PLOS ONE

RESEARCH ARTICLE

A comparison of Monte Carlo-based Bayesian parameter estimation methods for stochastic models of genetic networks

Inés P. Mariño[1,2]*, Alexey Zaikin[2,3,4], Joaquín Míguez[5]

1 Departamento de Biología y Geología, Física y Química Inorgánica, Universidad Rey Juan Carlos, Móstoles, Madrid 28933, Spain, **2** Institute for Women's Health, University College London, London WC1E 6BT, United Kingdom, **3** Department of Mathematics, University College London, London WC1E 6BT, United Kingdom, **4** Department of Applied Mathematics, Lobachevsky State University of Nizhny Novgorod, Nizhniy Novgorod, Russia, **5** Departamento de Teoría de la Señal y Comunicaciones, Universidad Carlos III de Madrid, Leganés, Madrid, Spain

* ines.perez@urjc.es

Check for updates

OPEN ACCESS

Citation: Mariño IP, Zaikin A, Míguez J (2017) A comparison of Monte Carlo-based Bayesian parameter estimation methods for stochastic models of genetic networks. PLoS ONE 12(8): e0182015. https://doi.org/10.1371/journal.pone.0182015

Editor: Art F. Y. Poon, Western University, CANADA

Received: October 2, 2016

Accepted: July 11, 2017

Published: August 10, 2017

Copyright: © 2017 Mariño et al. This is an open access article distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

Data Availability Statement: All relevant data are within the paper and its Supporting Information files.

Funding: This research has been partially supported by the Spanish Ministry of Economy and Competitiveness (projects TEC2015-69868-C2-1-R ADVENTURE and FIS2013-40653-P), the Spanish Ministry of Education, Culture and Sport (mobility award PRX15/00378), the Office of Naval Research (ONR) Global (Grant Award no. N62909-15-1-2011), the Cancer Research UK and the Eve Appeal

Abstract

We compare three state-of-the-art Bayesian inference methods for the estimation of the unknown parameters in a stochastic model of a genetic network. In particular, we introduce a stochastic version of the paradigmatic synthetic multicellular clock model proposed by Ullner *et al.*, 2007. By introducing dynamical noise in the model and assuming that the partial observations of the system are contaminated by additive noise, we enable a principled mechanism to represent experimental uncertainties in the synthesis of the multicellular system and pave the way for the design of probabilistic methods for the estimation of any unknowns in the model. Within this setup, we tackle the Bayesian estimation of a subset of the model parameters. Specifically, we compare three Monte Carlo based numerical methods for the approximation of the posterior probability density function of the unknown parameters given a set of partial and noisy observations of the system. The schemes we assess are the particle Metropolis-Hastings (PMH) algorithm, the nonlinear population Monte Carlo (NPMC) method and the approximate Bayesian computation sequential Monte Carlo (ABC-SMC) scheme. We present an extensive numerical simulation study, which shows that while the three techniques can effectively solve the problem there are significant differences both in estimation accuracy and computational efficiency.

Introduction

The field of systems biology is rich in problems that demand sophisticated computational tools for estimation, detection and prediction. As a consequence, we are witnessing the development of a rigorous engineering discipline to create, control and programme cellular behaviour [1]. The resulting branch of research, known as synthetic biology, has undergone a dramatic growth throughout the past decade and is poised to transform biotechnology and medicine. A core issue in synthetic biology is the analysis of networks of interacting biomolecules [2],

Gynaecological Cancer Research Fund (grant ref. A12677) supported by the National Institute for Health Research (NIHR) University College London Hospitals (UCLH) Biomedical Research Centre. AZ thanks support from Russian Science Foundation (16-12-00077).

Competing interests: The authors have declared that no competing interests exist.

which carry out many essential functions in living cells. However, the design principles underlying the functioning of such intracellular networks remain poorly understood. To develop new models and to simplify significantly the associated engineering processes, one needs new inference methods that enable the accurate calibration of potentially complex models by estimating any unknown parameters. It is expected that the ability to design complex synthetic networks will lead both to the engineering of new cellular behaviours and to an improved understanding of naturally occurring networks.

A particular system that has drawn considerable attention is the so-called repressilator [3] which is an oscillating network that periodically induces the synthesis of a green fluorescent protein as a readout of its state in individual cells and can be considered as a synthetic biological clock. Mathematical models, consisting of systems of nonlinear differential equations, that describe the dynamics of the original repressilator and subsequent extensions of it have appeared in the literature [3–9] and sparked interest from researchers in physics, engineering and mathematics.

In this paper we investigate the application of state-of-the-art Bayesian inference methods for the estimation of the unknown parameters in a stochastic model of a genetic network. In particular, we introduce a stochastic version of the chaotic, continuous-time modified repressilator model of [8], which consists of a set of stochastic differential equations (SDEs) driven by Wiener noise processes. These equations depend on a number of unknown parameters, which we model as random variables. We convert the system of SDEs into an (approximate) discrete-time state space model using a standard Euler-Maruyama scheme and then consider the problem of computing the posterior probability distribution of the unknown parameters in the model conditional on a sequence of partial observations that consist of noisy measurements of a small subset of the (dynamic) state variables. This setup resembles the scenario considered in [8] but (i) the system in this paper is stochastic, while in [8] only a deterministic model was studied, and (ii) we pose a data-poor problem, with the collected observations being low dimensional (2-dimensional, for a 14-dimensional state space), noisy and sparse in time, whereas in [8] data were assumed available continuously in time and noise-free. The randomness in the proposed model dynamics can potentially account for experimental uncertainties in the synthesis of the biological system. It also enables the application of probabilistic methods for the calibration of the model and its simulation and forecasting.

Within this probabilistic framework we investigate the Bayesian estimation of the unknown model parameters, i.e., the approximation of their posterior probability distribution conditional on the available observations. In particular, we compare three state-of-the-art Monte Carlo methods for Bayesian inference: the particle Metropolis-Hastings (PMH) method [10], the nonlinear population Monte Carlo (NPMC) algorithm [11] and the approximate Bayesian computation sequential Monte Carlo (ABC-SMC) scheme [8, 12]. PMH is a Markov chain Monte Carlo (MCMC) [13] method that relies on a built-in particle filtering approximation [14] to compute the likelihood of the unknown parameters, which cannot be obtained in closed-form for general nonlinear systems (such as the stochastic repressilator). The algorithm produces a sequence of parameter values that form a Markov chain and the limit probability of this chain is precisely the posterior distribution of the unknown parameters. The NPMC scheme relies on an importance sampling (IS) [15] approximation of the posterior parameter distribution, rather than a Markov chain approach. It employs the same particle filtering approximation of the parameter likelihoods as the PMH method but its key feature is the computation of non-linearly transformed importance weights (unlike conventional IS methods) in order to reduce the variance of the parameter estimates. Preliminary results on the application of the NPMC method to the coupled-repressilator model can be found in the conference communication [16]. Finally, the ABC-SMC algorithm is a likelihood-free procedure that relies on

the computation of distances between observed and synthetic data in order to weight candidate values for the unknown parameters.

We have carried out an extensive numerical comparison of the three methods, taking into account both the accuracy of the parameter estimates and their computational cost. We have considered scenarios where

- the signals are generated from a stochastic coupled repressilator schemes and
- noisy observations are collected from a deterministic (conventional) coupled repressilator system.

While the three schemes effectively solve the problem, the NPMC and PMH algorithm attain a clearly smaller estimation error than the ABC-SMC scheme for the same computational cost. When the computational budget is kept low, the PMH and NPMC algorithms attain very similar estimation errors, with some advantage for the PMH method in the stochastic-signal scenario. As we increase the computational effort, the NPMC algorithm outperforms the PMH scheme in our experiments.

In the sequel, we introduce the stochastic modified repressilator model to be investigated and describe the probabilistic inference methods. Then we present and discuss the results of an extensive set of computer simulations.

Methods

Intercellular network model

Modified stochastic repressilator. The standard repressilator is a "genetic clock" built around three genes, where the protein product of each gene represses the expression of another one in a cyclic manner [3]. It produces nearly harmonic oscillations in protein levels. In the original repressilator design, the gene *lacI* expresses protein LacI, which inhibits transcription of the gene *tetR*. The product of the latter, TetR, inhibits transcription of the gene *cI*. Finally, the protein product CI of the gene *cI* inhibits expression of *lacI* and completes the cycle. We can see this mechanism in the left side of Fig 1, where the genes are represented in light blue colour.

Fig 1 represents a modification of the repressilator, introduced in [5], that includes an additional feedback loop involving a small autoinducer (AI) molecule produced by CI that can diffuse through the cell membrane, and the protein LuxR, which responds to the AI by activating the transcription of a second copy of the repressilator gene *lacI*. Placing the gene *cI* under inhibitory control of the repressilator protein TetR leads to a repressive and phase-repulsive coupling that, in turn, generates rich dynamical patterns, including chaotic oscillations [5]. Phase repulsive coupling is common in many biological systems, e.g., in neural activity, in the brain of songbirds or in the respiratory system.

In this paper we study a model consisting of two modified repressilators with identical parameters, driven by Wiener-type noise and coupled by the fast diffusion of the AI across the cell membranes. The resulting mRNA dynamics in continuous time $t \in \mathbb{R}$ is described by a stochastic Hill-type equation with coefficient m, namely

$$\mathrm{d}a_i = -\left(a_i - \frac{\alpha}{1 + C_i^{\mathrm{m}}}\right)\mathrm{d}t + \sigma_a a_i \mathrm{d}W_i^a, \quad (1)$$

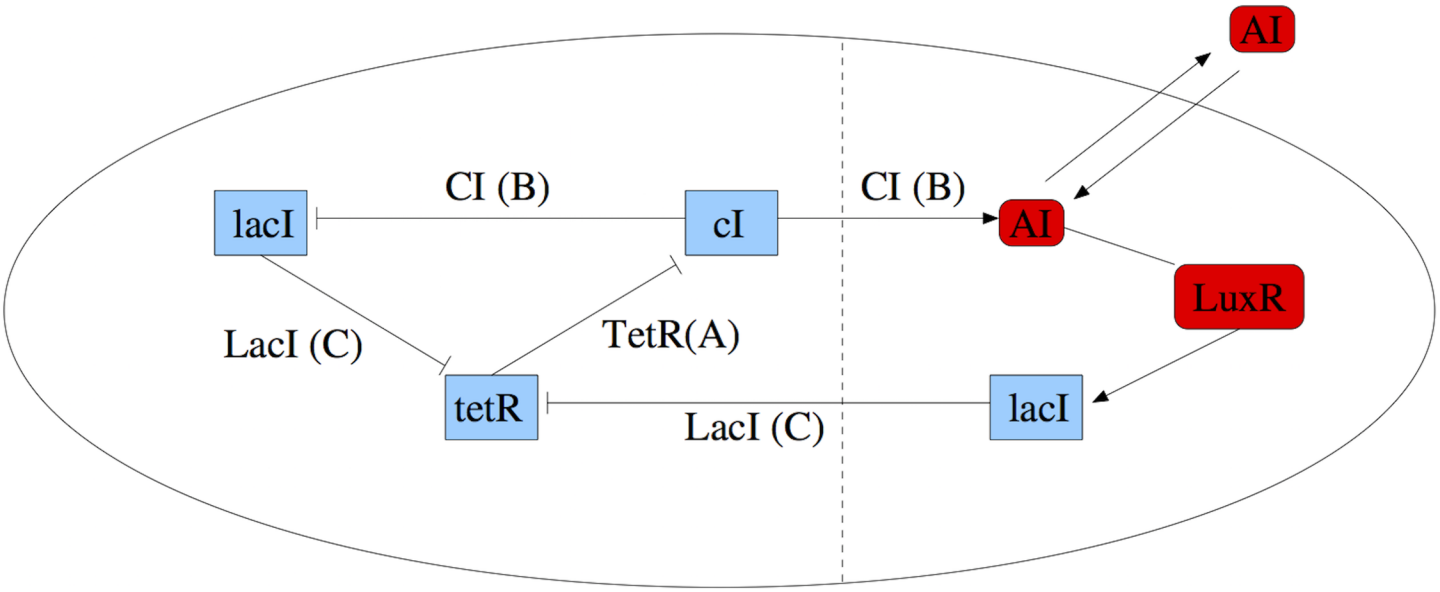

Fig 1. Plot of the modified repressilator. The elements added to the standard repressilator are depicted on the right hand side of the dashed vertical line. Genes and molecules are represented in light blue and red colours, respectively. The oval depicts the cell membrane and the long lines ending in small vertical bars represent the corresponding inhibitory couplings.

https://doi.org/10.1371/journal.pone.0182015.g001

$$\mathrm{d}b_i = -\left(b_i - \frac{\alpha}{1 + A_i^{\mathrm{m}}}\right)\mathrm{d}t + \sigma_b b_i \mathrm{d}W_i^b, \tag{2}$$

$$\mathrm{d}c_i = -\left(c_i - \frac{\alpha}{1 + B_i^{\mathrm{m}}} - \frac{\kappa S_i}{1 + S_i}\right)\mathrm{d}t + \sigma_c c_i \mathrm{d}W_i^c, \tag{3}$$

where the subscript $i = 1, 2$ specifies the cell; a_i, b_i, and c_i are time-varying state variables (stochastic processes) representing the concentrations of mRNA molecules transcribed from the genes of *tetR*, *cI*, and *lacI*, respectively; the constant parameter α is the dimensionless transcription rate in the absence of a repressor; the constant parameter κ is the maximum transcription rate of the LuxR promoter; S_i is a state variable representing the concentration of the AI molecule inside cell i, and W_i^a, W_i^b, W_i^c, $i = 1, 2$, are independent standard Wiener processes scaled by the constant non-negative factors σ_a, σ_b and σ_c, respectively. The additional time-varying states A_i, B_i, and C_i, $i = 1, 2$, are stochastic processes representing the concentration of the proteins TetR, CI, and LacI, respectively, whose dynamics obey the SDEs

$$\mathrm{d}A_i = \beta_a(a_i - A_i)\mathrm{d}t + \sigma_A A_i \mathrm{d}W_i^A, \tag{4}$$

$$\mathrm{d}B_i = \beta_b(b_i - B_i)\mathrm{d}t + \sigma_B B_i \mathrm{d}W_i^B, \tag{5}$$

$$\mathrm{d}C_i = \beta_c(c_i - C_i)\mathrm{d}t + \sigma_C C_i \mathrm{d}W_i^C. \tag{6}$$

The equations above show that the dynamics of the proteins is linked to the amount of the responsible mRNA, and the constant parameters β_a, β_b and β_c describe the ratio between mRNA and the protein lifetimes (i.e, the inverse degradation rates). Similar to Eqs (1)–(3), the

dynamics is driven by independent standard Wiener processes W_i^A, W_i^B and W_i^C, $i = 1, 2$, with constant scale factors $\sigma_A, \sigma_B, \sigma_C \geq 0$. The model is made dimensionless by measuring time in units of the mRNA lifetime (assumed equal for all genes) and the mRNA and protein levels in units of their Michaelis constant. The mRNA concentrations are additionally rescaled by the ratio of their protein degradation and translation rates [4, 5].

The term $\frac{\kappa S_i}{1+S_i}$ on the right-hand side of Eq (3) represents activated production of *lacI* by the AI molecule, whose concentration inside cell i is denoted by S_i. The dynamics of CI and LuxI can be considered identical, given that their production is controlled by the same protein (TetR). Hence, the synthesis of the AI is controlled by the concentration B_i of the protein CI. Taking also into account the intracellular degradation of the AI and its diffusion, the dynamics of S_i is modelled as

$$\mathrm{d}S_i = -(k_{s0}S_i - k_{s1}B_i + \eta(S_i - S_e))\mathrm{d}t + \sigma_S S_i \mathrm{d}W_i^S, \tag{7}$$

where k_{s0}, k_{s1} and η are constant parameters, the latter being a diffusion coefficient that depends on the permeability of the membrane to the AI. The variable S_e is the extracellular concentration of the AI molecule. It is common to apply a quasi-steady-state approximation to the dynamics of S_e [4, 5], which leads to $S_e = Q\bar{S} \equiv Q\frac{1}{N}\sum_{i=1}^{N} S_i$, where $Q = \frac{\delta N}{V_{ext}\left(k_{se}+\frac{\delta N}{V_{ext}}\right)}$, $N = 2$ is the number of cells, V_{ext} is the total extracellular volume, k_{se} is the extracellular AI degradation rate, and δ is the product of the membrane permeability and the surface area.

This model can produce a range of dynamic regimes. We achieve an underlying chaotic behaviour for this model when the constant parameters are set as [5]

$$(Q, \mathsf{m}, \alpha, \beta_a, \beta_b, \beta_c, \eta, \kappa, k_{s0}, k_{s1}) = (0.85, 2.6, 216, 0.85, 0.1, 0.1, 2, 25, 1, 0.01). \tag{8}$$

We will refer to these values as standard. Note that we focus on a system with underlying chaotic dynamics because this makes the parameter estimation problem more challenging. Due to the system sensitivity to small variations of the parameters, two choices of parameter sets, one closer than the other to the standard values, may appear equally "incompatible" with a sequence of observations produced by a system endowed with the standard parameters. Methods that work well in the chaotic regime can be expected to work well in other scenarios.

Numerical integration and state space model. In order to integrate the 14-dimensional SDE described by Eqs (1)–(7) numerically, we apply the Euler-Maruyama discretisation with integration step $\mathsf{h} << 1$, that can be explicitly written as

$$a_{i,m+1} = a_{i,m} - \mathsf{h}\left(\mathsf{a}_{\mathsf{i,m}} - \frac{\alpha}{1+\mathsf{C}_{\mathsf{i,m}}^{\mathsf{m}}}\right) + \sigma_{\mathsf{a}}\mathsf{a}_{\mathsf{i,m}}\mathsf{w}_{\mathsf{i,m}}^{(1)}, \tag{9}$$

$$b_{i,m+1} = b_{i,m} - \mathsf{h}\left(\mathsf{b}_{\mathsf{i}}(\mathsf{n}) - \frac{\alpha}{1+\mathsf{A}_{\mathsf{i,m}}^{\mathsf{m}}}\right) + \sigma_{\mathsf{b}}\mathsf{b}_{\mathsf{i,m}}\mathsf{w}_{\mathsf{i,m}}^{(2)}, \tag{10}$$

$$c_{i,m+1} = c_{i,m} - \mathsf{h}\left(\mathsf{c}_{\mathsf{i,m}} - \frac{\alpha}{1+\mathsf{B}_{\mathsf{i,m}}^{\mathsf{m}}} - \frac{\kappa\mathsf{S}_{\mathsf{i,m}}}{1+\mathsf{S}_{\mathsf{i,m}}}\right) + \sigma_{\mathsf{c}}\mathsf{c}_{\mathsf{i,m}}\mathsf{w}_{\mathsf{i,m}}^{(3)}, \tag{11}$$

$$A_{i,m+1} = A_{i,m} + \mathsf{h}\beta_{\mathsf{a}}(\mathsf{a}_{\mathsf{i,m}} - \mathsf{A}_{\mathsf{i,m}}) + \sigma_{\mathsf{A}}\mathsf{A}_{\mathsf{i,m}}\mathsf{w}_{\mathsf{i,m}}^{(4)}, \tag{12}$$

$$B_{i,m+1} = B_{i,m} + \mathsf{h}\beta_{\mathsf{b}}(\mathsf{b}_{\mathsf{i,m}} - \mathsf{B}_{\mathsf{i,m}}) + \sigma_{\mathsf{B}}\mathsf{B}_{\mathsf{i,m}}\mathsf{w}^{(5)}_{\mathsf{i,m}}, \tag{13}$$

$$C_{i,m+1} = C_{i,m} + \mathsf{h}\beta_{\mathsf{c}}(\mathsf{c}_{\mathsf{i,m}} - \mathsf{C}_{\mathsf{i,m}}) + \sigma_{\mathsf{C}}\mathsf{C}_{\mathsf{i,m}}\mathsf{w}^{(6)}_{\mathsf{i,m}}, \tag{14}$$

$$S_{i,m+1} = S_{i,m} - \mathsf{h}(\mathsf{k}_{\mathsf{s0}}\mathsf{S}_{\mathsf{i,m}} - \mathsf{k}_{\mathsf{s1}}\mathsf{B}_{\mathsf{i}}(\mathsf{n}) + \eta(\mathsf{S}_{\mathsf{i,m}} - \mathsf{S}_{\mathsf{e,m}})) + \sigma_{\mathsf{S}}\mathsf{S}_{\mathsf{i,m}}\mathsf{w}^{(7)}_{\mathsf{i,m}}, \tag{15}$$

where $i = 1, 2$ and $\{w^{(1)}_{i,m}, \ldots, w^{(7)}_{i,m}\}$ are independent Gaussian random variables (r.v.'s) with zero mean and variances $\sigma^2_a, \sigma^2_b, \sigma^2_c, \sigma^2_A, \sigma^2_B, \sigma^2_C$ and σ^2_S, respectively.

The system described by Eqs (9)–(15) can be compactly written as the multidimensional difference equation

$$\bar{\mathbf{x}}_m = F_\theta(\bar{\mathbf{x}}_{m-1}, \mathbf{w}_m), \tag{16}$$

where $F_\theta : \mathbb{R}^{14} \to \mathbb{R}^{14}$ is a function that accounts for both the deterministic and the stochastic part of the model and depends on a vector of unknown parameters θ (modelled as random), $\bar{\mathbf{x}}_m = [\bar{\mathbf{x}}^\top_{1,m}, \bar{\mathbf{x}}^\top_{2,m}]^\top$ is the 14×1 state of the system at discrete time $m \in \mathbb{Z}$, $\bar{\mathbf{x}}_{i,m}$ are the 7×1 state vectors associated to the two cells, $i = 1, 2$, and $\mathbf{w}_m = [\mathbf{w}^\top_{1,m}, \mathbf{w}^\top_{2,m}]^\top$ is an independent and identically distributed (i.i.d.) sequence of 14×1 zero-mean Gaussian vectors. Each 7×1 subvector $\mathbf{w}_{i,m}$, $i = 1, 2$, has the same distribution and can be written as $\mathbf{w}_{i,m} = [w^{(1)}_{i,m}, \ldots, w^{(7)}_{i,m}]^\top$. In particular, note that, for each cell,

$$\bar{\mathbf{x}}_{i,m} = [a_{i,m}, b_{i,m}, c_{i,m}, A_{i,m}, B_{i,m}, C_{i,m}, S_{i,m}]^\top, \tag{17}$$

with the continuous-time state variables evaluated at time $t = m\mathsf{h}$, e.g., $a_{i,m} = a_i(t = m\mathsf{h})$. The same as in [8], all constant parameters are assumed known except $\theta = [Q, \mathsf{m}, \alpha, \beta_a]^\top$, which are unknown and modelled as r.v.'s. We assume uniform and independent a priori probability distributions for each one of these parameters, namely $Q \sim \mathcal{U}(0, 1)$, $\mathsf{m} \sim \mathcal{U}(1, 5)$, $\alpha \sim \mathcal{U}(50, 300)$ and $\beta_a \sim \mathcal{U}(0, 1)$. The parameter vector θ, therefore, takes values on the set $\mathsf{S} = (0, 1) \times (1, 5) \times (50, 300) \times (0, 1)$. We denote the conditional (on θ) Markov kernel that determines the state transition from time $m - 1$ to time m as $\bar{\mathcal{K}}_\theta(\mathsf{d}\mathbf{x}_m|\mathbf{x}_{m-1})$. In particular, for a Borel set $A \subset \mathbb{R}^{14}$, $\bar{\mathcal{K}}_\theta(A|\mathbf{x}_{m-1})$ is the probability of moving from the point $\bar{\mathbf{x}}_{m-1}$ in the state space to some $\bar{\mathbf{x}}_m \in A$.

Partial and noisy observations of the system are collected every m_0 discrete time steps, i.e., every $t_o = m_0\mathsf{h}$ continuous time units. Only the variables a_i, $i = 1, 2$, are observable, hence the observations have the form

$$\mathbf{y}_n = \begin{bmatrix} a_{1,nm_o} \\ a_{2,nm_o} \end{bmatrix} + \sigma_y \boldsymbol{\epsilon}_n, \quad n = 1, 2, \ldots \tag{18}$$

where $\boldsymbol{\epsilon}_n$ is a sequence of independent 2×1 Gaussian random vectors (with zero mean and identity covariance matrix) and $\sigma_y > 0$ is a known constant parameter that determines the noise power.

In order to put the states and the observations on the same time scale, we define the sequence of states $\{\mathbf{x}_n\}_{n \geq 0}$ as $\mathbf{x}_n \triangleq \bar{\mathbf{x}}_{nm_0}$ and introduce the composite Markov kernel

$$\mathcal{K}_\theta(\mathsf{d}\mathbf{x}_n|\mathbf{x}_{n-1}) = \bar{\mathcal{K}}_\theta(\mathsf{d}\mathbf{x}_n|\bar{\mathbf{x}}_{nm_0-1})\bar{\mathcal{K}}_\theta(\mathsf{d}\bar{\mathbf{x}}_{nm_0-1}|\bar{\mathbf{x}}_{nm_0-2}) \cdots \bar{\mathcal{K}}_\theta(\mathsf{d}\bar{\mathbf{x}}_{(n-1)m_0+1}|\mathbf{x}_{n-1}). \tag{19}$$

For a Borel set $A \subset \mathbb{R}^{14}$, $\mathcal{K}_\theta(A|\mathbf{x}_{n-1})$ is the probability of the event $\mathbf{x}_n \in A$ (where $\mathbf{x}_n = \bar{\mathbf{x}}_{nm_0}$) conditional on $\mathbf{x}_{n-1} = \bar{\mathbf{x}}_{(n-1)m_0}$ (and on the parameter vector θ).

The pair of sequences $\mathbf{x}_n$ and $\mathbf{y}_n$ yield a discrete-time, Markov state space model [17] conditional on the choice of parameters θ. The model is specified by the prior probability distribution of the state $\mathbf{x}_0$, which we denote as $\mathcal{K}_0(\mathrm{d}\mathbf{x}_0)$, the dynamics of the state sequence $\mathbf{x}_n$, which is given by the Markov kernel $\mathcal{K}_\theta(\mathrm{d}\mathbf{x}_n|\mathbf{x}_{n-1})$, and the conditional pdf of the observations $\mathbf{y}_n$ given the states $\mathbf{x}_n$, which we denote as $l_n(\mathbf{y}_n|\mathbf{x}_n)$. We note that, in this model, the latter density is independent of the parameters θ. Also, since $\mathbf{y}_n - [a_{1,nm_0}, a_{2,nm_0}]^\top = \boldsymbol{\epsilon}_n$, the form of the function $l_n(\mathbf{y}_n|\mathbf{x}_n)$ is given by the pdf of the noise term $\boldsymbol{\epsilon}_n$. We often refer to $l_n(\mathbf{y}_n|\mathbf{x}_n)$ as the likelihood of the state $\mathbf{x}_n$.

Algorithms

In a Bayesian probabilistic setup, the unknown parameters are modelled as a random vector and the aim is to approximate its posterior probability distribution, conditional on the available observations $\mathbf{y} = \{\mathbf{y}_1, \mathbf{y}_1, \ldots, \mathbf{y}_R\}$ for some fixed $R > 0$. We denote the posterior pdf of the unknown parameters as $p(\theta|\mathbf{y})$ and note that it can be readily factored, using Bayes' theorem, as $p(\theta|\mathbf{y}) \propto \ell(\mathbf{y}|\theta)p_0(\theta)$, where $\ell(\mathbf{y}|\theta)$ is proportional to the conditional pdf of the observations $\mathbf{y}$ given the parameters θ (i.e., the likelihood of θ) and $p_0(\theta)$ is the prior density of θ, which has been chosen to be uniform on the set S, as described in the previous section.

For the case of general state space models, an additional difficulty encountered when trying to estimate the unknown model parameters (denoted θ in our setup) is that the likelihood $\ell(\mathbf{y}|\theta)$ is intractable. In the last few years, though, it has become a common approach to approximate this likelihood via particle filtering (PF) (see, e.g., [10, 11, 18–20]). To be specific, we let $\ell^N(\mathbf{y}|\theta)$ stand for the approximation of $\ell(\mathbf{y}|\theta)$ computed using a standard bootstrap filter (BF) [21, 22] with N particles (see S1 Appendix for full details). One key feature of this approach is that $\ell^N(\mathbf{y}|\theta)$ can be proved to be an unbiased estimator of $\ell(\mathbf{y}|\theta)$ [23].

Particle Metropolis-Hastings (PMH). The PMH is a member of the class of particle MCMC methods [10] that have become very popular in recent years. It can be seen as a conventional Metropolis-Hastings algorithm [15] where the likelihood of each candidate value of θ is approximated via particle filtering and, hence, its acceptance probability is also approximate. Given a Markov kernel $\mathcal{M}(\theta'|\theta)$, the PMH algorithm generates a chain on the space of the parameter θ as follows:

1. Initialisation: Draw $\theta_0 \sim p_0(\theta)$ from the prior distribution of the parameters.
2. At the m-th iteration, and given the previous sample θ_{m-1}:
   a. Draw a tentative new element $\tilde{\theta}_m$ from the kernel $\mathcal{M}(\theta|\theta_{m-1})$.
   b. Compute the (approximate) likelihood $\ell^N(\mathbf{y}|\tilde{\theta}_m)$ and prior density $p_0(\tilde{\theta}_m)$. The acceptance probability for $\tilde{\theta}_m$ is
   $$\alpha_m = \min\left(1, \frac{\ell^N(\mathbf{y}|\tilde{\theta}_m)p_0(\theta_{m-1})M(\tilde{\theta}_m|\theta_{m-1})}{\ell^N(\mathbf{y}|\theta_{m-1})p_0(\tilde{\theta}_m)M(\theta_{m-1}|\tilde{\theta}_m)}\right) \tag{20}$$
   c. Draw $u_m \sim \mathcal{U}(0,1)$. If $u_m < \alpha_m$ then $\theta_m = \tilde{\theta}_m$, else $\theta_m = \theta_{m-1}$.

We use this procedure to generate a chain of length L, $\theta_0, \theta_1, \ldots, \theta_{L-1}$. It can be proved that, as $n \to \infty$, the probability distribution of θ_n converges to the posterior pdf $p(\theta|\mathbf{y})$ [10] despite

the approximation $\ell^N(\mathbf{y}|\theta) \simeq \ell(\mathbf{y}|\theta)$ in the algorithm. Since the chain requires a number of iterations to converge to its limit distribution (this is often referred to in the literature as the "burn in" period), in practice we need to discard some samples at the beginning of the chain. Herein we assume that we use the second half of the chain to estimate θ. In other words, if we aim at approximating the expected value of θ given the observations, often denoted $E[\theta|\mathbf{y}]$, we approximate it as the sample mean of $\theta_{L/2}, \ldots, \theta_{L-1}$, i.e.

$$\hat{\theta}_L = \frac{2}{L}\sum_{i=L/2}^{L-1}\theta_i \tag{21}$$

where we assume, for convenience, that L is even. Note that we can also approximate other posterior statistics of θ (given $\mathbf{y}$). For eample, the posterior covariance of θ can be estimated as

$$\hat{\Sigma}_L = \frac{2}{L}\sum_{i=L/2}^{L-1}(\theta_i - \hat{\theta}_L)(\theta_i - \hat{\theta}_L)^T. \tag{22}$$

Nonlinear population Monte Carlo (NPMC). The NPMC algorithm of [11] is an iterative importance sampling (IS) scheme that seeks to approximate a target probability distribution, in our case given by the pdf $p(\theta|\mathbf{y})$, using weighted Monte Carlo samples. This algorithm generates a sequence of proposal pdf's $q_k(\theta)$, $k = 1, \ldots, K$, from which samples can be drawn and importance weights (IWs) can be computed. This sequence of proposals is expected to yield increasingly better approximations of the target as the algorithm converges. The key feature of the NPMC method, which departs from the classical PMC technique of [24], is to compute a set of *transformed* importance weights (TIWs) by applying a nonlinear function to the standard IWs. The aim of this transformation is to mitigate the well known problem of the degeneracy of the IWs (common to many IS methods, see [11, 25]) by controlling the weight variability. Some basic results regarding the convergence in probability of estimators based on TIWs are given in [11]. A specific analysis for the case of state space models, where the likelihood function $\ell(\mathbf{y}|\theta)$ can only be estimated via particle filtering, can be found in [26].

The NPMC algorithm with K iterations, M Monte Carlo samples per iteration, plain Gaussian proposals $\{q_k\}_{k\geq 1}$, and approximate likelihoods is outlined below. Recall that θ is the vector of unknown model parameters.

Initialisation. Draw M i.i.d. samples $\theta_0^1, \theta_0^2, \ldots, \theta_0^M$ from the prior pdf $p_0(\theta)$. Then,

1. compute non-normalised IWs $\tilde{\mathsf{w}}_0^i \propto \ell^N(\mathbf{y}|\theta_0^i)$, $i = 1, \ldots, M$,
2. compute TIWs as $\hat{\mathsf{w}}_0^i = \mathcal{T}_M(i, \{\tilde{\mathsf{w}}_0^j\}_{j=1}^M)$, where $\mathcal{T}_M : \{1, \ldots, M\} \times \{\tilde{\mathsf{w}}_0^j\}_{j=1}^M \to [0, +\infty)$ is a nonlinear map (to be specifically described below),
3. normalise the TIWs, $\mathsf{w}_0^i = \frac{\hat{\mathsf{w}}_0^i}{\sum_{j=1}^M \hat{\mathsf{w}}_0^j}$, $i = 1, \ldots, M$.

Iterative step. For $k = 1, \ldots, K$, take the following steps:

1. Let $q_k(\theta) = \mathcal{N}(\theta|\mu_k, \Sigma_k)$ be a multivariate Gaussian pdf with mean vector and covariance matrix obtained, respectively, as

$$\mu_k = \sum_{i=1}^M \mathsf{w}_{k-1}^i \theta_{k-1}^i \quad \text{and} \quad \Sigma_k = \sum_{i=1}^M \mathsf{w}_{k-1}^i (\theta_{k-1}^i - \mu_k)(\theta_{k-1}^i - \mu_k)^\top. \tag{23}$$

Note that the random variates θ_{k-1}^i, $i = 1, \ldots, M$, are 4×1 vectors.

2. Draw θ_k^i, $i = 1, \ldots, M$, i.i.d. samples from $q_k(\theta)$.

3. Compute IWs, $\tilde{\mathsf{w}}_k^i = \frac{\ell^N(\mathbf{y}|\theta_k^i)p_0(\theta_k^i)}{q_k(\theta_k^i)}$, $i = 1, \ldots, M$.

4. Compute TIWs, $\hat{\mathsf{w}}_k^i = \mathcal{T}_M(i, \{\tilde{\mathsf{w}}_k^j\}_{j=1}^M)$, $i = 1, \ldots, M$, using the same nonlinear map as for $k = 0$.

5. Normalise the TIWs, $\mathsf{w}_k^i = \frac{\hat{\mathsf{w}}_k^i}{\sum_{j=1}^M \hat{\mathsf{w}}_k^j}$, $i = 1, \ldots, M$.

Following [11], the nonlinear map $\mathcal{T}_M$ of choice is a "clipping" transformation. In particular, let $i_1, i_2, \ldots, i_M$ be a permutation of the indices $1, 2, \ldots, M$ such that the IWs become ordered, namely $\tilde{\mathsf{w}}_k^{i_1} \geq \tilde{\mathsf{w}}_k^{i_2} \geq \cdots \geq \tilde{\mathsf{w}}_k^{i_M}$. The clipping transformation $\mathcal{T}_M$, with parameter $1 \leq M_c \leq \sqrt{M}$, flattens the M_c largest IWs and makes them equal to the M_c-th non-normalised IW, $\tilde{\mathsf{w}}_k^{i_{M_c}}$. Specifically, for each $i = 1, \ldots, M$, we obtain

$$\hat{\mathsf{w}}_k^j = \mathcal{T}_M(j, \{\tilde{\mathsf{w}}_k^l\}_{l=1}^M) = \begin{cases} \tilde{\mathsf{w}}_k^{i_{M_c}}, & \text{if } \tilde{\mathsf{w}}_k^j \geq \tilde{\mathsf{w}}_k^{M_c}, \\ \tilde{\mathsf{w}}_k^j, & \text{if } \tilde{\mathsf{w}}_k^j < \tilde{\mathsf{w}}_k^{M_c}, \end{cases} . \tag{24}$$

Other choices of $\mathcal{T}_M$ are possible (e.g., tempering schemes [11]).

At any iteration k of the algorithm we may use the samples $\{\theta_k^i\}_{i=1}^M$ and the normalised IWs or TIWs for estimating θ or approximating any of its statistics. Hereafter, we assume that the normalised TIWs $\{\mathsf{w}_k^i\}_{i=1}^M$ are employed. Hence, the expected value of θ given the observations $\mathbf{y}$ is estimated (at the k-th iteration) as

$$\hat{\theta}_k^M = \sum_{i=1}^M \mathsf{w}_k^i \theta_k^i \simeq E[\theta|\mathbf{y}] \tag{25}$$

where the superscript M indicates the number of samples used in the estimates. Similarly, the covariance matrix of θ conditional on the data $\mathbf{y}$ can be approximated as

$$\hat{\Sigma}_k^M = \sum_{i=1}^M \mathsf{w}_k^i (\theta_k^i - \hat{\theta}_k^M)(\theta_k^i - \hat{\theta}_k^M)^T. \tag{26}$$

Approximate Bayesian computation sequential Monte Carlo (ABC-SMC). ABC methods have been conceived with the aim of approximating posterior probability distributions, conditional on the available observations, without having to calculate likelihood functions [12, 27]. The computation of the likelihood is replaced by a comparison between simulated data and actual observations. The ABC principle involves three general steps:

- Draw random candidate values for the parameter vector, say $\theta^1, \theta^2, \ldots$, from the prior distribution of the parameters $p_0(\theta)$,
- Use these candidates to simulate synthetic realisations of the observable variables, $\mathbf{y}(\theta^1)$, $\mathbf{y}(\theta^2), \ldots$, by means of the dynamic model equations and taking a fixed initial condition.
- Compare the actual data (observations) and the synthetic data, $\mathbf{y}(\theta^1), \mathbf{y}(\theta^2), \ldots$ using some suitable distance d, i.e., evaluate $d(\mathbf{y}, \mathbf{y}(\theta^1)), d(\mathbf{y}, \mathbf{y}(\theta^2)), \ldots$

Samples that yield a small enough distance, typically below a prescribed tolerance, ϵ, are accepted, and those yielding large distances are discarded. Random candidates are generated and tried until a prescribed number of them are accepted.

The sequential Monte Carlo (SMC) version of ABC is a more sophisticated sampling algorithm that generates a sequence of "populations", i.e., sets of randomly generated parameter values. Each population is associated to prescribed tolerance on the deviation between the actual and synthetic observations. Hence, for a ABC-SMC algorithm generating T populations with J members each we need to specify tolerances $\epsilon_1, \epsilon_2, \ldots, \epsilon_T$ and the candidate parameter vectors accepted in the t-th population, denoted $\{\theta_t^j\}_{j=1}^J$, all share the feature $d(\mathbf{y}, \mathbf{y}(\theta_t^j)) < \epsilon_t$. If the difference $\epsilon_{t-1} - \epsilon_t > 0$ is small, the intuition is that it should not be hard to generate $\{\theta_t^j\}_{j=1}^J$ from $\{\theta_{t-1}^j\}_{j=1}^J$. Afther the T-th population is generated, we expect to have a population $\{\theta_T^j\}_{j=1}^J$ such that $d(\mathbf{y}, \mathbf{y}(\theta_T^j)) < \epsilon_T$ for all j.

The ABC-SMC algorithm generating a sequence of T populations, J samples per population, with tolerances $\epsilon_1, \epsilon_2, \ldots, \epsilon_T$ and Markov kernel $K_t(.|.)$, $t = 1, \ldots, T$, is outlined below.

1. Initialisation: set the population indicator $t = 1$ and the sample indicator $j = 1$. Select the initial condition $\mathbf{x}_0$.
2. If $t = 1$, draw $\theta^\star$ from the prior $p_0(\theta)$. Else, for $t > 1$ draw $\theta^\star$ from the mixture density

$$g_t(\theta) = \sum_{j=1}^{J} w_{t-1}^j K_t(\theta|\theta_{t-1}^j) \tag{27}$$

   where $K_t(.|\theta')$ is a symmetric kernel centred around θ' and $w_{t-1}^1, \ldots, w_{t-1}^J$ are importance weights such that $\sum_{j=1}^J w_{t-1}^j = 1$.
3. If $p_0(\theta^\star) = 0$, then the parameter $\theta^\star$ is off the support set S. Return to step 2.
4. Simulate a dataset $\mathbf{y}(\theta^\star)$ from the state Eqs (9)-(15) with initial condition $\mathbf{x}_0$.
5. If $d(\mathbf{y}, \mathbf{y}(\theta^\star)) \geq \epsilon_t$ reject $\theta^\star$ and return to step 2.
6. Otherwise, if $d(\mathbf{y}, \mathbf{y}(\theta^\star)) < \epsilon_t$, set $\theta_t^j = \theta^\star$ and compute the weight

$$\tilde{w}_t^j = \begin{cases} 1, \text{if} \quad t = 1 \\ \dfrac{p_0(\theta_t^j)}{\sum_{l=1}^J w_{t-1}^l K_t(\theta_t^j|\theta_{t-1}^l)}, \text{if} \quad t > 0 \end{cases} \tag{28}$$

7. If $j < J$, set $j = j + 1$ and go to step 2.
8. Normalize the weights: $w_t^j = \frac{\tilde{w}_t^j}{\sum_{l=1}^J \tilde{w}_t^l}$, $j = 1, \ldots, J$.
9. If $t < T$, then set $t = t + 1$, set $j = 1$ and return to step 2. Otherwise stop.

The posterior estimate of θ is computed as

$$\hat{\theta}_{T,J}^{ABC} = \sum_{j=1}^{J} w_T^j \theta_T^j. \tag{29}$$

We remark that

- The algorithm performance can be optimised by tuning the number of populations, T, and the tolerances $\epsilon_1 > \epsilon_2 > \ldots > \epsilon_T$. However, the latter are limited by the power of the observation noise, σ_y^2. For example, if $d(\mathbf{y}, \mathbf{y}(\theta^\star)) = \frac{1}{n}\sum_{k=0}^n ||\mathbf{y}_k - \mathbf{y}_k(\theta^\star)||^2$ and we assume that the real data has been produced using exactly $\theta^\star$, then $d(\mathbf{y}, \mathbf{y}(\theta^\star)) \to 2\sigma_y^2$ as $n \to \infty$.

- The running time of the ABC-SMC algorithm is random, because of the accept/reject steps 3 and 5. To avoid that the algorithm get stalled at step 2 (sampling), in practice it is common to set a maximum number of draws that can be tried before proceeding to the next population. This implies that some populations may have less than J samples.

Results and discussion

We have carried out computer simulations to assess both the proposed model, i.e., the stochastic modified repressilator model described by Eqs (9)–(15), and the performance of the Monte Carlo based Bayesian inference methods. For all the simulations presented here, the true model parameters are set to their standard values (see Eq (8)) in order to generate synthetic (i.e., simulated) trajectories for the dynamic variables, $\bar{\mathbf{x}}_{i,m}$, $i = 1, 2$ and $m = 0, 1, \ldots$, and sequences of synthetic observations $\mathbf{y}_n$, $n = 1, 2, \ldots$, according to Eq (18). This choice of parameters yields an underlying chaotic dynamics of the state variables, as will be shown below. The integration step of the Euler-Mayurama scheme is $\mathsf{h} = 10^{-3}$ time units. When needed, observations are generated every $m_o = 20$ discrete-time steps of model Eqs (9)–(15) (equivalently, every $m_o\mathsf{h} = 0.02$ continuous time units). The observational noise $\boldsymbol{\epsilon}_n$ in Eq (18) is assumed to be zero-mean Gaussian with identity covariance matrix $\mathbf{I}_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, and the standard deviation parameter is $\sigma_y = 1$.

All the computer experiments to be presented here have been carried out using Matlab R2016a (9.0.0.341360, 64 bits) running on an Apple iMac equipped with a 4 GHz quad-core IntelCore i7 (turbo boost up to 4.2 GHz) and 32 GB of RAM.

When the parameters σ_a, σ_b, σ_c, σ_A, σ_B, σ_C and σ_S, which control the variance of the process noise variables $w_{i,m}^{(j)}$, $j = 1, \ldots, 7$, $i = 1, 2$, are set to zero, we recover the deterministic modified repressilator of [8], which displays chaotic behaviour. For this case, we have run a long simulation of the noiseless system (10,000 continuous time units) and used the results to obtain phase diagrams. In particular, Fig 2(a) shows the phase diagram for the variable b_1 versus a_1, while Fig 2(b) depicts the phase diagram of a_2 versus a_1. What we observe are two views of the multidimensional chaotic attractor generated by this system. When we add dynamical noise in the state equations Eqs (9)–(15), by setting $\sigma_a^2 = \sigma_b^2 = \sigma_c^2 = \sigma_A^2 = \sigma_B^2 = \sigma_C^2 = \sigma_S^2 = 0.02^2$, we obtain an stochastic dynamical system. However, if we repeat the experiment to generate long trajectories (with the same initial conditions and the same duration) we obtain two similar phase diagrams, as shown in Fig 2(c) and 2(d). Indeed, these figures simply depict perturbed versions of the original deterministic attractor. This illustrates the fact that the underlying chaotic dynamics is preserved in the stochastic model, which can account for slight perturbations or uncertainties in the physical system as well.

In the sequel, we compare the performance of the NPMC, PMH and the ABC-SMC parameter estimation methods described in the *Algorithms* section above. For the first set of computer experiments, we simulate trajectories of the coupled stochastic repressilator model in Eqs (9)–(15) with dynamical noise variance $\sigma_a^2 = \sigma_b^2 = \sigma_c^2 = \sigma_A^2 = \sigma_B^2 = \sigma_C^2 = \sigma_S^2 = 0.05^2$ and random initial conditions. To be specific, the initial condition of each simulated trajectory is drawn from a multivariate Gaussian distribution with mean

$$\begin{aligned}(a_{1,0}, b_{1,0}, c_{1,0}, A_{1,0}, B_{1,0}, C_{1,0}, S_{1,0}, a_{2,0}, b_{2,0}, c_{2,0}, A_{2,0}, B_{2,0}, C_{2,0}, S_{2,0}) \quad &= \\ (4.5, 6, 3, 4.2, 19, 4.3, 0.1, 7.3, 1.5, 3.4, 7, 6.5, 3.6, 0.08)\end{aligned} \tag{30}$$

and covariance matrix $\sigma_0^2\mathbf{I}$, where $\sigma_0^2 = 0.05^2$. Once the system state trajectory is generated, we

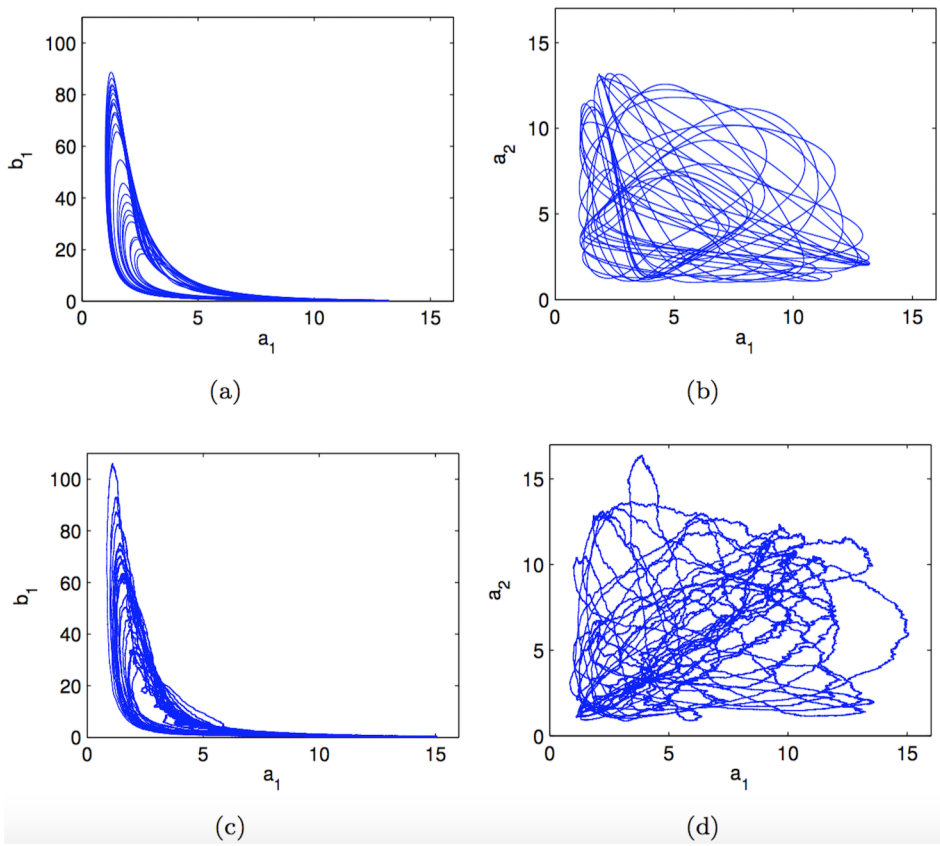

Fig 2. Comparison of 2-dimensional phase space diagrams for the deterministic and the stochastic repressilator models. (a) b_1 versus a_1 and (b) a_2 versus a_1 for the deterministic model; (c) b_1 versus a_1 and (d) a_2 versus a_1 for the stochastic model with common variance 0.02^2 for the dynamical noise in every state equation.

https://doi.org/10.1371/journal.pone.0182015.g002

produce synthetic observations over an interval of 80 continuous time units (which amounts to $80/\mathsf{h} = 80 \times 10^3$ time steps in the Euler-Maruyama scheme). Since observations are assumed to be collected every $m_o = 20$ discrete steps, this yields a sequence of 4×10^3 2-dimensional observation vectors contaminated with zero mean Gaussian noise with unit variance (namely, $\sigma_y^2 = 1$). In order to compute the likelihood approximation $\ell^N(\theta)$, which is necessary to obtain the weights in the NPMC algorithm and the acceptance probability in the PMH method, we run a bootstrap filter with $N = 100$ particles. We recall that the latter algorithm is detailed in the supplementary material S1 Appendix.

We first consider the estimation of the posterior probability density functions (pdf's) of the unknown parameters given the ground truth signal and the sequence of observations produced as described above. Fig 3 shows the kernel density estimators obtained for the Q parameter *in a single typical run* of the PMH (left), PMH (middle) and ABC-SMC (right) algorithms. In the three plots, the actual value of the parameter Q is indicated with a dashed vertical line. For the NPMC algorithm with $N = 200$ samples per iteration, we additionally plot

- the a priori uniform pdf (solid red line)
- the pdf estimated after the first iteration of the algorithm (dashed black line), and
- the pdf estimated after 20 iterations of the algorithm (solid blue line).

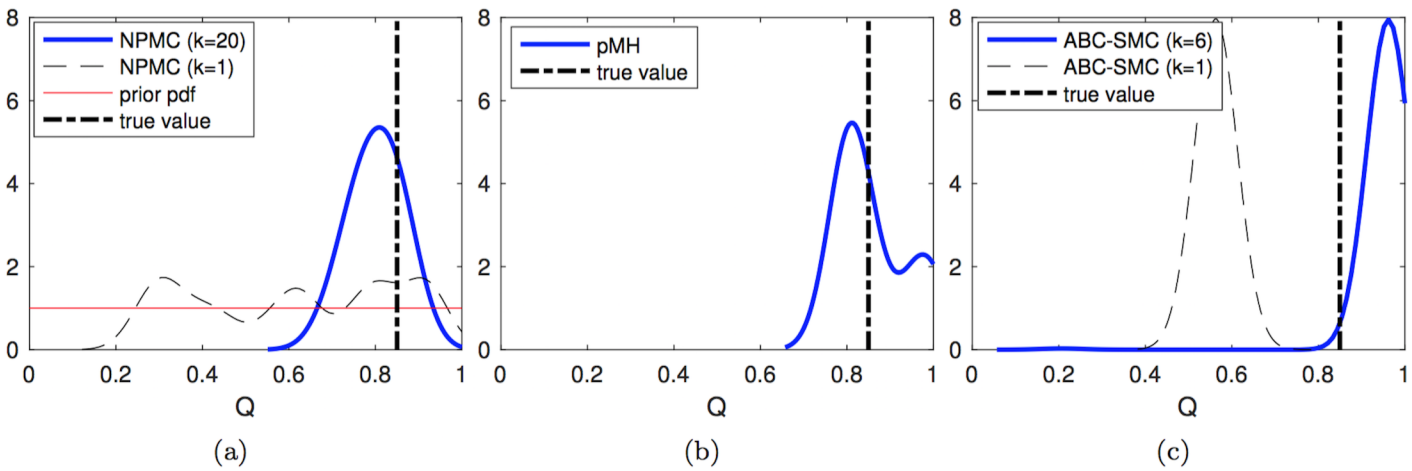

Fig 3. Estimated posterior pdf of *Q*. Posterior pdf's computed from the outcome of: (a) the NPMC algorithm with $M = 200$ samples per iteration, over 20 iterations, (b) the PMH algorithm with scale parameter $\sigma = 0.04$ generating a chain of length $L = 4{,}000$ elements, and (c) the ABC-SMC method with 5 stages, tolerances $\square_{1:5} = \{3.0, 2.4, 2.3, 2.2, 2.1\}$ and 800 accepted samples per stage. The true parameter value is shown with a vertical dashed line. For the NPMC method, the prior pdf, the pdf after the first iteration and the pdf after the last iteration are shown. For the ABC-SMC scheme, the pdf's computed from the first and last population are displayed.

https://doi.org/10.1371/journal.pone.0182015.g003

For the PMH algorithm, the figure shows the posterior pdf estimate computed from the last 2,000 samples of a chain of total length $L = 20 \times 200 = 4{,}000$, so that the complexity is approximately the same as with the NPMC scheme. The last plot shows the results for the ABC-SMC method with 5 populations. In particular, the dashed black line displays the pdf estimate from the first population and the solid blue line corresponds to the density estimate obtained from the fifth population. All pdf estimates have been obtained via the ksdensity function of Matlab R2016a, which uses a Gaussian kernel and calculates the bandwidth automatically as a function of the number of available samples.

We observe how the pdf estimate produced by the NPMC method improves significantly from the first to the last (20-th) iteration. Both the NPMC and PMH methods yield final density estimates which fit the ground truth value fairly accurately: both density estimators have their mode close to the true value of Q and the probability is also rather tightly concentrated around this value (it falls toward 0 quickly). For the ABC-SMC algorithm, there is a also an improvement from the first to the last population, but the final density estimate still leaves the true value of Q on the left tail of the function.

Figs 4–6 display the kernel density estimators, computed in the same manner, for the unknown parameters m, α and β_a. The results are similar as for parameter Q. Both the NPMC and PMH algorithms yield comparably fit pdf estimators with a similar computational cost of $20 \times 200 = 4{,}000$ random draws in total. The density estimators produced by the NMPC algorithm improve significantly through the iterations. The ABC-SMC algorithm yields similarly good density estimates for α and β_a, with considerable improvement from the first to the fifth population as well, but there is a poor outcome for parameter m. In this latter case, the density estimate after the first iteration is actually better than the last one (after the fifth iteration). While this effect does not necessarily occur in every simulation run, we show below that the ABC-SMC method indeed has the poorest average performance of the three schemes.

From the plots in Figs 3–6, we observe that the probability mass of the pdf estimates tends to concentrate around the region where the actual parameter value is located, e.g., for the

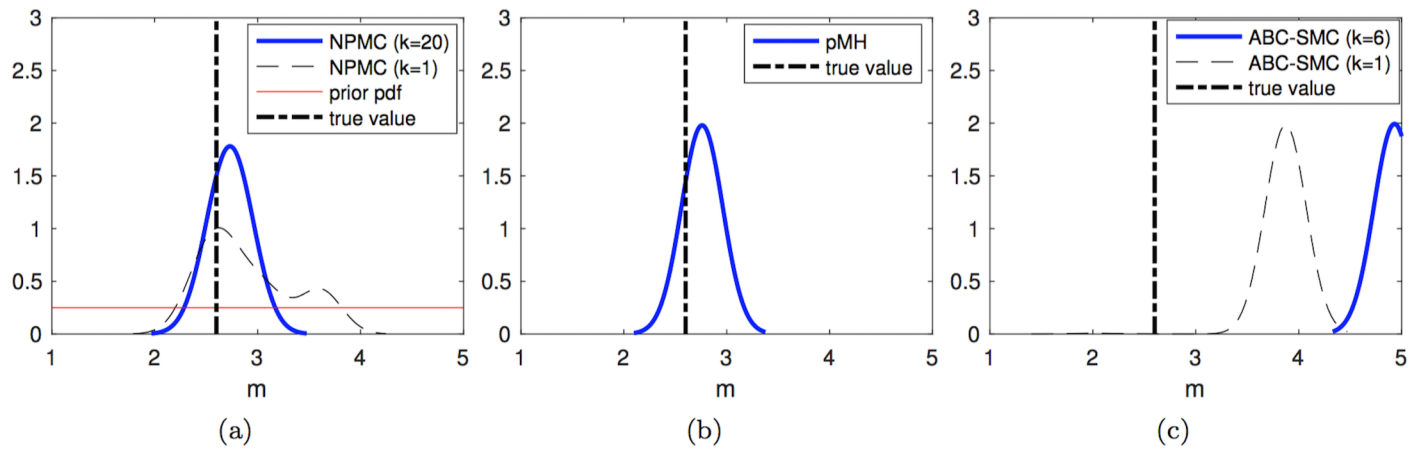

Fig 4. Estimated posterior pdf of m**.** Posterior pdf's computed from the outcome of: (a) the NPMC algorithm with $M = 200$ samples per iteration, over 20 iterations, (b) the PMH algorithm with scale parameter $\sigma = 0.04$ generating a chain of length $L = 4{,}000$ elements, and (c) the ABC-SMC method with 5 stages, tolerances $\square_{1:5} = \{3.0, 2.4, 2.3, 2.2, 2.1\}$ and 800 accepted samples per stage. The true parameter value is shown with a vertical dashed line. For the NPMC method, the prior pdf, the pdf after the first iteration and the pdf after the last iteration are shown. For the ABC-SMC scheme, the pdf's computed from the first and last population are displayed.

https://doi.org/10.1371/journal.pone.0182015.g004

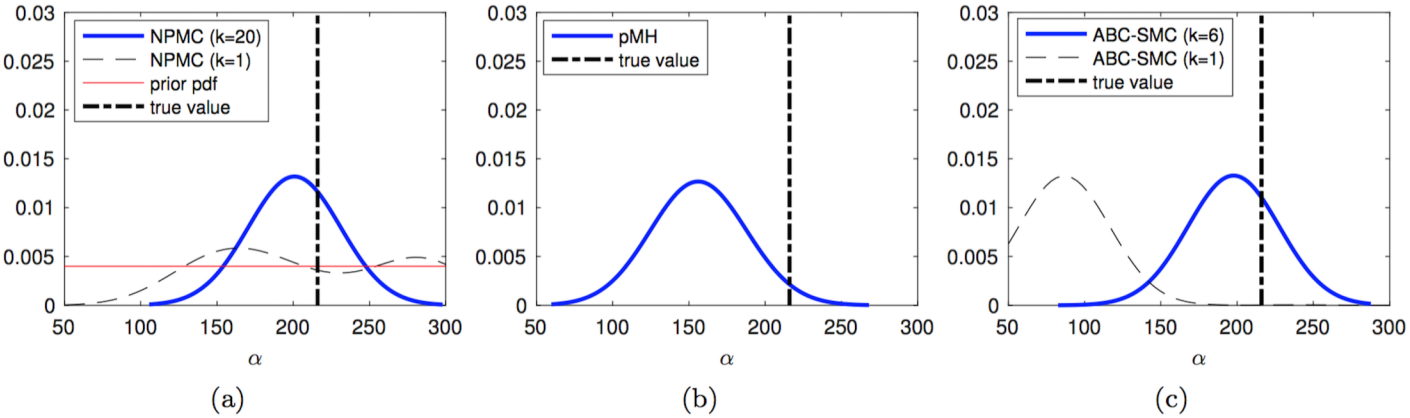

Fig 5. Estimated posterior pdf of α. Posterior pdf's computed from the outcome of: (a) the NPMC algorithm with $M = 200$ samples per iteration, over 20 iterations, (b) the PMH algorithm with scale parameter $\sigma = 0.04$ generating a chain of length $L = 4{,}000$ elements, and (c) the ABC-SMC method with 5 stages, tolerances $\square_{1:5} = \{3.0, 2.4, 2.3, 2.2, 2.1\}$ and 800 accepted samples per stage. The true parameter value is shown with a vertical dashed line. For the NPMC method, the prior pdf, the pdf after the first iteration and the pdf after the last iteration are shown. For the ABC-SMC scheme, the pdf's computed from the first and last population are displayed.

https://doi.org/10.1371/journal.pone.0182015.g005

NPMC and PMH algorithms and parameter Q, or for the ABC-SMC algorithm and parameter α. However, there is no perfect match (the mode of the estimated pdf does not exactly coincide with the ground-truth value of the parameter) and we can also see some cases, e.g., the density estimate produced by the ABC-SMC scheme for parameter m, where the probability mass is placed far away from the true parameter value.

To explain these mismatches, let us recall that the approximate statistics generated by the NPMC and PMH algorithms converge to the true value of these statistics as the computational effort is increased. For example, if we are interested in the posterior mean of θ, then the

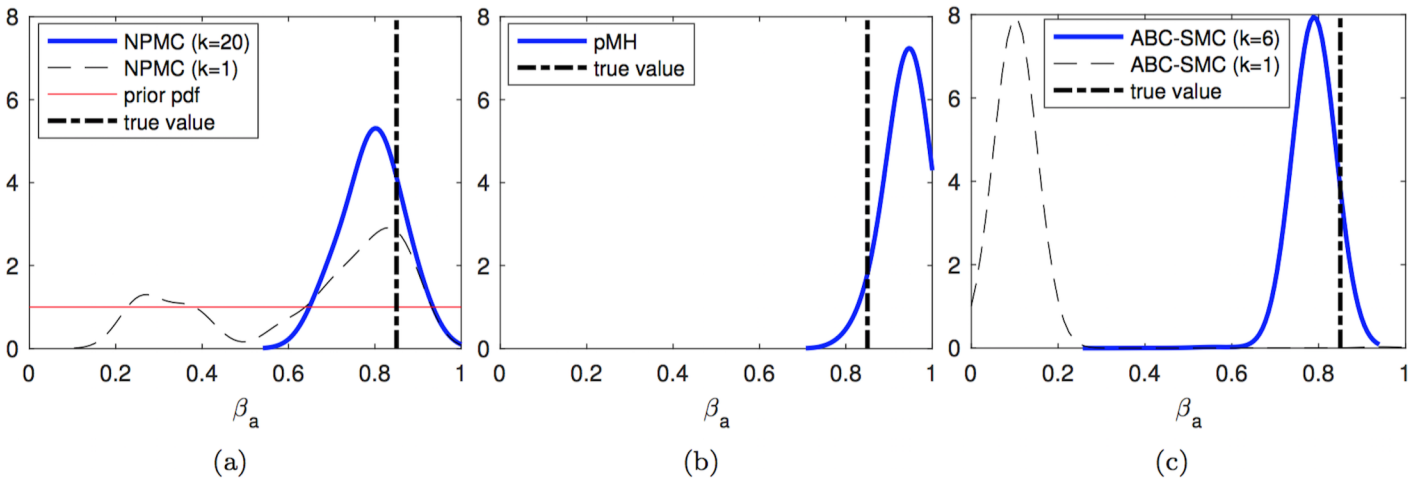

Fig 6. Estimated posterior pdf of β_a. Posterior pdf's computed from the outcome of: (a) the NPMC algorithm with $M = 200$ samples per iteration, over 20 iterations, (b) the PMH algorithm with scale parameter $\sigma = 0.04$ generating a chain of length $L = 4{,}000$ elements, and (c) the ABC-SMC method with 5 stages, tolerances $\square_{1:5} = \{3.0, 2.5, 2.4, 2.3, 2.2, 2.1\}$ and 800 accepted samples per stage. The true parameter value is shown with a vertical dashed line. For the NPMC method, the prior pdf, the pdf after the first iteration and the pdf after the last iteration are shown. For the ABC-SMC scheme, the pdf's computed from the first and last population are displayed.

https://doi.org/10.1371/journal.pone.0182015.g006

NPMC estimate has the form

$$\hat{\theta}_k^M = \sum_{i=1}^{M} \mathsf{w}_k^i \theta_k^i \tag{31}$$

after the k-th iteration. It can be proved [26] that $\lim_{M\to\infty} \hat{\theta}_k^M = E[\theta|\mathbf{y}]$, where $E[\theta|\mathbf{y}]$ is the expected value of the parameter vector θ conditional on the observations $\mathbf{y}$. However, depending on the available data (especially, the dimension of vector $\mathbf{y}$), the posterior mean $E[\theta|\mathbf{y}]$ can be significantly different from the *true* value of θ used to generate the synthetic data.

A simple way to illustrate this issue is to approximate the likelihood of two different parameter vectors, say $\theta_* = [Q_*, \mathsf{m}_*, \alpha_*, \beta_{a*}] = [0.85, 2.6, 216, 0.85]$ the ground truth, and $\theta' = \theta_* + [0, 0, -10, 0]$ a mismatched version, and see that, for a common and fixed observation vector, they are approximately the same (actually, $\ell^N(\theta') > \ell^N(\theta_*)$, even if the difference is small). This is shown in Fig 7, which, for a fixed sequence $\mathbf{y} = \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_n, \ldots\}$, depicts the approximate log-likelihood $\log(\ell^N(\mathbf{y}_{1:n}|\theta_*))$ and $\log(\ell^N(\mathbf{y}_{1:n}|\theta'))$ versus n. The number of particles in the BF is set to $N = 600$ in this case to ensure that we obtain low-variance estimates.

Next, we aim at a comparison of the three parameter estimation schemes in terms of their normalised mean square error (NMSE). Assume that we run J independent simulations and we let

$$\hat{\theta}(j) = (\hat{Q}(j), \hat{m}(j), \hat{\alpha}(j), \hat{\beta}_a(j)) \tag{32}$$

be the estimate of the unknown parameter vector output by a given algorithm in the j-th

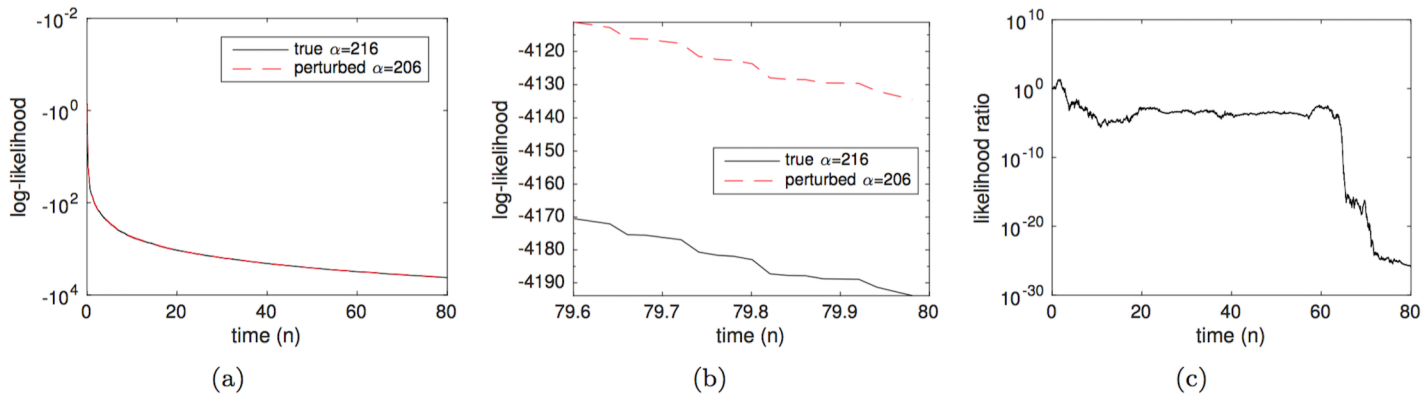

Fig 7. Comparison of the approximate likelihood of the true parameter vector $\theta_* = [0.85, 2.6, 216, 0.85]$ and perturbed version $\theta' = [\mathbf{0.85}, \mathbf{2.6}, \underline{\mathbf{206}}, \mathbf{0.85}]$**.** (a) Approximate log-likelihoods, $\ell(\mathbf{y}_{1:n}|\theta_*)$ and $\ell(\mathbf{y}_{1:n}|\theta')$ versus time n. (b) Zoom of plot (a) for a shorter time interval, showing that the perturbed parameter θ' yields a higher likelihood for the observation sequence $\mathbf{y}$ generated in this computer experiment. (c) The likelihood ratio $\frac{\ell^N(\mathbf{y}_{1:n}|\theta_*)}{\ell^N(\mathbf{y}_{1:n}|\theta')}$ versus time n, showing that $\ell^N(\mathbf{y}_{1:n}|\theta') > \ell^N(\mathbf{y}_{1:n}|\theta_*)$.

https://doi.org/10.1371/journal.pone.0182015.g007

simulation trial. Then, we calculate the empirical NMSE for that estimation algorithm as

$$\mathrm{NMSE} = \frac{1}{4J}\sum_{j=1}^{J}\left(\frac{(\hat{Q}(j)-Q_*)^2}{Q_*^2}+\frac{(\hat{\mathsf{m}}(j)-\mathsf{m}_*)^2}{\mathsf{m}_*^2}+\frac{(\hat{\alpha}(j)-\alpha_*)^2}{\alpha_*^2}+\frac{(\hat{\beta}_a(j)-\beta_{a*})^2}{\beta_{a*}^2}\right), \quad (33)$$

i.e., the NMSE is the average empirical quadratic error, normalised per parameter.

The three schemes for Bayesian parameter estimation admit some tuning that may affect their performance. For the NPMC algorithm, there is the choice of the proposal pdf and the clipping parameter M_c. For all the experiments, we assume that the k-th iteration proposal q_k, is Gaussian (with mean and covariance computed using the TIWs from the $(k-1)$-th iteration) and $M_c = \lfloor\sqrt{M}\rfloor$, which is its maximum admissible value that guarantees asymptotic convergence [11]. The number of samples in the BF (to compute the likelihood $\ell^N(\mathbf{y}|\theta)$) is set to $N = 100$, which has been found to yield a good trade-off between computational cost and accuracy in the calculation of the weights.

Based on the results from [8] and a number of additional simulation trials (not shown in the paper) we have selected a configuration for the ABC-SMC algorithm with 5 populations, tolerances $\epsilon_{1:5} = \{3.0, 2.4, 2.3, 2.2, 2.1\}$ and a target of J accepted samples per population (with take $J = 200$, 800 and 1,600 for different experiments). This is the configuration, with $J = 800$ samples, already used for the pdf estimates shown in Figs 3–6. We have tried other sequences of tolerances with very similar results. The configuration of choice yields a computational cost which is similar to the NPMC algorithm with $M = 200$ and $K = 20$, and the PMH scheme with $L = 4{,}000$ entries in the Markov chain. The distance $d(\mathbf{y}, \mathbf{y}(\theta'))$ is quadratic, namely,

$$d(\mathbf{y},\mathbf{y}(\theta')) = \frac{1}{K}\sum_{n=0}^{K-1}\|\,\mathbf{y}_n - \mathbf{y}_n(\theta')\,\|^2 \quad (34)$$

and, since the observations vectors are 2×1 and the observation noise has variance $\sigma_y^2 = 1$, the expected distance is $E[d(\mathbf{y},\mathbf{y}(\theta'))] \geq 2\sigma_y^2 = 2$ even in the case of perfect parameter estimation

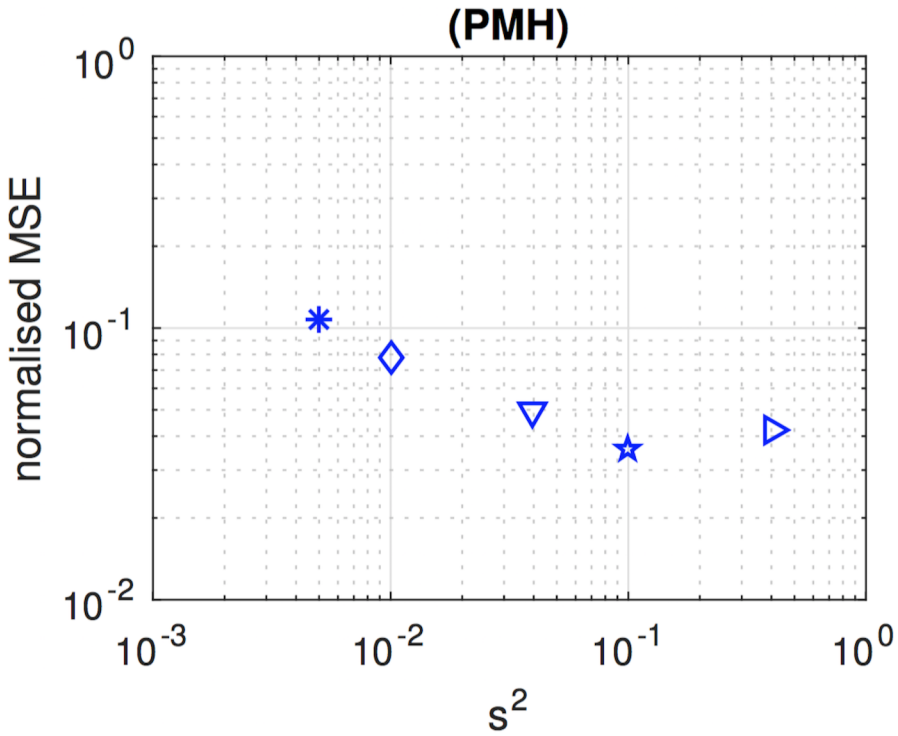

Fig 8. Performance of the PMH algorithm with varying scale parameter. NMSE attained by the PMH algorithm with scale parameter $s^2 \in \{0.005, 0.010, 0.040, 0.100, 0.400\}$. This parameter controls the variance of the truncated Gaussian kernel used to generate candidate values of the Markov chain. The accuracy of the algorithm is sensitive to the choice of s^2 as it determines the pace of mixing [15] of the Markov kernel $\mathcal{M}(\theta_m|\theta_{m-1})$.

https://doi.org/10.1371/journal.pone.0182015.g008

and perfect initialisation. Therefore, the tolerance of the last population should always be greater than 2.

As for the PMH algorithm, we have assumed that the candidate samples $\tilde{\theta}_m$ are drawn from a truncated Gaussian kernel $\mathcal{M}(\theta_m|\theta_{m-1})$, constructed from the Gaussian distribution with mean θ_{m-1}, covariance matrix

$$\Sigma = \sigma^2 \begin{pmatrix} 0.01 & 0 & 0 & 0 \\ 0 & 100 & 0 & 0 \\ 0 & 0 & 0.01 & 0 \\ 0 & 0 & 0 & 0.01 \end{pmatrix}, \tag{35}$$

and support restricted to the set $\mathsf{S} = (0,1) \times (1,5) \times (50,300) \times (0,1)$. The likelihood needed to determine the acceptance probability of a sample $\tilde{\theta}$ is approximated as $\ell^N(\mathbf{y}|\tilde{\theta})$, using the BF with $N = 100$ particles, the same as for the NPMC scheme. Note that, given Σ above, the parameters are sampled independently with variance proportional to their support sets (the support of α being much larger). The scale parameter σ^2 has a direct impact on the performance of the PMH method and, hence, we have carried out computer simulations to optimise it.

Fig 8 shows the variation of the NMSE, for the PMH algorithm with $L = 4{,}000$, as we increase the value of the scale parameter from $\sigma^2 = 0.005$ up to $\sigma^2 = 0.4$. The plot has been

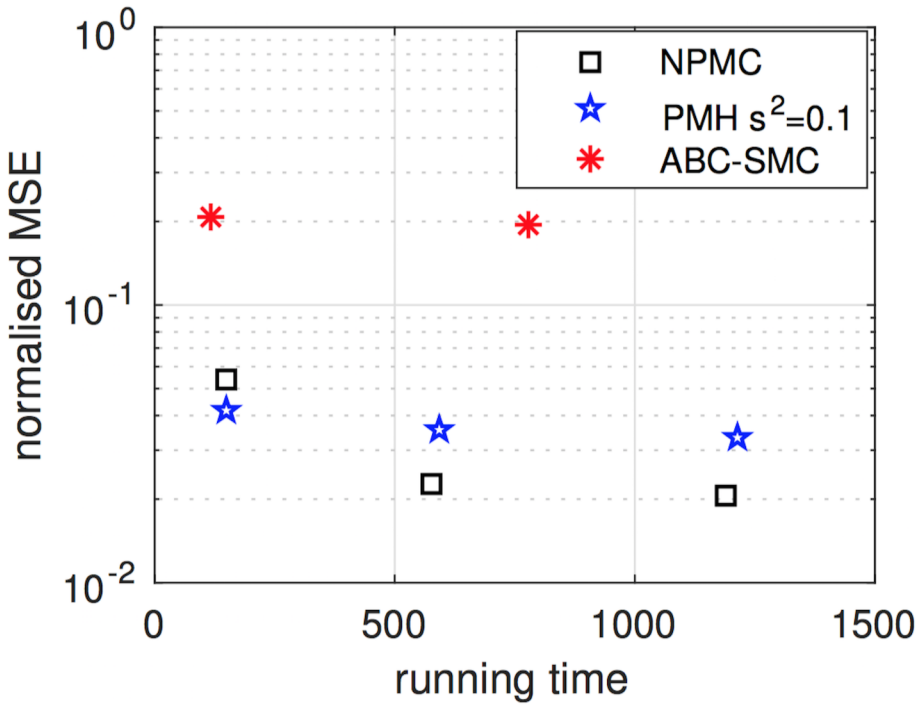

Fig 9. Normalised MSE of the PMH, NPMC and ABC-SMC algorithms versus running time. The figure displays the average normalised MSE per parameter attained by the NPMC scheme (with $M = 50$, 200 and 400 samples per iteration, after $K = 20$ iterations), the PMH algorithm with chains of length $L = 1{,}000$, $L = 4{,}000$ and $L = 8{,}000$, and scale parameter $s^2 = 0.1$, and the ABC-SMC method with $J = 200$ and $J = 800$ samples per population. The lowest error is achieved by the NPMC algorithm, which demands a running time slightly shorter than the PMH schemes and much shorter than the (average) running time of the ABC-SMC method. Time is given in minutes.

https://doi.org/10.1371/journal.pone.0182015.g009

obtained as an average of 40 independent simulation runs. For each run, a different sequence of state realisations and observations is generated, and then the PMH algorithm is run five times for the same observations with different values of σ^2. The figure shows that the NMSE keeps improving as we increase σ^2 until it worsens for $\sigma^2 = 0.4$. From this experiment, we choose $\sigma^2 = 0.1$ as the scale parameter for the PMH scheme that we compare with the other two algorithms.

Fig 9 displays the NMSE versus running time (in minutes) attained by

- the NPMC scheme with $K = 20$ iterations and $M = 50$, 200, 400 samples per iteration,
- the PMH method with chain lengths of $L = 50 \times 20 = 1{,}000$, $L = 200 \times 20 = 4{,}000$ and $L = 400 \times 20 = 8{,}000$, and scale parameter $\sigma^2 = 0.1$, and
- the ABC-SMC algorithm with tolerances $\epsilon_{1:5} = \{3.0, 2.4, 2.3, 2.2, 2.1\}$ and targets of $J = 200$ and $J = 800$ samples per population.

The values of M, K, L and J are chosen to attain comparable running times for the three methods. Note, however, that while the running times for the PMH and NPMC algorithms are deterministic given the parameters L, M and K, the running time of the ABC-SMC algorithm is random (even for fixed J), since an a priori unknown proportion of samples is expected to be rejected. Therefore, the times shown in the figure for this algorithm are an average (over 40

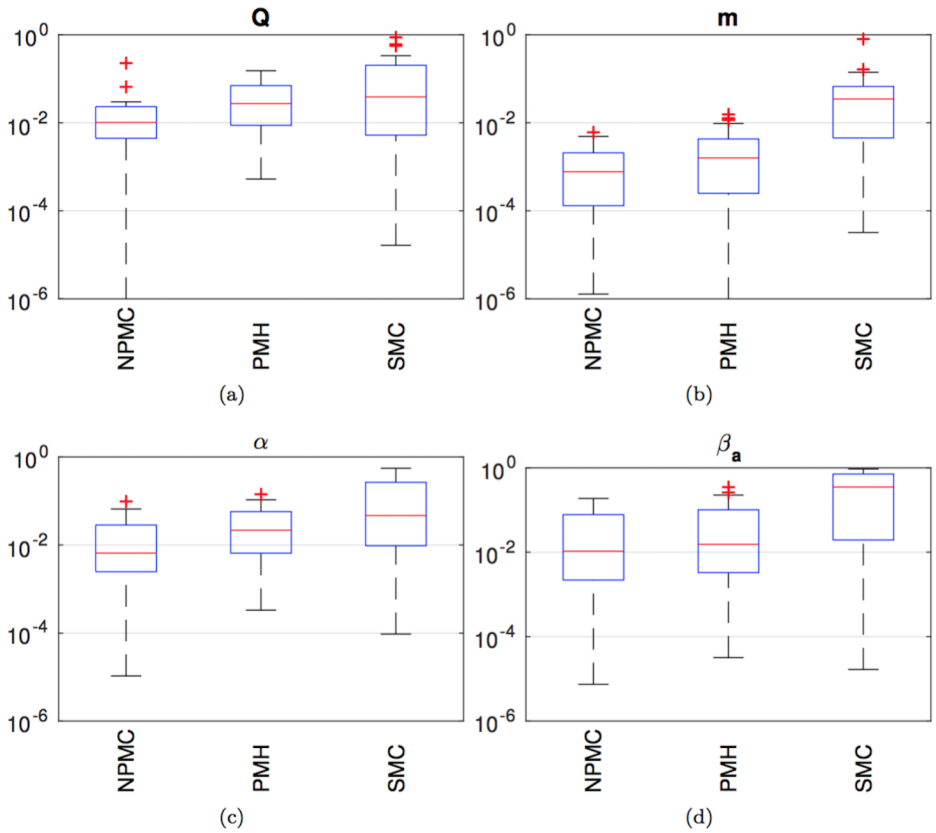

Fig 10. Box plots of the empirical NMSE for the NMPC algorithm with $M = 200$ and $K = 20$, the PMH scheme with scale factor $\sigma^2 = 0.1$ and $L = M \times K = 4{,}000$ samples and the ABC-SMC method. (a) Parameter Q. (b) Parameter m. (c) Parameter α. (d) Parameter β_a. For each box, the red central mark is the median NMSE, the edges of the blue box are the 25th and 75th percentiles, the black whiskers extend to the most extreme data-points which are not considered outliers. Outliers are plotted individually as red crosses.

https://doi.org/10.1371/journal.pone.0182015.g010

independent simulations). Moreover, to avoid that the ABC-SMC procedure gets stalled at any particular population, a limit of $4{,}000 \times J$ random draws per population has been imposed (which amounts to a minimum acceptance rate of $\approx 25 \times 10^{-5}$). If this limit is reached, the population is taken as complete even if containing less than J samples. Even with this limitation, the ABC-SMC has the largest (average) running time, and hence the highest computational cost, of all the algorithms tested.

The figure shows that the PMH algorithm attains the best performance when the computational budget is kept minimal ($M = 50$ and $L = 1{,}000$ for the NPMC and PMH algorithms, respectively) but the NPMC scheme yields the overall best results (for comparable running times) as the computational effort is moderately increased. Both the NPMC and PMH schemes outperform the ABC-SMC method clearly for this example. The most efficient choice for this experiment appears to be the NPMC scheme with $M = 200$ samples per iteration and $K = 20$ iterations. Note that the error decreases only slightly as we increase the number of samples to $M = 400$, even if the running time is duplicated.

Fig 10 displays box-plots of the NMSE outcomes of the same 40 independent simulation runs carried out to obtain Fig 9, but only for the NPMC ($M = 200$ samples), PMH ($L = 4{,}000$,

scale factor $\sigma^2 = 0.1$) and ABC-SMC ($J = 800$) algorithms, which demand comparable running times. For each box, the red central mark is the median, the edges of the blue box are the 25th and 75th percentiles, the black whiskers extend to the most extreme data-points which are not considered outliers, and the outliers are plotted individually as red crosses. A point is taken to be an outlier if it is larger than $\mathsf{q}_{75\%} + \frac{3}{2}(\mathsf{q}_{75\%} - \mathsf{q}_{25\%})$, where $\mathsf{q}_{75\%}$ and $\mathsf{q}_{25\%}$ are the 75% and 25% percentiles, respectively.

We observe four plots in Fig 10, labeled (a), (b), (c) and (d), and corresponding to the parameters Q, m, α and β_a, respectively. The outcomes are similar for all four parameters. The NMPC and PMH method perform similarly, with the median of the NPMC errors being slightly lower for all parameters and the dispersion ($|\mathsf{q}_{75\%} - \mathsf{q}_{25\%}|$) being similar or slightly smaller for NPMC compared to PMH. The ABC-SMC algorithm displays a consistently higher median NMSE and a slightly larger dispersion than the other two schemes.

The purpose of introducing a stochastic model of the coupled repressilator system is, as explained in the Introduction to this paper, twofold. On one hand, it provides a formalism to account for uncertainties in the design and realisation of the system. On the other hand, it sets up a probabilistic framework that enables the application of the NPMC and PMH computational schemes (and possibly other Bayesian techniques) for parameter estimation. From the latter point of view, it is of interest to test whether these methods are still useful when the underlying signals are produced by a deterministic, rather than stochastic model. With this aim, we have repeated the same computer experiments as in Figs 8–10 when the realisations of the state variables (a_i, b_i, c_i, A_i, B_i, C_i) are produced by the deterministic system given by Eqs (9)–(15) with null variances (i.e., $\sigma_a = \sigma_b = \sigma_c = \sigma_A = \sigma_B = \sigma_C = \sigma_S = 0$). We still generate noisy observations, with $\sigma_y^2 = 1$ as in the previous experiments, to account for observational noise.

Fig 11 shows the NMSE achieved by the PMH method with varying scale parameter σ^2 for the deterministic coupled repressilator with noisy observations. Compared to the results with the stochastic system, we observe that the best performance is attained with a smaller kernel variance, namely with a scale factor $\sigma^2 = 0.04$ (versus $\sigma^2 = 0.1$ in Fig 8).

Fig 12 displays the empirical NMSE of the competing algorithms when the state trajectories are realisations of the deterministic coupled repressilator model. As before, we compare the NPMC algorithm with $M = 50$, 200 and 800 samples per iteration (and $K = 20$ iterations), the PMH algorithm with chain lengths $L = 1{,}000$, $L = 4{,}000$ and $L = 8{,}000$, and scale parameter $\sigma^2 = 0.04$ (optimised for this scenario) and the ABC-SMC algorithm with $J = 200$, 800 and 1,600 samples per population, and otherwise the same configuration as for the simulations with the stochastic model.

Although the results are similar to Fig 9 (corresponding to the stochastic model), there are some significant differences in Fig 12. First, we observe that the average running time of the ABC-SMC method ($\approx$580 minutes for $J = 800$) reduces considerably and becomes slightly smaller than the running time of the PMH algorithms ($\approx$596 minutes for $L = 4{,}000$) and the NPMC scheme (with $M = 200$, which takes $\approx$586 minutes). The average error of the ABC-SMC scheme is, however, still higher than the error of the other methods. The other relevant outcome is that the NMPC algorithm is now consistently better than the PMH method, including the case with minimum running time (which corresponds to $M = 50$ samples per iteration for the NPMC algorithm and $L = 1{,}000$ for the PMH scheme).

Finally, Fig 13 shows the box plots for the NMSE values obtained in the set of 40 independent simulation runs, for the NPMC ($M = 200$), PMH ($\sigma^2 = 0.04$, $L = 4{,}000$) and ABC-SMC ($J = 800$) methods, which demand a similar running time. The results are plotted separately for the parameters Q (a), m (b), α (c) and β_a (d). We see that the median NMSE of the NPMC algorithm is slightly better than the median NMSE of the PMH scheme for all parameters

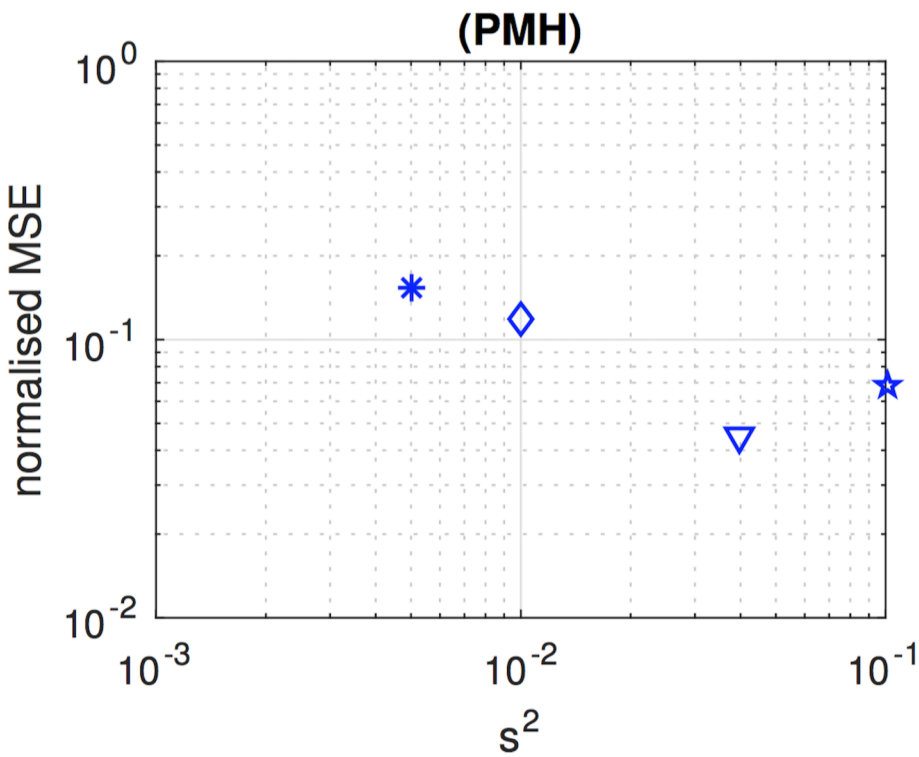

Fig 11. Performance of the PMH algorithm with varying scale parameter for the deterministic coupled repressilator with noisy observations. Empirical NMSE attained by the PMH algorithm with scale parameter $s^2 \in \{0.005, 0.01, 0.04, 0.1\}$. This parameter controls the variance of the truncated Gaussian kernel used to generate candidate values of the Markov chain. Compared to the case with stochastic trajectories, the best performance is attained with a smaller value of s^2 (hence, by a Markov kernel with less variance). Results obtained averaging 40 independent simulation runs.

https://doi.org/10.1371/journal.pone.0182015.g011

except β_a. For all four parameters, the NPMC algorithm yields a smaller dispersion of the errors.

Conclusions

We have proposed a stochastic version of the coupled repressilator model of [5] that enables

1. a mathematically principled manner of describing experimental uncertainties in the synthesis of multicellular clocks, and
2. the design of probabilistic methods for the estimation of unknown parameters in the model, even if the underlying dynamics is chaotic, which makes the problem more challenging.

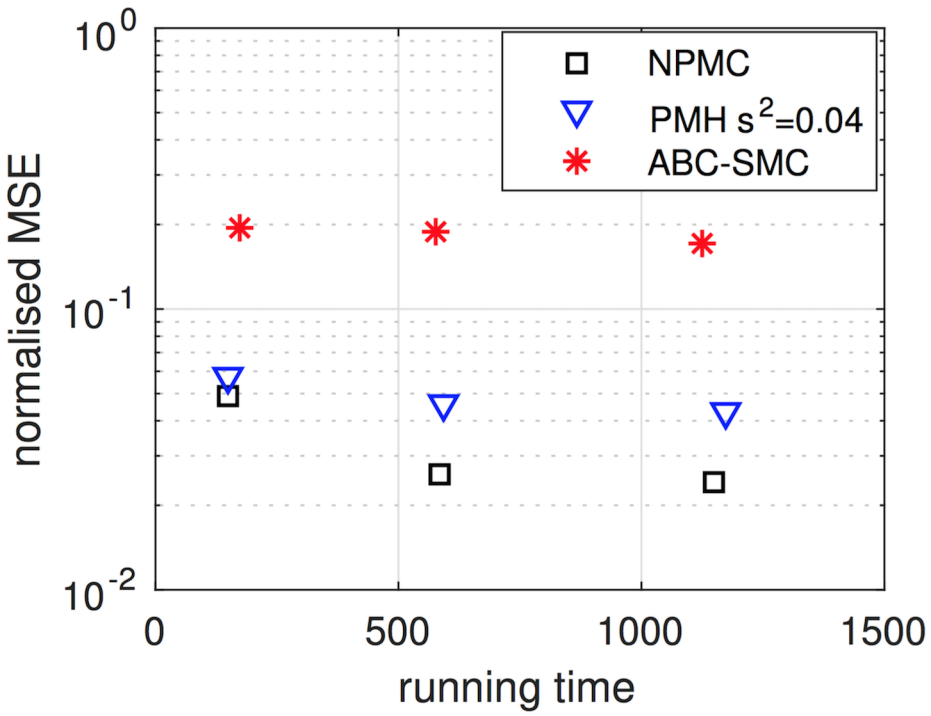

Fig 12. NMSE of the PMH, NPMC and ABC-SMC algorithms versus running time, for noiseless trajectories. The figure displays the average NMSE per parameter attained by the NPMC scheme (with $M = 50$, 200 and 400 samples per iteration, after $K = 20$ iterations), the PMH algorithm with scale parameter $s^2 = 0.04$ (optimised for this scenario) and chain lengths $L = 1{,}000$, $L = 4{,}000$ and $L = 8{,}000$, and the ABC-SMC method with $J = 200$, 800 and 1,600. The ground truth signal is generated from a deterministic coupled repressilator system, although the observations are noisy. The lowest error is achieved by the NPMC algorithm, which demands a running time slightly shorter than the PMH schemes and slightly longer than the (average) running time of the ABC-SMC method. Time is given in minutes.

https://doi.org/10.1371/journal.pone.0182015.g012

In particular, we have compared three Bayesian methods for computational model inference that enable the calculation of a posteriori probability distributions for the set of unknown parameters given a sequence of noisy observations of just two state variables. We have presented an extensive computer simulation study that illustrates the relationship between the deterministic and stochastic repressilator models and demonstrates the relative efficiency and accuracy of the nonlinear population Monte Carlo (NPMC), particle Metropolis-Hastings (PMH) and approximate Bayesian computation sequential Monte Carlo (ABC-SMC) algorithms. The best trade-off between accuracy and computational cost is attained by the NPMC algorithm, both for the deterministic and the stochastic coupled repressilator models, although the PMH scheme can attain a similar performance. The ABC-SMC method, on the other

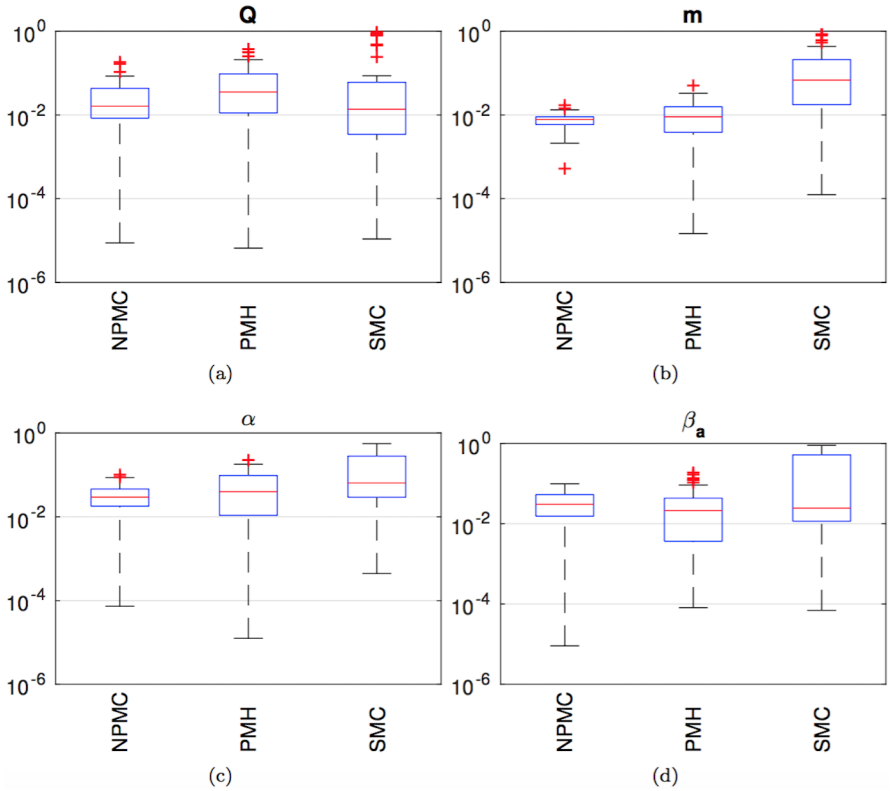

Fig 13. Box plots of the empirical NMSE for the NMPC algorithm with $M = 200$ and $K = 20$, the PMH scheme with scale factor $\sigma^2 = 0.1$ and $L = M \times K = 4{,}000$ samples and the ABC-SMC method. State trajectories are noiseless, generated from a deterministic coupled repressilator model. Observations are noisy. (a) Parameter Q. (b) Parameter m. (c) Parameter α. (d) Parameter β_a. For each box, the red central mark is the median NMSE, the edges of the blue box are the 25th and 75th percentiles, the black whiskers extend to the most extreme data-points which are not considered outliers. Outliers are plotted individually as red crosses.

https://doi.org/10.1371/journal.pone.0182015.g013

hand, has a two drawbacks. It demands knowledge of the initial condition of the system and its accuracy is directly limited by the variance of the observational noise.

Supporting information

S1 Appendix. The bootstrap filter.
(PDF)

S1 Code. Coding supplement (zip file). These are the matlab scripts used for the simulations in the paper. The main script is repress.m. The first part of the code includes the main simulation parameters. Notation is essentially the same as in the paper.
(ZIP)

Author Contributions

Conceptualization: IPM AZ JM.

Data curation: IPM JM.

Formal analysis: IPM JM.

Funding acquisition: IPM AZ JM.

Investigation: IPM AZ JM.

Methodology: IPM JM.

Project administration: IPM AZ JM.

Resources: IPM AZ JM.

Software: IPM AZ JM.

Supervision: IPM AZ JM.

Validation: IPM JM.

Visualization: IPM AZ JM.

Writing – original draft: IPM JM.

Writing – review & editing: IPM AZ JM.

References

1. Cameron DE, Bashor CJ, Collins JJ. A brief history of synthetic biology. Nature Reviews Microbiology. 2014; 12(5):381–390. https://doi.org/10.1038/nrmicro3239 PMID: 24686414
2. Chabrier-Rivier N, Chiaverini M, Danos V, Fages F, Schächter V. Modeling and querying biomolecular interaction networks. Theoretical Computer Science. 2004; 325(1):25–44. https://doi.org/10.1016/j.tcs.2004.03.063
3. Elowitz MB, Leibler S. A synthetic oscillatory network of transcriptional regulators. Nature. 2000; 403 (6767):335–338. https://doi.org/10.1038/35002125 PMID: 10659856
4. Garcia-Ojalvo J, Elowitz MB, Strogatz SH. Modeling a synthetic multicellular clock: repressilators coupled by quorum sensing. Proceedings of the National Academy of Sciences of the United States of America. 2004; 101(30):10955–10960. https://doi.org/10.1073/pnas.0307095101 PMID: 15256602
5. Ullner E, Zaikin A, Volkov EI, García-Ojalvo J. Multistability and clustering in a population of synthetic genetic oscillators via phase-repulsive cell-to-cell communication. Physical Review Letters. 2007; 99 (14):148103. https://doi.org/10.1103/PhysRevLett.99.148103 PMID: 17930726
6. Ullner E, Koseska A, Kurths J, Volkov E, Kantz H, García-Ojalvo J. Multistability of synthetic genetic networks with repressive cell-to-cell communication. Physical Review E. 2008; 78(3):031904. https://doi.org/10.1103/PhysRevE.78.031904
7. Koseska A, Ullner E, Volkov E, Kurths J, Garcia-Ojalvo J. Cooperative differentiation through clustering in multicellular populations. Journal of theoretical biology. 2010; 263(2):189–202. https://doi.org/10.1016/j.jtbi.2009.11.007 PMID: 19932703
8. Mariño IP, Ullner E, Zaikin A. Parameter Estimation Methods for Chaotic Intercellular Networks. PLoS ONE. 2013; 8(11):e79892. https://doi.org/10.1371/journal.pone.0079892 PMID: 24282513
9. Evans RD, Ricardez-Sandoval LA. Multi-scenario modelling of uncertainty in stochastic chemical systems. Journal of Computational Physics. 2014; 273:374–392. https://doi.org/10.1016/j.jcp.2014.05.028
10. Andrieu C, Doucet A, Holenstein R. Particle Markov chain Monte Carlo methods. Journal of the Royal Statistical Society B. 2010; 72:269–342. https://doi.org/10.1111/j.1467-9868.2009.00736.x
11. Koblents E, Míguez J. A population Monte Carlo scheme with transformed weights and its application to stochastic kinetic models. Statistics and Computing. 2015; 25(2):407–425. https://doi.org/10.1007/s11222-013-9440-2

12. Toni T, Welch D, Strelkowa N, Ipsen A, Stumpf MP. Approximate Bayesian computation scheme for parameter inference and model selection in dynamical systems. Journal of the Royal Society Interface. 2009; 6(31):187–202. https://doi.org/10.1098/rsif.2008.0172
13. Gilks WR, Richardson S, Spiegelhalter D. Markov Chain Monte Carlo in Practice Chapman and Hall/CRC; 1996
14. Cappé O, Godsill SJ, Moulines E. An overview of existing methods and recent advances in sequential Monte Carlo. Proceedings of the IEEE. 2007; 95(5):899–924. https://doi.org/10.1109/JPROC.2007.893250
15. Robert CP, Casella G. Monte Carlo Statistical Methods. Springer; 2004.
16. Mariño Ip, Míguez J. A nonlinear population Monte Carlo scheme for Bayesian parameter estimation in a stochastic intercellular network model. Proceedings of the IEEE 6th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing. 2015;497–500.
17. West M, Harrison J. Bayesian Forecasting, 2nd ed. New York: Springer-Verlag; 1996.
18. Chopin N, Jacob PE, Papaspiliopoulos O. SMC2: an efficient algorithm for sequential analysis of state space models. Journal of the Royal Statistical Society: Series B (Statistical Methodology). 2012.
19. Maroulas V, Stinis P. Improved particle filters for multi-target tracking. Journal of Computational Physics. 2012; 231(2):602–611. https://doi.org/10.1016/j.jcp.2011.09.023
20. Andrieu C, Roberts GO. The pseudo-marginal approach for efficient Monte Carlo computations. Annals of Statistics. 2009; 37:697–725. https://doi.org/10.1214/07-AOS574
21. Gordon N, Salmond D, Smith AFM. Novel Approach to Nonlinear and Non-Gaussian Bayesian State Estimation. IEE Proceedings-F. 1993; 140(2):107–113.
22. Doucet A, de Freitas N, Gordon N. An introduction to sequential Monte Carlo methods. In: Doucet A, de Freitas N, Gordon N, editors. Sequential Monte Carlo Methods in Practice. Springer; 2001. p. 4–14.
23. Del Moral P. Feynman-Kac Formulae: Genealogical and Interacting Particle Systems with Applications. Springer; 2004.
24. Cappé O, Gullin A, Marin JM, Robert CP. Population Monte Carlo. Journal of Computational and Graphical Statistics. 2004; 13(4):907–929. https://doi.org/10.1198/106186004X12803
25. Doucet A, Godsill S, Andrieu C. On sequential Monte Carlo Sampling methods for Bayesian filtering. Statistics and Computing. 2000; 10(3):197–208. https://doi.org/10.1023/A:1008935410038
26. Míguez J, Mariño IP, Vázquez MA. Analysis of a nonlinear importance sampling scheme for Bayesian parameter estimation in state-space models arXiv:1702.03146.
27. Beaumont MA, Zhang W, Balding DJ. Approximate Bayesian computation in population genetics Genetics Soc America. 2002; 162(4):2025–2035.