.$$

$$\left. (1 - k_3z)^{-(k+1)}(m-1) \left(\frac{k_1 + k_2z}{1 - k_3z} \right)^{m-2} \frac{k_2 + k_1k_3}{(1 - k_3z)^2} \right] +$$

$$\gamma_2^{(k)} \frac{m!}{(m-2)!} k_3^{k-2}(k_2 + k_1k_3)^2 \left[\left(\frac{k_1 + k_2z}{1 - k_3z} \right)^{m-2} k_3(k+2)(1 - k_3z)^{-(k+3)} + \right.$$

$$\left. (1 - k_3z)^{-(k+2)}(m-2) \left(\frac{k_1 + k_2z}{1 - k_3z} \right)^{m-3} \frac{k_2 + k_1k_3}{(1 - k_3z)^2} \right] + \cdots +$$

$$\gamma_{k-1}^{(k)} \frac{m!}{(m-(k-1))!} k_3^{k-(k-1)}(k_2 + k_1k_3)^{k-1} \left[\left(\frac{k_1 + k_2z}{1 - k_3z} \right)^{m-(k-1)} k_3(k+k-1) \cdot \right.$$

$$(1 - k_3z)^{-(k+(k-1)-1)} +$$

$$\left. (1 - k_3z)^{-(k+(k-1))}(m-(k-1)) \left(\frac{k_1 + k_2z}{1 - k_3z} \right)^{m-k} \frac{k_2 + k_1k_3}{(1 - k_3z)^2} \right] +$$

$$\gamma_k^{(k)} \frac{m!}{(m-k)!} k_3^{k-k}(k_2 + k_1k_3)^k \left[\left(\frac{k_1 + k_2z}{1 - k_3z} \right)^{m-k} k_3(k+k) \right.$$

$$\left. (1 - k_3z)^{-(k+k+1)} + (1 - k_3z)^{-(k+2)}(m-k) \left(\frac{k_1 + k_2z}{1 - k_3z} \right)^{m-(k+1)} \frac{k_2 + k_1k_3}{(1 - k_3z)^2} \right]$$

Expanding the terms gives,

$$
\begin{aligned}
&G_m^{(k+1)}(z,t) = \tfrac{\partial}{\partial z} G_m^{(k)}(z,t) \\
&= \gamma_1^{(k)}(k+1)\frac{m!}{(m-1)!}k_3^k(k_2+k_1k_3)\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-1}(1-k_3z)^{-(k+2)} \\
&+\gamma_1^{(k)}\frac{m!}{(m-2)!}k_3^{k-1}(k_2+k_1k_3)^2\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-2}(1-k_3z)^{-(k+3)} \\
&+\gamma_2^{(k)}(k+2)\frac{m!}{(m-2)!}k_3^{k-1}(k_2+k_1k_3)^2\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-2}(1-k_3z)^{-(k+3)} \\
&+\gamma_2^{(k)}\frac{m!}{(m-3)!}k_3^{k-2}(k_2+k_1k_3)^3\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-3}(1-k_3z)^{-(k+4)}+\cdots \\
&+\gamma_{k-1}^{(k)}(k+k-1)\frac{m!}{(m-(k-1))!}k_3^2(k_2+k_1k_3)^{k-1}\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-(k-1)}(1-k_3z)^{-(k+k)} \\
&+\gamma_{k-1}^{(k)}\frac{m!}{(m-k)!}k_3(k_2+k_1k_3)^k\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-k}(1-k_3z)^{-(k+k+1)} \\
&+\gamma_k^{(k)}(k+k)\frac{m!}{(m-k)!}k_3(k_2+k_1k_3)^k\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-k}(1-k_3z)^{-(k+k+1)} \\
&+\gamma_k^{(k)}\frac{m!}{(m-(k+1))!}(k_2+k_1k_3)^{k+1}\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-(k+1)}(1-k_3z)^{-(k+k+2)}
\end{aligned}
$$

Grouping like terms and reorganizing algebraically gives,

$$
\begin{aligned}
&G_m^{(k+1)}(z,t) = \tfrac{\partial}{\partial z} G_m^{(k)}(z,t) \\
&= \gamma_1^{(k)}(k+1)\frac{m!}{(m-1)!}k_3^{(k+1)-1}(k_2+k_1k_3)\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-1}(1-k_3z)^{-(k+1+1)} \\
&+[\gamma_1^{(k)}+\gamma_2^{(k)}(k+1+1)]\frac{m!}{(m-2)!}k_3^{(k+1)-2}(k_2+k_1k_3)^2\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-2}(1-k_3z)^{-(k+1+2)} \\
&+[\gamma_2^{(k)}+\gamma_3^{(k)}(k+1+2)]\frac{m!}{(m-3)!}k_3^{(k+1)-3}(k_2+k_1k_3)^3\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-3}(1-k_3z)^{-(k+1+3)} \\
&\quad+\cdots+ \\
&[\gamma_{k-1}^{(k)}+\gamma_k^{(k)}(k+1+k-1)]\frac{m!}{(m-k)!}k_3^{(k+1)-k}(k_2 \\
&\qquad\qquad +k_1k_3)^k\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-k}(1-k_3z)^{-(k+1+k)} \\
&+\gamma_k^{(k)}\frac{m!}{(m-(k+1))!}(k_2+k_1k_3)^{k+1}\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-(k+1)}(1-k_3z)^{-(k+1+k+1)}
\end{aligned}
$$

$$
= \sum_{j=1}^{k+1}\gamma_j^{(k+1)}\frac{m!}{(m-j)!}k_3^{(k+1)-j}(k_2+k_1k_3)^j\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-j}(1-k_3z)^{-(k+1+j)}
$$

where $\gamma_j^{(k+1)} = \gamma_{j-1}^{(k)} + (k+1+j-1)\gamma_j^{(k)}$ for $j = 2,3,\cdots,k$ and $\gamma_1^{(k+1)} = (k+1)\gamma_1^{(k)}$, and $\gamma_{k+1}^{(k+1)} = \gamma_k^{(k)} = 1$ as required.

Case 2: $k+1 \geq m$

$$G_m^{(k+1)}(z,t) = \tfrac{\partial}{\partial z} G_m^{(k)}(z,t)$$

$$= \frac{\partial}{\partial z}\left[\sum_{j=1}^{m} \gamma_j^{(k)} \frac{m!}{(m-j)!} k_3^{k-j}(k_2+k_1k_3)^j \left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-j} (1-k_3z)^{-(k+j)}\right]$$

$$= \frac{\partial}{\partial z}\left[\gamma_1^{(k)} \frac{m!}{(m-1)!} k_3^{k-1}(k_2+k_1k_3)\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-1} (1-k_3z)^{-(k+1)}\right]$$

$$+\frac{\partial}{\partial z}\left[\gamma_2^{(k)} \frac{m!}{(m-2)!} k_3^{k-2}(k_2+k_1k_3)^2\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-2} (1-k_3z)^{-(k+2)}\right] + \cdots +$$

$$\frac{\partial}{\partial z}\left[\gamma_{m-1}^{(k)} \frac{m!}{(m-(m-1))!} k_3^{k-(m-1)}(k_2+k_1k_3)^{m-1}\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-(m-1)} (1-k_3z)^{-(k+m-1)}\right] +$$

$$\frac{\partial}{\partial z}\left[\gamma_m^{(k)} \frac{m!}{(m-m)!} k_3^{k-m}(k_2+k_1k_3)^m\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-m} (1-k_3z)^{-(k+m)}\right]$$

$$= \gamma_1^{(k)} \frac{m!}{(m-1)!} k_3^{k-1}(k_2+k_1k_3)\left[\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-1} \frac{\partial}{\partial z}(1-k_3z)^{-(k+1)} +\right.$$

$$\left.(1-k_3z)^{-(k+1)} \frac{\partial}{\partial z}\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-1}\right] +$$

$$\gamma_2^{(k)} \frac{m!}{(m-2)!} k_3^{k-2}(k_2+k_1k_3)^2\left[\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-2} \frac{\partial}{\partial z}(1-k_3z)^{-(k+2)} +\right.$$

$$\left.(1-k_3z)^{-(k+2)} \frac{\partial}{\partial z}\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-2}\right] + \cdots +$$

$$\gamma_{m-1}^{(k)} \frac{m!}{(m-(m-1))!} k_3^{k-(m-1)}(k_2+k_1k_3)^{m-1}\left[\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-(m-1)} \frac{\partial}{\partial z}(1-k_3z)^{-(k+m-1)}\right.$$

$$+$$

$$\left.(1-k_3z)^{-(k+m-1)} \frac{\partial}{\partial z}\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-(m-1)}\right] +$$

$$\gamma_m^{(k)} \frac{m!}{(m-m)!} k_3^{k-m}(k_2+k_1k_3)^m\left[\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-m} \frac{\partial}{\partial z}(1-k_3z)^{-(k+m)} +\right.$$

$$\left.(1-k_3z)^{-(k+m)} \frac{\partial}{\partial z}\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-m}\right]$$

$$= \gamma_1^{(k)} \frac{m!}{(m-1)!} k_3^{k-1}(k_2+k_1k_3)\left[\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-1} k_3(k+1)(1-k_3z)^{-(k+2)} + \right.$$
$$\left.(1-k_3z)^{-(k+1)}(m-1)\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-2}\frac{k_2+k_1k_3}{(1-k_3z)^2}\right] +$$
$$\gamma_2^{(k)} \frac{m!}{(m-2)!} k_3^{k-2}(k_2+k_1k_3)^2\left[\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-2} k_3(k+2)(1-k_3z)^{-(k+3)} + \right.$$
$$\left.(1-k_3z)^{-(k+2)}(m-2)\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-3}\frac{k_2+k_1k_3}{(1-k_3z)^2}\right] + \cdots +$$
$$\gamma_{m-1}^{(k)} \frac{m!}{(m-(m-1))!} k_3^{k-(m-1)}(k_2+k_1k_3)^{m-1}\left[\left(\frac{k_1+k_2z}{1-k_3z}\right)k_3(k+m-1)\right.$$
$$\left.(1-k_3z)^{-(k+m)} + (1-k_3z)^{-(k+m-1)}\frac{k_2+k_1k_3}{(1-k_3z)^2}\right] +$$
$$\gamma_m^{(k)} \frac{m!}{(m-m)!} k_3^{k-m}(k_2+k_1k_3)^m\left[k_3(k+m)(1-k_3z)^{-(k+1+m)}\right]$$

Expanding the terms gives,

$G_m^{(k+1)}(z,t) = \frac{\partial}{\partial z} G_m^{(k)}(z,t)$

$$= \gamma_1^{(k)}(k+1)\frac{m!}{(m-1)!}k_3^k(k_2+k_1k_3)\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-1}(1-k_3z)^{-(k+2)}$$
$$+\gamma_1^{(k)}\frac{m!}{(m-2)!}k_3^{k-1}(k_2+k_1k_3)^2\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-2}(1-k_3z)^{-(k+3)}$$
$$+\gamma_2^{(k)}(k+2)\frac{m!}{(m-2)!}k_3^{k-1}(k_2+k_1k_3)^2\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-2}(1-k_3z)^{-(k+3)}$$
$$+\gamma_2^{(k)}\frac{m!}{(m-3)!}k_3^{k-2}(k_2+k_1k_3)^3\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-3}(1-k_3z)^{-(k+4)}$$
$$+\gamma_3^{(k)}(k+3)\frac{m!}{(m-3)!}k_3^{k-2}(k_2+k_1k_3)^3\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-3}(1-k_3z)^{-(k+4)} + \cdots$$
$$+\gamma_{m-2}^{(k)}\frac{m!}{(m-(m-1))!}k_3^{k-(m-2)}(k_2+k_1k_3)^{m-1}\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-(m-1)}(1-k_3z)^{-(k+m)}$$
$$+\gamma_{m-1}^{(k)}(k+m-1)\frac{m!}{(m-(m-1))!}k_3^{k-(m-2)}(k_2+k_1k_3)^{m-1}\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-(m-1)}\cdot$$
$$(1-k_3z)^{-(k+m)}$$
$$+\gamma_{m-1}^{(k)}\frac{m!}{(m-m)!}k_3^{k-(m-1)}(k_2+k_1k_3)^m(1-k_3z)^{-(k+1+m)}$$
$$+\gamma_m^{(k)}(k+m)\frac{m!}{(m-m)!}k_3^{k-(m-1)}(k_2+k_1k_3)^m(1-k_3z)^{-(k+1+m)}$$

Grouping like terms and reorganizing algebraically gives,

$$G_m^{(k+1)}(z,t) = \tfrac{\partial}{\partial z} G_m^{(k)}(z,t)$$

$$= \gamma_1^{(k)}(k+1)\frac{m!}{(m-1)!}k_3^{(k+1)-1}(k_2+k_1k_3)\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-1}(1-k_3z)^{-(k+1+1)}$$

$$+[\gamma_1^{(k)}+\gamma_{m,2}^{(k)}(k+1+1)]\frac{m!}{(m-2)!}k_3^{(k+1)-2}(k_2+k_1k_3)^2\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-2}(1-k_3z)^{-(k+1+2)}$$

$$+[\gamma_2^{(k)}+\gamma_3^{(k)}(k+1+2)]\frac{m!}{(m-3)!}k_3^{(k+1)-3}(k_2+k_1k_3)^3\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-3}(1-k_3z)^{-(k+1+3)}$$

$$+\cdots$$

$$+[\gamma_{m-2}^{(k)}+\gamma_{m-1}^{(k)}(k+1+m-2)]\frac{m!}{(m-(m-1))!}k_3^{(k+1)-(m-1)}(k_2+k_1k_3)^{(m-1)}\cdot$$

$$\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-(m-1)}(1-k_3z)^{-(k+1+m-1)}$$

$$+[\gamma_{m-1}^{(k)}+\gamma_m^{(k)}(k+1+m-1)]\frac{m!}{(m-m)!}k_3^{(k+1)-m}(k_2+k_1k_3)^m\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-m}\cdot$$

$$(1-k_3z)^{-(k+1+m)}$$

$$= \sum_{j=1}^{m}\gamma_j^{(k+1)}\frac{m!}{(m-j)!}k_3^{(k+1)-j}(k_2+k_1k_3)^j\left(\frac{k_1+k_2z}{1-k_3z}\right)^{m-j}(1-k_3z)^{-(k+1+j)};$$

where $\gamma_j^{(k+1)} = \gamma_{j-1}^{(k)} + (k+1+j-1)\gamma_j^{(k)}$ for all $2 \le j \le m$, $\gamma_1^{(k+1)} = (k+1)\gamma_1^{(k)}$, Q.E.D.

and $\gamma_1^{(1)} = 1$ as required.

Remark. Given Theorem 1, we can directly derive the exact transition function of the B-D-C process $X(t)$ from the *nth* derivative of its PGF given by equation 1 (w.r.t. z) as indicated in Corollary 1.1.

Corollary 1.1. The analytical form of the transition function of the process $X(t)$ for $m,n \geq 0$ is given as:

$$P_{m,n}(t) = \begin{cases} \sum_{j=1}^{\min(m,n)} \gamma_j^{(n)} \frac{m!}{n!(m-j)!} k_3^{n-j}(k_2+k_1k_3)^j k_1^{m-j} & m,n \geq 1 \\ k_1^m + C(t) & n=0, \quad m \geq 1 \\ 1 & m,n=0 \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

Proof. Suppose $m,n\geq 0$

Case 1: $m,n\geq 1$

$$\begin{aligned} P_{m,n}(t) = P\{X(t)=n|X(0)=m\} &= \frac{G_m^{(n)}(0,t)}{n!} \\ &= \sum_{j=1}^{\min(m,n)} \gamma_j^{(n)} \frac{m!}{n!\,(m-j)!} k_3^{n-j}(k_2+k_1k_3)^j \left(\frac{k_1+k_2\cdot 0}{1-k_3\cdot 0}\right)^{m-j} (1-k_3\cdot 0)^{-(n+j)} \\ &= \sum_{j=1}^{\min(m,n)} \gamma_j^{(n)} \frac{m!}{n!\,(m-j)!} k_3^{n-j}(k_2+k_1k_3)^j k_1^{m-j}. \end{aligned}$$

Case 2: Let $n=0$ and $m\geq 1$

Given the generating function $G_m(z,t)$,

$$\begin{aligned} P_{m,0}(t) &= G_m(0,t) \\ &= \left(\frac{k_1+k_2\cdot 0}{1-k_3\cdot 0}\right)^m + C(t) = k_1^m + C(t). \end{aligned}$$

Case 3: Suppose $m=n=0$

Qualitatively, given that there are $m=0$ parasites at time $t=0$, there is a certain probability of having $n=0$ parasites at any time $t>0$ since no birth can occur. Now, since

$$P_{m,0}(t) = k_1^m + C(t)$$

; where

$$C(t) = 1 - \left(\frac{k_1+k_2}{1-k_3}\right)^m \quad \text{for} \quad k_1,k_2,k_3>0,$$

$$\implies P_{0,0}(t) = P\{X(t)=0|X(0)=0\} = P_{m,0}(t)|_{m=0} = k_1^0 + 1 - \left(\frac{k_1+k_2}{1-k_3}\right)^0 = 1 \quad \text{Q. E. D.}$$

Hence, the transition function $P_{m,n}(t)$ given by Corollary 1.1 is true for $m,n\geq 0$ as required.

Remark. Given the *nth* derivative of the B-D-C process $X(t)$ as defined under Theorem 1, we can further derive at least the first to third theoretical moments of $X(t)$ as indicated in Corollary 1.2.

Corollary 1.2. The expected value, variance and the 3rd uncentered moment of the B-D-C process $X(t)$ can be derived from equation 1 such that

$$\begin{aligned} E\{X(t)|X(0)=m\} &= \frac{\partial}{\partial z}G_m(z,t)\bigg|_{z=1} \\ &= \frac{m(k_2+k_1k_3)}{(1-k_3)^2}\left(\frac{k_1+k_2}{1-k_3}\right)^{m-1}. \end{aligned} \tag{4}$$

$$Var\{X(t)|X(0)=m\} = \frac{\partial^2}{\partial z^2}G_m(z,t)\bigg|_{z=1} + E\{X(t)|X(0)=m\} - [E\{X(t)|X(0)=m\}]^2, \tag{5}$$

where

$$\frac{\partial^2}{\partial z^2}G_m(z,t)\bigg|_{z=1} = \frac{2mk_3(k_2+k_1k_3)}{(1-k_3)^3}\left(\frac{k_1+k_2}{1-k_3}\right)^{m-1} + \frac{m(m-1)(k_2+k_1k_3)^2}{(1-k_3)^4}\left(\frac{k_1+k_2}{1-k_3}\right)^{m-2}.$$

Additionally,

$$E\{X^3(t)|X(0)=m\} = \frac{\partial^3}{\partial z^3}G_m(z,t)\bigg|_{z=1} + 3E\{X^2(t)|X(0)=m\} - 2E\{X(t)|X(0)=m\} \tag{6}$$

with

$$\begin{aligned} \frac{\partial^3}{\partial z^3}G_m(z,t)\bigg|_{z=1} = {} & \frac{6m(k_2+k_1k_3)k_3^2}{(1-k_3)^4}\left(\frac{k_1+k_2}{1-k_3}\right)^{m-1} + \frac{6m(m-1)(k_2+k_1k_3)^2k_3}{(1-k_3)^5}\left(\frac{k_1+k_2}{1-k_3}\right)^{m-2} \\ & + \frac{m(m-1)(m-2)(k_2+k_1k_3)^3}{(1-k_3)^6}\left(\frac{k_1+k_2}{1-k_3}\right)^{m-3}, \end{aligned}$$

and

$$E\{X^2(t)|X(0)=m\} = Var\{X(t)|X(0)=m\} + [E\{X(t)|X(0)=m\}]^2. \tag{7}$$

The 1st, 2nd and 3rd uncentered theoretical moments are useful for the generalised method of moments estimation of the B-D-C process described in section 2.

1.3 Numerical validation of the derived transition function of the B-D-C process

The derived transition function $P_{m,n}(t)$ and its theoretical moments (mean and variance functions) were further validated numerically based on 1 million exact stochastic simulations of the B-D-C process (for pseudo-code and R codes of the exact SSA of the B-D-C process, see Algorithm 1 and Appendix C) at given parameter values ($\lambda = 0.512$, $\mu = 0.35$ and $\rho = 0.003$) and $X_0 = 2$ as a case study. Algorithm 1 is a novel B-D-C simulation algorithm developed (in the current study) by adapting the standard Monte Carlo stochastic simulation technique to also include host survival status. The analytical transition probabilities (computed from corollary 1.1) and their corresponding Monte Carlo estimates ($\hat{p}_{2,n}(t) = k_n(t)/N$, where $k_n(t)$ is the frequency of occurrence for each value of $n \geq 0$ at time t and N is the total number of simulations) were compared over time (Figure 1). Figure 1 is a goodness-of-fit plot of the analytical and Monte Carlo estimates of the transition probabilities for different values $n \geq 0$ at six time points ($t = 1$ to $t = 6$). The estimated 95% confidence intervals for the Monte Carlo estimates of the transition probabilities were very small in size due to the smaller values of their respective standard errors as a result of the large number of simulations; and thus, not presented. However, the analytical transition probabilities were within the estimated confidence intervals at least 90% of the time (i.e., coverage probability=0.90). This is because, in practice, a 95% confidence level (based on normal approximation) does not necessarily guarantee a 95% coverage probability. The sampling distribution of the true mean and variance functions (equations 4 and 5) of the B-D-C process were respectively compared with their corresponding Monte Carlo estimates (Figure 2).

Remark. We can deduce from Figures 1 and 2 that the Monte Carlo estimates of the transition probabilities, mean and variance of the B-D-C process $X(t)$ are consistent with the theoretical values. Hence, the derived transition function can be used to accurately estimate transition probabilities of the B-D-C process.

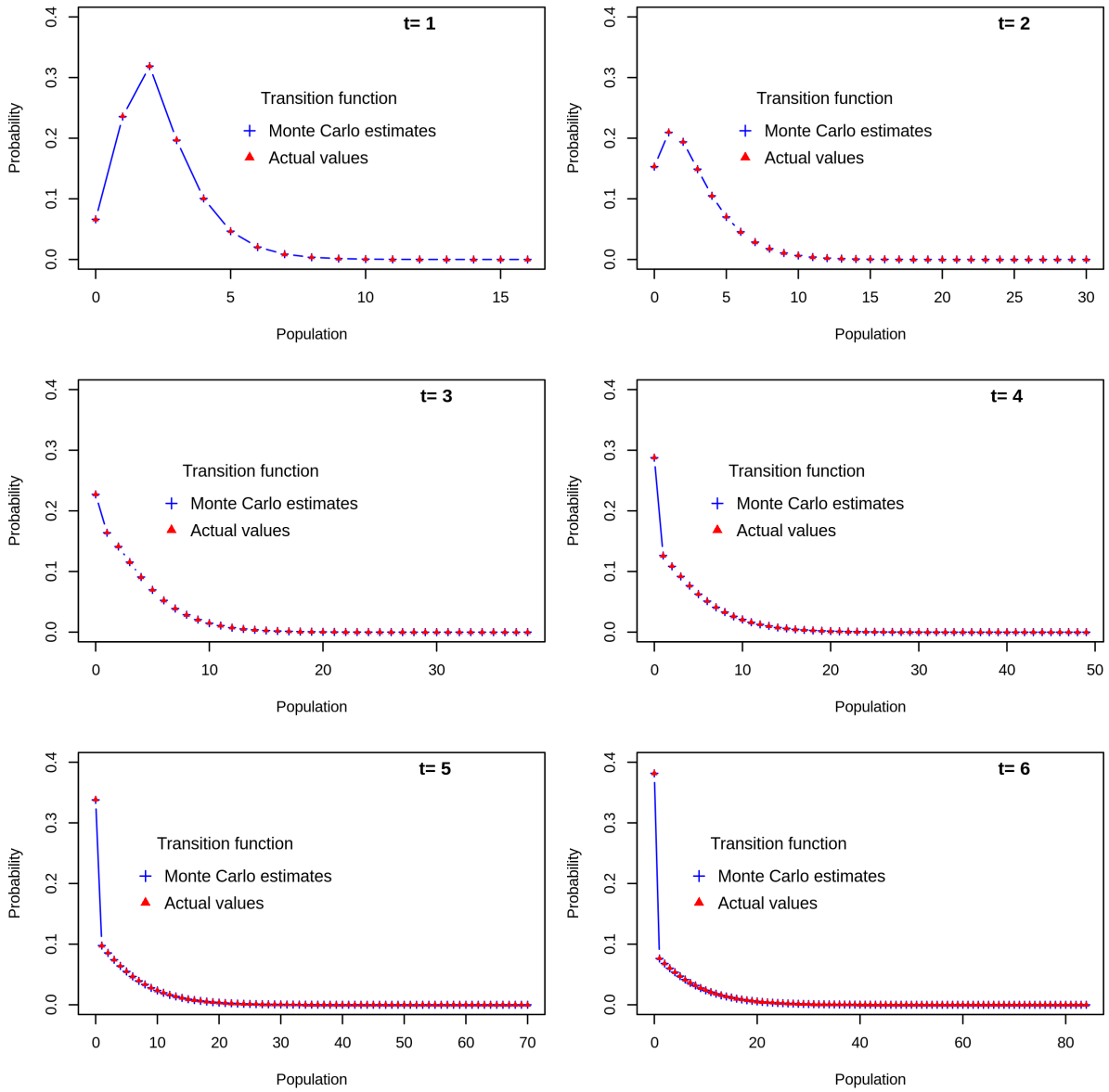

Figure 1: Comparison between analytical and numerical estimates of transition probabilities of the B-D-C process at given parameter values ($\lambda = 0.512$, $\mu = 0.35$ and $\rho = 0.003$) from $t = 1$ to $t = 6$ (in days).

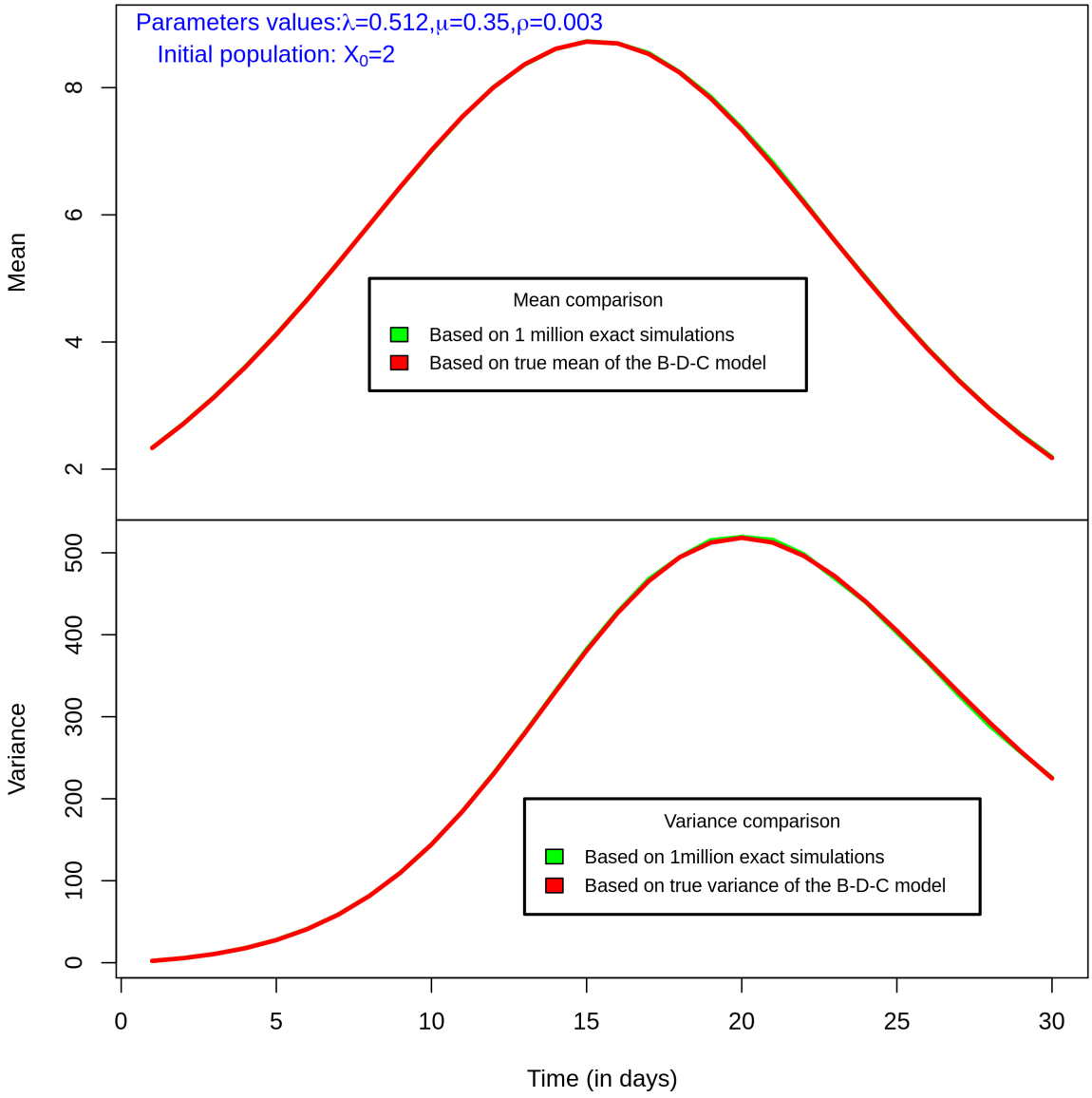

Figure 2: Comparison between analytical and Monte Carlo estimates of the mean and variance of the B-D-C process over time ($0 \leq t \leq 30$ days).

Algorithm 1: Exact SSA of the B-D-C process (pseudo-code)

Input: X, λ, μ, ρ, t, t_{final} and host survival status (s).

Output: Parasite numbers and survival status (alive: $s=1$; dead: $s=2$) recorded at discrete times ($t=1,2,\cdots,t_{\text{final}}$).

while $t < t_{final}$ *and* $s=1$ **do**
  Set initial time $t=t_0$, state $X=X_0$ and $s=1$.
  Calculate rates corresponding to birth (a_1), death (a_2) and catastrophe (a_3); such that $a_1=\lambda X$, $a_2=\mu X$ and $a_3=\rho X$.
  Compute the total rate, $a_0=\sum_{j=1}^{3} a_j$, for $j=1,2,3$ (from step 3).
  Determine the event to occur using a random number u from $Uniform(0,a_0)$
  if $0 \le u \le a_1$ **then**
    set $X=X+1$
  else
  end
  else if $a_1 < u \le a_1+a_2$ **then**
    set $X=X-1$
  else
    set $X=0$ and $s=2$ (then stop the simulation).
  end
  Generate time increment τ from $Exponential(a_0)$, and update the time such that $t=t+\tau$.
  Record (X,s) at the desired discrete times.
end

1.4 Behaviour of the transition, mean and variance functions of the B-D-C process

We further explored the behaviour of the exact transition function by varying the values of m ($1 \leq m \leq 10$) and n ($1 \leq n \leq 10$) at pre-specified parameter values ($\lambda = 0.512$, $\mu = 0.35$ and $\rho = 0.003$). Figure 3 shows that for any fixed value of $m > 0$ and $n > 0$, transition probabilities decrease over time for $t > 0$ and the probability of transitioning to state n decreases with increasing values of m. The behaviour of the B-D-C process's mean and variance functions was also explored over time at $m = 2$ and different parameter values by varying the rates in a full factorial design (Figure 4). This combination of parameter values was pre-specified to determine how the choice of values can affect the mean (or parasite population) distribution over time and help determine instances where the parasite population from the B-D-C process is low, moderate, or high (for the sake of three different *in silico* simulation experiments to be carried out to assess the computational speed and accuracy of the B-D-C parameter estimation techniques proposed under the next section 2). Generally, the variance of the process $X(t)$ is far greater than its mean at any value of $t > 1$ and any given parameter values.

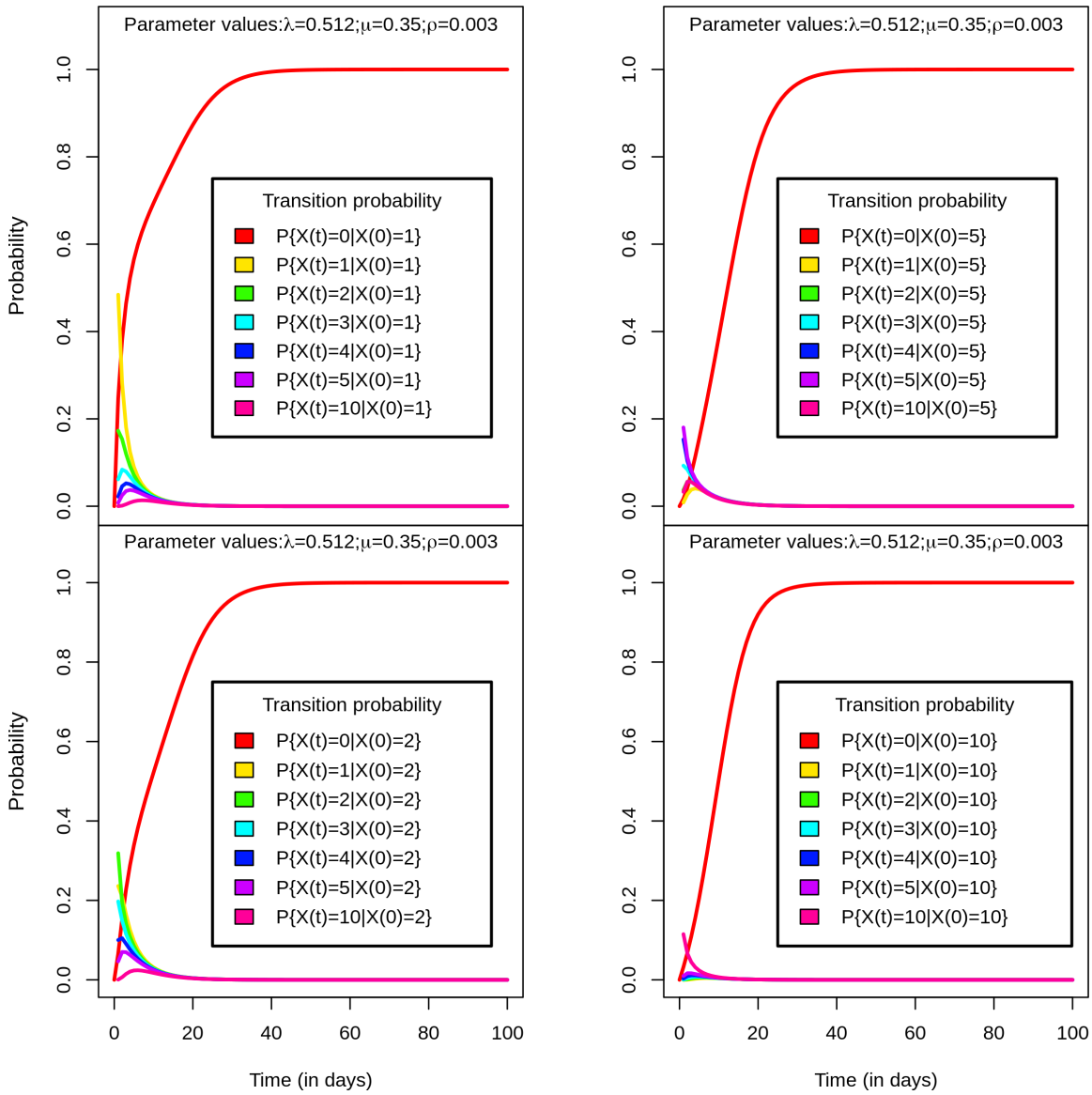

Figure 3: Exact transition probabilities of the B-D-C process at pre-specified parameter values ($\lambda = 0.512$, $\mu = 0.35$ and $\rho = 0.003$) for different values of m and n ($0 \leq m, n \leq 10$) over time ($0 \leq t \leq 100$ days).

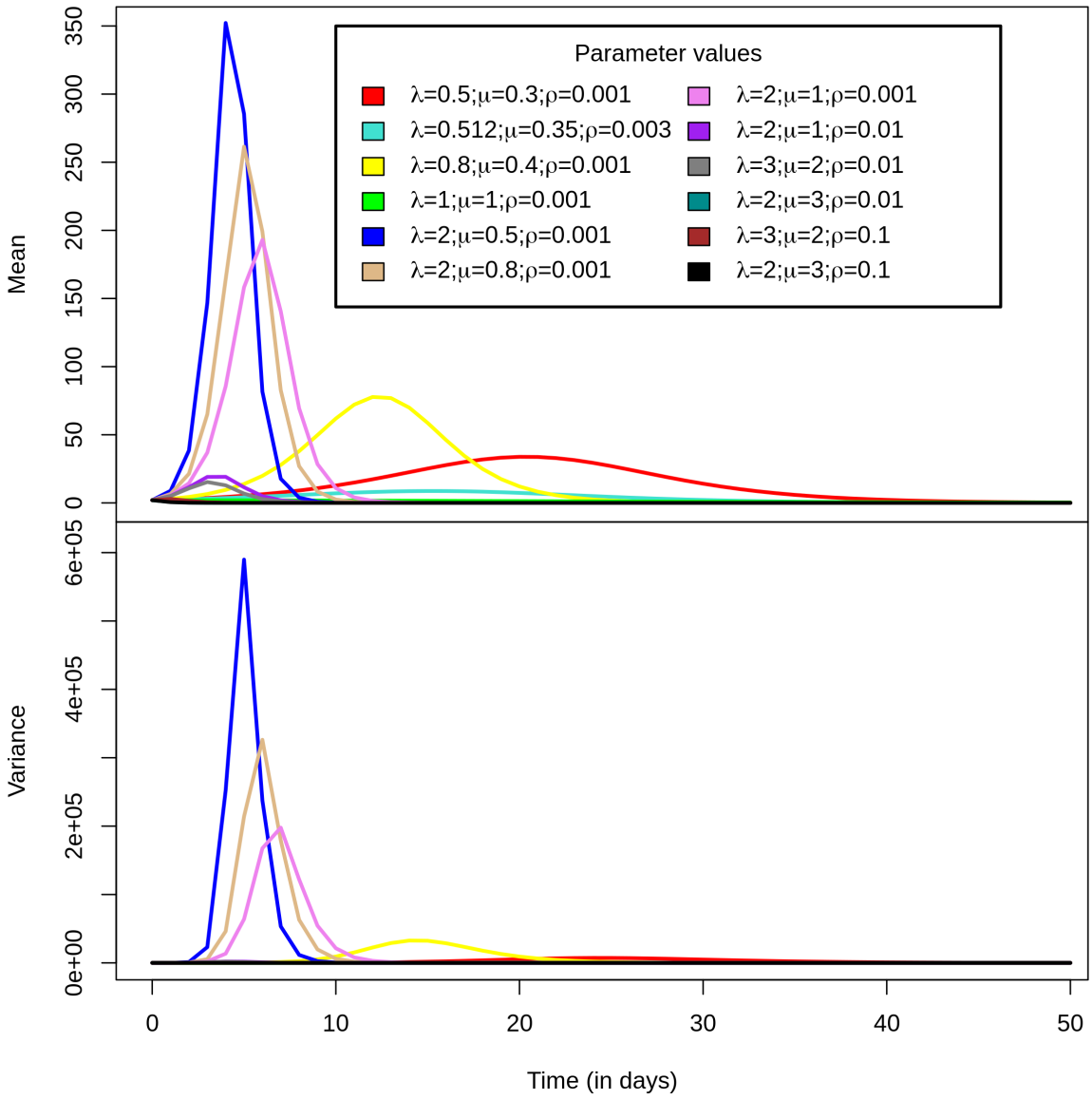

Figure 4: Exact mean and variance of the B-D-C process at different pre-specified parameter values ($0.5 \leq \lambda \leq 3$; $0.3 \leq \mu \leq 3$; $0.001 \leq \rho \leq 0.1$) over time ($0 \leq t \leq 50$ days).

2 Parameter estimation of the B-D-C process

We estimated the parameters of the B-D-C model by comparing between three estimation methods: Maximum likelihood estimation (MLE), Generalised method of moments (GMM) and Embedded Galton-Watson approach (GW), based on three different simulation experiments repeated 100 times (with $X_0 = 2$); such that the parasite population size were low ($\lambda = 3$, $\mu = 2$ and $\rho = 0.1$), moderate ($\lambda = 2$, $\mu = 1$ and $\rho = 0.01$) or high ($\lambda = 0.5$, $\mu = 0.3$ and $\rho = 0.001$). The main motivation is to identify which estimation method achieves good estimates (based on their bias, variance and mean square error of the estimates, respectively) but with faster computational time for the different simulation experiments. These parameter estimates will improve the summary statistics of a modified ABC based on a complex stochastic simulation model for the gyrodactylid-host system (see the main paper). The estimation methods were set-up to distinguish between the two zero states of the B-D-C process at any time t due to natural death of parasites or catastrophic extinction, such that

$$P\{X(t) = 0 \text{ and host alive} | X(0) = m\} = k_1^m$$

and

$$P\{X(t) = 0 \text{ and host dead} | X(0) = m\} = 1 - \left(\frac{k_1 + k_2}{1 - k_3}\right)^m,$$

with k_1, k_2 and k_3 defined as before.

For each *in silico* simulation experiment (Cases 1-3), we simulated data for 50 independent hosts. We recorded parasite numbers for each host over a 17-day period (odd-numbered days from day 1 to 17) to represent one simulation realisation. The simulation is set-up like the observed empirical data for the sophisticated stochastic model. We fit the MLE (described in section 2.1) as well as the GMM and GW estimators (described in sections 2.2 and 2.3, respectively) to the 50 simulated data for 100 different simulation realisations, respectively. The bias, variance and mean square error of the parameter estimates are computed for comparison. For any biased estimator, we explored its consistency as the sample size becomes large. The three different *in silico* simulation experiments were considered in determining whether the estimation methods will give reasonable estimates irrespective of the parasite population size over time as well as investigate the computational speed in such instances. A major assumption made for these parameter estimation methods is that data from different simulations and *in silico* simulation experiments are independent and identically distributed.

2.1 Maximum likelihood estimator

Suppose $X = \{X(t_1), X(t_2), \cdots X(t_9)\}$ is the number of parasites on a host at time t_i, $i = 1, 2, 3, \cdots, 9$ and follows the B-D-C process with transition function defined in Corollary 1.1. Let n_r be the total number of hosts for each realisation for $r = 1, 2, \cdots, 100$ (n_r was set at 50 for each simulation realisation). Suppose n_{ki} is the number of parasites on the kth host for $k = 1, 2, \cdots, n_r$ at time t_i, $i = 1, 2, 3, \cdots, 9$. By employing the Markov property, the likelihood function corresponding to each simulation realisation for r is given as

$$\begin{aligned} L(X, \lambda, \mu, \rho) &= \prod_{k=1}^{n_r} P\{X(t_9) = n_{k9}, X(t_8) = n_{k8}, \cdots, X(t_1) = n_{k1} | X(t_0) = n_{k0}\} \\ &= \prod_{k=1}^{n_r} P\{X(t_1) = n_{k1} | X(0) = n_{k0}\} \times P\{X(t_2) = n_{k2} | X(t_1) = n_{k1}\} \times \cdots \times \\ &\qquad P\{X(t_9) = n_{k9} | X(t_8) = n_{k8}\} \\ &= \prod_{k=1}^{n_r} \prod_{i=1}^{9} P\{X(t_i - t_{i-1}) = n_{ki} | X(0) = n_{k0}\}. \end{aligned} \tag{8}$$

Taking logarithm of $L(X, \lambda, \mu, \rho)$ (equation 8) gives the log-likelihood function $l(X, \lambda, \mu, \rho)$ (equation 9):

$$l(X, \lambda, \mu, \rho) = \sum_{k=1}^{n_r} \sum_{i=1}^{9} log\left(P\{X(t_i - t_{i-1}) = n_{ki} | X(t_0) = n_{k0}\}\right) \quad \text{for} \quad r = 1, 2, \cdots, 100. \tag{9}$$

Now, suppose $\theta = [\lambda, \mu, \rho]^T$ and $\Theta = \{\theta \in \mathbb{R}^3 | \lambda > 0,\ \mu > 0,\ \rho > 0\}$ is the parameter space of θ, then the MLE ($\hat{\theta}_{\text{MLE}}$) is obtained such that

$$\hat{\theta}_{\text{MLE}} = \arg\max_{\theta \in \Theta} l(X, \theta). \tag{10}$$

The maximum likelihood estimates were obtained numerically in R across the entire parameter space Θ by maximising the log-likelihood function for each simulation experiment (Cases 1-3). Due to round-off errors in storing the transition probabilities for large parasite numbers in R version 3.6.3 [11], arbitrary precision arithmetic in Julia version 1.5.3 [1] was employed to store these probabilities as double-precision binary floating-point numbers (i.e., 64-bit floating-point numbers with 15 decimal digits of precision); however, arbitrary-precision arithmetic is considerably slow. Additionally, the probabilities were efficiently computed recursively by pre-calculating various components of the transition function before computing the log-likelihood function (for the Julia codes of the recursive transition function and log-likelihood function, see Appendix D).

2.2 Generalised method of moments estimator

Let $\varphi_d(\theta,t_i) = E\{X_r^d(t_i)|X(0)=m\}$ represent the first d exact theoretical moments and $\bar{\varphi}_d(X,t_i)$ denoting their corresponding sample moments for $d=1,2,3$ for simulation realisation r such that

$$\varphi_1(\theta,t_i) = \frac{m(k_2+k_1k_3)}{(1-k_3)^2}\left(\frac{k_1+k_2}{1-k_3}\right)^{m-1},$$

$$\varphi_2(\theta,t_i) = \frac{2mk_3(k_2+k_1k_3)}{(1-k_3)^3}\left(\frac{k_1+k_2}{1-k_3}\right)^{m-1} + \frac{m(m-1)(k_2+k_1k_3)^2}{(1-k_3)^4}\left(\frac{k_1+k_2}{1-k_3}\right)^{m-2} + \varphi_1(\theta,t_i),$$

$$\begin{aligned}\varphi_3(\theta,t_i) = {} & \frac{6m(k_2+k_1k_3)k_3^2}{(1-k_3)^4}\left(\frac{k_1+k_2}{1-k_3}\right)^{m-1} + \frac{6m(m-1)(k_2+k_1k_3)^2k_3}{(1-k_3)^5}\left(\frac{k_1+k_2}{1-k_3}\right)^{m-2} \\ & + \frac{m(m-1)(m-2)(k_2+k_1k_3)^3}{(1-k_3)^6}\left(\frac{k_1+k_2}{1-k_3}\right)^{m-3} + 3\varphi_2(\theta,t_i) - 2\varphi_1(\theta,t_i);\end{aligned}$$

and their corresponding sample moments at time t_i, $i=1,2\cdots,9$ is given by

$$\bar{\varphi}_d(X,t_i) = \frac{1}{n_r}\sum_{k=1}^{n_r} X_k^d(t_i) \text{ for } d=1,2,3 \text{ and } r=1,2,\cdots,100.$$

Suppose $\varphi_4(\theta,t_i)$ is the theoretical probability of catastrophe and let $\bar{\varphi}_4(X,t_i)$ be its sample estimate at time t_i such that

$$\varphi_4(\theta,t_i) = 1-\left(\frac{k_1+k_2}{1-k_3}\right)^m \qquad \text{and} \qquad \bar{\varphi}_4(X,t_i) = \frac{\omega_{ir}}{n_r};$$

where ω_{ir} is the number of hosts who died at time t_i out of the n_r total number of hosts who were alive at time 0 in the rth simulation realisation. Here, we assume an infected host dies and parasite population extinction occur at a rate proportional to the number of parasites $X(t_i)$ on host at time t_i.

Let $g(X,\theta)$ be a 9×4 matrix with entries $g_{ij}(X,\theta)$ (for $1\le i\le 9$ and $1\le j\le 4$) defined such that

$$g_{ij}(X,\theta) = \varphi_j(\theta,t_i) - \bar{\varphi}_j(X,t_i) \tag{11}$$

and satisfying the 9×1 unconditional moment conditions:

$$E\left[g(X,\theta)\right] = 0. \tag{12}$$

A two-step generalised method of moments technique originally proposed by Hansen [7] was employed to numerically obtain efficient GMM. Now, suppose that

$$\bar{g}_j(\theta) = \frac{1}{9}\sum_{i=1}^{9} g_{ij}(X,\theta) \;\text{ for }\; j = 1,2,3,4. \tag{13}$$

Then, the GMM estimator is given by

$$\hat{\theta}_{\text{GMM}} = \arg\min_{\theta\in\Theta} \bar{g}(\theta)^T \Omega(\theta^*)^{-1} \bar{g}(\theta), \tag{14}$$

where $\Omega(\theta^*)^{-1}$ is a positive-definite weighting matrix (a diagonal matrix with leading diagonal representing the multiplicative inverse of the variance of jth column entries of matrix g evaluated at θ^* as defined in equation 15), and θ^* is the initial parameter estimates at the first estimation step with the weighting matrix being a 4×4 identity matrix. For the weighting matrix at the second estimation step, we assume that the column entries or components (g_j) of matrix $g(X,\theta^*)$ are uncorrelated for $j = 1,2,3,4$. The 2-step algorithm for obtaining $\hat{\theta}_{\text{GMM}}$ is therefore:

1. Estimate $\theta^* = \arg\min_{\theta\in\Theta} \bar{g}(\theta)^T \bar{g}(\theta)$.
2. Estimate the weighting matrix $\Omega(\theta^*)^{-1}$ given θ^*

$$\Omega(\theta^*)^{-1} = \begin{array}{c} \\ g_1 \\ g_2 \\ g_3 \\ g_4 \end{array} \begin{array}{c} \begin{array}{cccc} g_1 & g_2 & g_3 & g_4 \end{array} \\ \begin{pmatrix} \frac{1}{\sigma^2_{g_1(\theta^*)}} & 0 & 0 & 0 \\ 0 & \frac{1}{\sigma^2_{g_2(\theta^*)}} & 0 & 0 \\ 0 & 0 & \frac{1}{\sigma^2_{g_3(\theta^*)}} & 0 \\ 0 & 0 & 0 & \frac{1}{\sigma^2_{g_4(\theta^*)}} \end{pmatrix} \end{array}.$$

where $\sigma^2_{g_j(\theta^*)}$ is the variance of jth column entries of matrix $g(X,\theta^*)$ such that

$$\sigma^2_{g_j(\theta^*)} = \frac{\sum_{i=1}^{9} \left[g_{ij}(X,\theta^*) - \bar{g}_j(\theta^*)\right]^2}{8} \qquad \text{for} \quad j = 1,2,3,4. \tag{15}$$

3. Compute the required estimates $\hat{\theta}_{\text{GMM}}$ given by equation 14.

By the weak law of large numbers,

$$\bar{\varphi}_j(X,t_i) \xrightarrow{p} \varphi_j(\theta,t_i) \;\text{ as }\; n_r \to \infty \;\text{ for }\; 1\le i\le 9,\;\; 1\le j\le 4 \;\text{ and }\; 1\le r\le 100.$$

Hence, if the moment conditions are met, GMM estimates are consistent as the sample size increases.

2.3 Embedded Galton-Watson estimation approach

A single trajectory of a discretely-observed linear birth-death process (LBDP) with birth rate λ, death rate μ and equal inter-observation times Δt corresponds to a Galton-Watson (GW) process with offspring mean $m = m(\Delta t)$ and variance $\sigma^2 = \sigma^2(\Delta t)$ [2, 8]; where

$$\hat{m} = \frac{\sum_{i=1}^{\mathcal{T}} Z_i}{\sum_{i=1}^{\mathcal{T}} Z_{i-1}} \quad \text{and} \quad \hat{\sigma}^2 = \frac{1}{\mathcal{T}} \sum_{i=1}^{\mathcal{T}} Z_{i-1} \left(\frac{Z_i}{Z_{i-1}} - \hat{m} \right)^2$$

with $Z_0 = Z(0)$ denoting the initial population size at time $t = 0$, the successive population sizes represented as $Z_1 = Z(\Delta t), Z_2 = Z(2\Delta t), \cdots, Z_{\mathcal{T}} = Z(\mathcal{T}\Delta t)$, and $\mathcal{T}$ being the final observed time. The GW process is thus the most basic (branching process) model for a population changing through time. It assumes that given an initial number of individuals or parasites ($Z_0 = 1$) at the zeroth generation (or at $t = 0$), each individual in the population at any time $t \geq 1$ can give birth (at a constant rate $\lambda > 0$) to offspring with the same probability distribution, independently of one another; whereas, each individual can die (at a constant rate $\mu > 0$) until population extinction may occur when the offspring mean $m < 1$. Also, analytical estimates of the birth and death rates (λ and μ) given a finite number of independent trajectories (or initial population size > 1) of a discretely-observed LBDP with equal inter-observation times conditioned on population non-extinction have been proposed in the general case by Davison et al. [2] as an embedded GW process (i.e., the more general or extended case with $Z_0 \geq 1$).

Now, let suppose $Z_{i,k}$ is the number of parasites on each kth surviving host at time t_i for $i = 1, 2, \cdots, 9$, $k = 1, 2, \cdots, s_r$, $r = 1, 2, \cdots, 100$ and equal inter-observation times $\Delta t = t_i - t_{i-1}$ (where s_r is the total number of surviving hosts for each rth simulation realisation). Suppose $Z_{i,k}$ with birth rate $\lambda > 0$ and death rate $\mu > 0$, is a GW process with mean $m(\Delta t) > 1$ and variance $\sigma^2(\Delta t) > 0$ (where $Z_{0,k} = 2$ in our case). Then, the analytical estimates of the birth and death rates for $k > 1$ independent trajectories of the LBDP and equal Δt ($\Delta t = 2$ in our case) is given as

$$\hat{\lambda} = \frac{\log \hat{m}}{2\Delta t} \left\{ \frac{\hat{\sigma}^2}{\hat{m}(\hat{m} - 1)} + 1 \right\} \quad \text{and} \quad \hat{\mu} = \frac{\log \hat{m}}{2\Delta t} \left\{ \frac{\hat{\sigma}^2}{\hat{m}(\hat{m} - 1)} - 1 \right\}; \tag{16}$$

where

$$\hat{m} = \frac{\sum_{k=1}^{s_r} \sum_{i=1}^{9} Z_{i,k}}{\sum_{k=1}^{s_r} \sum_{i=1}^{9} Z_{i-1,k}} \quad \text{and} \quad \hat{\sigma}^2 = \frac{1}{9 s_r} \sum_{k=1}^{s_r} \sum_{i=1}^{9} Z_{i-1,k} \left(\frac{Z_{i,k}}{Z_{i-1,k}} - \hat{m} \right)^2.$$

From equation 16, it can be observed that when the mean $\hat{m} < 1$ (case of subcritical

process), $\hat{\lambda} < 0$; and at $\hat{m} = 1$ (case of critical process), $\hat{\lambda}$ and $\hat{\mu}$ are undefined. According to Davison et al. [2], $\hat{\lambda} = \hat{\mu} = \frac{\hat{\sigma}^2}{2\Delta t}$ when $\hat{m} = 1$. In the non-critical case where the offspring mean $\hat{m} > 1$, the birth and death rates of the B-D-C process are estimated based on equation 16 by conditioning on non-extinction or survival of host. To estimate the catastrophic rate (ρ) when $\hat{m} > 1$, we estimate ρ using maximum likelihood estimation given by equation 18 based on the GW estimates of the birth and death rates ($\hat{\lambda}$ and $\hat{\mu}$ from equation 16). Suppose hosts die in the time interval $(t_{k_{i-1}}, t_{k_i}]$ at rate ρ, then the log-likelihood function is given as

$$l(\rho; \hat{\lambda}, \hat{\mu}) = \sum_{k=1}^{n_r - s_r} \log\left[C(t_{k_i}) - C(t_{k_{i-1}})\right] \quad \text{for} \quad r = 1, 2, \cdots, 100; \tag{17}$$

where $C(t)$ is the theoretical probability of catastrophe given by

$$C(t) = 1 - \left(\frac{k_1 + k_2}{1 - k_3}\right)^m,$$

with $k_1 = \frac{\nu_0\nu_1(1-\sigma)}{\nu_1 - \sigma\nu_0}$, $k_2 = \frac{\nu_1\sigma - \nu_0}{\nu_1 - \sigma\nu_0}$, $k_3 = \frac{1-\sigma}{\nu_1 - \sigma\nu_0}$, $\sigma = \mathrm{e}^{-\hat{\lambda}(\nu_1 - \nu_0)t}$, $\nu_0 = \frac{(\hat{\lambda}+\hat{\mu}+\rho) - \sqrt{(\hat{\lambda}+\hat{\mu}+\rho)^2 - 4\hat{\mu}\hat{\lambda}}}{2\hat{\lambda}}$, and $\nu_1 = \frac{(\hat{\lambda}+\hat{\mu}+\rho) + \sqrt{(\hat{\lambda}+\hat{\mu}+\rho)^2 - 4\hat{\mu}\hat{\lambda}}}{2\hat{\lambda}}$. The MLE estimator of the catastrophe rate given the GW estimates of the birth and death rates is obtained such that

$$\hat{\rho} = \arg\max_{\rho \in \Theta} l(\rho; \hat{\lambda}, \hat{\mu}). \tag{18}$$

Remark. The fastest computational time is expected to be achieved from the analytical estimation of the birth and death rates ($\hat{\lambda}$ and $\hat{\mu}$, respectively) using the GW estimator given by equation 16 together with the MLE estimator of the catastrophe rate ρ given by equation 18 compared to the computational time required for the MLE and GMM estimators given by equations equations 10 and 14, respectively. For instances where the offspring mean $\hat{m} \leq 1$ (for critical and subcritical cases), we estimated the B-D-C model parameters (λ, μ and ρ) using the GMM estimator given by equation 14 to obtain more consistent and less biased estimates within a fast computational time. Generally, GMM estimation is relatively faster in computational time than MLE. In this study, the GW estimation approach is employed if and only if the offspring mean $m > 1$ within the simulation experiments.

2.4 Comparison between the estimation methods

The best estimation methods are identified in this study by exploring the trade-off between estimation accuracy and computational speed. The MLE and GMM estimates were computed with their corresponding 95% confidence interval (C.I) for the three simulation experiments based on 50 independent hosts, respectively (Table 1). The GW

estimation in this current study, only held for simulation Case 1 (where the true parameter values were set at $\lambda = 0.5$, $\mu = 0.3$ and $\rho = 0.001$) since it was the only instance where the offspring mean m from the Galton-Watson estimation approach > 1 (due to large parasite numbers obtained in simulation Case 1). Thus, GW estimation was not employed for simulation experiments two and three since the offspring mean $m \leq 1$. The GW estimates with their 95% confidence interval for simulation Case 1 are also presented in Table 1. The estimation methods' performance was compared by estimating the bias, variance, and mean square error of estimates (based on equations 19–21) for each parameter of the B-D-C process (Table 1). Given $\theta = [\lambda, \mu, \rho]^T$, suppose the true parameter values of the simulated data is $\theta_0 = [\lambda_0, \mu_0, \rho_0]^T$ and its corresponding estimate is $\hat{\theta} = [\hat{\lambda}, \hat{\mu}, \hat{\rho}]^T$; then the bias, variance and mean square error of estimates for each parameter θ_i for $i = 1, 2, 3$ are computed such that

$$bias(\hat{\theta}_i) = E(\hat{\theta}_i) - \theta_{0i} = \frac{1}{100}\sum_{r=1}^{100}\hat{\theta}_{ir} - \theta_{0i}, \tag{19}$$

$$Var(\hat{\theta}_i) = E[(\hat{\theta}_i - E(\hat{\theta}_i))^2], \tag{20}$$

and

$$MSE(\hat{\theta}_i) = E[(\hat{\theta}_i - \theta_{0i})^2] = Var(\hat{\theta}_i) + [bias(\hat{\theta}_i)]^2. \tag{21}$$

For simulation Case 1, the three estimation methods generally performed well based on their bias, variance and mean square error estimates (Table 1); however, the bias of MLE was consistently lower and relatively efficient (less variance) in estimating the B-D-C model parameters based on the three simulation experiments. Nevertheless, the Galton-Watson estimation approach was significantly faster in computational speed than MLE and GMM estimation methods (Table 4). The computational time required for MLE was significantly higher than the GMM estimation for simulation Cases 1 and 2, where parasite numbers were relatively higher than that of simulation Case 3 (Figure 6). This finding is due to the complexity of the B-D-C process's transition and log-likelihood functions. The consistency, efficiency and computational time of GW and GMM estimates were also investigated as the sample size increases for each simulation experiment at sample sizes of 50, 100 and 500 (Tables 2–4). The GW and GMM estimates were relatively efficient as the sample size increased from 50 to 500. The B-D-C process's mean behaviour was also bounded based on the MLE and GMM for all three *in silico* simulation experiments as goodness-of-fit plots (Figure 5). The mean trajectory based on the expected MLE and GMM estimates and actual parameter values perfectly averaged the mean trajectories evaluated at MLEs from the 100 realisations, respectively, across the three simulation cases. Based on the computational speed and accuracy measures of the respective

estimation methods, the Galton-Watson estimation approach (given by equations 16 and 18) and the GMM estimator (given by equation 14) can be used together to obtain fast and reasonably accurate estimates of the B-D-C model parameters in refining the modified ABC algorithm for the sophisticated stochastic model. The estimation must be done such that the Galton-Watson method should be used when the offspring mean $m > 1$ and GMM employed when $m \leq 1$.

Table 1: Maximum likelihood, Generalised method of moments and Galton-Watson estimates based on the three different *in silico* simulation experiments (Cases 1-3) of 50 hosts respectively.

Simulation	Method	Parameters	$\hat{\theta}$	θ_0	bias($\hat{\theta}$)	Var($\hat{\theta}$)	MSE($\hat{\theta}$)	95% C.I
Case 1	MLE	λ	0.503	0.5	0.003	0.001	0.001	0.496-0.509
		μ	0.303	0.3	0.003	0.001	0.001	0.296-0.309
		ρ	0.001	0.001	7.29×10^{-5}	1.42×10^{-7}	1.48×10^{-7}	0.0009-0.0011
	GMM	λ	0.526	0.5	0.026	0.140	0.141	0.452-0.599
		μ	0.330	0.3	0.029	0.139	0.141	0.256-0.403
		ρ	0.001	0.001	-3.97×10^{-5}	1.28×10^{-7}	1.29×10^{-7}	0.0009-0.0011
	GW	λ	0.507	0.5	0.007	0.020	0.020	0.479-0.534
		μ	0.322	0.3	0.022	0.020	0.020	0.294-0.349
		ρ	0.0073	0.001	0.006	4.81×10^{-6}	4.46×10^{-5}	0.0068-0.0077
Case 2	MLE	λ	1.998	2	-0.002	0.031	0.031	1.964-2.033
		μ	1.009	1	0.001	0.031	0.031	0.975-1.044
		ρ	0.010	0.01	0.000	4.10×10^{-6}	4.13×10^{-6}	0.009-0.011
	GMM	λ	1.879	2	-0.121	0.474	0.489	1.744-2.014
		μ	0.932	1	-0.068	0.206	0.211	0.843-1.021
		ρ	0.010	0.01	2.06×10^{-4}	5.17×10^{-6}	5.21×10^{-6}	0.009-0.011
Case 3	MLE	λ	2.953	3	-0.047	0.222	0.224	2.861-3.045
		μ	1.969	2	-0.031	0.176	0.177	1.887-2.050
		ρ	0.107	0.1	0.007	0.001	0.001	0.101-0.112
	GMM	λ	2.728	3	-0.272	1.299	1.373	2.501-2.952
		μ	1.833	2	-0.167	0.700	0.728	1.669-1.997
		ρ	0.104	0.1	0.004	0.001	0.001	0.099-0.109

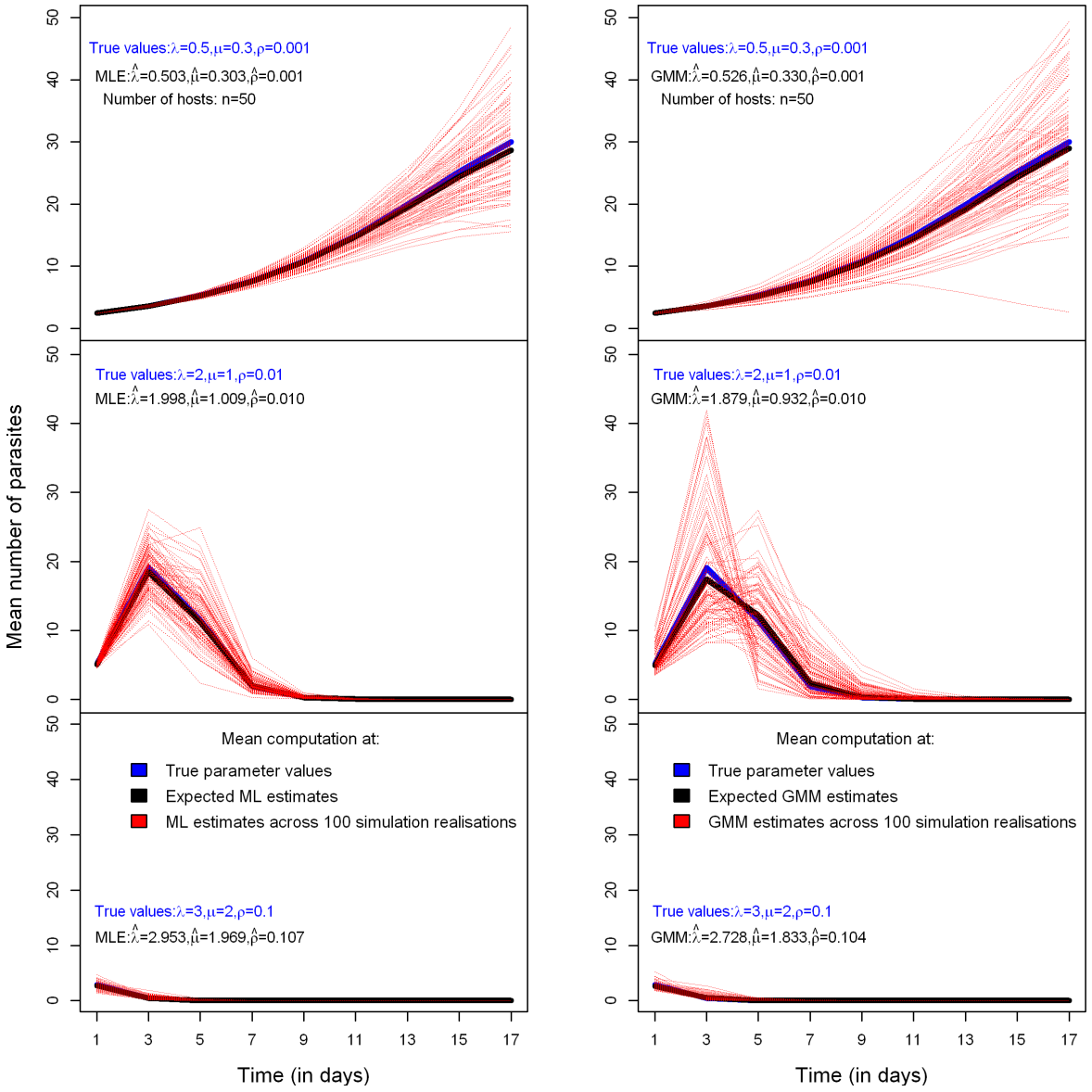

Figure 5: Bounded mean behaviour of the B-D-C process based on the MLE and GMM estimates over time across the three different *in silico* simulation experiments ($n = 50$ hosts).

Table 2: Effect of sample size on the GMM estimates based on the three different *in silico* simulation experiments (Cases 1-3).

Simulation	Sample size	Parameters	$\hat{\theta}$	θ_0	bias($\hat{\theta}$)	Var($\hat{\theta}$)	MSE($\hat{\theta}$)	95% C.I
Case 1	50	λ	0.526	0.5	0.026	0.140	0.141	0.452-0.599
		μ	0.330	0.3	0.029	0.139	0.141	0.256-0.403
		ρ	0.001	0.001	-3.97×10^{-5}	1.28×10^{-7}	1.29×10^{-7}	0.0009-0.0011
	100	λ	0.494	0.5	-0.006	0.026	0.026	0.462-0.526
		μ	0.294	0.3	-0.006	0.029	0.029	0.261-0.327
		ρ	0.001	0.001	-9.78×10^{-6}	7.39×10^{-6}	7.40×10^{-8}	0.0009-0.0011
	500	λ	0.496	0.5	-0.004	0.003	0.003	0.485-0.507
		μ	0.296	0.3	-0.004	0.003	0.003	0.285-0.307
		ρ	0.001	0.001	2.09×10^{-6}	1.18×10^{-8}	1.18×10^{-8}	0.0009-0.0011
Case 2	50	λ	1.879	2	-0.121	0.474	0.489	1.744-2.014
		μ	0.932	1	-0.068	0.206	0.211	0.843-1.021
		ρ	0.010	0.01	2.06×10^{-4}	5.17×10^{-6}	5.21×10^{-6}	0.009-0.011
	100	λ	1.808	2	-0.192	0.347	0.383	1.693-1.923
		μ	0.884	1	-0.116	0.140	0.154	0.810-0.957
		ρ	0.010	0.01	0.001	3.46×10^{-6}	3.69×10^{-6}	0.009-0.0011
	500	λ	1.927	2	-0.073	0.112	0.117	1.862-1.993
		μ	0.961	1	-0.039	0.046	0.048	0.919-1.003
		ρ	0.010	0.01	3.21×10^{-5}	3.59×10^{-7}	3.60×10^{-7}	0.009-0.011
Case 3	50	λ	2.728	3	-0.272	1.299	1.373	2.501-2.952
		μ	1.833	2	-0.167	0.700	0.728	1.669-1.997
		ρ	0.104	0.1	0.004	0.001	0.001	0.099-0.109
	100	λ	2.579	3	-0.421	0.628	0.806	2.423-2.734
		μ	1.728	2	-0.272	0.294	0.368	1.622-1.834
		ρ	0.100	0.1	-0.001	3.75×10^{-4}	3.76×10^{-4}	0.095-0.102
	500	λ	2.916	3	-0.084	0.243	0.250	2.820-3.013
		μ	1.953	2	-0.047	0.116	0.118	1.889-2.020
		ρ	0.100	0.1	-2.27×10^{-4}	8.06×10^{-5}	8.07×10^{-5}	0.098-0.102

Table 3: Effect of sample size on the Galton-Watson estimates (based on simulation Case 1).

Sample size	Parameters	$\hat{\theta}$	θ_0	bias($\hat{\theta}$)	Var($\hat{\theta}$)	MSE($\hat{\theta}$)	95% C.I
50	λ	0.507	0.5	0.007	0.020	0.020	0.479-0.534
	μ	0.322	0.3	0.022	0.020	0.020	0.294-0.349
	ρ	0.0073	0.001	0.006	4.81×10^{-6}	4.46×10^{-5}	0.0068-0.0077
100	λ	0.505	0.5	0.005	0.007	0.007	0.488-0.521
	μ	0.319	0.3	0.019	0.007	0.007	0.302-0.335
	ρ	0.0074	0.001	0.006	2.86×10^{-6}	4.36×10^{-5}	0.0070-0.0077
500	λ	0.528	0.5	0.028	0.001	0.002	0.520-0.535
	μ	0.341	0.3	0.041	0.001	0.003	0.334-0.348
	ρ	0.0067	0.001	0.001	3.58×10^{-7}	3.30×10^{-5}	0.0065-0.0068

Table 4: Computational time (in secs) between MLE, GMM and GW estimation based on simulation Case 1 (where true value of $\lambda = 0.5$, $\mu = 0.3$ and $\rho = 0.001$) at different sample sizes (n).

Estimation method	$n = 50$	$n = 100$	$n = 500$
Galton-Watson approach	9.736	19.046	88.531
Generalised method of moments	1086.836	1852.000	6202.532
Maximum likelihood	114584.588	-	-

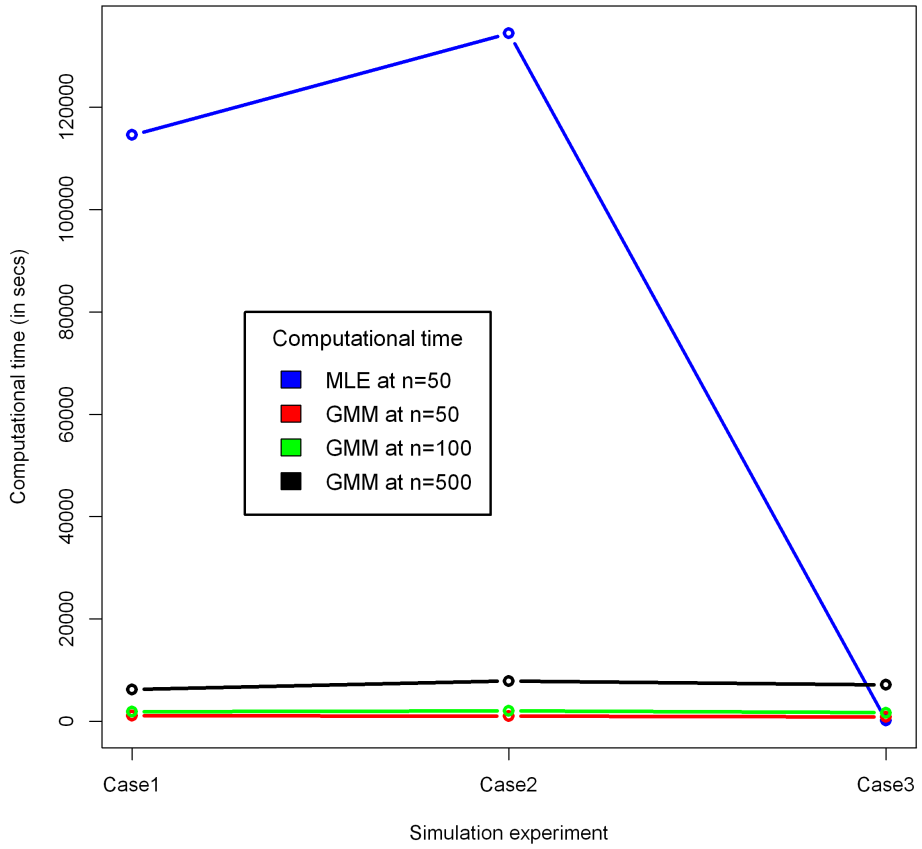

Figure 6: Comparison between MLE computational time and that of GMM at different sample sizes across the three simulation experiments (Cases 1-3).

3 B-D-C hybrid τ-leaping algorithm

Tau-leaping stochastic simulation is a fast approximate method of the exact stochastic simulation algorithm (SSA) originally proposed by Feller [4]. The $\tau-$leaping algorithm simulates a stochastic system such that all events are carried out during a time step before updating the event or transition rates [5]. The principle behind $\tau-$leaping is analogous to the standard Euler's method for solving deterministic systems (or differential equations). The only difference is that the one-step Euler's method makes use of fixed change in the system's state such that $x(t+\tau) = x(t) + \tau x'(t)$ where $x'(t) = f(t,x)$ and τ is the fixed step size; whereas $\tau-$leaping update the state using $x(t+\tau) = x(t) + P(\tau x'(t))$. Here, $P(\tau x'(t))$ is a Poisson random variable with fixed rate $\tau x'(t)$ and event or transition rate $x'(t)$ to approximate the number of transitions in the time interval $[t, t+\tau)$. However, the accuracy of $\tau-$leaping depends on how well the leap condition is satisfied during a time step, rising to different leaping methods [5, 6].

The leap condition requires a good choice of the leap size τ such that the state change is fixed or does not vary significantly within the time interval $[t, t+\tau)$ to avoid unexpected increment in the transition rates.

The size of τ determines the extent to which the system's history axis is leapt down. The choice of τ or not using τ ($\tau = 0$) is all about the trade-off between speed and accuracy. When the population size of the system increases, the exact SSA is relatively slower and thus, the leap size $\tau > 0$ is chosen to allow multiple events while satisfying the leap condition during a time step. However, if only a small number of transitions are leapt over, then using the exact SSA is much preferred (i.e., when $\tau = 0$). Different choice of the leap size for accelerating stochastic simulations have been proposed in the literature [10, 12]; but this current study only focuses on leap-size selection methods proposed by Gillespie [5] and Gillespie and Petzold [6], respectively.

To successfully develop a tau-leaping method, we require to find the largest value of the leap size (τ) that satisfies the leap condition. Let $\mathbf{x}$ represent the state of the process $X(t)$, $\theta = [\lambda, \mu, \rho]^T$ denote the parameters of the B-D-C process, $a_j(\mathbf{x}) = \theta_j \mathbf{x}$ for $j = 1, 2, 3$ be the propensity or rate function, and v represent the state-change vector (which takes values -1, 0 or +1); then the leap condition assumes that the value of τ must be small enough such that the change in propensity function or event rates $|a_j(\mathbf{x} + \Lambda(\tau, \mathbf{x})) - a_j(\mathbf{x})|$ is bounded above by a pre-specified error control parameter ϵ ($0 < \epsilon \ll 1$) of the sum of all event rates [5] such that

$$|a_j(\mathbf{x} + \Lambda(\tau, \mathbf{x})) - a_j(\mathbf{x})| \leq \epsilon a_0(\mathbf{x}), \quad \forall j = 1, 2, 3; \tag{22}$$

where $\Lambda(\tau, \mathbf{x}) \approx \sum_{j=1}^{3} P_j(a_j(\mathbf{x}), \tau) v_j$ (with $P_j(a_j(\mathbf{x}), \tau)$ denoting a Poisson random variable with rate $a_j(\mathbf{x})\tau$) and $a_0(\mathbf{x})$ is the total rate. However, since the state $\mathbf{x}$ cannot change by more than 1 within any infinitesimal time interval $[t, t+\tau)$ during the simulation of continuous-time Markov processes, we separately set-up the catastrophe event (where the entire population can hit 0 within the infinitesimal time interval) different from the birth and death events using standard Monte Carlo technique (based on a uniform random number). Hence, the name "B-D-C hybrid τ-leaping algorithm".

3.1 Procedure for selecting the Leap size

Two different procedures have been proposed by for selecting the leap size (τ) for processes with N-dimensional states such that the leap condition (equation 22) is satisfied [5, 6]. For the purpose of this study, the state $\mathbf{x}$ of the B-D-C process is one-dimensional and thus, the leap size selection stated in Lemma 2 based on τ-selection by Gillespie [5] and

Lemma 3 based on a new τ-selection by Gillespie and Petzold [6] are defined for a Markov process with one-dimensional state.

Lemma 2. Let $a_0(\mathbf{x}) = \sum_{j=1}^{M} a_j(\mathbf{x})$, $\xi(\mathbf{x}) = \sum_{j=1}^{M} a_j(\mathbf{x})v_j$, and $b_j = \frac{da_j(\mathbf{x})}{dx}$ for $j = 1, 2, \cdots M$. According to Gillespie [5], a choice for τ satisfying the leap condition (equation 22) at a given value of ϵ is

$$\tau = \min_{j\in[1,M]} \left\{ \frac{\epsilon a_0(\mathbf{x})}{|\xi(\mathbf{x})b_j|} \right\}, \tag{23}$$

where v_j is the state-change vector for $j = 1, 2, \cdots M$ and M is the total number of events the process $X(t)$.

Lemma 3. Given the M^2 functions $f_{jj'} = \frac{da_j(\mathbf{x})}{dx} v_{j'}$ for $j, j' = 1, 2, \cdots M$; suppose the $2M$ functions $\delta_j(\mathbf{x})$ and $\sigma_j^2(\mathbf{x})$, are defined such that

$$\delta_j(\mathbf{x}) = \sum_{j'=1}^{M} f_{jj'} a_{j'}(\mathbf{x}) \quad \text{and} \quad \sigma_j^2(\mathbf{x}) = \sum_{j'=1}^{M} f_{jj'}^2 a_{j'}(\mathbf{x}).$$

According to Gillespie and Petzold [6], the largest value for τ which satisfies the leap condition (equation 22) at a given value of ϵ is

$$\tau = \min_{j\in[1,M]} \left\{ \frac{\epsilon a_0(\mathbf{x})}{|\delta_j(\mathbf{x})|}, \frac{\epsilon^2 a_0^2(\mathbf{x})}{\sigma_j^2(\mathbf{x})} \right\}. \tag{24}$$

Remark. We propose a choice of τ value as presented in Theorems 2 and 3 (proposed for the first time in the current study) based on Lemmas 2 and 3, respectively, for the B-D-C hybrid τ-leaping algorithm by exploring the trade-off between speed and accuracy. Since we have set-up the catastrophe event differently from the birth and death events in the τ-leaping algorithm, the leap sizes (given by equations 25 and 26) are defined based on only the birth and death events of the B-D-C process. For easy comparison, the B-D-C hybrid τ-leaping algorithms motivated by Gillespie [5] and Gillespie and Petzold [6] studies are named "HTL2001" and "HTL2003", respectively, in subsequent sections (described fully in section 3.2).

Theorem 2. *Given the optimal leap size (defined by equation 23) and a pre-specified error bound (ϵ), the value of τ for simulating the B-D-C process (HTL2001) with birth rate λ and death rate μ is*

$$\tau_{HTL2001} = \frac{\epsilon(\lambda+\mu)}{|\lambda-\mu|\max(\lambda,\mu)}. \tag{25}$$

Proof of Theorem 2 .

Let λ and μ be the birth and death rates of the B-D-C process $X(t) = \mathbf{x}$. Suppose the propensity or even rate functions corresponding to birth and death events are given by:

$a_1(\mathbf{x}) = \lambda\mathbf{x}$ and $a_2(\mathbf{x}) = \mu\mathbf{x}$, with the state-change vector $v = [+1, -1]$ and error bound ϵ. From Lemma 2, $a_0(\mathbf{x}) = (\lambda+\mu)\mathbf{x}$; $\xi(\mathbf{x}) = a_1(\mathbf{x}) - a_2(\mathbf{x}) = (\lambda-\mu)\mathbf{x}$; $b_1(\mathbf{x}) = \lambda$ and $b_2(\mathbf{x}) = \mu$.

Then from equation 23,

$$\begin{aligned}\tau_{\text{HTL2001}} &= \min_{j\in[1,2]}\left\{\frac{\epsilon a_0(\mathbf{x})}{|\xi(\mathbf{x})b_j|}\right\} = \min\left\{\frac{\epsilon(\lambda+\mu)\mathbf{x}}{|(\lambda-\mu)\mathbf{x}\lambda|}, \frac{\epsilon(\lambda+\mu)\mathbf{x}}{|(\lambda-\mu)\mathbf{x}\mu|}\right\} \\ &= \min\left\{\frac{\epsilon(\lambda+\mu)}{|(\lambda-\mu)\lambda|}, \frac{\epsilon(\lambda+\mu)}{|(\lambda-\mu)\mu|}\right\} \\ &= \frac{\epsilon(\lambda+\mu)}{|(\lambda-\mu)|\max(\lambda,\mu)} \qquad \text{Q. E. D.}\end{aligned}$$

as required by equation 25.

Theorem 3. *Given the optimal leap size (defined by equation 24) and a pre-specified error bound (ϵ), the value of τ for simulating the B-D-C process (HTL2003) with birth rate λ and death rate μ is*

$$\tau_{HTL2003} = \min\left\{\frac{\epsilon(\lambda+\mu)}{|(\lambda-\mu)|\max(\lambda,\mu)}, \frac{\epsilon^2(\lambda+\mu)^2\mathbf{x}}{(\lambda+\mu)\max(\lambda^2,\mu^2)}\right\}. \tag{26}$$

Proof of Theorem 3 .

Let λ and μ be the birth and death rates of the B-D-C process $X(t) = \mathbf{x}$. Suppose the propensity or event rate functions corresponding to birth and death events are given by: $a_1(\mathbf{x}) = \lambda\mathbf{x}$ and $a_2(\mathbf{x}) = \mu\mathbf{x}$, with the state-change vector $v = [+1, -1]$, error bound ϵ and $a_0(\mathbf{x}) = (\lambda+\mu)\mathbf{x}$. Then from Lemma 3,

$$f_{jj'} = \begin{array}{c} \\ 1 \\ 2 \end{array}\begin{array}{c} \begin{array}{cc} 1 & 2 \end{array} \\ \begin{pmatrix} \lambda & -\lambda \\ \mu & -\mu \end{pmatrix} \end{array},$$

$$\begin{aligned}\delta_1(\mathbf{x}) &= \sum_{j'=1}^{2} f_{1j'}a_{j'}(\mathbf{x}) = f_{11}a_1(\mathbf{x}) + f_{12}a_2(\mathbf{x}) \\ &= \lambda(\lambda\mathbf{x}) + (-\lambda)(\mu\mathbf{x}) = \lambda^2\mathbf{x} - \lambda\mu\mathbf{x} \\ &= (\lambda^2 - \lambda\mu)\mathbf{x}, \\ \delta_2(\mathbf{x}) &= \sum_{j'=1}^{2} f_{2j'}a_{j'}(\mathbf{x}) = f_{21}a_1(\mathbf{x}) + f_{22}a_2(\mathbf{x}) \\ &= \mu(\lambda\mathbf{x}) + (\mu)(\mu\mathbf{x}) = \lambda\mu\mathbf{x} - \mu^2\mathbf{x} \\ &= (\lambda\mu - \mu^2)\mathbf{x},\end{aligned}$$

$$\sigma_1^2(\mathbf{x}) = \sum_{j'=1}^{2} f_{1j'}^2 a_{j'}(\mathbf{x}) = f_{11}^2 a_1(\mathbf{x}) + f_{12}^2 a_2(\mathbf{x})$$
$$= \lambda^2(\lambda\mathbf{x}) + \lambda^2(\mu\mathbf{x}) = (\lambda^3 + \lambda^2\mu)\mathbf{x},$$

and

$$\sigma_2^2(\mathbf{x}) = \sum_{j'=1}^{2} f_{2j'}^2 a_{j'}(\mathbf{x}) = f_{21}^2 a_1(\mathbf{x}) + f_{22}^2 a_2(\mathbf{x})$$
$$= \mu^2(\lambda\mathbf{x}) + \mu^2(\mu\mathbf{x}) = (\lambda\mu^2 + \mu^3)\mathbf{x}.$$

Then from equation 24,

$$\tau_{\text{HTL2003}} = \min_{j\in[1,2]} \left\{ \frac{\epsilon a_0(\mathbf{x})}{|\delta_j(\mathbf{x})|}, \frac{\epsilon^2 a_0^2(\mathbf{x})}{\sigma_j^2(\mathbf{x})} \right\} = \min \left\{ \frac{\epsilon a_0(\mathbf{x})}{|\delta_1(\mathbf{x})|}, \frac{\epsilon^2 a_0^2(\mathbf{x})}{\sigma_1^2(\mathbf{x})}, \frac{\epsilon a_0(\mathbf{x})}{|\delta_2(\mathbf{x})|}, \frac{\epsilon^2 a_0^2(\mathbf{x})}{\sigma_2^2(\mathbf{x})} \right\}$$
$$= \min \left\{ \frac{\epsilon(\lambda+\mu)\mathbf{x}}{|\lambda^2 - \lambda\mu|\mathbf{x}}, \frac{\epsilon^2(\lambda+\mu)^2\mathbf{x}^2}{(\lambda^3+\lambda^2\mu)\mathbf{x}}, \frac{\epsilon(\lambda+\mu)\mathbf{x}}{|\lambda\mu - \mu^2|\mathbf{x}}, \frac{\epsilon^2(\lambda+\mu)^2\mathbf{x}^2}{(\lambda\mu^2+\mu^3)\mathbf{x}} \right\}$$
$$= \min \left\{ \frac{\epsilon(\lambda+\mu)}{\max(|\lambda^2 - \lambda\mu|, |\lambda\mu - \mu^2|)}, \frac{\epsilon^2(\lambda+\mu)^2\mathbf{x}}{\max(\lambda^3+\lambda^2\mu, \lambda\mu^2+\mu^3)} \right\}$$
$$= \min \left\{ \frac{\epsilon(\lambda+\mu)}{|(\lambda-\mu)|\max(\lambda,\mu)}, \frac{\epsilon^2(\lambda+\mu)^2\mathbf{x}}{(\lambda+\mu)\max(\lambda^2,\mu^2)} \right\} \qquad \text{Q. E. D.}$$

as required by equation 26.

3.2 Pseudo-codes of the B-D-C hybrid τ-leaping algorithms

In the B-D-C τ-leaping algorithms (HTL2001 and HTL2003, proposed for the B-D-C process for the first time in this study), we forego the leaping method and switch to the exact SSA (Algorithm 1) whenever the leap size τ is very small such that, τ is at most a few multiples of the expected time to the next event in the exact SSA (1/total rate). The pseudo-codes of the τ-leaping algorithms are given by Algorithms 2 and 3, respectively (for R codes, see Appendix E). The main difference between the two B-D-C hybrid τ-leaping algorithms HTL2001 (Algorithm 2) and HTL2003 (Algorithm 3) is the choice of the optimal leap size estimator and its leap condition.

Algorithm 2: B-D-C hybrid τ-leaping algorithm (HTL2001 pseudo-code)

Input: X, λ, μ, ρ, t, t_{final} ϵ, and host survival status (s).

Output: Parasite numbers and survival status (alive: $s = 1$; dead: $s = 2$) recorded at discrete times ($t = 1, 2, \cdots, t_{\text{final}}$).

while $t < t_{final}$ *and* $s = 1$ **do**
  Set initial time $t = t_0$, state $X = X_0$ and $s = 1$.
  Calculate rates corresponding to birth (a_1), death (a_2) and catastrophe (a_3); such that $a_1 = \lambda X$, $a_2 = \mu X$ and $a_3 = \rho X$.
  Compute the total rate, $a_0 = \sum_{j=1}^{3} a_j$, for $j = 1, 2, 3$ (from step 3).
  Compute the leap size $\tau = \frac{\epsilon(\lambda+\mu)}{|(\lambda-\mu)|\max(\lambda,\mu)}$.
  if $\tau > \frac{2}{a_0}$ **then**
    set $t = t + \tau$ and choose a random number r from $Uniform(0, 1)$.
    if $r < a_3\tau$ **then**
      set $X = 0$ and $s = 2$ (catastrophe event occurs)
    else
      set $X = X + Poisson(a_1\tau) - Poisson(a_2\tau)$ (birth and death events occur)
    end
  else
    Execute exact SSA (Algorithm 1)
  end

  Record (X, s) at the desired discrete times.
end

Algorithm 3: B-D-C hybrid τ-leaping algorithm (HTL2003 pseudo-code)

Input: X, λ, μ, ρ, t, t_{final} ϵ, and host survival status (s).

Output: Parasite numbers and survival status (alive: $s=1$; dead: $s=2$) recorded at discrete times ($t=1,2,\cdots,t_{\text{final}}$).

while $t < t_{final}$ *and* $s=1$ **do**
  Set initial time $t=t_0$, state $X=X_0$ and $s=1$.
  Calculate rates corresponding to birth (a_1), death (a_2) and catastrophe (a_3); such that $a_1=\lambda X$, $a_2=\mu X$ and $a_3=\rho X$.
  Compute the total rate, $a_0=\sum_{j=1}^{3} a_j$, for $j=1,2,3$ (from step 3).
  Compute the leap size $\tau=\min\left\{\frac{\epsilon(\lambda+\mu)}{|(\lambda-\mu)|\max(\lambda,\mu)}, \frac{\epsilon^2(\lambda+\mu)^2\mathbf{x}}{(\lambda+\mu)\max(\lambda^2,\mu^2)}\right\}$.
  if $\tau > \frac{1}{10a_0}$ **then**
    set $t=t+\tau$ and choose a random number r from $Uniform(0,1)$.
    if $r < a_3\tau$ **then**
      set $X=0$ and $s=2$ (catastrophe event occurs)
    else
      set $X=X+Poisson(a_1\tau)-Poisson(a_2\tau)$ (birth and death events occur)
    end
  else
    Execute exact SSA (Algorithm 1)
  end

  Record (X,s) at the desired discrete times.
end

3.3 Effects of the error bound on the accuracy and speed of the τ-leaping algorithms

We compared the two B-D-C hybrid τ-leaping algorithms by exploring a balance between simulation accuracy and computational speed at 15 different error bound values (for $0\leq \epsilon \leq 0.1$) based on the three different *in silico* simulation experiments (Cases 1-3) in a full factorial design. For each simulation experiment, 3 million simulations were conducted. We quantified the simulation accuracy of the two τ-leaping algorithms (HTL2001 and HTL2003) by estimating the squared error loss between the true mean number of parasites and its corresponding predictions across all observed time points at each error bound (ϵ =0, 0.002, 0.004, 0.006, 0.008, 0.01, 0.02, 0.03, 0.04, 0.05, 0.06, 0.07, 0.09 and 0.1). We also computed total standard error of the population mean (total deviation in the sampling distribution) and computational time (speed) at each ϵ value for comparison. We proposed an error threshold for the two τ-leaping algorithms based on the simulation

speed and accuracy for the three simulation cases.

3.3.1 Relationship between the leap size, the state and the leap conditions

The relationship between the leap size, the state and the leap conditions were explored for the two tau-leaping algorithms and across the three different *in silico* experiments (Figure 7). It was consistently shown across the different simulation Cases that as the state variable x increases, the leap size of algorithm HTL2003 increases and converges towards the leap size of algorithm HTL2001. However, the leap condition decreases if the state value increases. Tau-leaping started in algorithm HTL2001 at state $x > 40$ for simulation Case 1, state $x > 50$ for simulation Case 2 and state $x > 30$ for simulation Case 3. For algorithm HTL2003, leaping started at state $x > 20$ for simulation Case 1, state $x > 30$ for simulation Case 2 and state $x > 20$ for simulation Case 3. Consequently, starting the tau-leaping algorithms earlier after state $x \geq 10$ or setting $\tau = 0$ for a small value of x, say, could improve the computational speed of the $\tau-$leaping algorithms (HTL2001 and HTL2003), but may have some cost of simulation accuracy.

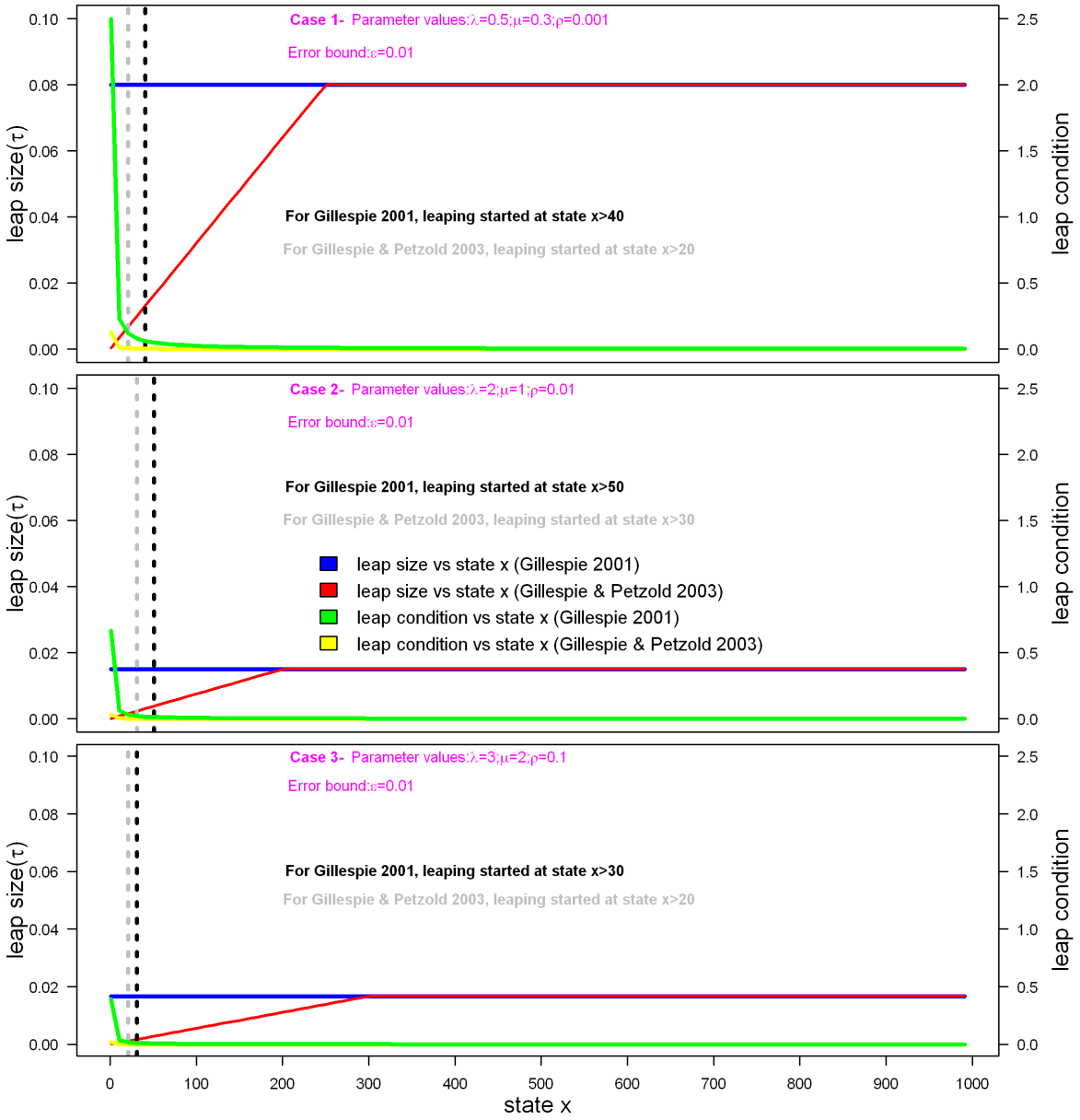

Figure 7: Relationship between the leap size, the state variable and the leap conditions.

3.3.2 Mean comparison at different error values

Figures 8–10 present the Monte Carlo estimates of the mean parasite numbers between the two B-D-C hybrid τ-leaping algorithms for $\epsilon \leq 0.01$ and their corresponding true mean values over time across the three simulation experiments based on 3 million simulations, respectively. Although 95% confidence intervals were obtained for the estimates, the interval width was very small in size due to small standard errors associated with the large number of simulations. Hence, the 95% confidence intervals are not presented here; nonetheless, the true mean was found in the estimated intervals over 90% of the time. The true mean and the Monte Carlo mean estimates were consistent across the observed time

points (t=1-30 days) for both τ-leaping algorithms at $\epsilon \leq 0.01$. The τ-leaping methods were fairly accurate for small error values ($0 \leq \epsilon \leq 0.01$) compared to error values > 0.01 .

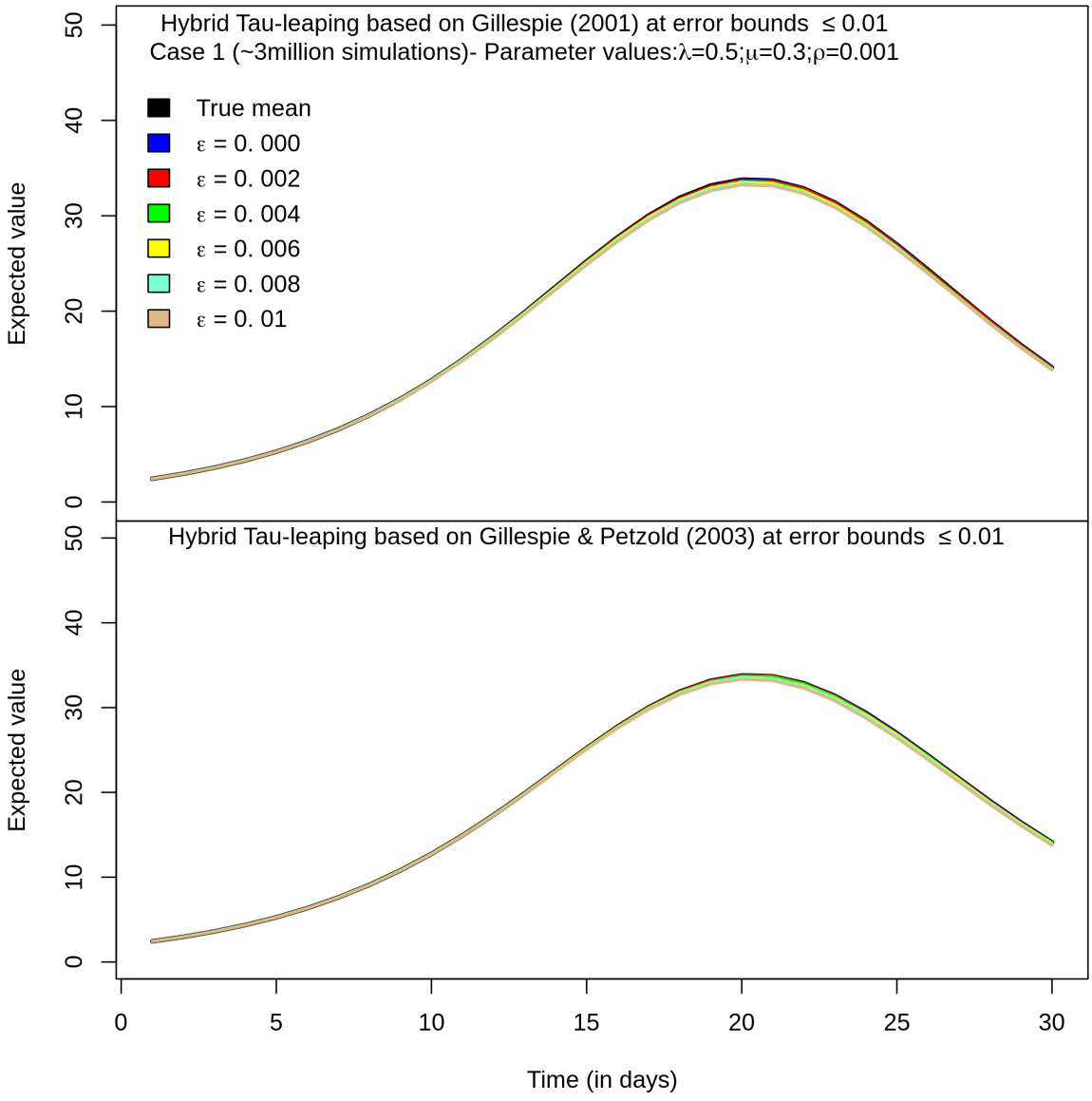

Figure 8: Comparing the mean behaviour of parasites between the two B-D-C hybrid τ-leaping algorithms at given parameter values ($\lambda = 0.5$, $\mu = 0.3$, $\rho = 0.001$) at $\epsilon \leq 0.01$ (Case 1).

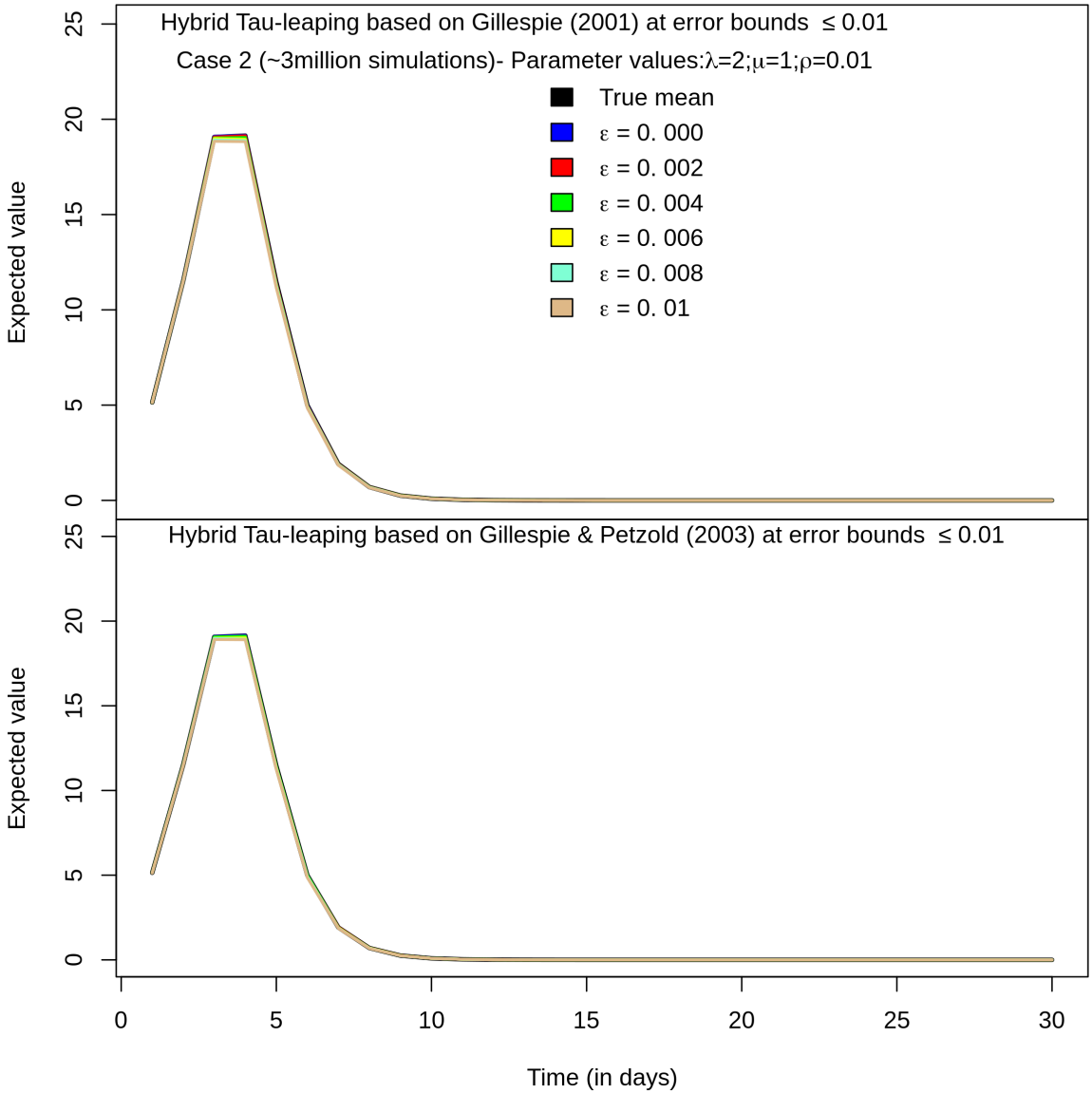

Figure 9: Comparing the mean behaviour of parasites between the two B-D-C hybrid τ-leaping algorithms at given parameter values ($\lambda = 2$, $\mu = 1$, $\rho = 0.01$) at $\epsilon \leq 0.01$ (Case 2).

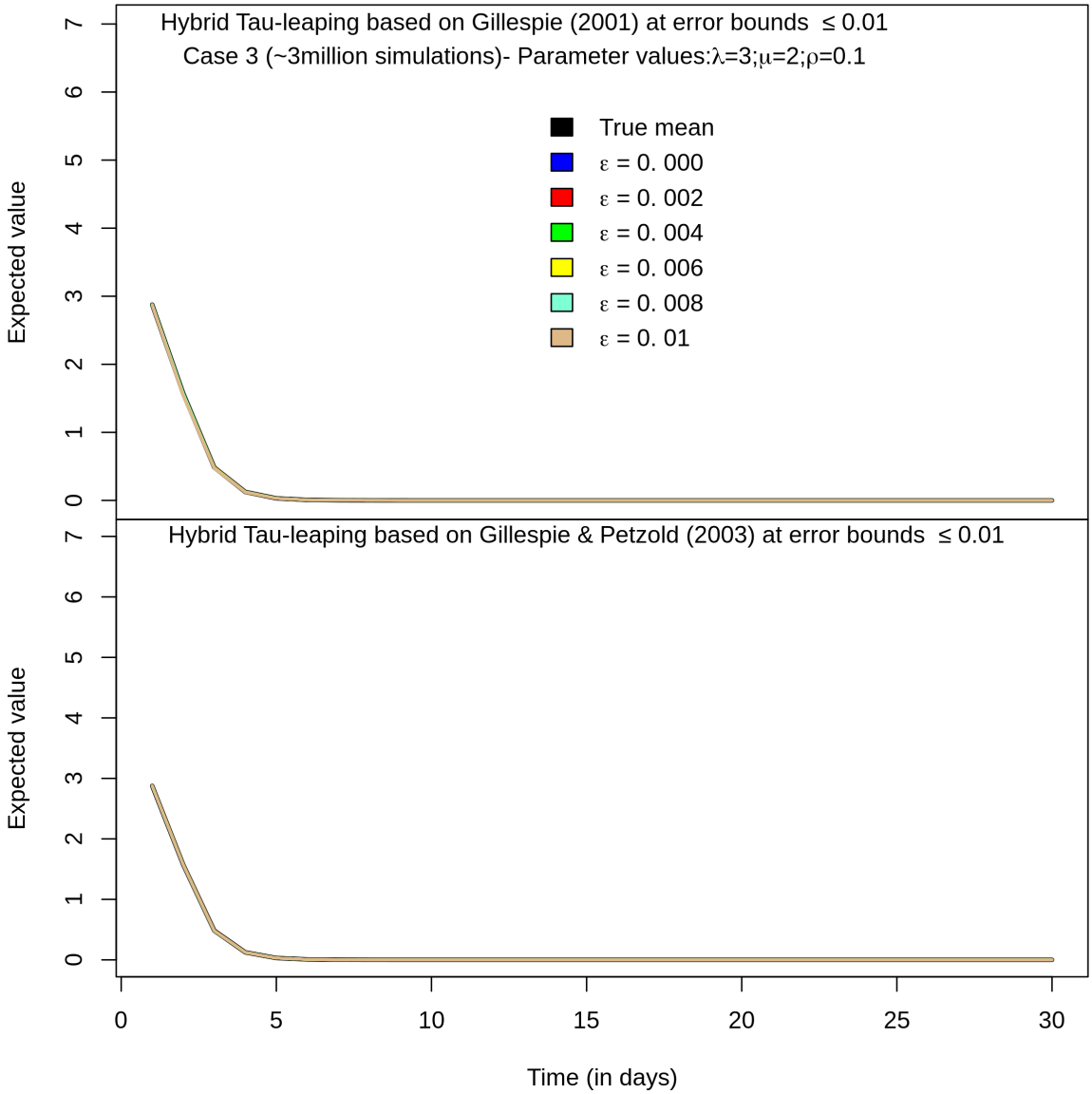

Figure 10: Comparing the mean behaviour of parasites between the two B-D-C hybrid τ-leaping algorithms at given parameter values ($\lambda = 3$, $\mu = 2$, $\rho = 0.1$) at $\epsilon \leq 0.01$ (Case 3).

3.3.3 Simulation accuracy and deviations in sampling distribution at different error values

We examined the simulation accuracy by estimating the squared error loss and the total standard error of the population mean at each value of ϵ across the three simulation experiments (Figures 11–13). The simulation accuracy decreased with increasing value of the error bound, while the small deviation in the sampling distribution of the simulated data was achieved for larger values of ϵ. In the first and second simulation experiments (where parasite numbers were relatively higher), there was no significant difference in the error loss and standard error estimates between the two τ-leaping algorithms across the different error bounds. However, simulation Case 3 (where parasite numbers were relatively low) revealed a significant difference in simulation accuracy and standard error at $\epsilon > 0.01$; the HTL2003 algorithm performed better than the HTL2001 algorithm, but the sampling variations from the HTL2003 algorithm was higher comparatively.

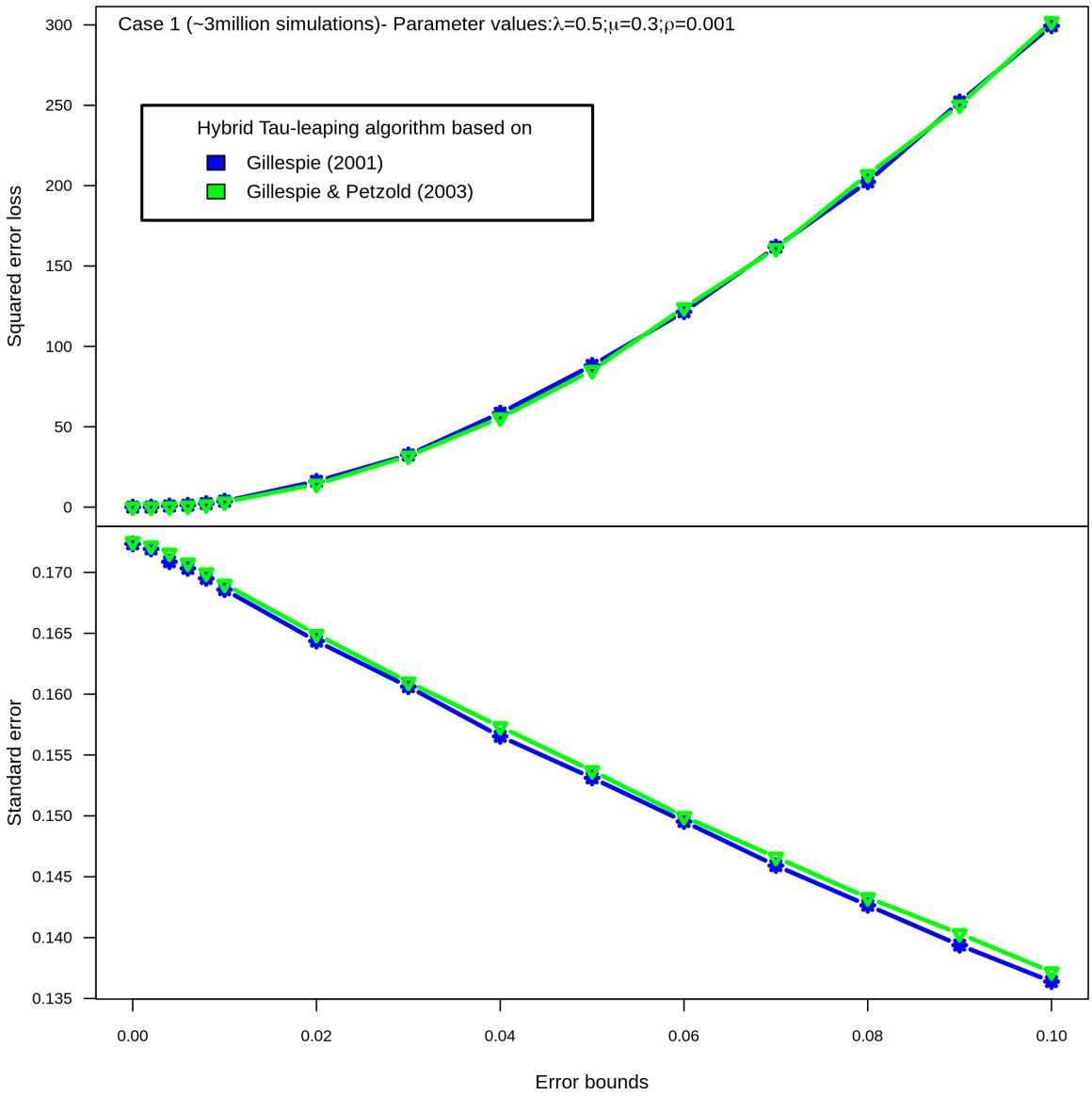

Figure 11: Comparing the simulation accuracy between the two B-D-C hybrid τ-leaping algorithms at given parameter values ($\lambda = 0.5$, $\mu = 0.3$, $\rho = 0.001$) across the error bound values (Case 1).

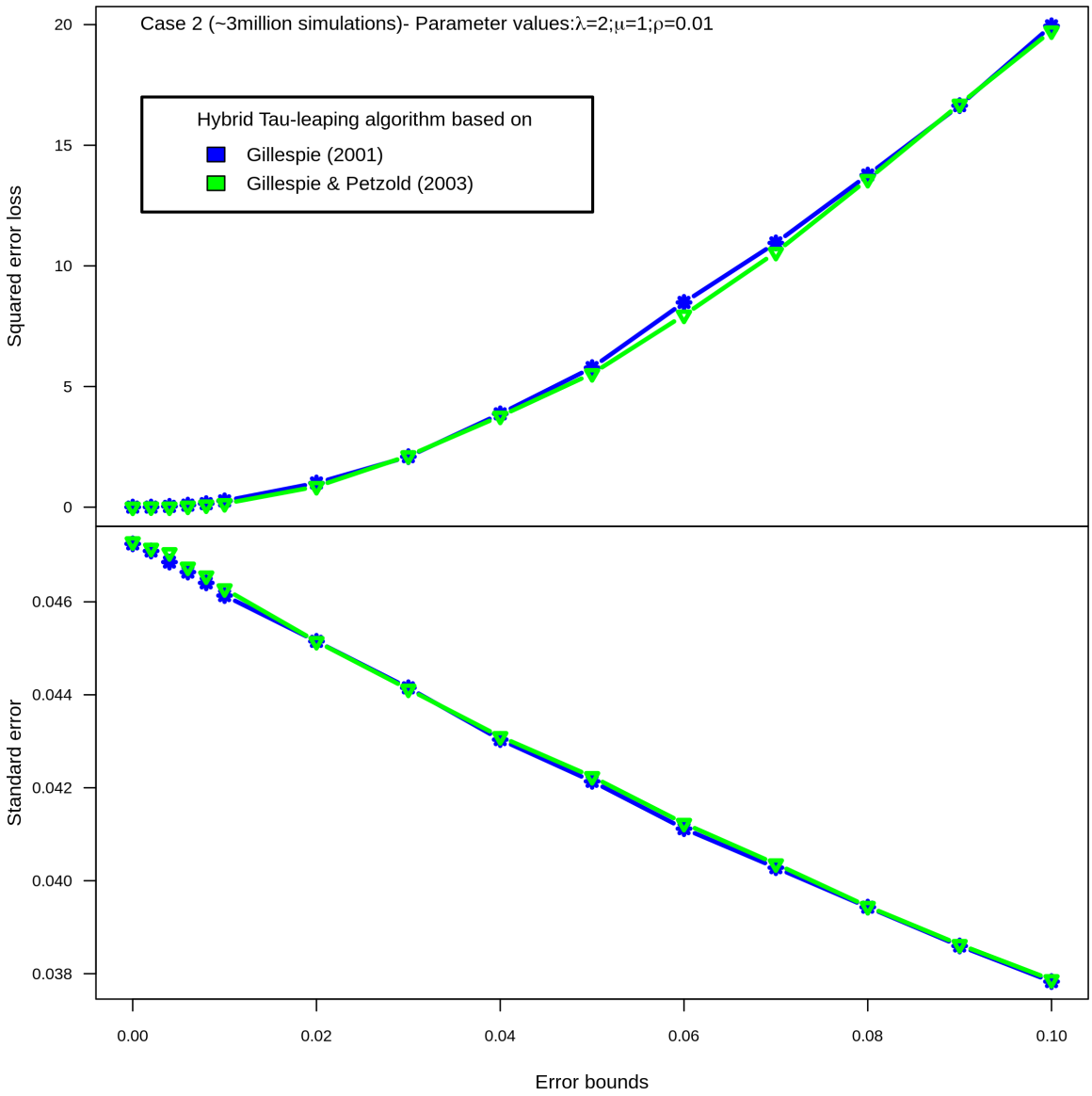

Figure 12: Comparing the simulation accuracy between the two B-D-C hybrid τ-leaping algorithms at given parameter values ($\lambda = 2,\ \mu = 1,\ \rho = 0.01$) across the error bound values (Case 2).

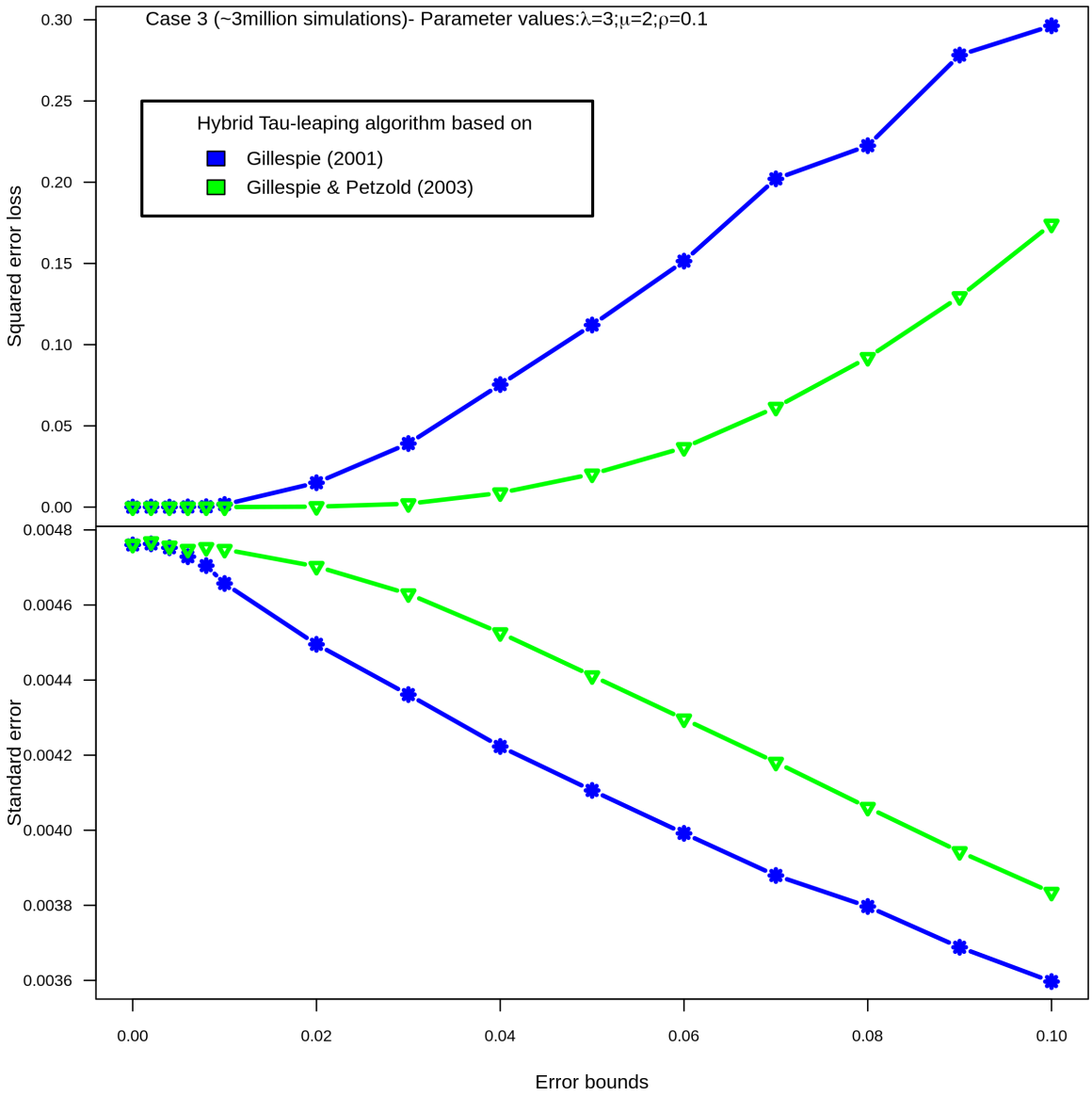

Figure 13: Comparing the simulation accuracy between the two B-D-C hybrid τ-leaping algorithms at given parameter values ($\lambda = 3$, $\mu = 2$, $\rho = 0.1$) across the error bound values (Case 3).

3.3.4 Computational speed versus simulation accuracy across different error values

We proposed a good choice of the error bound (based on the 15 different ϵ values) by exploring the trade-off between computational speed and simulation accuracy across the three simulation experiments for both τ-leaping algorithms (Figures 14–16). It can be observed from Figures 14–16 that $\epsilon = 0.01$ can be a good error bound choice for both $\tau-$leaping algorithms so as to achieve accurate simulations with relatively low computational time. Nonetheless, HTL2001 was much faster (with small computational time) than HTL2003 at any each ϵ value ($0 \leq \epsilon \leq 0.1$) across all the simulation

experiments. From Figure 7, the computational speed of the algorithms can be improved further.

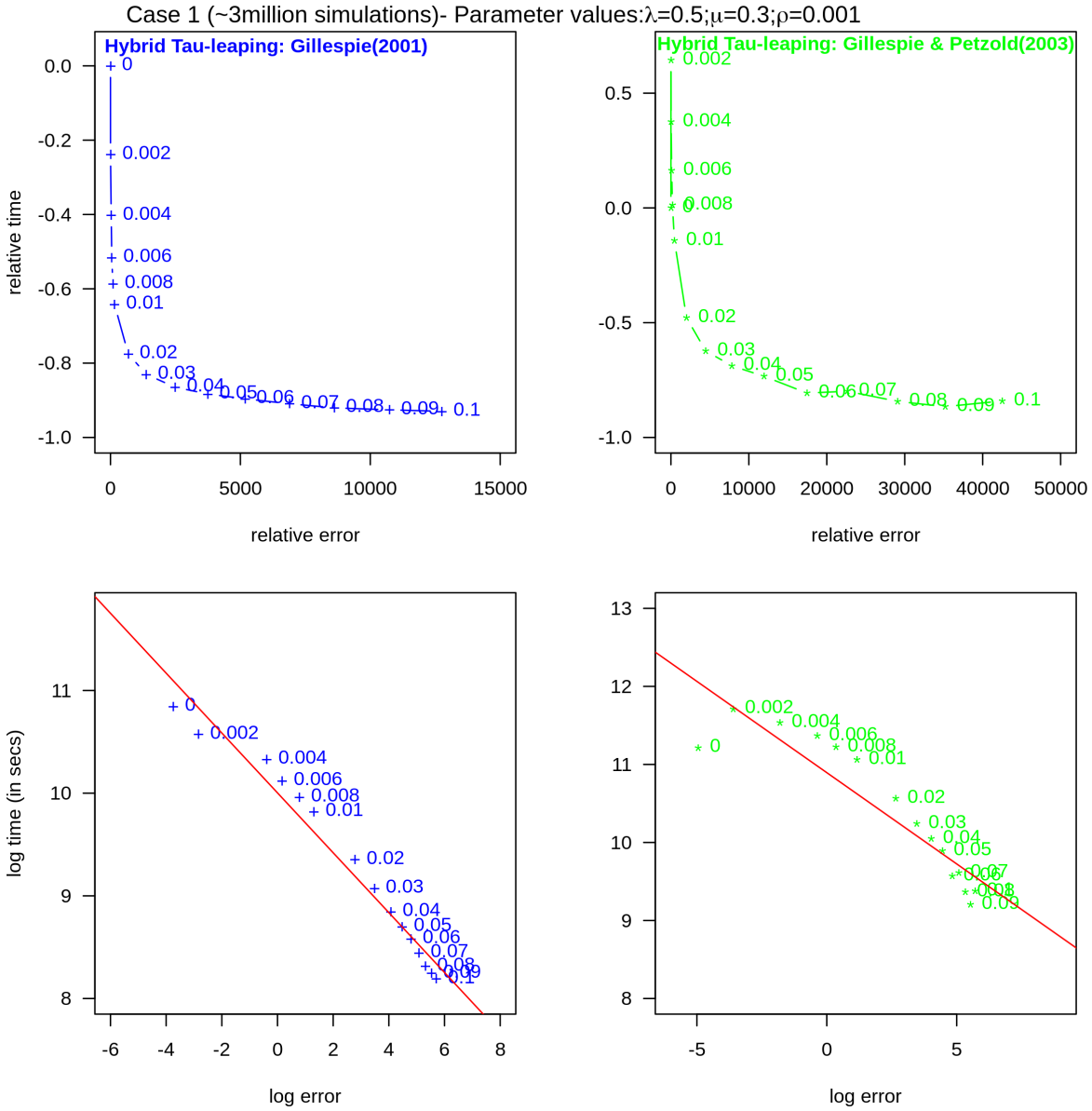

Figure 14: Plot of computational speed against simulation accuracy between the two B-D-C hybrid τ-leaping algorithms across the error bound values (Case 1).

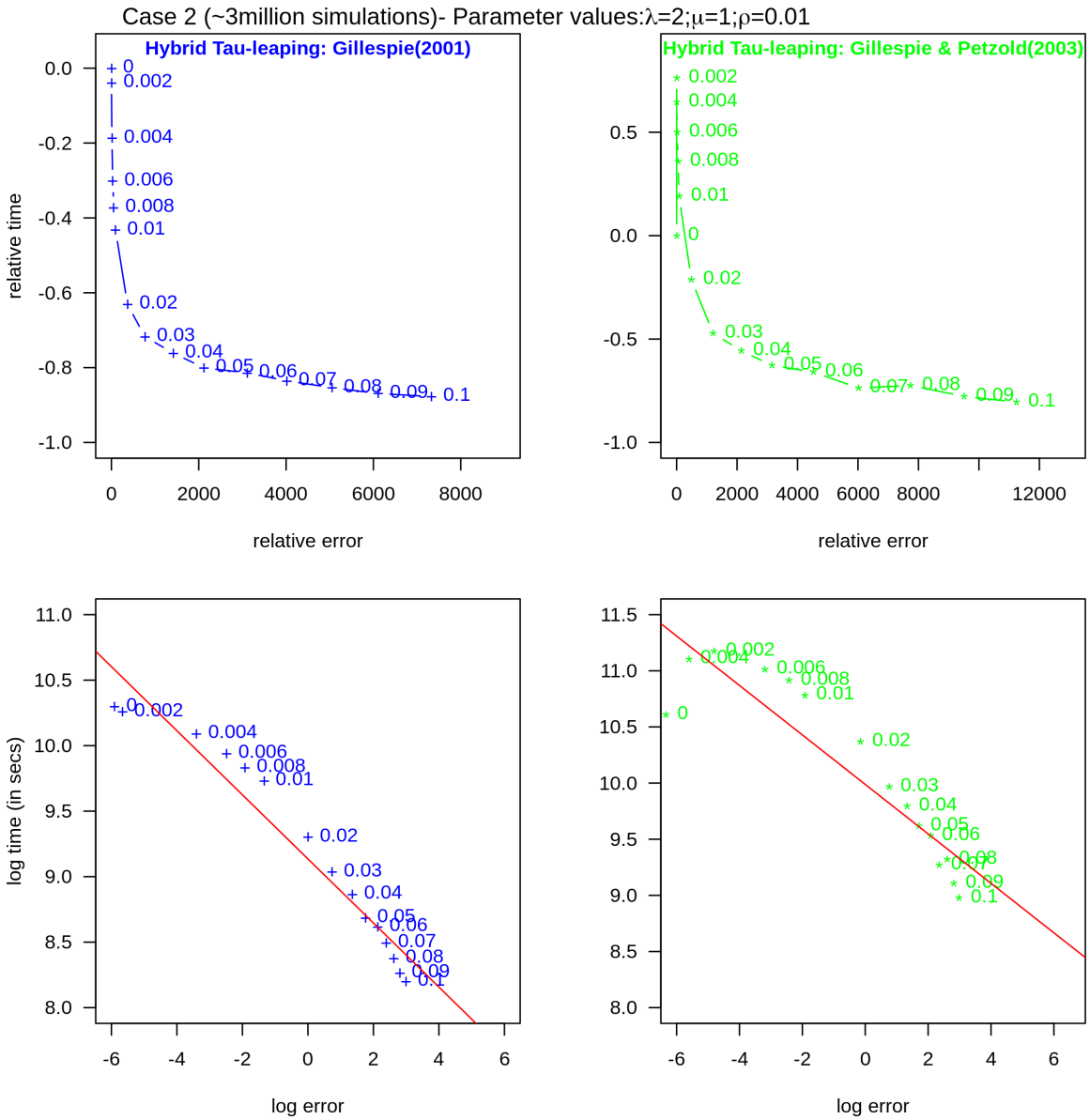

Figure 15: Plot of computational speed against simulation accuracy between the two B-D-C hybrid τ-leaping algorithms across the error bound values (Case 2).

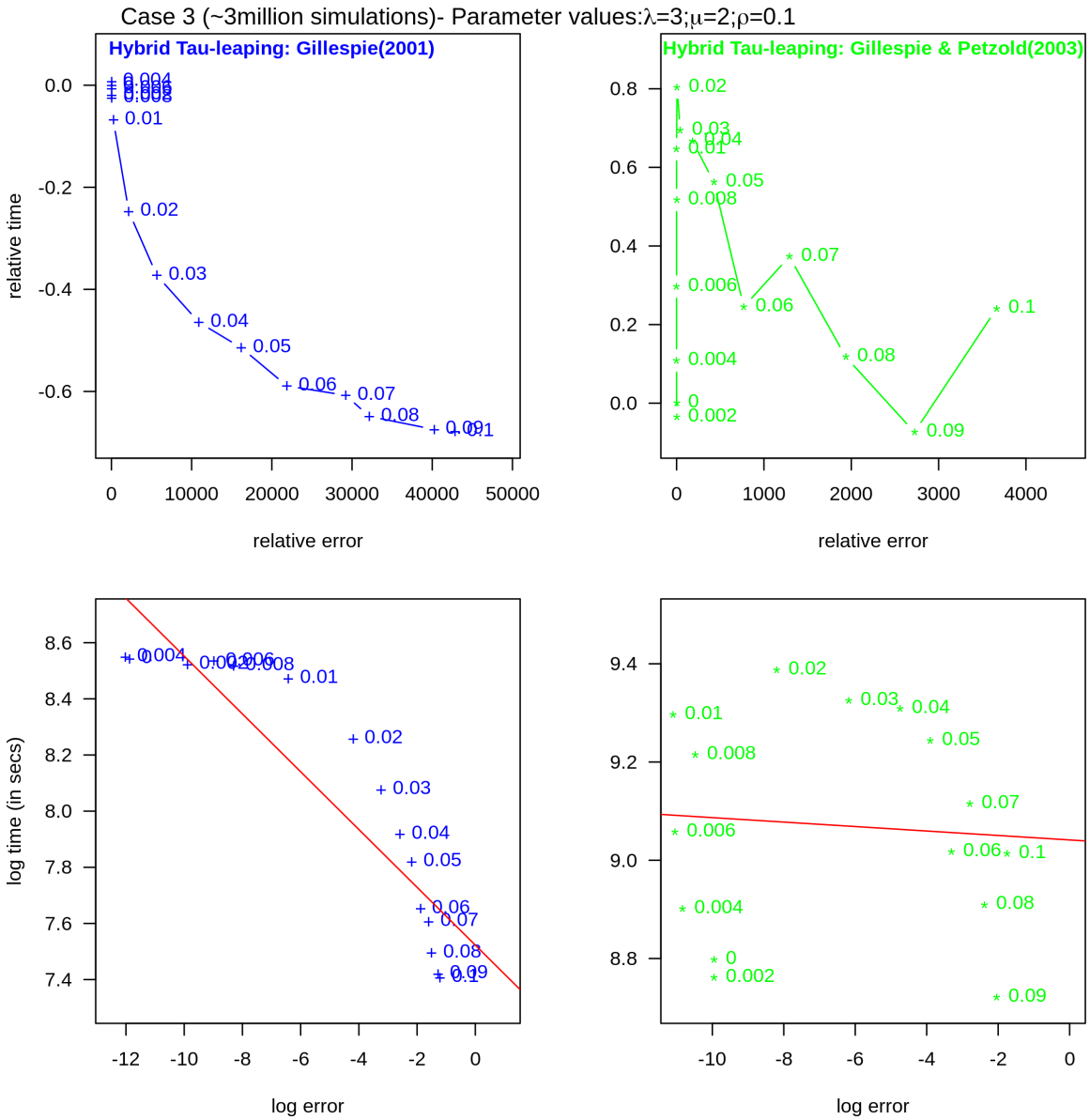

Figure 16: Plot of computational speed against simulation accuracy between the two B-D-C hybrid τ-leaping algorithms across the error bound values (Case 3).

Bibliography

1. Bezanson, J., Edelman, A., Karpinski, S., and Shah, V. B. (2017). Julia: A fresh approach to numerical computing. *SIAM review*, 59(1):65–98.

2. Davison, A. C., Hautphenne, S., and Kraus, A. (2021). Parameter estimation for discretely observed linear birth-and-death processes. *Biometrics*, 77(1):186–196.

3. Di Crescenzo, A., Giorno, V., Nobile, A. G., and Ricciardi, L. M. (2008). A note on birth–death processes with catastrophes. *Statistics & probability letters*, 78(14):2248–2257.

4. Feller, W. (1940). On the integro-differential equations of purely discontinuous Markoff processes. *Transactions of the American Mathematical Society*, 48:488–515.

5. Gillespie, D. T. (2001). Approximate accelerated stochastic simulation of chemically reacting systems. *Journal of Chemical Physics*, 115(4):1716–1733.

6. Gillespie, D. T. and Petzold, L. R. (2003). Improved lead-size selection for accelerated stochastic simulation. *Journal of Chemical Physics*, 119(16):8229–8234.

7. Hansen, L. P. (1982). Large sample properties of generalized method of moments estimators. *Econometrica: Journal of the Econometric Society*, pages 1029–1054.

8. Haris, T. E. (2002). *The theory of Branching Processes*. Dover, Mineola, NY.

9. Karlin, S. and Tavaré, S. (1982). Linear birth and death processes with killing. *Journal of Applied Probability*, 19(3):477–487.

10. Lipková, J., Arampatzis, G., Chatelain, P., Menze, B., and Koumoutsakos, P. (2019). S-leaping: An adaptive, accelerated stochastic simulation algorithm, bridging *tau*-leaping and r-leaping. *Bulletin of mathematical biology*, 81(8):3074–3096.

11. R Core Team (2019). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. http://www.R-project.org/.

12. Rathinam, M., Petzold, L. R., Cao, Y., and Gillespie, D. T. (2003). Stiffness in stochastic chemically reacting systems: The implicit tau-leaping method. *Journal of Chemical Physics*, 119(24):12784–12794.

Appendix C: R Codes for Exact SSA of the B-D-C process

C.1: Function for updating events of the B-D-C process via exact SSA

```
SSA_update_event=function(X,fish_status,rate,total_rate){
 #Let b,d & c be the birth, death & catastrophe parameters
 #X be the number of parasites
 #fish_status <- 1 # fish starts out alive

 if(total_rate == 0) {
  return(list(X = X, t_incr = Inf)) # zero population
 }

 #Determine event occurence from single draw
 u<-runif(1,0, total_rate)
 if (u<abs(rate[1])){
  #birth of parasites
  X<-X+1
 } else if(u<abs(rate[1]+rate[2])){
  #death of parasites
  X<-X-1
 }  else {
  #catastrophe or death of fish
  X<-0
  fish_status<-2
 }
 t_incr <- rexp(1, total_rate) # time increment
 #Returns parasite numbers,time step and survival status
 return(list(X = X,t_incr=t_incr,fish_status=fish_status))

}
```

C.2: Function for exact stochastic simulation (SSA)

```
#Function for exact simulation of the B-D-C process
Exact_BDC<-function(X0,b,d,c,ti=0,tmax=30){
 #Let ti be the initial time (set at 0)
 #tfinal be the final simulation time
 rate<-numeric(3) #event rates
 #stop simulation if total population exceeds this limit
 pop_max <- 10000
 #Time variable; ti<- 0; tmax<-30;
 save_ti <- 1:tmax #Discrete times to store simulation
 save_TF <- rep(FALSE, length(save_ti))
 #parasite pop over time
 pop <-matrix(NA,1,length(save_ti))
 # host host status at each time point
 alive <- rep(2, length(save_ti))
 alive_ti <- 1 #fish starts out alive
 X<-X0;pop_ti <- sum(X)
 while(sum(save_TF) < length(save_ti)){
  #Calculate rate of events
  #probability of birth
  rate[1]<- b*X
  #probability of death
  rate[2]<- d*X
  #probability of catastrophe
  rate[3]<- c*X
  #tota rate
  total_rate<- rate[1]+rate[2]+rate[3]
  if(sum(pop_ti) > pop_max){
   cat("Popmax_exceeded","\n")
   break
  }
  if(alive_ti == 2)  break
  output <-SSA_update_event(X,fish_status=alive_ti,
  rate=rate,total_rate=total_rate)
  #Update time to next event
  ti <- ti + output$t_incr
  #break if there is negative population
  if (X < 0) break
  # Events to occur
  save_new <- which((ti >= save_ti) & !save_TF)
  for (i in save_new){
   pop[,i]<- pop_ti
   alive[i] <- alive_ti
  }
  save_TF <- (ti >= save_ti)
  X<- output$X
  pop_ti <- sum(X)
  alive_ti <- output$fish_status
 }
 #Returns the parasite numbers & survival status over time
 return(list(pop=pop,alive = alive))
}
```

Appendix D: Julia codes for computing the log-likelihood function

D.1: Computing constants of the B-D-C transition function

```
module BDCfit #begin module
using PolynomialRoots
export logL
function BDCconsts(lambda, mu, rho, t)
#lambda, mu, rho are B-D-C parameters respectively
# Computing constants of BDC process at time t
rts = sort(real(roots([mu,-(lambda+mu+rho),lambda])))
v0 = rts[1]
v1 = rts[2]
sigma = exp(-lambda*(v1 - v0)*t)
k1 = v0*v1*(1 - sigma)/(v1 - sigma*v0)
k2 = (v1*sigma - v0)/(v1 - sigma*v0)
k3 = (1 - sigma)/(v1 - sigma*v0)
return [k1, k2, k3]
end
function gamma_n_j(nmax)
# calculates gamma^n_j for n = 1, ..., nmax & j = 1, ...,n
# used by ProbBDC
gnj = zeros(BigInt, nmax, nmax)
gnj[1,1] = 1
if nmax > 1
for n = 2:nmax
gnj[n,1] = n*gnj[n-1,1]
end
for j = 2:nmax
for n = j:nmax
gnj[n,j] = gnj[n-1,j-1] + (n+j-1)*gnj[n-1,j]
end
end
end
return gnj
end
function delta_m_j(mmax, k1, k2, k3)
# calculates delta^m_j for n = 1, ..., mmax & j = 1, ...,n
# used by ProbBDC;k1, k2, k3 will be output from BDCconsts
k = (k2 + k1*k3)/k1/k3
dmj = zeros(BigFloat, mmax, mmax)
dmj[1,1] = k
if mmax == 1
return dmj
else
for m = 2:mmax
dmj[m,1] = k*m
for j = 2:m
dmj[m,j] = k*(m - j + 1)*dmj[m,j-1]
end
end
return dmj
end
end
```

D.2: Computing the B-D-C transition function

```
function ProbBDC(lambda, mu, rho, t, mmax, nmax)
# P(X_t=n | X_0=m) for -1 <= m <= mmax and
#-1 <= n <= nmax
# where -1 indicates extinction by catastrophe
cc = BDCconsts(lambda, mu, rho, t)
k1 = cc[1]
k2 = cc[2]
k3 = cc[3]
k4 = (k1 + k2)/(1 - k3)
P = zeros(Float64, mmax+2, nmax+2)
P[1,1] = 1
P[2,2] = 1
k1_powers = zeros(BigFloat, mmax)
k3_powers = zeros(BigFloat, nmax)
k4_powers = zeros(BigFloat, mmax)
facts = zeros(BigFloat, nmax)
k1_powers[1] = k1
k4_powers[1] = k4
P[3,1] = Float64(1 - k4)
P[3,2] = Float64(k1)
for m = 2:mmax
k1_powers[m] = k1*k1_powers[m-1]
k4_powers[m] = k4*k4_powers[m-1]
P[m+2,1] = Float64(1 - k4_powers[m])
P[m+2,2] = Float64(k1_powers[m])
end
k3_powers[1] = k3
facts[1] = 1
for n = 2:nmax
k3_powers[n] = k3*k3_powers[n-1]
facts[n] = n*facts[n-1]
end
gnj = gamma_n_j(nmax)
dmj = delta_m_j(mmax, k1, k2, k3)
for m = 1:mmax
for n = 1:nmax
x = BigFloat(0)
for j = 1:(min(m,n))
x = x + gnj[n,j]*dmj[m,j]
end
P[m+2,n+2] = Float64(x*k1_powers[m]*k3_powers[n]/facts[n])
end
end
return P
end
```

D.3: Computing the B-D-C log-likelihood function

```
function logL(lambda, mu, rho, x)
# calculate the log likelihood for params:
# lambda, mu, rho and data x
# each row of x are population at times: t= 1, 3, 5,7,..17
# assume population at time 0 is 2;
# state -1 indicates catastrophe
mmax1 = 2
nmax1 = Int64(max(maximum(x[:,1]), 2))
P1 = ProbBDC(lambda, mu, rho, 1, mmax1, nmax1)
mmax2 = Int64(max(maximum(x[:,1:8]), 2))
nmax2 = Int64(max(maximum(x), 2))
P2 = ProbBDC(lambda, mu, rho, 2, mmax2, nmax2)
el = 0
for i = 1:size(x, 1) # logL for observation i
# time 0 to time 1 transition
el = el + log(P1[4, Int64(x[i,1]+2)])
for j = 1:8
# time 2j-1 to time 2j+1 transition
el = el + log(P2[Int64(x[i,j]+2), Int64(x[i,j+1]+2)])
end
end
return el
end

end #module
```

E.1: Function for updating B-D-C Hybrid τ-leaping simulation

(see Appendix E)

```
#Function to update tau-leaping

tauleap_update<-function(X,tau,fish_status,
rate,total_rate){
 #Inputs:
 #X=parasite number, tau=leap size, rate=event rates
 #fish_status=survival status, total_rate=total rate
 if(runif(1) < rate[3]*tau){ # catastrophe

  X <- 0
  fish_status<-2
 }else{ # births and deaths
  X <- X + rpois(1,  rate[1]*tau) - rpois(1,rate[2]*tau)
 }
 #Returns the parasite numbers & survival status
 return(list(X = X,fish_status=fish_status))
}
```

Appendix E: R Codes for B-D-C Hybrid τ-leaping algorithms

E.2: Function for τ-leaping based on Gillespie 2001

```
HTL2001<-function(X0,b,d,c,error,ti=0,tmax=30){
 #ti<-0 #initial time, X0=initial population size
 #tmax<-30  #final time
 rate<-numeric(3) #store event rates
 save_ti <- 1:tmax  #Times to simulate
 # host fish status at each time point
 alive <- rep(2, length(save_ti))
 alive_ti <- 1 #fish starts out alive
 save_TF <- rep(FALSE, length(save_ti))
 # parasite pop at observed time point
 pop <-matrix(NA,1,length(save_ti))
 X<-X0;pop_ti <- sum(X)
 while(ti<tmax){
  #Computing event rates (birth,death & catastrophe)
  rate[1]<- b*X;rate[2]<- d*X;rate[3]<- c*X
  #representing a0(x) or total rate
  total_rate<- rate[1]+rate[2]+rate[3]
  #Computing tau on Gillespie (2001)
  tau<-(error*(b+d))/(abs(b-d)*max(b,d))
  #Switching condition
  leap_condition<- 2/total_rate  #leap condition
  #Running Tau-leaping
  if(tau>leap_condition){#Execute tau-leaping
   ti <- ti + tau #update time
   output<-tauleap_update(X,tau=tau,fish_status=alive_ti,
   rate=rate,total_rate=total_rate)
   X<-output$X
  } #end of tau-leaping
  #Running exact SSA algorithm if tau<=leap_condition
  else {#Execute exact SSA
   output<SSA_update_event(X,fish_status=alive_ti,
   rate=rate,total_rate=total_rate)
   X<- output$X;ti <- ti +output$t_incr# update time
   if (X < 0) break #break if there is negative population
   if (alive_ti == 2) break
   # saving output
   save_new <- which((ti >= save_ti) & !save_TF)
   for (i in save_new){
    pop[,i]<- pop_ti; alive[i] <- alive_ti
   }
   save_TF <- (ti >= save_ti)
   X<- X;pop_ti<- sum(X);alive_ti <- output$fish_status
  }
  #Returns parasite numbers & survival status over time
  return(list(pop=pop,alive=alive))
 }
```

E.3: Function for τ-leaping based on Gillespie and Petzold (2003)

```
HTL2003<-function(X0,b,d,c,error,ti=0,tmax=30){
 #ti<-0 #initial time, X0=initial population size
 #tmax<-30  #final time
 rate<-numeric(3) #store event rates
 Leap_sizes<- NULL #store leap size
 save_ti <- 1:tmax  #Times to simulate
 # host fish status at each time point
 alive <- rep(2, length(save_ti))
 alive_ti <- 1 #fish starts out alive
 save_TF <- rep(FALSE, length(save_ti))
 # parasite pop at observed time point
 pop <-matrix(NA,1,length(save_ti))
 X<-X0;pop_ti <- sum(X)
 while(ti<tmax){
  #Computing event rates (birth,death & catastrophe)
  rate[1]<- b*X;rate[2]<- d*X;rate[3]<- c*X
  #representing a0(x) or total rate
  total_rate<- rate[1]+rate[2]+rate[3]
  #Computing tau on Gillespie & Petzold 2003
  Leap_sizes[[1]]<- (error*(b+d))/(abs(b-d)*max(b,d))
  Leap_sizes[[2]]<- X*(error*(b+d))^2/((b+d)*max(b^2,d^2))
  tau<- min(Leap_sizes[[1]],Leap_sizes[[2]])#leap size
  #Switching condition
  leap_condition<- (1/(10*total_rate)) #leap condition
  #Running Tau-leaping
  if(tau>leap_condition){#Execute tau-leaping
   ti <- ti + tau #update time
   output<-tauleap_update(X,tau=tau,fish_status=alive_ti,
   rate=rate,total_rate=total_rate)
   X<-output$X
  } #end of tau-leaping
  #Running exact SSA algorithm if tau<=leap_condition
  else {#Execute exact SSA
   output<SSA_update_event(X,fish_status=alive_ti,
   rate=rate,total_rate=total_rate)
   X<- output$X; ti <- ti +output$t_incr# update time
   if (X < 0) break#break if there is negative values
   if (alive_ti == 2) break
   # saving output
   save_new <- which((ti >= save_ti) & !save_TF)
   for (i in save_new){
    pop[,i]<- pop_ti;alive[i] <- alive_ti
   }
   save_TF <- (ti >= save_ti)
   X<- X;pop_ti<- sum(X);alive_ti <- output$fish_status
  }

  #Returns parasite numbers & survival status over time
  return(list(pop=pop,alive=alive))
 }
```