Computational Statistics (2023) 38:1357–1389
https://doi.org/10.1007/s00180-022-01273-w

ORIGINAL PAPER

On the estimation of partially observed continuous-time Markov chains

Alan Riva-Palacio[1] · Ramsés H. Mena[1] · Stephen G. Walker[2]

Received: 17 December 2020 / Accepted: 5 August 2022 / Published online: 18 August 2022

Abstract
Motivated by the increasing use of discrete-state Markov processes across applied disciplines, a Metropolis–Hastings sampling algorithm is proposed for a partially observed process. Current approaches, both classical and Bayesian, have relied on imputing the missing parts of the process and working with a complete likelihood. However, from a Bayesian perspective, the use of latent variables is not necessary and exploiting the observed likelihood function, combined with a suitable Markov chain Monte Carlo method, results in an accurate and efficient approach. A comprehensive comparison with simulated and real data sets demonstrate our approach when compared with alternatives available in the literature.

Keywords Bayesian estimation · Transition matrix · Credit risk scoring

1 Introduction

We consider the inference problem of a partially observed continuous-time Markov chain (CTMC), written as $X := \{X(t); t \leq \tau\}$, that take values on a finite state space, $\mathbb{S} := \{1, \ldots, m\}$. Such continuous-time discrete-state systems find applications in areas such as physics, Van Kampen (2007); ecology, Fukaya and Royle (2013); neuroscience, Sauer (2016); and finance, Pardoux (2008). Hence, the need for

The authors gratefully acknowledge the support of project *CONTEX 2018-9B* and *PAPIIT-UNAM IG100221*.

✉ Alan Riva-Palacio
alan@sigma.iimas.unam.mx

Ramsés H. Mena
ramses@sigma.iimas.unam.mx

Stephen G. Walker
s.g.walker@math.utexas.edu

[1] IIMAS, UNAM, Mexico City, Mexico

[2] University of Texas at Austin, Austin, USA

efficient inference procedures is required. The literature on CTMC is extensive, with an excellent exposition provided in, for example, the monograph by Norris (1998).

If the process is observed in full, that is all moves and times of moves between states are observed, then the likelihood is easy to derive, to evaluate and maximize. The time the process spends in each state $j \in \mathbb{S}$ is exponential with parameter $\lambda_j > 0$. After this time the process moves from state j to state k with probability $p_{j,k}$, noting that $p_{j,j} = 0$. We then write the vector of (λ_j) for convenience in matrix form with $\Lambda = \text{diag}\,(\lambda_1, \ldots, \lambda_m)$, a $m \times m$ matrix. We also let $P = (p_{j,k})$ be the $m \times m$ matrix of transition probabilities. The process can be characterized by the intensity matrix $G = -\Lambda + \Lambda P$.

With a fully observed process the likelihood can be written as

$$\mathcal{L}(G) = \left(\prod_{j \neq k} p_{j,k}^{N_{j,k}} \right) \left(\prod_{j=1}^{m} \lambda_j^{N_j}\, e^{-T_j \lambda_j} \right), \tag{1}$$

where $N_{j,k}$ denotes the number of transitions from state j to state k, N_j is the number of exits to state j and T_j is the total time spent in state j, up to the final time for which the process is observed. In this case, the maximum likelihood estimator is readily obtained, see e.g. Inamura (2006),

$$\widehat{\lambda}_j = \frac{N_j}{T_j}, \quad \text{and} \quad \widehat{p}_{j,k} = \frac{N_{j,k}}{N_j}, \quad j,k \in \{1, \ldots, m\},\ j \neq k.$$

All the above quantities exist provided the process visits all states in $\mathbb{S}$.

However, in most applications, fully observed processes are not typically available and X is only partially observed at specific time points; $X_p := \{X(t_i) = x_i\}$, for $i = 1, \ldots, n$. This problem of the partially observed CTMC has received considerable attention within the literature, and recent reviews can be found in Israel et al. (2001), Bladt and Sorensen (2005), and Pfeuffer (2017), for example. The latter of these references also provides an R package implementation of various proposals.

It is well known that the maximum likelihood estimator (MLE) approach has several drawbacks; even with the MLE possibly not existing. This existence issue worsens as gaps between the partially observed records increases; see Bladt and Sorensen (2005). Given all this, an EM algorithm treating the unobserved record lying between the partial observations as latent variables, is currently a popular approach, see dos Reis and Smith (2018) and Pfeuffer et al. (2019). On the other hand, a Bayesian treatment of the problem has also been presented; Bladt and Sorensen (2005) proposed a Gibbs sampler approach based on a rejection sampling algorithm for the unobserved states. A more sophisticated Gibbs sampling approach, which relies on the simulation of the CTMC over an interval given the start and end states was considered in Fearnhead and Sherlock (2006), and implemented in Pfeuffer (2017). Another approach is to use the Gibbs sampler in Rao and Teh (2013), which simulates the CTMC conditioned on possibly noisy observations.

Therefore, current approaches to tackle the inference problem for X_p prefer the use of a complete likelihood. We believe this has been in part motivated in order to achieve the existence of a simple looking complete likelihood, even if it is difficult to obtain; which in a Bayesian setting allows for the use of conjugate priors and avoidance of parameter tuning when performing Markov-chain–Monte-Carlo (MCMC) methods. However, problems with using latent variables for sampling the missing process in between observed states are the high computational cost for doing so and the possibly high correlations which slow the mixing of the sampler. An illustration of this problem in a simple scenario consisting of a binary Markov chain is presented in Appendix B.

There have been some successful applications of Bayesian inference for partially observed CTMCs without the use of latent variables though all use some form of restrictions compared to our approach. The paper of Amoros et al. (2019) considers a CTMC with only two states for the modeling of presence or absence of hepatocellular carcinoma. As the transition probabilities for the two states are known exactly, there is no need to impute missing data. So despite the overall setting of a hidden Markov model, the two states make their algorithm simpler. As well as Amoros et al. (2019), Sherlock et al. (2010) consider the two state case for which the partial and full likelihoods are available. In Georgoulas et al. (2017) the authors consider a pseudo marginal approach for population models based on CTMCs with infinite states which are then truncated into a finite state setting. As such it restricts attention to rate matrices which depend on a small number of parameters, usually less than the number of states. On the other hand, a full rate matrix has $d(d-2)$ free parameters where d is the number of states. Zhao et al. (2016) considers the use of CTMCs for phylogenetic protein modelling. In particular, they propose a generalized linear model based parameterization for the generator matrix and use data augmentation to complete the likelihood. On the other hand, Zhao et al. (2016) consider the use of constrained rate matrices instead of full rate matrices due to their particular application in phylogenetics. In fact, we do find a suitable *dependent* Metropolis proposal distribution based on a parameterization of the rate matrix in terms of probability vectors and scalars. A dependent Dirichlet proposal distribution is shown in our paper to work well. Also, in Sect. 2.5, they discuss the use of MH samplers for partially observed CTMCs with the following proposals: (1) additively or multiplicatively perturbing each entry of the generator matrix; or (2) setting an independent Dirichlet prior for the transition probabilities at each state. In our experiments, we found that in many settings it is better to use a random walk proposal based on a Dirichlet distribution rather than independent ones. As we will see, our new algorithm is applied to a fully specified rate matrix.

In the present work we focus on Bayesian inference for the partially observed CTMC without using latent variables; we use the likelihood function directly by evaluation of matrix exponentials and perform posterior inference via a Metropolis–Hastings approach where the generator matrices are fully specified and not constrained. The outline is as follows: Sect. 2 presents the theory and the proposed algorithm. Section 3 includes simulation and real data studies where we compare with, while Section 4 concludes with a brief discussion.

2 Bayesian procedures

Let a CTMC be partially observed at times $\{0, t_1, \dots, t_n\}$, and denote the observations as $\{X(0), X(t_1), \dots, X(t_n)\}$ where, for short, we write $X(t_i) = x_i$, and $\Delta_i = t_i - t_{i-1}$, with $t_0 = 0$. The probability that the process is in state k at time t after being observed in state j at time 0 is denoted by $Q_t(j, k)$. Such a transition probability can be expressed in terms of the generator matrix G as follows:

$$Q_t(j,k) = [\exp(t\,G)]_{(j,k)}, \tag{2}$$

which is the (j, k)th element of the matrix $\exp(tG)$; written in full as

$$\exp(tG) = \sum_{l=0}^{\infty} \frac{t^l}{l!} G^l \tag{3}$$

with $G^0 = I$, the $m \times m$ identity matrix. The likelihood then becomes

$$\mathcal{L}(G) = \prod_{i=1}^{n} Q_{\Delta_i}(x_{i-1}, x_i) = \prod_{i=1}^{n} [\exp(\Delta_i\, G)]_{(x_{i-1}, x_i)}. \tag{4}$$

For more on the theory presented here, see, for example, Grimmett and Stirzaker (1982).

Evaluation of the exponential of a matrix is a well studied topic, see Higham (2005), and it is readily available in most scientific computing programming languages, such as R, from the package “expm”. So computation of the likelihood is possible and simulation from the corresponding posterior distribution can be performed with a Metropolis–Hastings sampler.

On the other hand, maximization of the above likelihood is difficult due to the constraints involving the generator matrix G. Indeed, it appears to be a strategy that has not been directly tried. This issue motivated searches for suitable latent variables, which once found have provided a source of inference for both classical and Bayesian approaches alike.

In the next two subsections we consider Bayesian approaches; the first is the existing strategy involving latent variables and the second is our proposal which obviates the need for latent variables by working with the likelihood (4) directly. The former is currently the more popular approach as it is the natural Bayesian version of the EM methodology, see for example dos Reis and Smith (2018) and Pfeuffer et al. (2019).

2.1 Gibbs sampler and latent variables

In order to utilize the complete likelihood (1) given partially observed data, it is necessary to sample the complete trajectory of the CTMC between the discretely observed times where we know the states of the CTMC. An elaborate way is to sample the process forward from the start; i.e. between consecutive observed times (t_j, t_{j+1}) the plan is to sample the waiting time $\Delta'_1 = t'_1 - t_1$ at state x_j, the unobserved transition to a state x'_1, the waiting time $\Delta'_2 = t'_2 - t'_1$ at state x'_1, the unobserved transition to state x'_2 and so on; the simulated trajectory is accepted if at time t_{j+1} the process is at the observed state x_{j+1}. This would be how to sample the missing process conditional on the G. See Fig 1 for an illustration, where k states have been sampled in between known states at times t_j and t_{j+1}.

This is equivalent to a rejection sampling algorithm; i.e. sample the missing process and accept it if it hits the observed part of the process. With a few states this might work adequately, but obviously will run into efficiency issues when the number of states becomes large or the time elapsed between discrete observations is large. This rejection sampler approach was originally used by Bladt and Sorensen (2005) to implement a Gibbs sampler. An alternative more efficient version is given by Fearnhead and Sherlock (2006), where an algorithm for exact simulation of the CTMC between two known states, s_0 at time t_0 and s_e at time $t_e > t_0$, was presented. We note that such a simulation scheme relies on the evaluation of $[\exp(t_e - t_0)\,G)]_{(s_0, s_e)}$ to determine the number of unobserved transitions k in (t_0, t_e). We also note that matrix exponential evaluations are needed to calculate the likelihood (4). The usefulness of such an algorithm for the Gibbs sampling procedure of Bladt and Sorensen (2005) was discussed in Fearnhead and Sherlock (2006) and implemented in the R package of Pfeuffer (2017). In Algorithm 1 we present the corresponding pseudocode for simulation of CTMC bridges.

Algorithm 1 (Fearnhead and Sherlock (2006)) Simulation of a CTMC (X_t) with generator G over an interval $[t_0, t_e]$ given the start and end states s_0 and s_e.

```
Let $\rho = \max\{\lambda_1, \ldots, \lambda_m\}$, $\Delta = t_e - t_0$ and $M = \rho^{-1} G + I$. Simulate the number of
  transitions $N$ between times $t_0$ and $t_e$ given by
    $$\mathbb{P}[N = r] = \frac{\exp(-\rho\Delta)(\rho\Delta)^r\,[M^r]_{(s_0,s_e)}}{r!\,[\exp(G\Delta)]_{(s_0,s_e)}},$$
Simulate $t'_1, \ldots, t'_N$ uniformly from the interval $[t_0, t_e]$.
Let $t'_0 = t_0$ and $s'_0 = s_0$. Simulate the states $(s_1 \ldots, s_N)$ of the CTMC at times
  $(t'_1, \ldots, t'_N)$, respectively, from
    $$\mathbb{P}\left[X_{t'_j} = s \,|\, X_{t'_{j-1}} = s'_{j-1},\, X_{t_e} = s_e\right] = \frac{[M]_{(s'_{j-1},s)}\,\left[M^{r-j}\right]_{(s,s_e)}}{[M^{r-j+1}]_{(s'_{j-1},s_e)}}.$$
```

Algorithm 2 Gibbs sampler 1

```
for $k \in \{1,\ldots,l\}$ do
    Draw $\left\{X_t^{(k+1)},\ t_i \leq t \leq t_{i-1} \mid X_{t_{i-1}}^{(k)} = x_{i-1}, X_{t_i}^{(k)} = x_i\right\}$ from Algorithm 1 using the generator $G^{(k)}$.
    Draw the generator $G^{(k+1)}$ given $\left\{X_t^{(k+1)},\ 0 \leq t \leq t_e\right\}$ from the posterior distribution given by the complete likelihood (1) and selected prior distribution.
end for
```

A Bayesian approach using latent variables can be based on sampling the complete trajectory using a Gibbs sampler where we sample iteratively from

$$\left[\{X(t),\ t_i < t < t_{i+1}\} \mid G,\ X(t_i) = x_i,\ X(t_{i+1}) = x_{i+1}, \Delta_{i+1} = t_{i+1} - t_i\right],$$

for $i = 1, \ldots, n-1$, and then sample

$$[G \mid \{X(t),\ 0 \leq t \leq \tau\}],$$

which is based on (1), suitably multiplied by the prior. Pseudocode for the Gibbs sampler is given in Algorithm 2. An advantage of such a Gibbs sampling strategy is that the algorithm is automatic, in the sense that no tuning is required. Another alternative is to use the Gibbs sampler of Rao and Teh (2013), presented in Algorithm 3, where uniformization is used to draw a CTMC full trajectory conditioned on partial observations from a previously drawn full trajectory and the *forward filtering backward sampling* algorithm is used to resample the states. In contrast with the previous Gibbs sampler evaluation of matrix exponentials, which can be computationally expensive for large dimensions, this algorithm does not require such computations. The corresponding Gibbs sampler is presented in Algorithm 4.

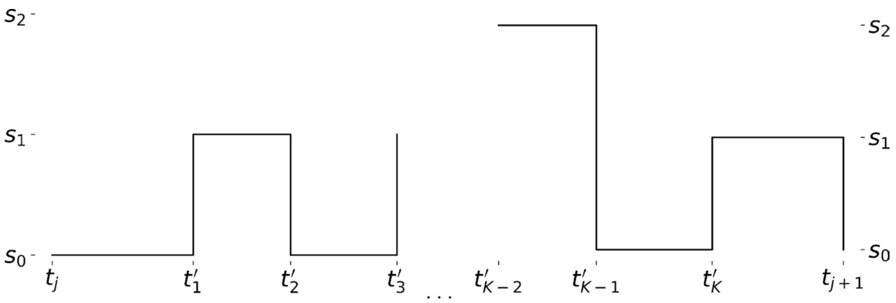


Fig. 1 Observed states x and y separated by a time of length t

Algorithm 3 (Rao and Teh (2013)) Simulation of a CTMC $\{X_t^{\text{new}}\,;\,t\in[t_0,t_e]\}$ with generator G given partial observations $X_p^{\text{new}}=\{X^{\text{new}}(t_i)=x_i\}_{i=0}^n$ and a previous draw $\left\{X_t^{\text{old}}\,;\,t\in[t_0,t_e]\right\}\mid G,X_p^{\text{old}}=\left\{X^{\text{old}}(t_i)=x_i\right\}_{i=0}^n$.

```
1: Let $\rho>\max\{\lambda_1,\ldots,\lambda_m\}$ and sample Poisson process $U_t$ in $[t_0,t_e]$ with piecewise constant rate
       $$R(t)=\rho+\lambda_{X_t^{\text{old}}}.$$
   Define $W$ as the sorted, in increasing order, union of the jump times of $U$ and $X^{\text{old}}$ in $[t_0,t_e]$. Denote the entries of $W$ as $w_j$.
2: Use the *forward filtering backward sampling* algorithm to sample a discrete time Markov chain $\{Z_j\,;\,j=1,\ldots,|W|\}$ given by a transition matrix $M=\frac{1}{\rho}G+I$. The likelihood associated to the partial observations is given by
       $$L_j(x)=\prod_{\{i\,;\,t_i\in[w_j,w_{j+1})\}}\mathbb{1}\left\{X^{\text{old}}(t_i)=x_i=x\right\}.$$
3: Return a CTMC trajectory $\{X_t^{\text{new}}\,;\,t\in[t_0,t_e]\}$ which starts with value $x_0$ at time $t_0$ and changes to state $Z_j=z_j$ at time $w_j$.
```

Algorithm 4 Gibbs sampler 2

```
1: Initialize a generator matrix $G^{(0)}$ and use Algorithm 1 to initialize $\left\{X_t^{(0)}\mid X_{t_0}^0=x_{(0)},\ldots,X_{t_n}^{(0)}=x_n\right\}$ using $G^{(0)}$.
2: for $k\in\{1,\ldots,l\}$ do
3:     Draw the generator $G^{(k)}$ given $\left\{X_t^{(k-1)},\ 0\le t\le t_e\right\}$ from the posterior distribution given by the complete likelihood (1) and selected prior distribution.
4:     Draw $\left\{X_t^{(k)}\mid X_{t_0}^{(k)}=x_0,\ldots,X_{t_n}^{(k)}=x_n\right\}$ using Algorithm 3 with $G^{(k)}$ and $\left\{X_t^{(k-1)}\mid 0\le t\le t_e\right\}$.
5: end for
```

2.2 Metropolis–Hastings sampler

In this subsection we propose a more direct Bayesian approach which does not rely on the introduction of latent variables. We do this by considering a Metropolis–Hastings sampler for the posterior distribution based on the likelihood (4). The Metropolis–Hastings algorithm works by introducing a proposal distribution for the parameters of interest. The proposals for the parameters in the matrices Λ and P are given as follows: Regarding the diagonal matrix Λ with j-th diagonal entry, denoted by λ_j, proposal values are given by $q(\lambda_j'|\lambda_j)$, for each $j\in\{1,\ldots,m\}$, to be a LogNormal distribution with mean $\log\lambda_j$ and standard deviation $\sigma_j^2=\sigma^2$.

For the stochastic matrix P, we consider three possible proposal distributions. Let $\boldsymbol{p}_j$ denote the vector consisting of the off-diagonal entries in each row of P.

The first proposal is given by $q(\boldsymbol{p}_j'|\boldsymbol{p}_j)$ to be a Dirichlet distribution with parameters $c_j\boldsymbol{p}_j$ with $c_j \in \mathbb{R}^+$. For such a Dirichlet proposal the mean for $\boldsymbol{p}_j'$ is $\boldsymbol{p}_j$. An alternative choice is to let $q(\boldsymbol{p}_j'|\boldsymbol{p}_j)$ be a Dirichlet distribution with parameter $c_j\boldsymbol{p}_j + \mathbf{1}$, where $\mathbf{1}$ is a vector consisting of entries 1. Such a Dirichlet distribution has mode $\boldsymbol{p}_j$. In practice we found this latter proposal is more robust. Finally we consider a proposal $q(p_{j,k}'|p_{j,k})$ to be a LogitNormal distribution with mean $\text{logit}(p_{j,k})$. If a draw from such a proposal is accepted then we have to normalize the vector $\boldsymbol{p}_j$ by the sum of its entries, so the matrix P remains a stochastic matrix. This proposal distribution is more suitable when some entries of P are zero or close to zero.

Algorithm 5 Metropolis–Hastings sampler

```
Draw $G'$ from the selected proposal distribution $q(G'|G^{(k)})$.
Let
    $$\alpha = \min\left\{1, \frac{\mathcal{L}(G')\,\pi(G')q(G^{(k)}|G')}{\mathcal{L}(G^{(k)})\,\pi(G^{(k)})q(G'|G^{(k)})}\right\}$$
  where $\mathcal{L}$ is the likelihood (4) and $\pi$ is the selected prior over $G$. Take
    $$G^{(k+1)} = \begin{cases} G' & \text{with probability } \alpha \\ G^{(k)} & \text{with probability } 1-\alpha. \end{cases}$$
```

Each time a proposal is made, we recompute the likelihood function in order to determine whether the proposal is accepted. We illustrate with the update for λ_1; we sample λ_1' from $q(\lambda_1'\,|\,\lambda_1)$ and accept this move with probability

$$\alpha = \min\left\{1, \frac{\mathcal{L}(G')\,\pi(\Lambda')q(\lambda_1\,|\,\lambda_1')}{\mathcal{L}(G)\,\pi(\Lambda)q(\lambda_1'\,|\,\lambda_1)}\right\},$$

where $G' = -\Lambda' + \Lambda' P$ and $\Lambda' = (\lambda_1', \lambda_2, \ldots, \lambda_m)$, while $G = -\Lambda + \Lambda P$ and $\Lambda = (\lambda_1, \lambda_2, \ldots, \lambda_m)$. A similar and obvious procedure follows for the other parameters making up Λ and P, a complete summary of the algorithm is presented in Appendix 1. In Algorithm 5 we give pseudocode for the Metropolis–Hastings sampler. The exponential of the matrix G is calculated using the scaling and squaring method of Higham (2005) as implemented in the Julia LinearAlgebra library https://docs.julialang.org/en/v1/stdlib/LinearAlgebra/. Alternatively, when the number of states becomes prohibitively large we can use the algorithm of Al-Mohy and Higham (2011) by calling the Python implementation of the SciPy library (Virtanen et al. 2020).

2.3 Prior selection

With such a Bayesian framework, the setting of prior distributions is relatively straightforward. For example, we assign independent gamma priors to each λ_j, with shape and rate parameters $\alpha_j, \beta_j \in \mathbb{R}^+$. Each row of P can be assigned a Dirichlet prior with parameters chosen so the prior is uniform on the simplex. In previous Bayesian analyses the prior distribution was assigned on the off-diagonal elements of the matrix G, e.g. Bladt and Sorensen (2005), Pfeuffer (2017). The main motivation of this was to assign gamma priors so that there is conjugacy when considering the complete likelihood (1). However, a prior assignment at the level of the matrices Λ and P is useful to elicit expert information given the interpretation of the matrices. The jth diagonal elements of Λ give the rates of the exponential times the Markov chain waits in state j and the jth row of P gives transition probabilities for the corresponding state changes. In particular, for the simulation studies where we compare Gibbs samplers with the Metropolis–Hastings algorithm, we will use gamma priors with equal shape parameters so that it coincides with the Dirichlet prior on the rows of P, with gamma priors on the diagonal entries of D, as discussed above.

3 Illustrations

We programmed the Metropolis–Hastings algorithm, and the two Gibbs samplers in the Julia programming language with the code available at https://github.com/alan7riva/CTMC. Overall, we observe that the Metropolis–Hastings algorithm is not only significantly simpler to code but it is also faster and has better mixing throughout our experiments.

3.1 Two state toy example

Here we consider a two state problem with generator matrix

$$G = \begin{pmatrix} -\lambda_1 & \lambda_1 \\ \lambda_2 & -\lambda_2 \end{pmatrix}$$

for $\lambda_1, \lambda_2 > 0$. In such a case we know explicitly $Q(t) = \exp(tG)$ and assuming that we discretely observe a Markov chain associated with this generator matrix G over a time mesh Δk, $\Delta \in \mathbb{R}^+$ and $k \in \{1, \dots, n\}$ for some $n \in \mathbb{N}$, the likelihood $\mathcal{L}(\lambda)$ is also computable; details are given in Appendix B. We drew simulations for the above CTMC with $\boldsymbol{\lambda} = (\lambda_1, \lambda_2) \in \{(1, 1), (2, 1)\}$. The prior distributions were $\lambda \sim \text{Gamma}(2, 1)$ for all the experiments. The variance in the LogNormal proposals for λ_i were set to 0.5. A CTMC was generated until 1000 transitions were obtained and we considered the discrete observations given by times Δk with $\Delta = 1.0$ and $1 \leq k \leq \min\{n\,; X(n\Delta) \text{ was fully observed}\}$. For the simulation studies with rates $\boldsymbol{\lambda}$ we obtained, respectively, 1014 and 1482 partial observations. In Figs. 2 and 3 we show the posterior fits of λ_1 and λ_2 respectively with 50000 iterations of MCMC after diagnosing convergence with the potential scale reduction factor of Gelman and Rubin (1992) below 1.01 calculated with 4 chains started at random.

In Tables 1 and 2 we show the computation time, effective sample sizes (ESS) and ESS per second of computation before diagnosing convergence, all averaged over the 4 chains started at random values; the same quantities are also presented

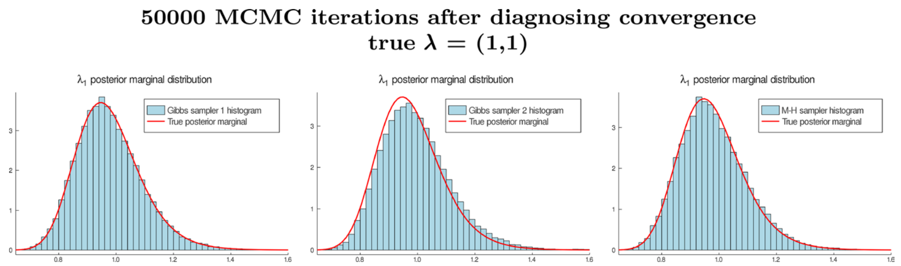


Fig. 2 Marginal histograms of λ_1 for Gibbs samplers 1, 2 and Metropolis–Hastings with 5000 iterations after diagnosing convergence, with potential scale reduction, compared with the true posterior distribution. True $\boldsymbol{\lambda} = (1, 1)$

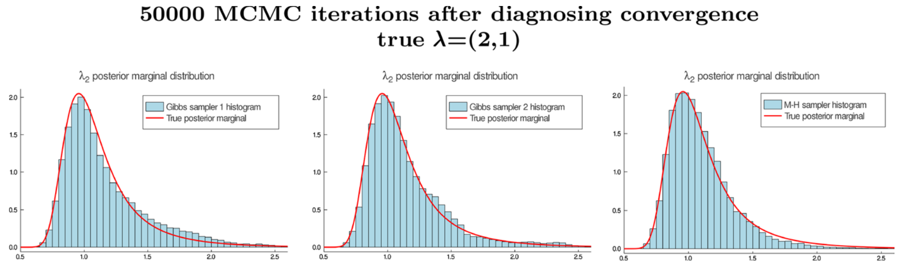


Fig. 3 Marginal histograms of λ_2 for Gibbs samplers 1, 2 and Metropolis–Hastings with 5000 iterations after diagnosing convergence, with potential scale reduction, compared with the the posterior distribution. True $\boldsymbol{\lambda} = (2, 1)$

Table 1 Gibbs and Metropolis–Hastings comparison for toy simulation study with $\boldsymbol{\lambda} = (1, 1)$

	MH	Gibbs 1	Gibbs 2
Iterations until diagnosing convergence	2023	1069	4457
Time (s) before diagnosing convergence (mean)	0.341	14.3	75.5
Time (s) after diagnosing convergence	6.29	524	864
ESS after diagnosing convergence λ_1	864	2291	1003
ESS/sec after diagnosing convergence λ_1	137	4.37	1.16
ESS after diagnosing convergence λ_2	871	2240	1007
ESS/sec after diagnosing convergence λ_2	138	4.28	1.17

for a 50000 iterations run started at the final value of one of the 4 pilot runs used to assess convergence, the choice of such run was sampled from a uniform distribution.

We observe that the Metropolis–Hastings sampler is significantly faster than the Gibbs samplers, resulting in bigger ESS per second. In our experiments Gibbs sampler 1 outperforms Gibbs sampler 2 due to the latter being considerably slow. For this reason in the following studies we focus our comparison only on Gibbs sampler 1.

Table 2 Gibbs and Metropolis–Hastings comparison for toy simulation study with $\boldsymbol{\lambda} = (2, 1)$

	MH	Gibbs 1	Gibbs 2
Iterations until diagnosing convergence	2521	17483	3891
Time (s) before diagnosing convergence (mean)	0.424	348	300
Time (s) after diagnosing convergence	5.41	733	11657
ESS after diagnosing convergence λ_1	166	36	59
ESS/sec after diagnosing convergence λ_1	30.7	0.14	0.005
ESS after diagnosing convergence λ_2	165	36	59
ESS/sec after diagnosing convergence λ_2	30.5	0.14	0.005

3.2 Five state simulation example

In this study, we draw observations from a CTMC with state space $\mathbb{S} = \{1, 2, 3, 4, 5\}$. In all the experiments presented in this section, the stochastic matrix $P = (p_{j,k})$ associated with the Markov Chain is chosen such that $p_{j,k} = 0.25$ for $j \neq k$. On the other hand we take $\Lambda = (\lambda, \lambda, \lambda, \lambda, \lambda)$ for 3 types of simulation using $\lambda \in \{0.25, 0.5, 1, 2, 4\}$. Let n be the total number of observations, for which we take the inter-arrival times $\Delta_i = \Delta$ for $i \in \{1, \dots, n\}$. For different values of λ and Δ we choose the number of observations $n(\lambda, \Delta) = 5000\lambda\Delta$, so we have about $\mathbb{E} \sum_{i=1}^{n} T_i / \Delta = 5000$ observations in each study. The prior distributions were $\lambda_i \sim \text{Exp}(1)$, $i \in \{1, \dots, 5\}$ and $p_{j,k} \sim \text{Gamma}(0.25, 1)$, $j \in \{1, \dots, 5\}$, $k \in \{1, \dots, 4\}$ pairwise independently for all the experiments. The variance in the LogNormal proposals for λ_i were set to 0.5. For the proposals of $\boldsymbol{p}_j$ we used a Dirichlet with common parameter 0.5. We ran the Gibbs and Metropolis–Hastings sampler; i.e. Algorithms 2 and 5, respectively, with 10000 iterations for each experiment. In Fig. 4, we present the mean generators obtained from each sampler after removing 3000 burn-in iterations with $\Delta = 1$. Respective figures for $\Delta \in \{0.25, 0.5, 2, 4\}$ are presented in the supplementary material. We observe that when the scale $1/\lambda$ is equal to the observation window Δ, both the Gibbs sampler 1 and Metropolis–Hastings chains produce a mean generator which fits well with the true values; with the Metropolis–Hastings performance being faster so allowing for better estimation if desired.

When the scale of the waiting times for the true process to change state are smaller than the observation window Δ, one could expect the Gibbs sampler, which uses as auxiliary variables the unobserved transitions, to perform better than the Metropolis–Hastings sampler. However, we observe that this is not the case; for instance, when $\Delta = 0.5$ with scale 0.125 we have that the Frobenius distance between the true generator matrix and the mean generator associated with the Metropolis–Hastings sampler is 5.79, while for the Gibbs sampler we get a distance of 6.63. Table 3 shows for each experiment the computation time and Frobenius distance to the true generator matrix for the Metropolis–Hastings and Gibbs samplers as well as for the EM algorithm, which is usually used in credit risk applications (see for instance Pfeuffer (2017)). All starting points are the same for the three methods. Another advantage of the Metropolis–Hasting algorithm

Table 3 Frobenius distance between estimated posterior mean and true generator matrix across 5 by 5 simulation studies

	Gibbs 1	M-H	EM
$\Delta = 4$			
$1/\lambda = 4$	0.0455	0.0448	0.0436
$1/\lambda = 2$	2.4357	1.4644	0.1516
$1/\lambda = 1$	2.0586	1.6259	1.0289
$\Delta = 2$			
$1/\lambda = 2$	0.1033	0.1048	0.1059
$1/\lambda = 1$	1.1454	1.005	0.5163
$1/\lambda = 0.5$	1.8128	2.1211	1.4434
$\Delta = 1$			
$1/\lambda = 1$	0.2393	0.2358	0.2235
$1/\lambda = 0.5$	2.5025	1.7987	0.7307
$1/\lambda = 0.25$	3.8773	3.5661	2.3316
$\Delta = 0.5$			
$1/\lambda = 0.5$	0.3224	0.3202	0.3136
$1/\lambda = 0.25$	1.0471	1.2246	0.9696
$1/\lambda = 0.125$	6.6398	5.7922	4.3873
$\Delta = 0.25$			
$1/\lambda = 0.25$	0.8714	0.8586	0.8617
$1/\lambda = 0.125$	4.9129	3.7802	4.0352
$1/\lambda = 0.0625$	14.9219	13.5827	8.0351

showcased by these experiments was again the improvement of computation times as well as their stability across experiments in comparison to the Gibbs sampler where depending on the observation window and true scale the algorithm could be slower due to the bridges simulations. Similar behavior was observed for Gibbs sampler 2. Table 4 shows computation time.

3.3 Credit risk analysis

In Pfeuffer (2017) a package for analyzing continuous-time Markov chain models with partially observed data is presented. In particular, they implement the Gibbs sampler of Bladt and Sorensen (2005) with a default setting to use the exact simulation of the Markov chain over an interval given the start and end states, as presented and discussed in Fearnhead and Sherlock (2006), rather than the acceptance–rejection sampling algorithm of Bladt and Sorensen (2005). The foremost application of the package is in credit risk where the Markov chain states $\{\mathrm{AAA}, \mathrm{AA}, \mathrm{A}, \mathrm{BBB}, \mathrm{BB}, \mathrm{B}, \mathrm{C}, \mathrm{D}\}$ correspond to credit ratings. In Fig. 5 we show a comparison of the new Metropolis–Hastings algorithm, using the LogitNormal proposal in P with the Gibbs sampler for the credit risk application presented in Pfeuffer (2017). The choice of the priors was taken so they coincide for both samplers. Here we take the priors for each λ_j as Gamma $(7, 5)$ and the Dirichlet distribution for each $\boldsymbol{p}_j$ as Dirichlet with common parameter 1. The variance in the

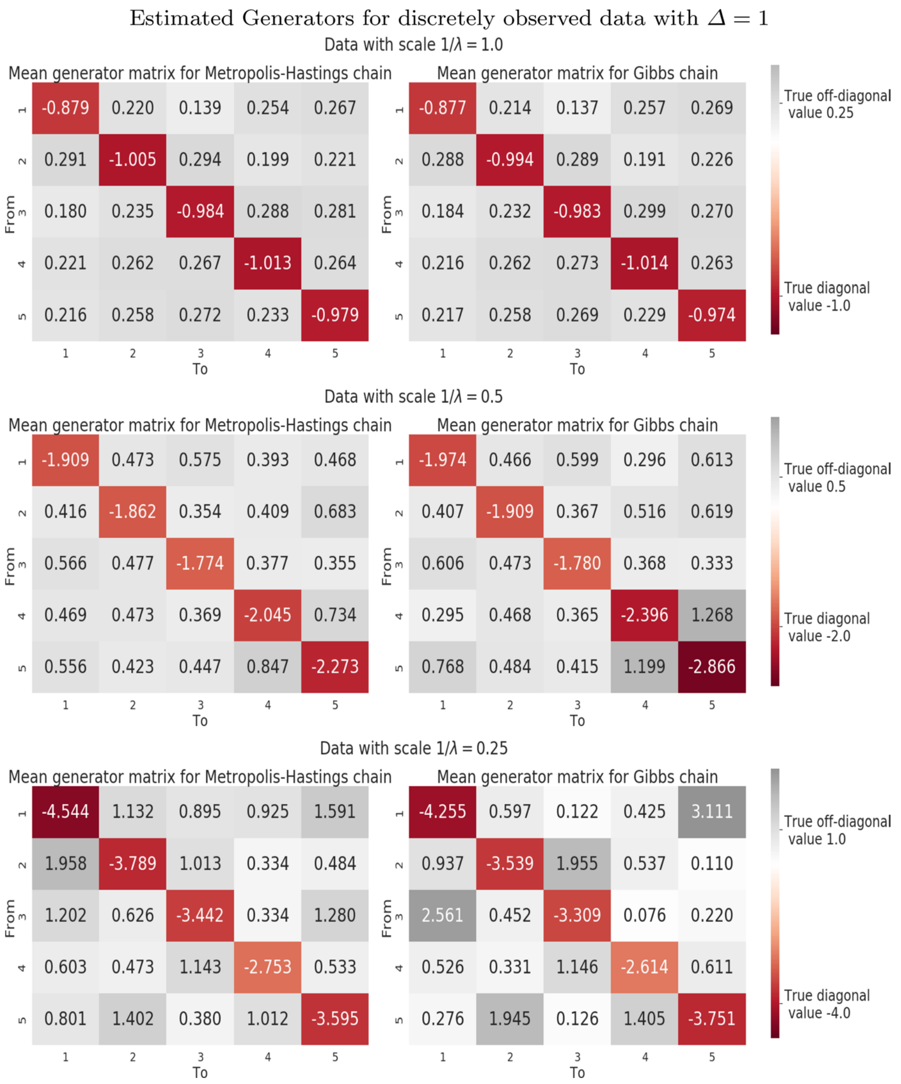
Table 4 Computation time for 5 by 5 simulation studies

	Gibbs 1 time	M-H time
$\Delta = 4$		
$1/\lambda = 4$	722	7.1
$1/\lambda = 2$	1055	7.8
$1/\lambda = 1$	1050	6.7
$\Delta = 2$		
$1/\lambda = 2$	778	7.2
$1/\lambda = 1$	807	7.3
$1/\lambda = 0.5$	909	8.7
$\Delta = 1$		
$1/\lambda = 1$	696	7.4
$1/\lambda = 0.5$	829	8.7
$1/\lambda = 0.25$	931	7.6
$\Delta = 0.5$		
$1/\lambda = 0.5$	1460	11.4
$1/\lambda = 0.25$	1485	12.3
$1/\lambda = 0.125$	1807	12
$\Delta = 0.25$		
$1/\lambda = 0.25$	707	7.1
$1/\lambda = 0.125$	757	6.7
$1/\lambda = 0.0625$	797	7.7

LogNormal and LogitNormal proposals are taken to be 0.5. We observe that the fitted values for the mean generator matrices, obtained from a run of length 100000 of which the first 10000 iterations were discarded as burn in, are close to the Gibbs mean values; the Frobenius distance between the matrices is less than 0.11. Hence, we have shown in a not too demanding scenario that the use of latent variables is not necessary and equal performance can be achieved using the observed likelihood.

4 Discussion

Bayesian inference for a partially observed CTMC, without the use of latent variables, has not, to the best of our knowledge, been considered before. We argue that the use of latent variables in this problem is unnecessary and leads to more complex and difficult implementation algorithms. On the other hand, the use of and convergence of the Metropolis–Hastings MCMC sampler is both simpler and faster, as showcased in Sect. 3.1. The Metropolis–Hastings approach is computationally simple yet efficient in comparison with Gibbs samplers based on imputing missing latent observations in the form of CTMC bridges between the partial observations. Both algorithms accurately target the posterior distributions, however high correlations in the bridge sampler steps cause slow mixing and small effective sample sizes per second in this particular setting. The only complicated aspect to our algorithm is the computation of the exponential of a



Estimated Generators for discretely observed data with $\Delta = 1$

Data with scale $1/\lambda = 1.0$

Mean generator matrix for Metropolis-Hastings chain

From \ To	1	2	3	4	5
1	-0.879	0.220	0.139	0.254	0.267
2	0.291	-1.005	0.294	0.199	0.221
3	0.180	0.235	-0.984	0.288	0.281
4	0.221	0.262	0.267	-1.013	0.264
5	0.216	0.258	0.272	0.233	-0.979

Mean generator matrix for Gibbs chain

From \ To	1	2	3	4	5
1	-0.877	0.214	0.137	0.257	0.269
2	0.288	-0.994	0.289	0.191	0.226
3	0.184	0.232	-0.983	0.299	0.270
4	0.216	0.262	0.273	-1.014	0.263
5	0.217	0.258	0.269	0.229	-0.974

Data with scale $1/\lambda = 0.5$

Mean generator matrix for Metropolis-Hastings chain

From \ To	1	2	3	4	5
1	-1.909	0.473	0.575	0.393	0.468
2	0.416	-1.862	0.354	0.409	0.683
3	0.566	0.477	-1.774	0.377	0.355
4	0.469	0.473	0.369	-2.045	0.734
5	0.556	0.423	0.447	0.847	-2.273

Mean generator matrix for Gibbs chain

From \ To	1	2	3	4	5
1	-1.974	0.466	0.599	0.296	0.613
2	0.407	-1.909	0.367	0.516	0.619
3	0.606	0.473	-1.780	0.368	0.333
4	0.295	0.468	0.365	-2.396	1.268
5	0.768	0.484	0.415	1.199	-2.866

Data with scale $1/\lambda = 0.25$

Mean generator matrix for Metropolis-Hastings chain

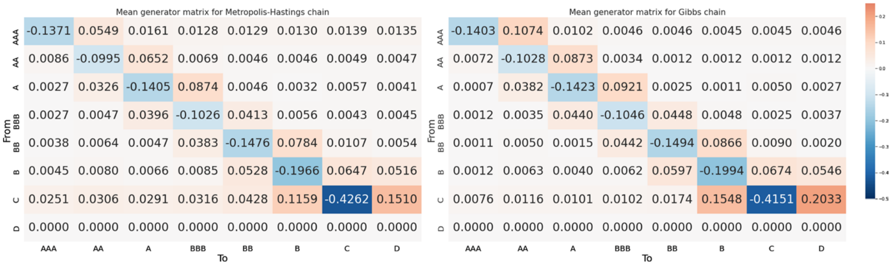
From \ To	1	2	3	4	5
1	-4.544	1.132	0.895	0.925	1.591
2	1.958	-3.789	1.013	0.334	0.484
3	1.202	0.626	-3.442	0.334	1.280
4	0.603	0.473	1.143	-2.753	0.533
5	0.801	1.402	0.380	1.012	-3.595

Mean generator matrix for Gibbs chain

From \ To	1	2	3	4	5
1	-4.255	0.597	0.122	0.425	3.111
2	0.937	-3.539	1.955	0.537	0.110
3	2.561	0.452	-3.309	0.076	0.220
4	0.526	0.331	1.146	-2.614	0.611
5	0.276	1.945	0.126	1.405	-3.751

Fig. 4 Comparison of estimated generator matrices for Metropolis–Hastings and Gibbs chains for $\Delta = 1$ with $\lambda = 4$ (first row), $\lambda = 2$ (second row) and $\lambda = 1$ (third row)

matrix for which adequate software is now currently available. Our approach is general as it allows for the inference of fully specified generator matrices, i.e. without further constraints than having negative values in the diagonal, non-negative values off the diagonal and zero sum rows. Two approaches were



Mean generator matrix for Metropolis-Hastings chain

From \ To	AAA	AA	A	BBB	BB	B	C	D
AAA	-0.1371	0.0549	0.0161	0.0128	0.0129	0.0130	0.0139	0.0135
AA	0.0086	-0.0995	0.0652	0.0069	0.0046	0.0046	0.0049	0.0047
A	0.0027	0.0326	-0.1405	0.0874	0.0046	0.0032	0.0057	0.0041
BBB	0.0027	0.0047	0.0396	-0.1026	0.0413	0.0056	0.0043	0.0045
BB	0.0038	0.0064	0.0047	0.0383	-0.1476	0.0784	0.0107	0.0054
B	0.0045	0.0080	0.0066	0.0085	0.0528	-0.1966	0.0647	0.0516
C	0.0251	0.0306	0.0291	0.0316	0.0428	0.1159	-0.4262	0.1510
D	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000

Mean generator matrix for Gibbs chain

From \ To	AAA	AA	A	BBB	BB	B	C	D
AAA	-0.1403	0.1074	0.0102	0.0046	0.0046	0.0045	0.0045	0.0046
AA	0.0072	-0.1028	0.0873	0.0034	0.0012	0.0012	0.0012	0.0012
A	0.0007	0.0382	-0.1423	0.0921	0.0025	0.0011	0.0050	0.0027
BBB	0.0012	0.0035	0.0440	-0.1046	0.0448	0.0048	0.0025	0.0037
BB	0.0011	0.0050	0.0015	0.0442	-0.1494	0.0866	0.0090	0.0020
B	0.0012	0.0063	0.0040	0.0062	0.0597	-0.1994	0.0674	0.0546
C	0.0076	0.0116	0.0101	0.0102	0.0174	0.1548	-0.4151	0.2033
D	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000

Fig. 5 Estimated generator matrices for Metropolis–Hastings and Gibbs chains for the credit risk data of Pfeuffer (2017)

proposed, one for generator matrices with non-zero off-diagonal entries and one for the possibility of such zero entries.

Appendix A: Metropolis-Hastings algorithm (5) details

The Metropolis–Hastings moves used in our work are the following:

- For $\lambda \in \Lambda = \{\lambda_1, \dots, \lambda_m\}$ let $g : \mathbb{R} \to \mathbb{R}^+$ be given by $g(x) = e^x$ and propose moves from λ to $\tilde{\lambda}$ by
$$\tilde{\lambda} = g(g^{-1}(\lambda) + Z)$$
with $Z \sim$ Norma$(0, 1)$ so $\tilde{\lambda}$ has a LogNormal distribution. The corresponding transition kernel is given by $q(\tilde{\lambda} \mid \lambda) \propto e^{-0.5(\log(\lambda)-\log(\tilde{\lambda}))^2}\,\tilde{\lambda}^{-1}$.
- For transition matrix P with no zeros in off-diagonal entries and $\boldsymbol{p} \in \left\{\left(p_{j,1}, \dots, p_{j,j-1}, p_{j,j+1}, \dots, p_{j,m}\right) : 1 \le j \le m\right\}$, let $c > 0$ and propose
$$\tilde{\boldsymbol{p}}|\boldsymbol{p} \sim \text{Dirichlet}(\mathbf{1} + c\boldsymbol{p})$$
with $\mathbf{1} = (1, \dots, 1)$ so $\tilde{\boldsymbol{p}}$ has mode $\boldsymbol{p}$.
- For transition matrix P with zeros in off-diagonal entries and $p \in \{p_{j,k} : 1 \le j \le m,\ 1 \le k \le m,\ j \ne k\}$, let $g : \mathbb{R} \to \mathbb{R}^+$ be a standard logistic function $g(x) = e^x/(1 + e^x)$ and propose moves from p to $\tilde{p}$ by
$$\tilde{p} = g(g^{-1}(p) + Z)$$
with $Z \sim$ Normal$(0, 1)$ so $\tilde{p}$ has a LogitNormal distribution. Denote $\boldsymbol{w}$ as the vector $\boldsymbol{p}$ with entry value p changed to $\tilde{p}$. Finally we do a normalization
$$\tilde{\boldsymbol{p}} = \boldsymbol{w} / \sum_{k=1}^{m} w.$$
The transformation $f(x_1, \dots, x_m) = \left(\frac{x_1}{\sum_{i=1}^m x_i}, \dots, \frac{x_{m-1}}{\sum_{i=1}^m x_i}, \sum_{i=1}^m x_i\right)$ is known to have Jacobian $\left(\sum_{i=1}^m x_i\right)^{-m+1}$. So the corresponding transition kernel is given by

$q(\tilde{\boldsymbol{p}}\,|\,\boldsymbol{p}) \propto \exp\left(-0.5(\operatorname{logit}(p_k)-\operatorname{logit}(\tilde{p}))^2\right)\frac{\left(\tilde{p}+\sum_{i\neq k}p_i\right)^{m-1}}{\tilde{p}(1-\tilde{p})}$. Note that we do not marginalize the random variable $S=\sum_{i=1}^{m} w_i$ so we have to draw simulations of the auxiliary variable $\tilde{p}$ to evaluate the above conditional density.

Appendix B: Second simulation study details

For a generator matrix

$$G=\begin{pmatrix}-\lambda_1 & \lambda_1\\ \lambda_2 & -\lambda_2\end{pmatrix}$$

with $\lambda_1, \lambda_2 > 0$, we have explicitly that

$$Q(t)=\exp(tG)=\begin{pmatrix}\frac{\lambda_2}{\lambda_1+\lambda_2}+\frac{\lambda_1}{\lambda_1+\lambda_2}e^{-(\lambda_1+\lambda_2)t} & \frac{\lambda_1}{\lambda_1+\lambda_2}-\frac{\lambda_1}{\lambda_1+\lambda_2}e^{-(\lambda_1+\lambda_2)t}\\ \frac{\lambda_2}{\lambda_1+\lambda_2}-\frac{\lambda_2}{\lambda_1+\lambda_2}e^{-(\lambda_1+\lambda_2)t} & \frac{\lambda_1}{\lambda_1+\lambda_2}+\frac{\lambda_2}{\lambda_1+\lambda_2}e^{-(\lambda_1+\lambda_2)t}\end{pmatrix}.$$

For discrete observation over a time mesh Δk, $\Delta \in \mathbb{R}^+$, and $k \in \{1,\ldots,n\}$ for some $n \in \mathbb{N}$ of the Markov chain associated G; let n_i^{eq} be the number of times the state remains equal for transitions starting in state i and n_i^{ch} the number of times the state changes for transitions starting in state i along the time mesh, with $i \in \{1,2\}$. The likelihood is readily seen to be

$$\begin{aligned}\mathcal{L}(\lambda)=&\left(\frac{\lambda_2}{\lambda_1+\lambda_2}+\frac{\lambda_1}{\lambda_1+\lambda_2}e^{-(\lambda_1+\lambda_2)\Delta}\right)^{n_1^{eq}}\left(\frac{\lambda_1}{\lambda_1+\lambda_2}+\frac{\lambda_2}{\lambda_1+\lambda_2}e^{-(\lambda_1+\lambda_2)\Delta}\right)^{n_2^{eq}}\\ &\times\left(\frac{\lambda_1}{\lambda_1+\lambda_2}-\frac{\lambda_1}{\lambda_1+\lambda_2}e^{-(\lambda_1+\lambda_2)\Delta}\right)^{n_1^{ch}}\left(\frac{\lambda_2}{\lambda_1+\lambda_2}-\frac{\lambda_2}{\lambda_1+\lambda_2}e^{-(\lambda_1+\lambda_2)\Delta}\right)^{n_2^{ch}}.\end{aligned}$$

Simulations for the above CTMC with $(\lambda_1,\lambda_2)=(1,1)$ and $(\lambda_1,\lambda_2=(2,1)$ were drawn. The prior distributions were $\lambda \sim \text{Gamma}(1,1)$ and $\left\{p_{j,k}\right\}_{j\neq k} \sim \text{Dirichlet}(1)$ for all the experiments. A CTMC was generated until 1000 transitions were obtained and we considered the discrete observations given by times Δk with $\Delta = 1$ and $1 \leq k \leq \min\{n\,;\,X(n\Delta) \text{ was fully observed}\}$.

In Figs. 6 and 7 we compare the true posterior distributions given the simulated data with the Gibbs sampler 1 realizations of λ_1 and λ_2; whereas in Figures 8 and 9 we do the same experiment for the Gibbs sampler 2 and in 10, 11 for the Metropolis–Hastings sampler. The choice of priors was taken as indicated above for all the three samplers.

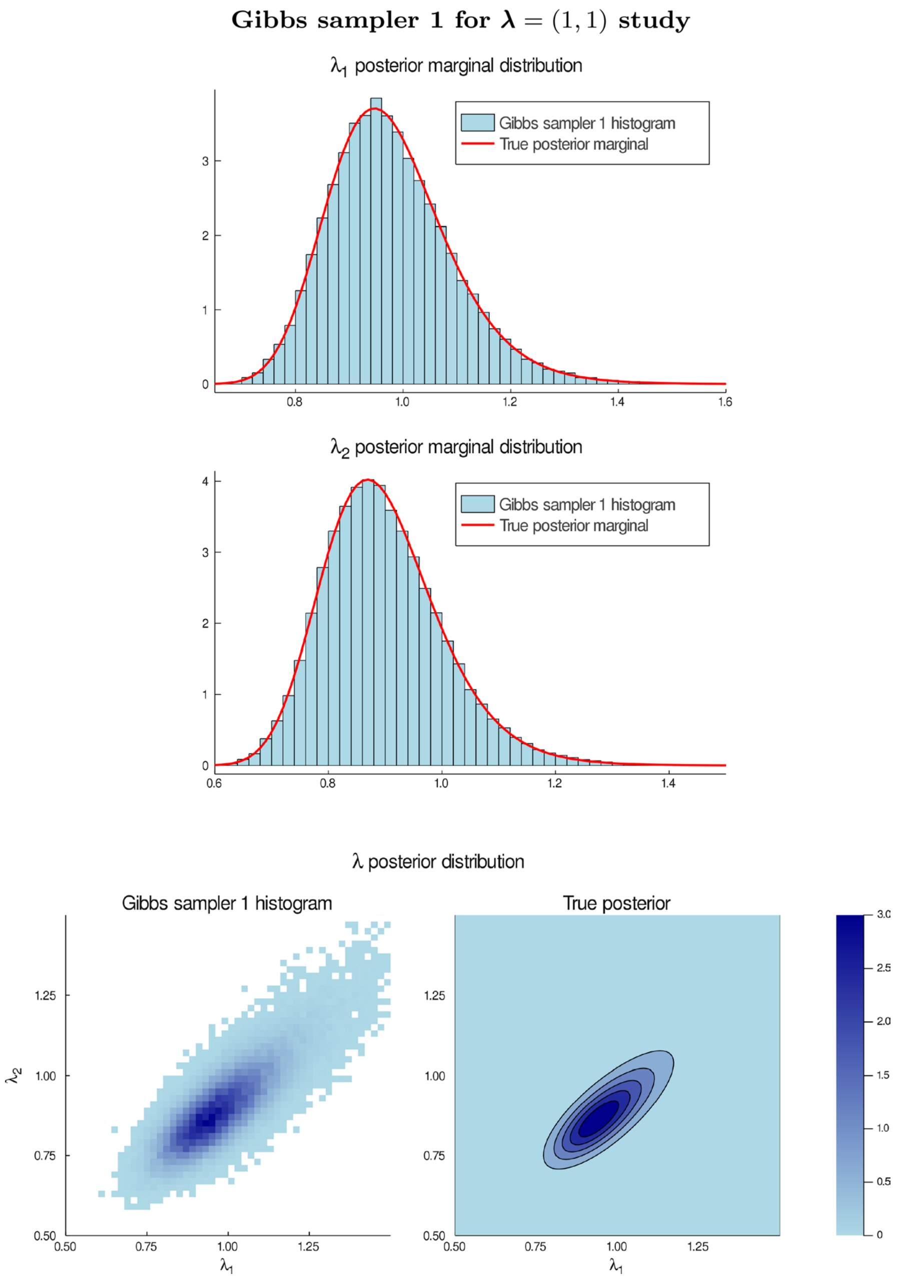


Fig. 6 Marginal and bivariate histograms of $\boldsymbol{\lambda}$ for the Gibbs sampler 1 ran for 50000 iterations after diagnosing stationarity compared with the the true posterior distribution. corresponding to $\boldsymbol{\lambda} = (1, 1)$

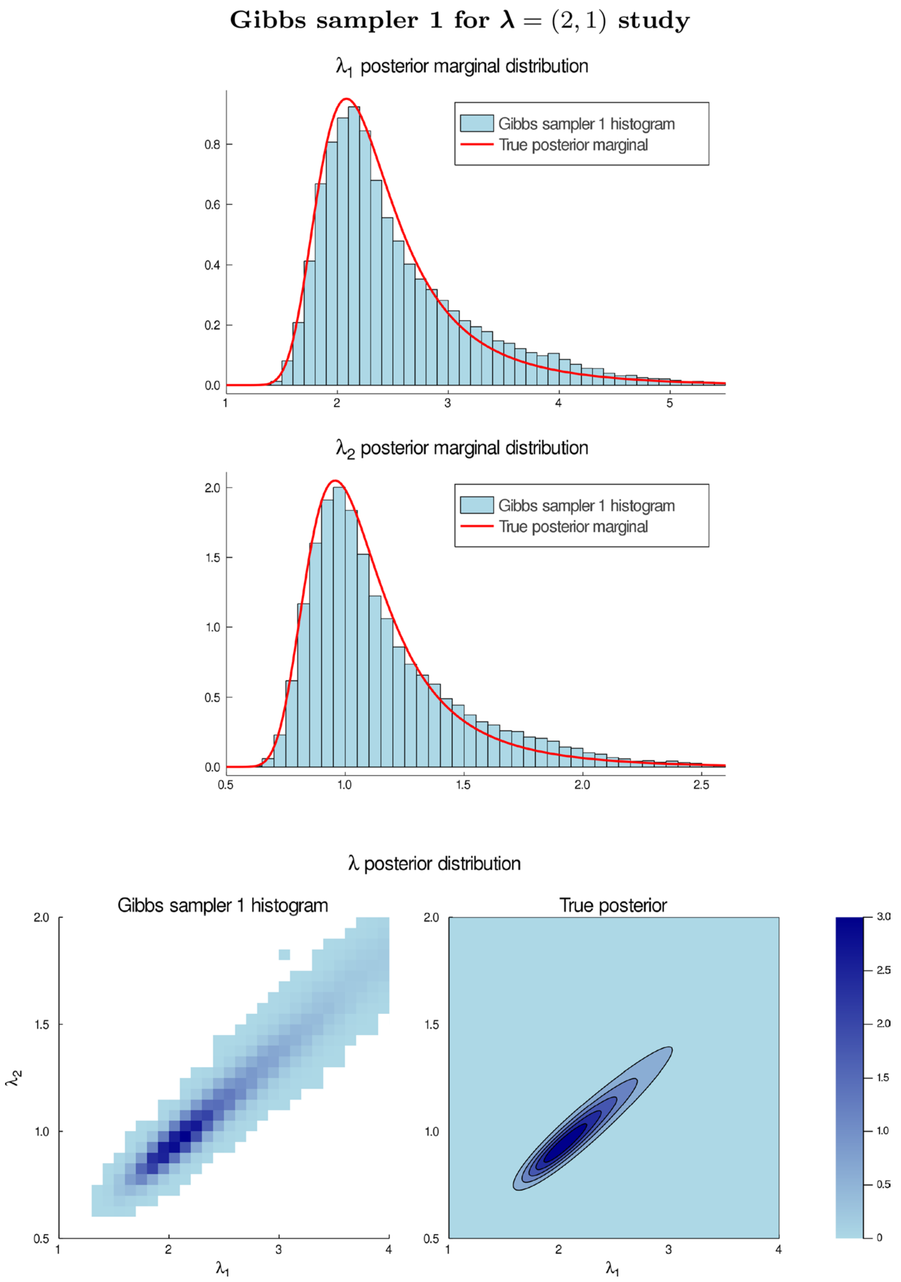


Fig. 7 Marginal and bivariate histograms of $\boldsymbol{\lambda}$ for the Gibbs sampler 1 ran for 50000 iterations after diagnosing stationarity compared with the the true posterior distribution. corresponding to $\boldsymbol{\lambda} = (2, 1)$

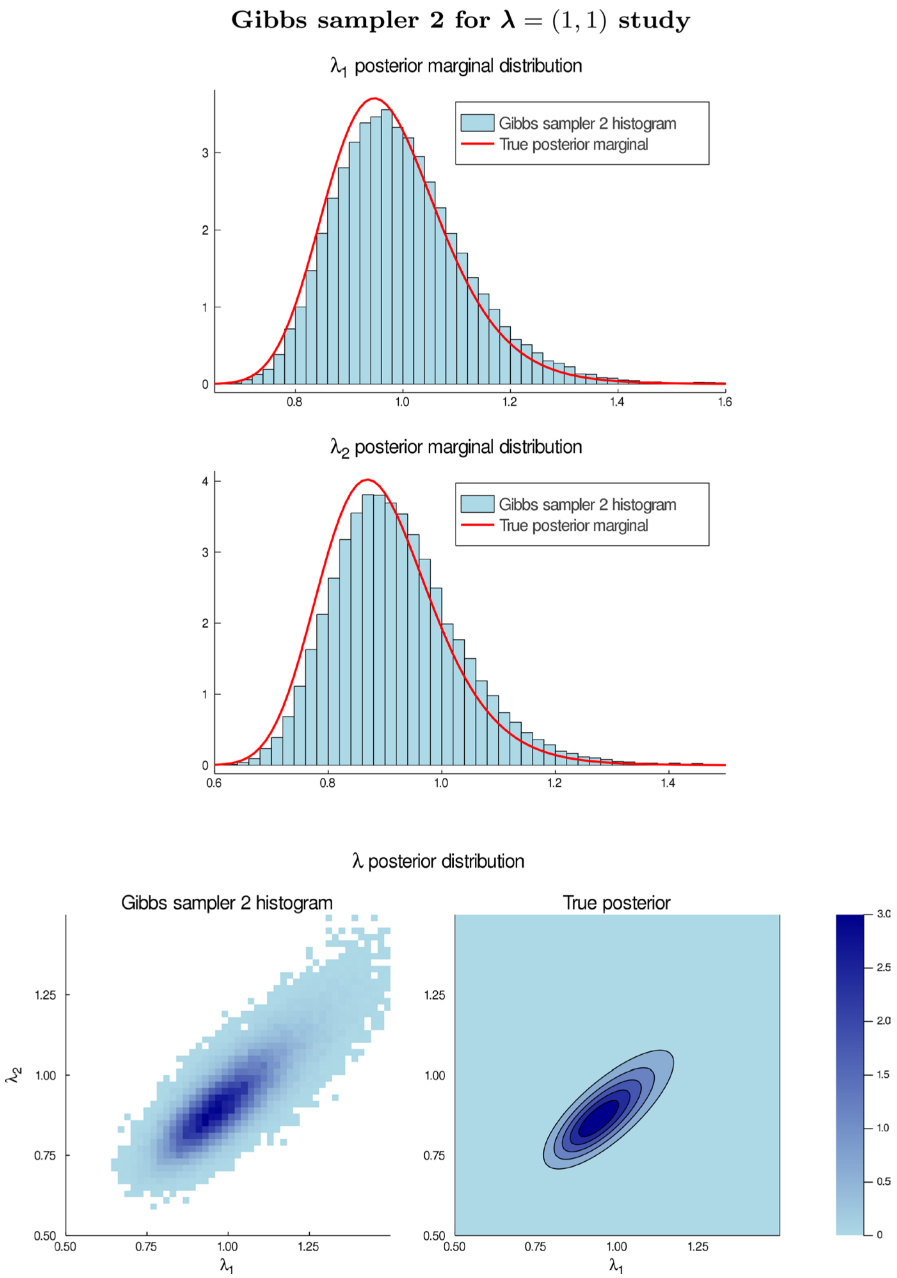


Fig. 8 Marginal and bivariate histograms of $\boldsymbol{\lambda}$ for the Gibbs sampler 2 ran for 50000 iterations after diagnosing stationarity compared with the the true posterior distribution. corresponding to $\boldsymbol{\lambda} = (1, 1)$

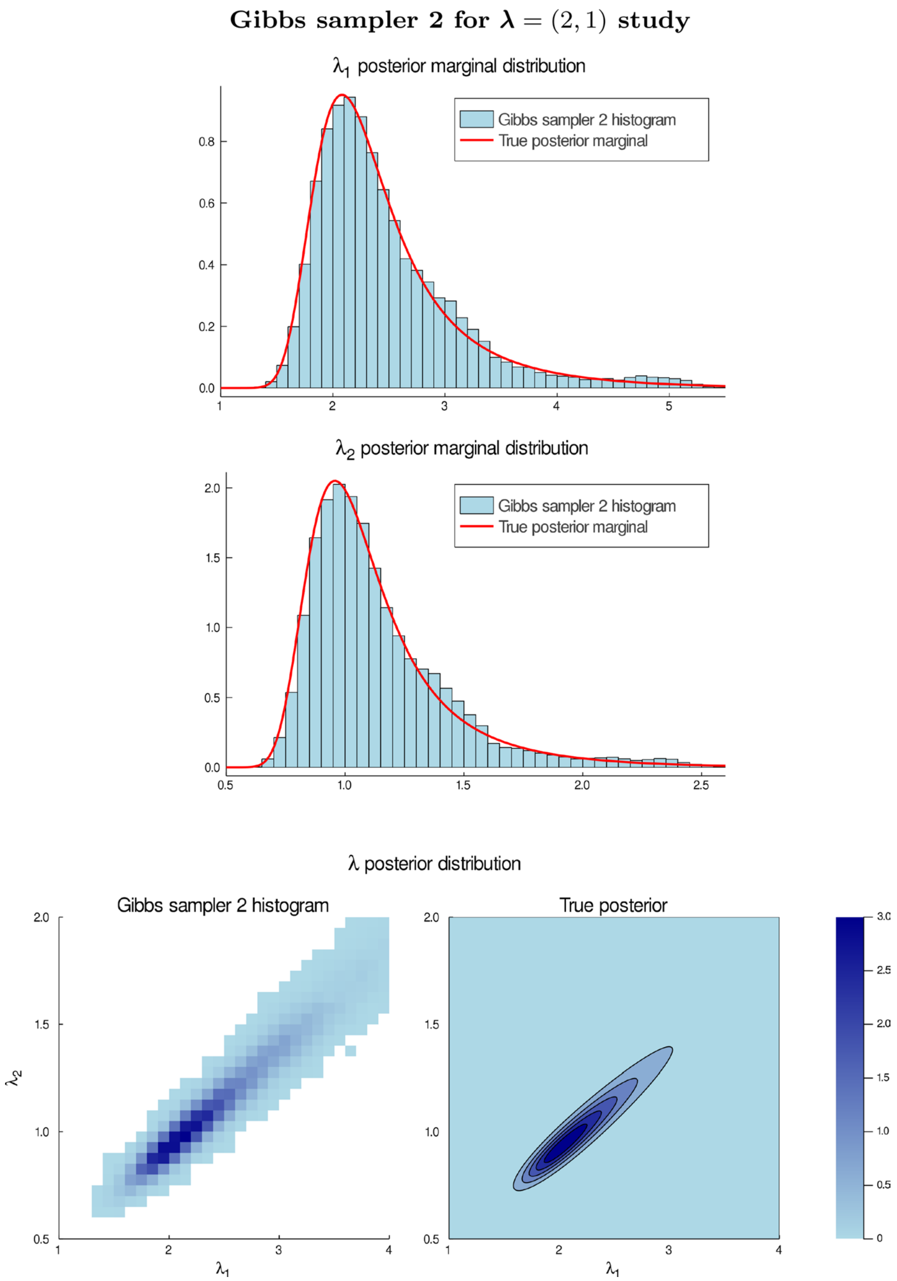


Fig. 9 Marginal and bivariate histograms of $\boldsymbol{\lambda}$ for the Gibbs sampler 2 ran for 50000 iterations after diagnosing stationarity compared with the the true posterior distribution. corresponding to $\boldsymbol{\lambda} = (2, 1)$

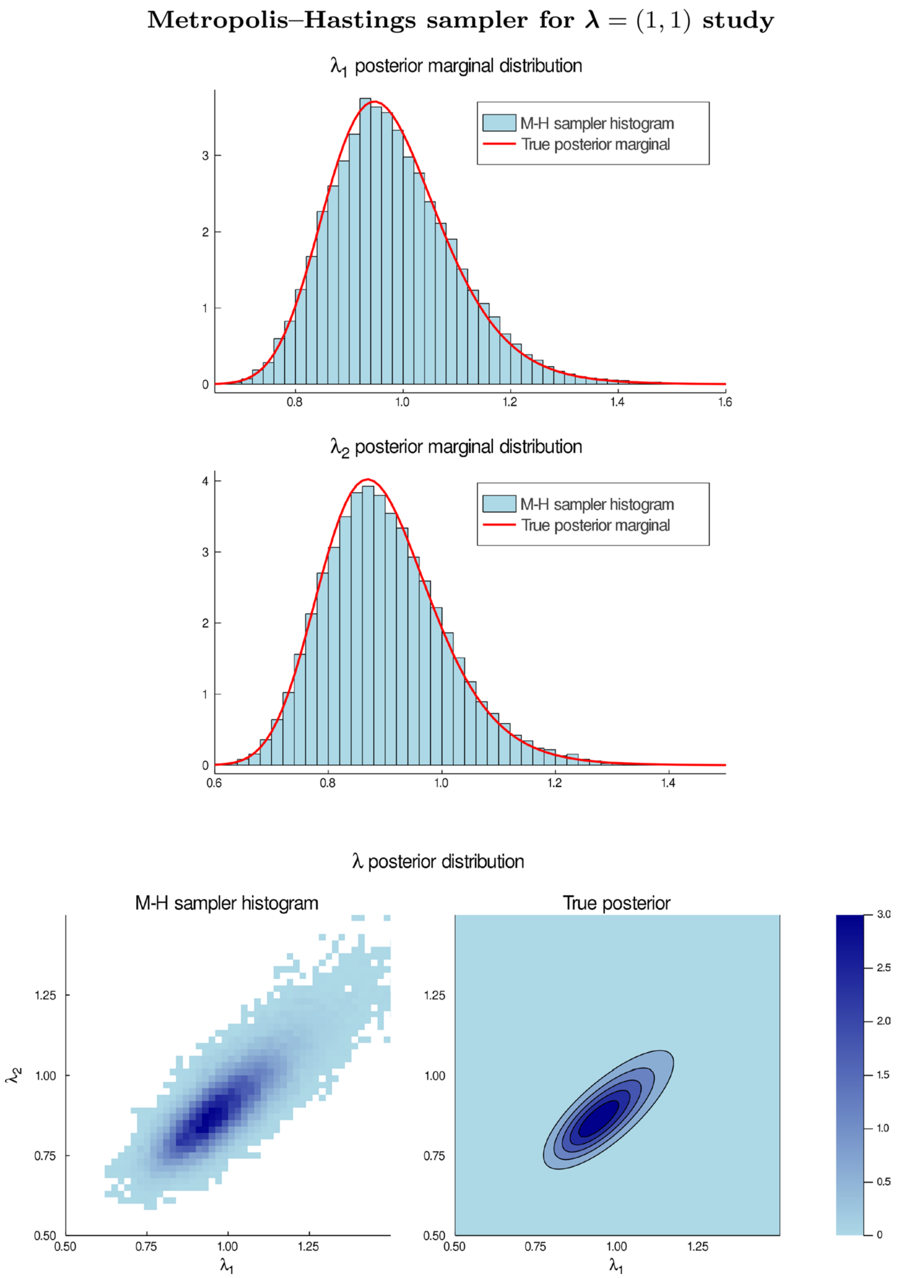


Fig. 10 Marginal and bivariate histograms of $\boldsymbol{\lambda}$ for the Metropolis–Hastings sampler ran for 50000 iterations after diagnosing stationarity compared with the the true posterior distribution. corresponding to $\boldsymbol{\lambda} = (1, 1)$

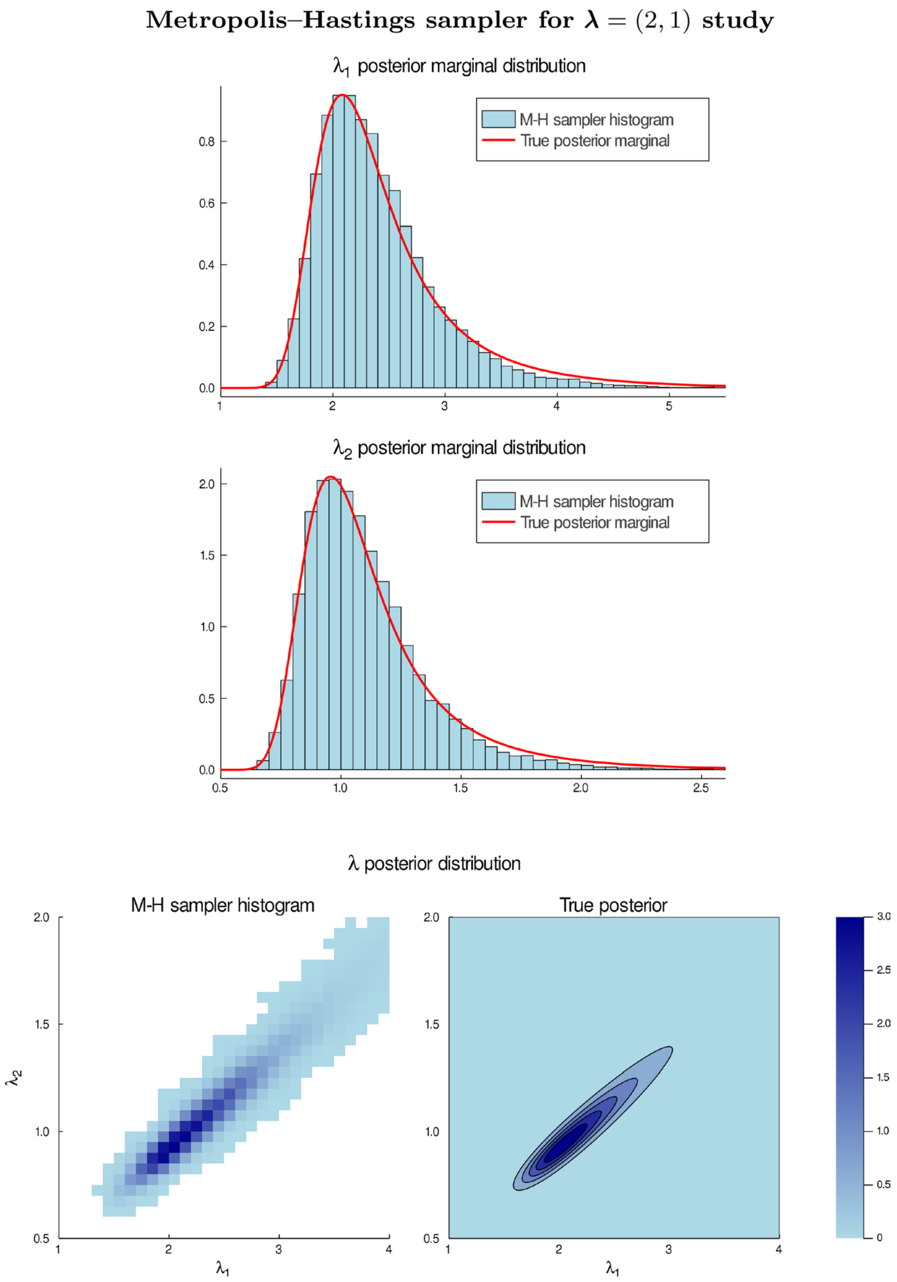


Fig. 11 Marginal and bivariate histograms of $\boldsymbol{\lambda}$ for the Metropolis–Hastings sampler ran for 50,000 iterations after diagnosing stationarity compared with the the true posterior distribution. corresponding to $\boldsymbol{\lambda} = (2, 1)$

The variance in the LogNormal proposals for λ_i in the Metropolis–Hastings were set to 0.0025. We ran 50000 iterations of each sampler after having diagnosed stationarity via the potential scale reduction factor of Gelman and Rubin (1992). We highlight that the run time for the Metropolis–Hastings algorithm is significantly faster than the Gibbs samplers as showed in Tables 1 and 2. This is due to the computational cost of simulating the conditioned trajectories between discretely observed times for the CTMC.

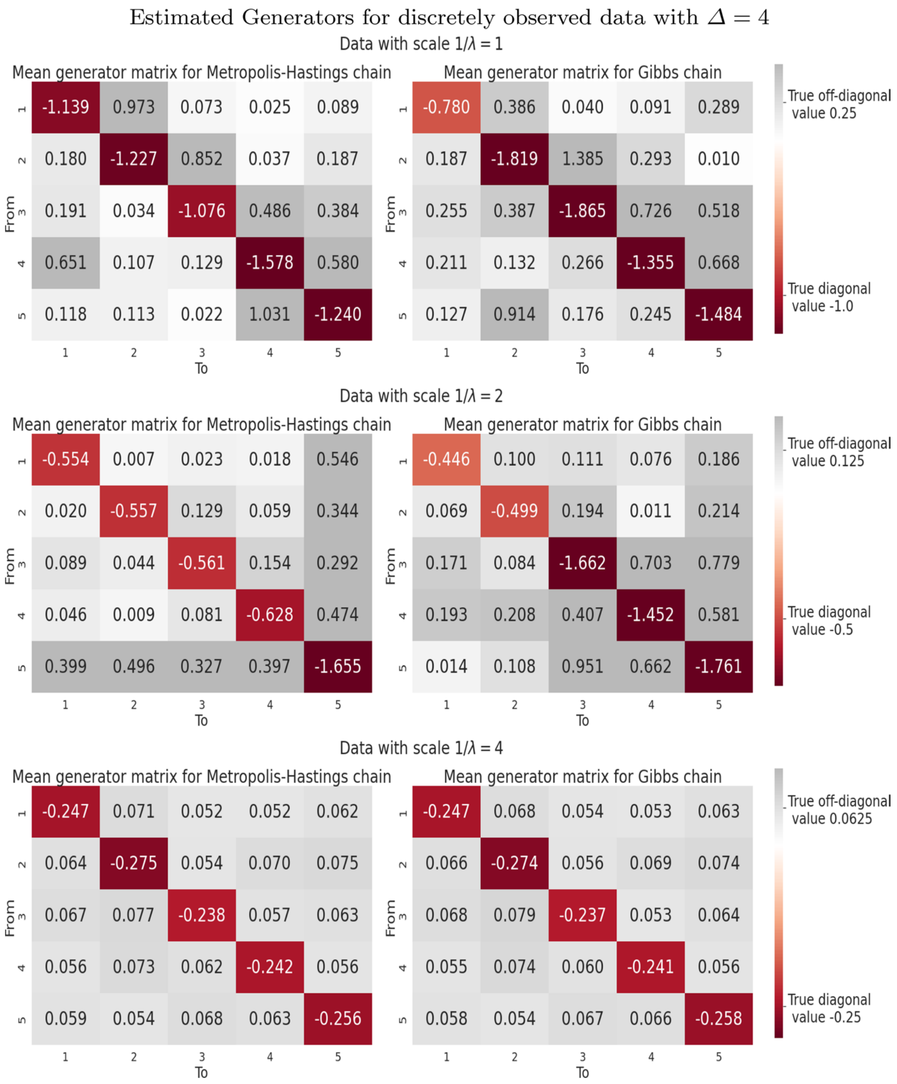


Fig. 12 Comparison of estimated generator matrices for Metropolis–Hastings and Gibbs chains for $\Delta = 4$ with $\lambda = 0.25$ (first row), $\lambda = 0.5$ (second row) and $\lambda = 1$ (third row)

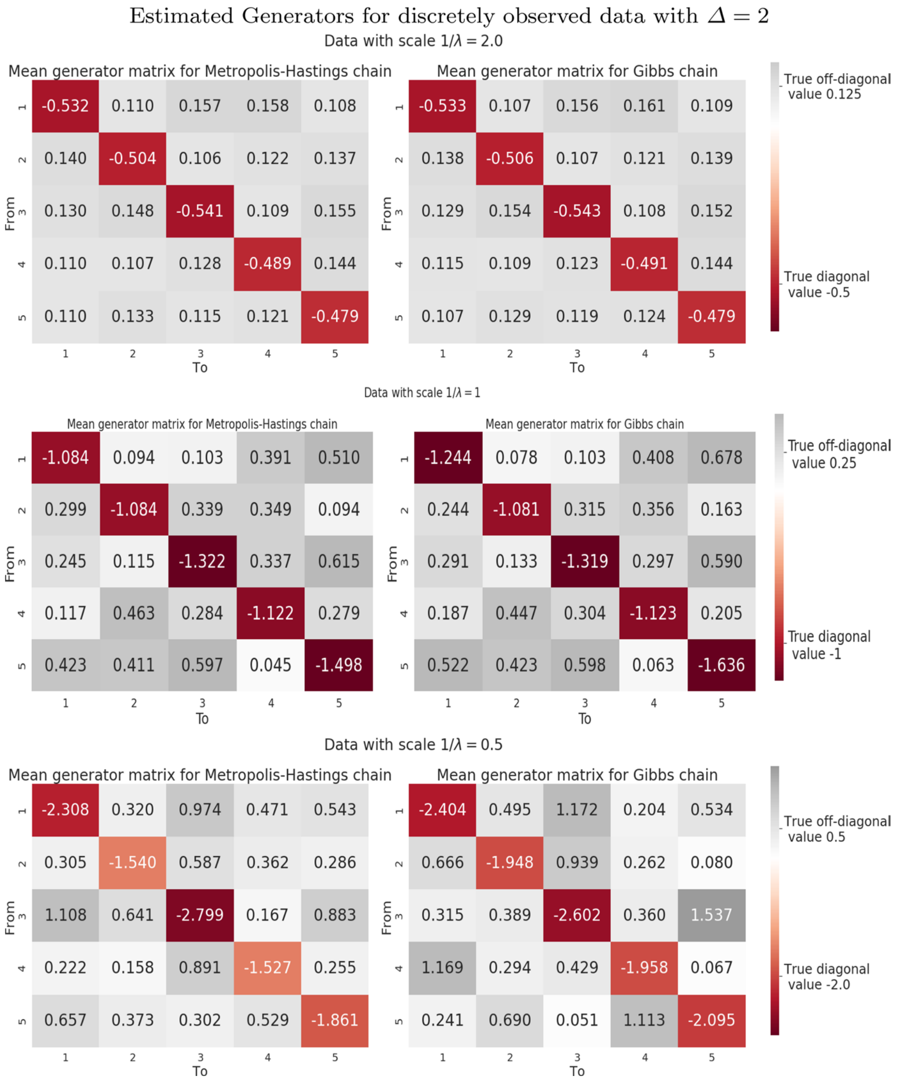
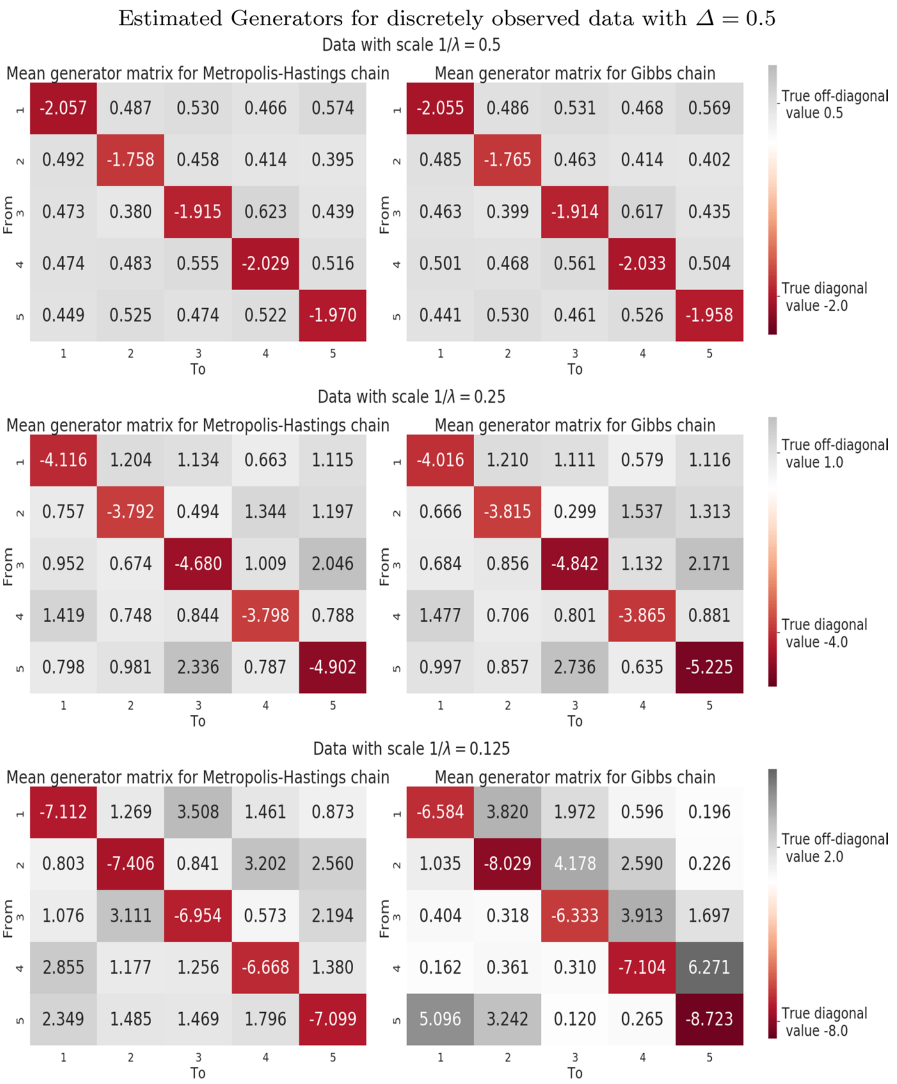


Fig. 13 Comparison of estimated generator matrices for Metropolis–Hastings and Gibbs chains for $\Delta = 2$ with $\lambda = 2$ (first row), $\lambda = 1$ (second row) and $\lambda = 0.5$ (third row)

Appendix C: Further comparisons for the third simulation study

In this appendix we present further mean generator comparisons between the Gibbs and Metropolis–Hastings samplers for discretely observed CTMCs. Also we present comparisons of generators fitted with the EM algorithm and our Metropolis–Hastings approach.


Estimated Generators for discretely observed data with $\Delta = 0.5$

Data with scale $1/\lambda = 0.5$

Mean generator matrix for Metropolis-Hastings chain

	1	2	3	4	5
1	-2.057	0.487	0.530	0.466	0.574
2	0.492	-1.758	0.458	0.414	0.395
3	0.473	0.380	-1.915	0.623	0.439
4	0.474	0.483	0.555	-2.029	0.516
5	0.449	0.525	0.474	0.522	-1.970

Mean generator matrix for Gibbs chain

	1	2	3	4	5
1	-2.055	0.486	0.531	0.468	0.569
2	0.485	-1.765	0.463	0.414	0.402
3	0.463	0.399	-1.914	0.617	0.435
4	0.501	0.468	0.561	-2.033	0.504
5	0.441	0.530	0.461	0.526	-1.958

Data with scale $1/\lambda = 0.25$

Mean generator matrix for Metropolis-Hastings chain

	1	2	3	4	5
1	-4.116	1.204	1.134	0.663	1.115
2	0.757	-3.792	0.494	1.344	1.197
3	0.952	0.674	-4.680	1.009	2.046
4	1.419	0.748	0.844	-3.798	0.788
5	0.798	0.981	2.336	0.787	-4.902

Mean generator matrix for Gibbs chain

	1	2	3	4	5
1	-4.016	1.210	1.111	0.579	1.116
2	0.666	-3.815	0.299	1.537	1.313
3	0.684	0.856	-4.842	1.132	2.171
4	1.477	0.706	0.801	-3.865	0.881
5	0.997	0.857	2.736	0.635	-5.225

Data with scale $1/\lambda = 0.125$

Mean generator matrix for Metropolis-Hastings chain

	1	2	3	4	5
1	-7.112	1.269	3.508	1.461	0.873
2	0.803	-7.406	0.841	3.202	2.560
3	1.076	3.111	-6.954	0.573	2.194
4	2.855	1.177	1.256	-6.668	1.380
5	2.349	1.485	1.469	1.796	-7.099

Mean generator matrix for Gibbs chain

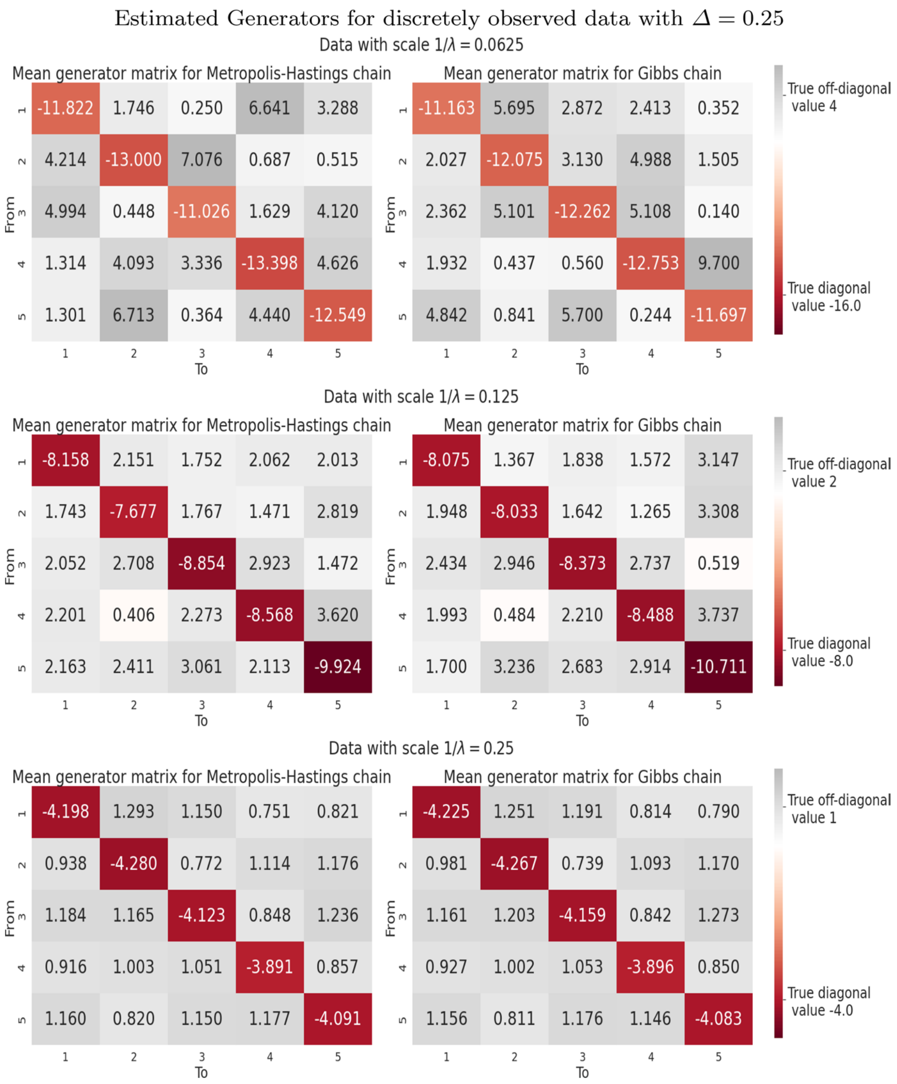
	1	2	3	4	5
1	-6.584	3.820	1.972	0.596	0.196
2	1.035	-8.029	4.178	2.590	0.226
3	0.404	0.318	-6.333	3.913	1.697
4	0.162	0.361	0.310	-7.104	6.271
5	5.096	3.242	0.120	0.265	-8.723

Fig. 14 Comparison of estimated generator matrices for Metropolis–Hastings and Gibbs chains for $\Delta = 0.5$ with $\lambda = 8$ (first row), $\lambda = 4$ (second row) and $\lambda = 2$ (third row)

C.1: Gibbs and Metropolis–Hastings comparisons

In Figs. 13, 15 and we present the mean generator comparisons for the Gibbs and Metropolis–Hastings samplers with simulation studies determined, respectively, by $\Delta = 0.5, 2$ (Figs. 12, 13, 14, 15, 16, 17, 18, 19, 20).

As mentioned in the main document the entry-wise distance of the Gibbs mean generator with the true generator tends to be greater than the one with respect to the Metropolis–Hasting mean generator; this is particularly illustrated in the mean



Data with scale 1/λ = 0.0625 — Mean generator matrix for Metropolis-Hastings chain

From \ To	1	2	3	4	5
1	-11.822	1.746	0.250	6.641	3.288
2	4.214	-13.000	7.076	0.687	0.515
3	4.994	0.448	-11.026	1.629	4.120
4	1.314	4.093	3.336	-13.398	4.626
5	1.301	6.713	0.364	4.440	-12.549

Data with scale 1/λ = 0.0625 — Mean generator matrix for Gibbs chain

From \ To	1	2	3	4	5
1	-11.163	5.695	2.872	2.413	0.352
2	2.027	-12.075	3.130	4.988	1.505
3	2.362	5.101	-12.262	5.108	0.140
4	1.932	0.437	0.560	-12.753	9.700
5	4.842	0.841	5.700	0.244	-11.697

Data with scale 1/λ = 0.125 — Mean generator matrix for Metropolis-Hastings chain

From \ To	1	2	3	4	5
1	-8.158	2.151	1.752	2.062	2.013
2	1.743	-7.677	1.767	1.471	2.819
3	2.052	2.708	-8.854	2.923	1.472
4	2.201	0.406	2.273	-8.568	3.620
5	2.163	2.411	3.061	2.113	-9.924

Data with scale 1/λ = 0.125 — Mean generator matrix for Gibbs chain

From \ To	1	2	3	4	5
1	-8.075	1.367	1.838	1.572	3.147
2	1.948	-8.033	1.642	1.265	3.308
3	2.434	2.946	-8.373	2.737	0.519
4	1.993	0.484	2.210	-8.488	3.737
5	1.700	3.236	2.683	2.914	-10.711

Data with scale 1/λ = 0.25 — Mean generator matrix for Metropolis-Hastings chain

From \ To	1	2	3	4	5
1	-4.198	1.293	1.150	0.751	0.821
2	0.938	-4.280	0.772	1.114	1.176
3	1.184	1.165	-4.123	0.848	1.236
4	0.916	1.003	1.051	-3.891	0.857
5	1.160	0.820	1.150	1.177	-4.091

Data with scale 1/λ = 0.25 — Mean generator matrix for Gibbs chain

From \ To	1	2	3	4	5
1	-4.225	1.251	1.191	0.814	0.790
2	0.981	-4.267	0.739	1.093	1.170
3	1.161	1.203	-4.159	0.842	1.273
4	0.927	1.002	1.053	-3.896	0.850
5	1.156	0.811	1.176	1.146	-4.083

Fig. 15 Comparison of estimated generator matrices for Metropolis–Hastings and Gibbs chains for $\Delta = 0.25$ with $\lambda = 4$ (first row), $\lambda = 8$ (second row) and $\lambda = 16$ (third row)

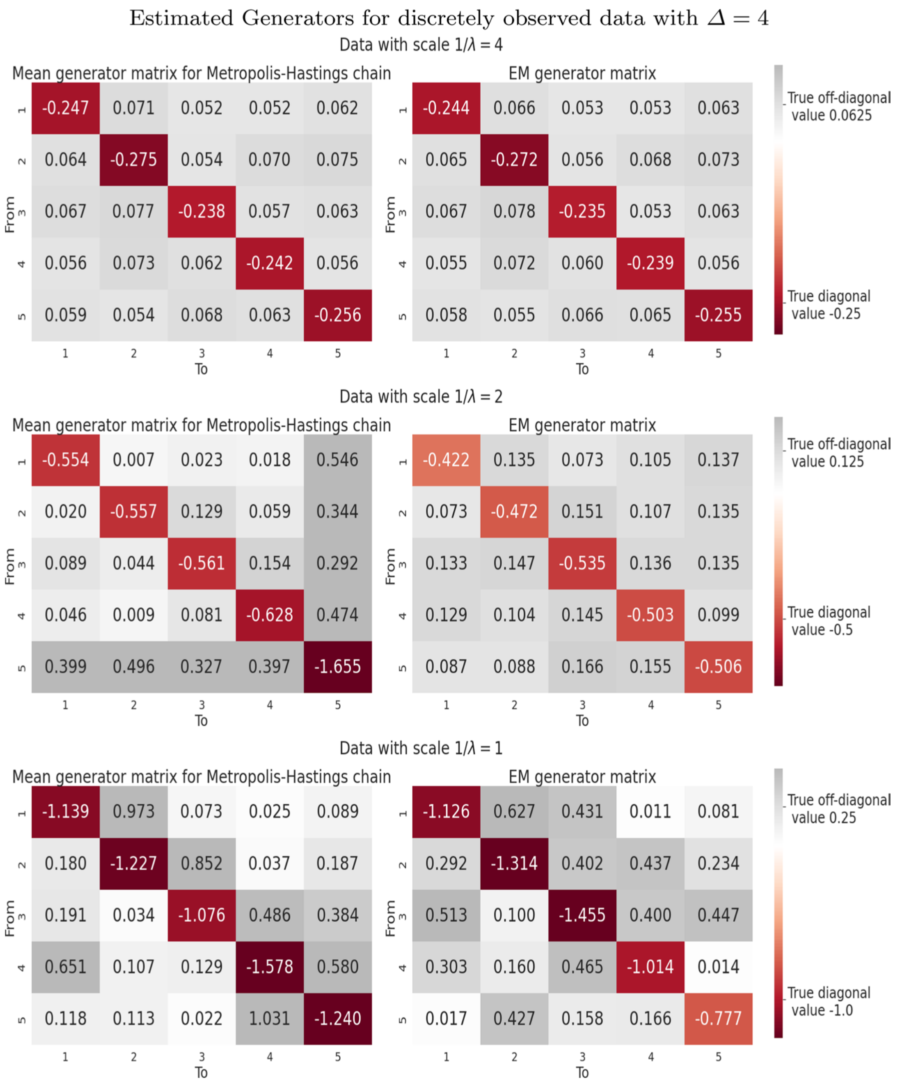
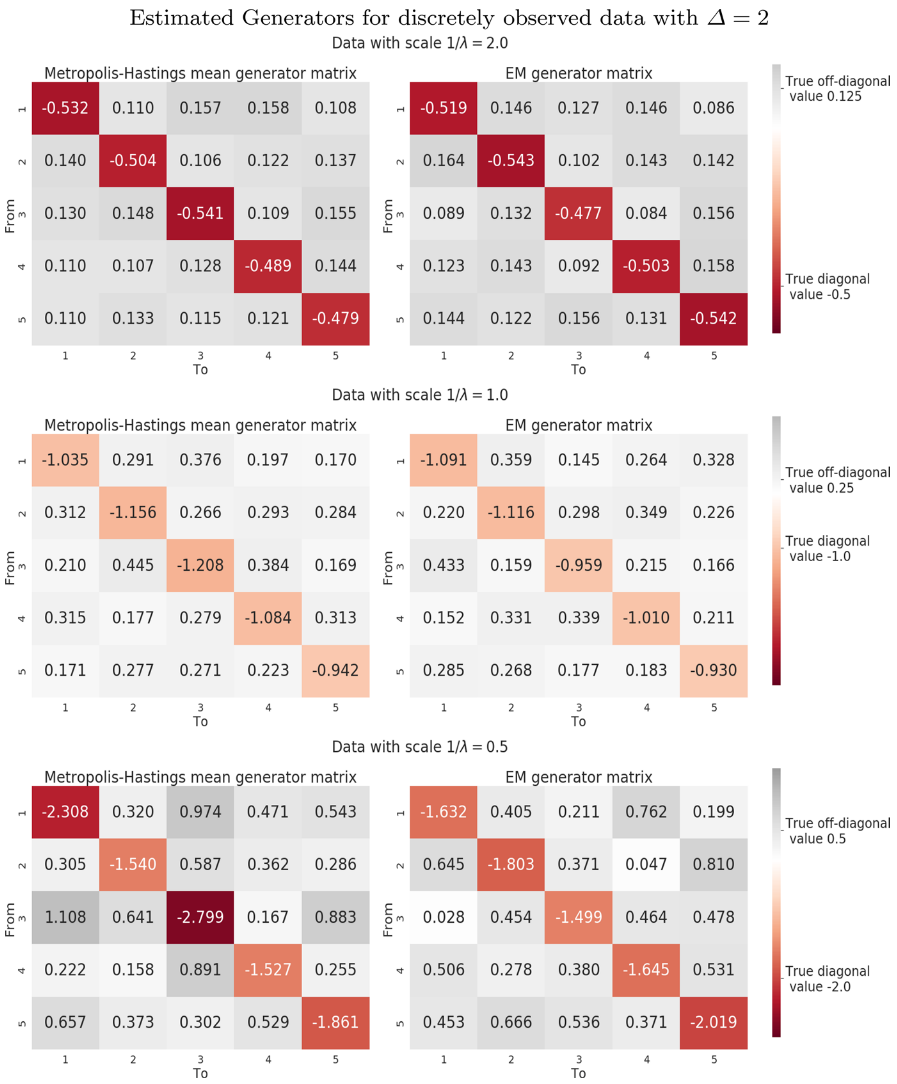


Fig. 16 Comparison of estimated generator matrices for Metropolis–Hastings and EM algorithms for $\Delta = 4$ with $\lambda = 1$ (first row), $\lambda = 0.5$ (second row) and $\lambda = 0.25$ (third row)



Estimated Generators for discretely observed data with $\Delta = 2$

Data with scale $1/\lambda = 2.0$

Metropolis-Hastings mean generator matrix

From \ To	1	2	3	4	5
1	-0.532	0.110	0.157	0.158	0.108
2	0.140	-0.504	0.106	0.122	0.137
3	0.130	0.148	-0.541	0.109	0.155
4	0.110	0.107	0.128	-0.489	0.144
5	0.110	0.133	0.115	0.121	-0.479

EM generator matrix

From \ To	1	2	3	4	5
1	-0.519	0.146	0.127	0.146	0.086
2	0.164	-0.543	0.102	0.143	0.142
3	0.089	0.132	-0.477	0.084	0.156
4	0.123	0.143	0.092	-0.503	0.158
5	0.144	0.122	0.156	0.131	-0.542

Data with scale $1/\lambda = 1.0$

Metropolis-Hastings mean generator matrix

From \ To	1	2	3	4	5
1	-1.035	0.291	0.376	0.197	0.170
2	0.312	-1.156	0.266	0.293	0.284
3	0.210	0.445	-1.208	0.384	0.169
4	0.315	0.177	0.279	-1.084	0.313
5	0.171	0.277	0.271	0.223	-0.942

EM generator matrix

From \ To	1	2	3	4	5
1	-1.091	0.359	0.145	0.264	0.328
2	0.220	-1.116	0.298	0.349	0.226
3	0.433	0.159	-0.959	0.215	0.166
4	0.152	0.331	0.339	-1.010	0.211
5	0.285	0.268	0.177	0.183	-0.930

Data with scale $1/\lambda = 0.5$

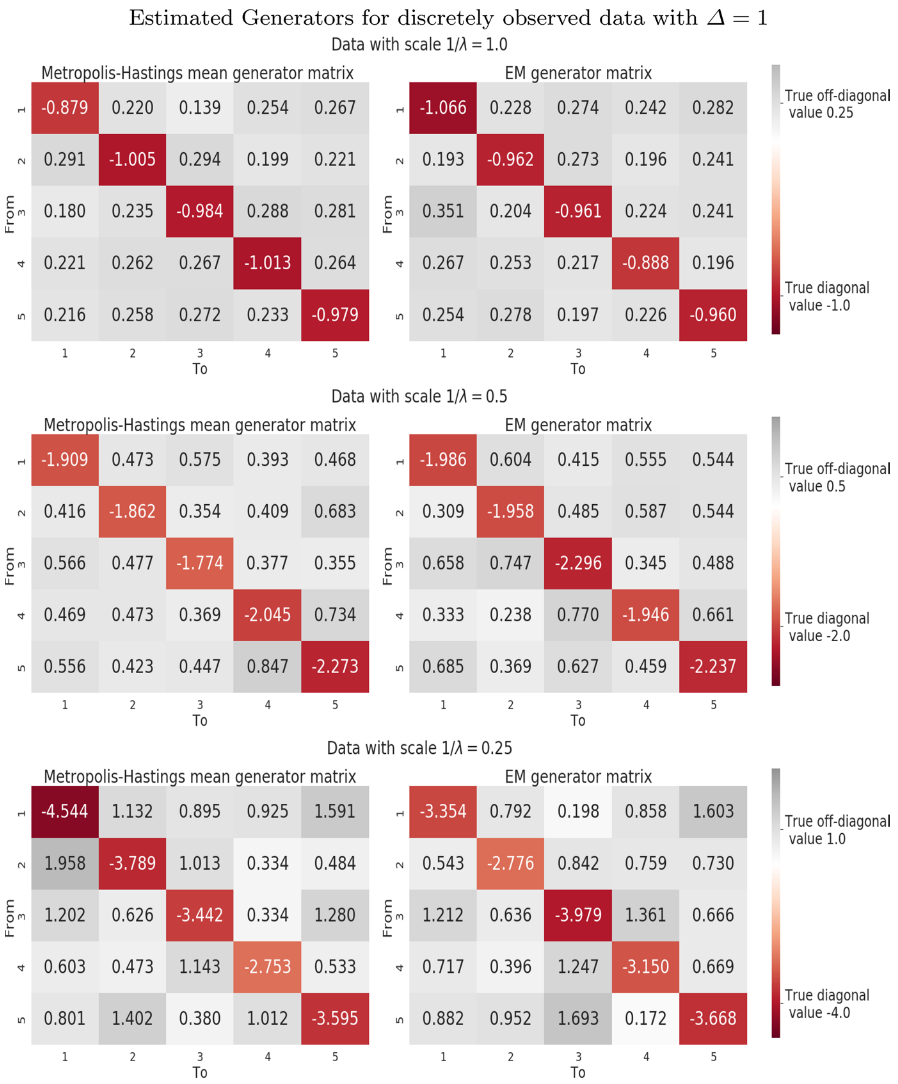
Metropolis-Hastings mean generator matrix

From \ To	1	2	3	4	5
1	-2.308	0.320	0.974	0.471	0.543
2	0.305	-1.540	0.587	0.362	0.286
3	1.108	0.641	-2.799	0.167	0.883
4	0.222	0.158	0.891	-1.527	0.255
5	0.657	0.373	0.302	0.529	-1.861

EM generator matrix

From \ To	1	2	3	4	5
1	-1.632	0.405	0.211	0.762	0.199
2	0.645	-1.803	0.371	0.047	0.810
3	0.028	0.454	-1.499	0.464	0.478
4	0.506	0.278	0.380	-1.645	0.531
5	0.453	0.666	0.536	0.371	-2.019

Fig. 17 Comparison of estimated generator matrices for Metropolis–Hastings and EM algorithms for $\Delta = 2$ with $\lambda = 2$ (first row), $\lambda = 1$ (second row) and $\lambda = 0.5$ (third row)



Data with scale $1/\lambda = 1.0$

Metropolis-Hastings mean generator matrix

From \ To	1	2	3	4	5
1	-0.879	0.220	0.139	0.254	0.267
2	0.291	-1.005	0.294	0.199	0.221
3	0.180	0.235	-0.984	0.288	0.281
4	0.221	0.262	0.267	-1.013	0.264
5	0.216	0.258	0.272	0.233	-0.979

EM generator matrix

From \ To	1	2	3	4	5
1	-1.066	0.228	0.274	0.242	0.282
2	0.193	-0.962	0.273	0.196	0.241
3	0.351	0.204	-0.961	0.224	0.241
4	0.267	0.253	0.217	-0.888	0.196
5	0.254	0.278	0.197	0.226	-0.960

Data with scale $1/\lambda = 0.5$

Metropolis-Hastings mean generator matrix

From \ To	1	2	3	4	5
1	-1.909	0.473	0.575	0.393	0.468
2	0.416	-1.862	0.354	0.409	0.683
3	0.566	0.477	-1.774	0.377	0.355
4	0.469	0.473	0.369	-2.045	0.734
5	0.556	0.423	0.447	0.847	-2.273

EM generator matrix

From \ To	1	2	3	4	5
1	-1.986	0.604	0.415	0.555	0.544
2	0.309	-1.958	0.485	0.587	0.544
3	0.658	0.747	-2.296	0.345	0.488
4	0.333	0.238	0.770	-1.946	0.661
5	0.685	0.369	0.627	0.459	-2.237

Data with scale $1/\lambda = 0.25$

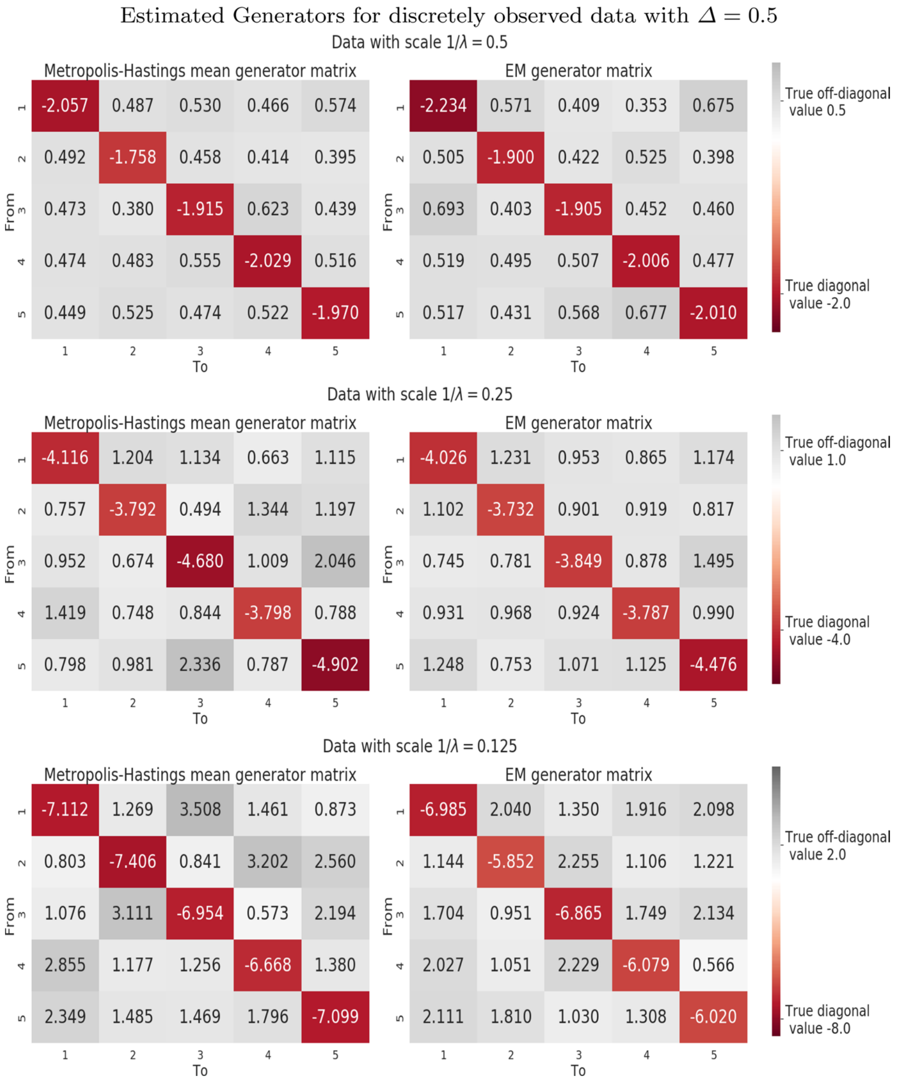
Metropolis-Hastings mean generator matrix

From \ To	1	2	3	4	5
1	-4.544	1.132	0.895	0.925	1.591
2	1.958	-3.789	1.013	0.334	0.484
3	1.202	0.626	-3.442	0.334	1.280
4	0.603	0.473	1.143	-2.753	0.533
5	0.801	1.402	0.380	1.012	-3.595

EM generator matrix

From \ To	1	2	3	4	5
1	-3.354	0.792	0.198	0.858	1.603
2	0.543	-2.776	0.842	0.759	0.730
3	1.212	0.636	-3.979	1.361	0.666
4	0.717	0.396	1.247	-3.150	0.669
5	0.882	0.952	1.693	0.172	-3.668

Fig. 18 Comparison of estimated generator matrices for Metropolis–Hastings and EM algorithms for $\Delta = 1$ with $\lambda = 4$ (first row), $\lambda = 2$ (second row) and $\lambda = 1$ (third row)



Estimated Generators for discretely observed data with $\Delta = 0.5$

Data with scale $1/\lambda = 0.5$

Metropolis-Hastings mean generator matrix

From \ To	1	2	3	4	5
1	-2.057	0.487	0.530	0.466	0.574
2	0.492	-1.758	0.458	0.414	0.395
3	0.473	0.380	-1.915	0.623	0.439
4	0.474	0.483	0.555	-2.029	0.516
5	0.449	0.525	0.474	0.522	-1.970

EM generator matrix

From \ To	1	2	3	4	5
1	-2.234	0.571	0.409	0.353	0.675
2	0.505	-1.900	0.422	0.525	0.398
3	0.693	0.403	-1.905	0.452	0.460
4	0.519	0.495	0.507	-2.006	0.477
5	0.517	0.431	0.568	0.677	-2.010

Data with scale $1/\lambda = 0.25$

Metropolis-Hastings mean generator matrix

From \ To	1	2	3	4	5
1	-4.116	1.204	1.134	0.663	1.115
2	0.757	-3.792	0.494	1.344	1.197
3	0.952	0.674	-4.680	1.009	2.046
4	1.419	0.748	0.844	-3.798	0.788
5	0.798	0.981	2.336	0.787	-4.902

EM generator matrix

From \ To	1	2	3	4	5
1	-4.026	1.231	0.953	0.865	1.174
2	1.102	-3.732	0.901	0.919	0.817
3	0.745	0.781	-3.849	0.878	1.495
4	0.931	0.968	0.924	-3.787	0.990
5	1.248	0.753	1.071	1.125	-4.476

Data with scale $1/\lambda = 0.125$

Metropolis-Hastings mean generator matrix

From \ To	1	2	3	4	5
1	-7.112	1.269	3.508	1.461	0.873
2	0.803	-7.406	0.841	3.202	2.560
3	1.076	3.111	-6.954	0.573	2.194
4	2.855	1.177	1.256	-6.668	1.380
5	2.349	1.485	1.469	1.796	-7.099

EM generator matrix

From \ To	1	2	3	4	5
1	-6.985	2.040	1.350	1.916	2.098
2	1.144	-5.852	2.255	1.106	1.221
3	1.704	0.951	-6.865	1.749	2.134
4	2.027	1.051	2.229	-6.079	0.566
5	2.111	1.810	1.030	1.308	-6.020

Fig. 19 Comparison of estimated generator matrices for Metropolis–Hastings and EM algorithms for $\Delta = 0.5$ with $\lambda = 8$ (first row), $\lambda = 4$ (second row) and $\lambda = 2$ (third row)

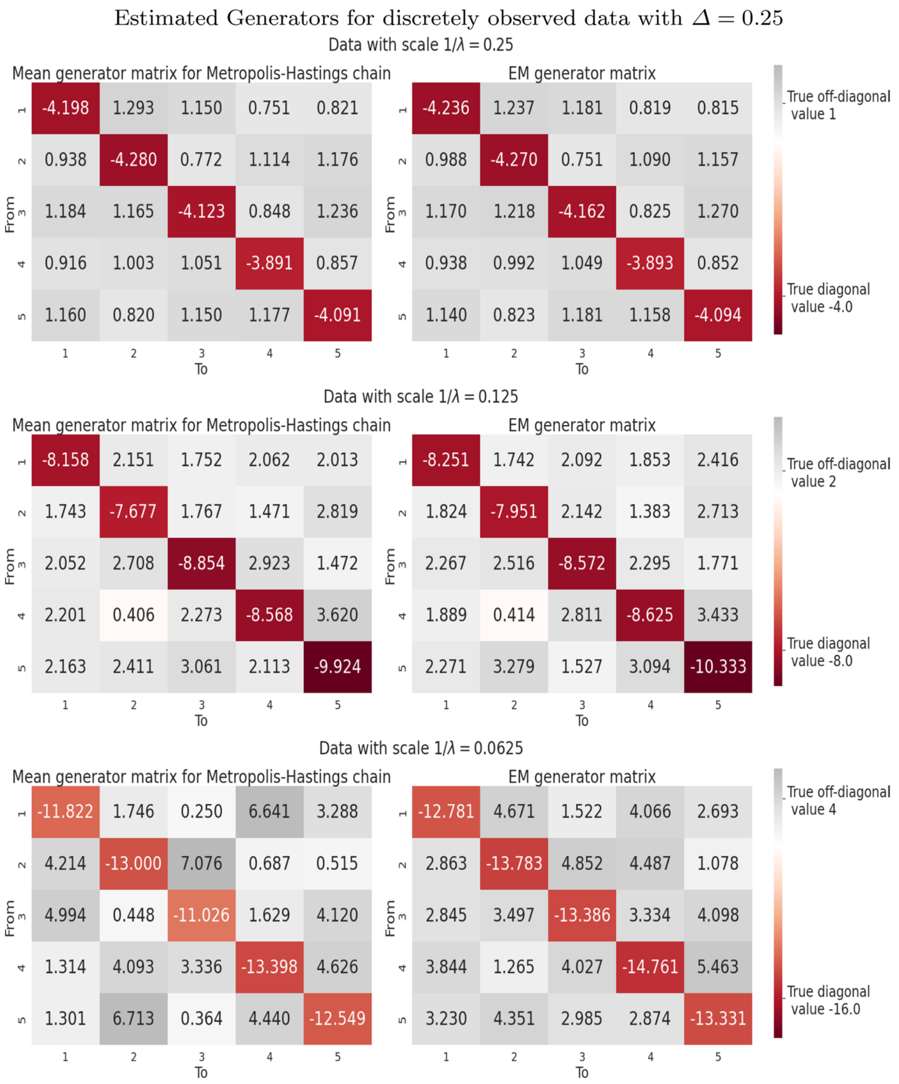


Fig. 20 Comparison of estimated generator matrices for Metropolis–Hastings and EM algorithms for $\Delta = 0.25$ with $\lambda = 16$ (first row), $\lambda = 8$ (second row) and $\lambda = 4$ (third row)

generators for $\Delta = 0.5$ as shown in Table 1 of the main document in terms of Frobenius distances.

C.2: EM and Metropolis–Hastings comparison

We use the EM algorithm of the ctmcd R package, see Pfeuffer (2017), to compare with the proposed Metropolis–Hastings algorithm in the context of the third simulation study.

Acknowledgements The authors are grateful for the support CONTEX project 2018-9 and PAPIIT-UNAM project IG100221. The authors are also grateful for the comments and suggestions of a referee which has greatly improved the paper.

References

Al-Mohy AH, Higham NJ (2011) Computing the action of the matrix exponential, with an application to exponential integrators. SIAM J Sci Comput 33(2):488–511

Amoros R, King R, Toyoda H, Kumada T, Johnson PJ, Bird TG (2019) A continuous-time hidden Markov model for cancer surveillance using serum biomarkers with application to hepatocellular carcinoma. Metron 77(2):67–86

Bladt M, Sorensen M (2005) Statistical inference for discretely observed Markov jump processes. J Roy Stat Soc B 67:395–410

dos Reis G, Smith G (2018) Robust and consistent estimation of generators in credit risk. Quant Financ 18:983–1001

Fearnhead P, Sherlock C (2006) An exact Gibbs sampler for the Markov-modulated Poisson process. J R Stat Soc: Ser B (Stat Methodol) 68:767–784

Fukaya K, Royle JA (2013) Markov models for community dynamics allowing for observation error. Ecology 94:2670–2677

Gelman A, Rubin DB (1992) Inference from iterative simulation using multiple sequences. Stat Sci 7(4):457–472

Georgoulas A, Hillston J, Sanguinetti G (2017) Unbiased Bayesian inference for population Markov jump processes via random truncations. Stat Comput 27(4):991–1002

Gillespie DT (1977) Exact stochastic simulation of coupled chemical reactions. J Phys Chem 81(25):2340–2361

Goulet V, Dutang C, Maechler M, Firth D, Shapira M, Stadelmann M (2021) Package ‘expm’

Grimmett GR, Stirzaker DR (1982) Probability and Random Processes. Oxford University Press

Higham NJ (2005) The scaling and squaring method for the matrix exponential revisited. SIAM J Matrix Anal Appl 26:1179–1193

Inamura Y (2006) Estimating continuous time transition matrices from discretely observed data. Bank of Japan, No.06-E07

Israel RB, Rosenthal JS, Wei JS (2001) Finding generators for Markov chains via empirical transition matrices, with applications to credit ratings. Math Financ 11:245–265

Norris JR (1998) Markov Chains. Cambridge University Press

Pardoux E (2008) Markov processes and applications. Algorithms, networks, genome and finance. Wiley

Pfeuffer M, Möstel L, Fischer M (2019) An extended likelihood framework for modelling discretely observed credit rating transitions. Quant Financ 19:93–104

Pfeuffepdr M (2017) ctmcd: An R Package for Estimating the Parameters of a Continuous-Time Markov Chain from Discrete-Time Data. R J 19:127–141

Plummer M (2003) JAGS: A program for analysis of Bayesian graphical models using Gibbs sampling. In: Proceedings of the 3rd international workshop on distributed statistical computing, vol 124, No. 125.10, pp 1–10

Rao V, Teh YW (2013) Fast MCMC sampling for Markov jump processes and extensions. J Mach Learn Res 14(1):3295–3320

Sherlock C, Fearnhead P, Roberts GO (2010) The random walk Metropolis: linking theory and practice through a case study. Stat Sci 25(2):172–190

Sauer M, Stannat W (2016) Reliability of signal transmission in stochastic nerve axon equations. J Comput Neurosci 40:103–111

Van Kampen NG (2007) Stochastic processes in physics and chemistry. North–Holland

Virtanen P, Gommers R, Oliphant TE, Haberland M, Reddy T, Cournapeau D, van Mulbregt P (2020) SciPy 1.0: fundamental algorithms for scientific computing in Python. Nat Methods 17(3):261-272

Zhao T, Wang Z, Cumberworth A, Gsponer J, de Freitas N, Bouchard-Côté A (2016) Bayesian analysis of continuous time Markov chains with application to phylogenetic modelling. Bayesian Anal 11(4):1203–1237

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.