Parameter Estimation Methods for Chaotic Intercellular Networks

Inés P. Mariño[1]*, Ekkehard Ullner[2], Alexey Zaikin[3,4]

1 Departamento de Física, Universidad Rey Juan Carlos, Móstoles, Madrid, Spain, 2 Department of Physics, Institute for Complex Systems and Mathematical Biology and Institute of Medical Sciences, University of Aberdeen, Aberdeen, United Kingdom, 3 Institute for Women Health and Department of Mathematics, University College London, London, United Kingdom, 4 Department of Mathematics, King Abdulaziz University, Jeddah, Saudi Arabia

Abstract

We have investigated simulation-based techniques for parameter estimation in chaotic intercellular networks. The proposed methodology combines a synchronization–based framework for parameter estimation in coupled chaotic systems with some state–of–the–art computational inference methods borrowed from the field of computational statistics. The first method is a stochastic optimization algorithm, known as accelerated random search method, and the other two techniques are based on approximate Bayesian computation. The latter is a general methodology for non–parametric inference that can be applied to practically any system of interest. The first method based on approximate Bayesian computation is a Markov Chain Monte Carlo scheme that generates a series of random parameter realizations for which a low synchronization error is guaranteed. We show that accurate parameter estimates can be obtained by averaging over these realizations. The second ABC–based technique is a Sequential Monte Carlo scheme. The algorithm generates a sequence of ''populations'', i.e., sets of randomly generated parameter values, where the members of a certain population attain a synchronization error that is lesser than the error attained by members of the previous population. Again, we show that accurate estimates can be obtained by averaging over the parameter values in the last population of the sequence. We have analysed how effective these methods are from a computational perspective. For the numerical simulations we have considered a network that consists of two modified repressilators with identical parameters, coupled by the fast diffusion of the autoinducer across the cell membranes.

Introduction

Most dynamical systems studied in the physical, biological and social sciences that exhibit a rich dynamical behavior can be modeled by sets of nonlinear differential equations. These mathematical models are a useful tool to predict complex behaviors using numerical simulations. However, for the vast majority of systems, and particularly for biological systems, we lack a reliable description of the parameters of the model. In this paper we are interested in parameter estimation for coupled intercellular networks displaying chaotic behavior, since models of spontaneously active neural circuits typically exhibit chaotic dynamics (for example, spiking models of spontaneous activity in cortical circuits [1–3] and the analogous spontaneously active firing-rate model networks [4,5]).

The problem of parameter estimation can be tackled in different ways, e.g., using multiple shooting methods [6–8] or some statistical procedures based on time discretizations and other approximations [9–13]. These methods involve the solution of high-dimensional minimization problems, since not only the unknown parameters but also the initial values of the trajectory segments between the sampling times need to be estimated [7,14].

This is specially difficult when working with chaotic systems since very complicated error landscapes with many local minima can appear. In particular, notice that chaotic systems have an exponential sensitivity to initial conditions, that is, completely different trajectories can be obtained for identical parameter values and very similar initial conditions of the system variables.

On the other hand, several authors have suggested to take advantage of synchronization techniques for coupled chaotic systems and turn them into accurate parameter estimation methods [14–29]. There is a variety of techniques that rely on the synchronization properties of chaotic systems in order to tackle the parameter estimation problem. For example, some authors have proposed to handle the parameters as additional variables whose dynamics is described by tailored differential equations designed to have a fixed point at the true parameter values [14–20]. Adaptive estimation techniques relying on the time discretization of the state trajectories [22,23,30] and Monte Carlo methods [29,31], including particle filters [32,33], have also been investigated. All these techniques address the estimation of the system parameters independently of the initial conditions of the dynamic variables.

In this paper we investigate techniques that combine a synchronization–based framework for parameter estimation in coupled chaotic systems with some state–of–the–art computational inference methods borrowed from the recent literature in computational statistics. In particular, we describe estimation methods based on the accelerated random search (ARS) optimization algorithm [29,31,34] and two approximate Bayesian computation (ABC) [35–39] schemes. ABC is a general methodology for non–parametric inference that can be applied to practically any system of interest. We investigate two ABC–based methods. The first one is a Markov Chain Monte Carlo (MCMC) scheme [37] that generates a series of random parameter realizations for which a low synchronization error is guaranteed. We show that accurate parameter estimates can be obtained by averaging over these realizations. The second ABC scheme is termed Sequential Monte Carlo (SMC) ABC [38]. It generates a sequence of "populations", i.e., sets of randomly generated parameter values, where the members of the $n-th$ population attain a synchronization error that is lesser than the error attained by members of the $(n-1)-th$ population. Again, we show how very accurate estimates can be obtained by averaging over the parameter values in the last population of the sequence. For the numerical simulations we consider a network of two coupled repressilators, since the repressilator is a paradigmatic gene regulatory system.

Methods

Structure of the Model

The repressilator is a prototype of a synthetic genetic clock built by three genes and the protein product of each gene represses the expression of another in a cyclic manner [40]. It can be constructed experimentally and produce near harmonic oscilla-

tions in protein levels. In the original repressilator design [40], 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. (See left-hand module in Fig. 1 of Ref. [41]).

Cell-to-cell communication was introduced to the repressilator by García-Ojalvo and coworkers [42] by introducing an additional feedback loop to the network scheme that is based on the quorum sensing mechanism. The additional genetic module, which might be placed on a separate plasmid involves two other proteins [42–44]: LuxI, which produces a small autoinducer (AI) molecule that can diffuse through the cell membrane, and *LuxR*, which responds to the autoinducer by activating transcription of a second copy of the repressilator gene *lacI*. The additional quorum sensing feedback loop can be connected to the basic repressilator in such a way that it reinforces the oscillations of the repressilator or competes with the overall negative feedback of the basic repressilator. The first one leads to phase attractive coupling for a robust synchronised oscillation [42] whereas the latter one evokes phase-repulsive influence [45–47], which is the key to multi-stability and a very rich dynamics including chaotic oscillations [41,48,49]. Placing the gene *luxI* under inhibitory control of the repressilator protein TetR (see Fig. 1 bottom in Ref. [41]) leads to the desired repressive and phase-repulsive coupling. We term this system modified repressilator. Phase repulsive coupling is common in several biological systems, e.g. in neural activity in the brain of songbirds [50], in the respiratory system [51], in the jamming avoidance response in electrical fish [52] and in the morphogenesis in Hydra regeneration and animal coat pattern formation [53].

In particular, in this work we are going to consider two modified repressilators with identical parameters, coupled by the fast diffusion of the autoinducer (AI) across the cell membranes. The
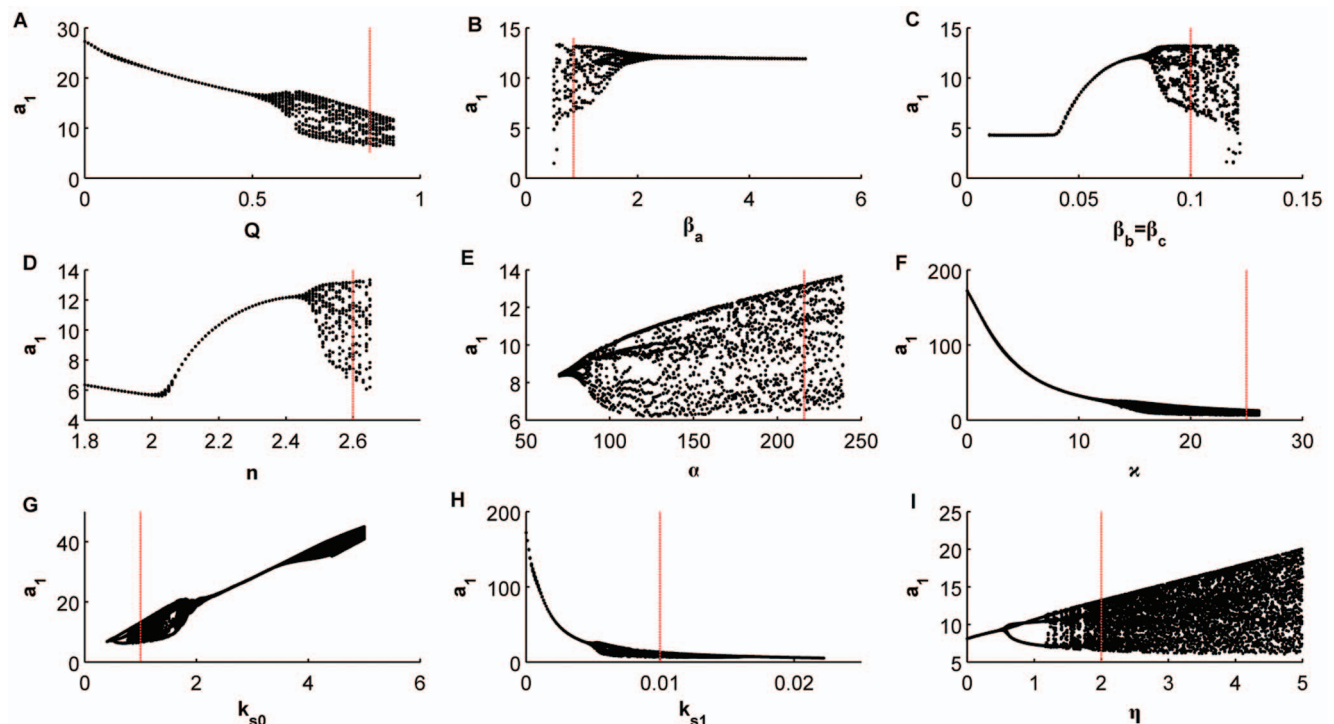

Figure 1. Bifurcation diagrams versus the different control parameters. (A) Q, (B) β_a, (C) β_b, (D) n, (E) α, (F) κ, (G) k_{s0}, (H) k_{s1}, (I) η.
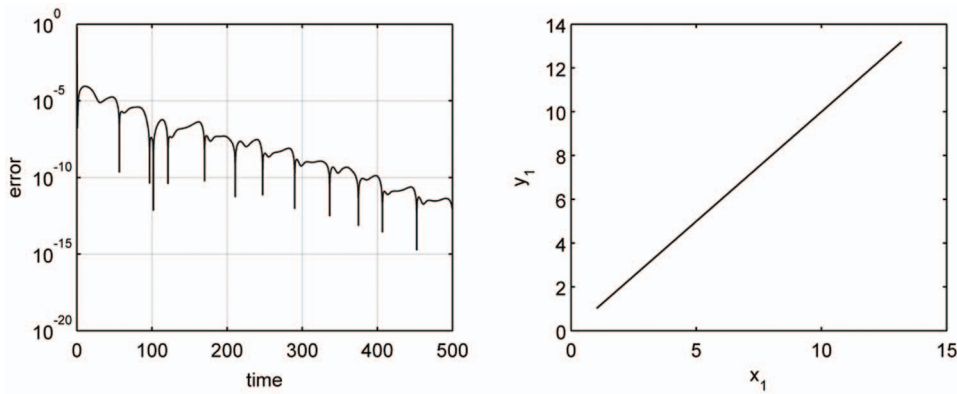doi:10.1371/journal.pone.0079892.g001

Figure 2. Synchronization between the primary and secondary systems. (left) Synchronization error, $|\bar{a}_1 - a_1|$, between both systems for $D = 20$. (right) The first variable of the secondary system ($y_1 = \bar{a}_1$) versus the first variable of the primary system ($x_1 = a_1$).
doi:10.1371/journal.pone.0079892.g002

mRNA dynamics is described by the following Hill-type kinetics with Hill coefficient n:

$$\dot{a}_i = -a_i + \frac{\alpha}{1 + C_i^n}, \tag{1}$$

$$\dot{b}_i = -b_i + \frac{\alpha}{1 + A_i^n}, \tag{2}$$

$$\dot{c}_i = -c_i + \frac{\alpha}{1 + B_i^n} + \frac{\kappa S_i}{1 + S_i}, \tag{3}$$

where the subscript $i = 1, 2$ specifies the cell, and a_i, b_i, and c_i represent the concentrations of mRNA molecules transcribed from the genes of *tetR*, *cI*, and *lacI*, respectively. The parameter α is the dimensionless transcription rate in the absence of a repressor. The parameter κ is the maximum transcription rate of the LuxR promoter.

The protein dynamics is given by

$$\dot{A}_i = \beta_a(a_i - A_i), \tag{4}$$

$$\dot{B}_i = \beta_b(b_i - B_i), \tag{5}$$

$$\dot{C}_i = \beta_c(c_i - C_i) \tag{6}$$

where variables A_i, B_i, and C_i denote the concentration of the proteins TetR, CI, and LacI, respectively. The dynamics of the proteins is linked to the amount of the responsible mRNA, and the parameters $\beta_{a,b,c}$ describe the ratio between mRNA and the protein lifetimes (inverse degradation rates). In what follows, we are going to assume $\beta_b = \beta_c$. Thus, we can consider it as a single parameter. 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 [42].

The third term 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, assuming equal lifetimes of the two proteins and given that their production is controlled by the same protein (TetR). Hence, the synthesis of the AI S_i can be considered to be controlled by the concentration B_i of the protein CI. Taking also into account the intracellular degradation of the AI and its diffusion toward or from the intercellular space, the dynamics of S_i is given by

$$\dot{S}_i = -k_{s0}S_i + k_{s1}B_i - \eta(S_i - S_e), \tag{7}$$

where the diffusion coefficient η depends on the permeability of the membrane to the AI. Because of the fast diffusion of the extracellular AI (S_e) compared to the repressilator period, we can apply the quasi-steady-state approximation to the dynamics of the external AI [42], which leads to

$$S_e = Q\bar{S} \equiv Q\frac{1}{N}\sum_{i=1}^{N} S_i. \tag{8}$$

The parameter Q is defined as

$$Q = (\delta N / V_{ext}) / (k_{se} + \delta N / V_{ext}) \tag{9}$$

where $N = 2$ 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.

We achieve chaotic behavior for the following parameter values [41]: $Q = 0.85$, $n = 2.6$, $\alpha = 216$, $\beta_a = 0.85$, $\beta_b = 0.1$, $\eta = 2$, $\kappa = 25$, $k_{s0} = 1$, and $k_{s1} = 0.01$. In particular, these are the values we are considering throughout this manuscript in order to assess the parameter estimation algorithms.

To see how the system behaves around these values we have plotted some bifurcation diagrams taking in each one a different parameter as a control parameter, whereas the rest of the parameters remain constant on the values mentioned above. Figure 1(A–I) represents the bifurcation diagrams versus the different control parameters. In particular, in each plot we are representing all maxima of the variable a_1 for some fixed initial

condition as a function of the corresponding control parameter. We can observe how for some values of the control parameters the behavior changes from periodic to chaotic and vice versa. The vertical red dashed lines correspond to the values we are considering to assess the parameter estimation algorithms. Notice how for these values the system can display chaotic behavior.

Problem Statement

We first introduce the notation to be used in the description of the parameter estimation methodologies. Let

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{p}) \tag{10}$$

represent the chaotic intercellular network (consisting of two coupled modified repressilators), with state variables

$$
\begin{aligned}
\mathbf{x} &= (x_1, \ldots, x_{14}) \\
&= (a_1, b_1, c_1, A_1, B_1, C_1, S_1, a_2, b_2, c_2, A_2, B_2, C_2, S_2)
\end{aligned} \tag{11}
$$

and unknown parameters

$$
\begin{aligned}
\mathbf{p} &= (p_1, \ldots, p_9) \\
&= (Q, \beta_a, n, \alpha, \beta_b, \kappa, k_{s0}, k_{s1}, \eta).
\end{aligned} \tag{12}
$$

The vector–valued function $\mathbf{f}$ can be explicitly written down by comparing Eq. (10) with Eqs. (1)-(7). In the sequel we refer to the system of Eq. (10) as *primary* system.

Since $\mathbf{f}$ in Eq. (10) is known, we can build a model with identical functional form but adjustable parameters and a coupling term, termed *secondary* system in the sequel, as

$$\dot{\mathbf{y}} = \mathbf{f}(\mathbf{y}, \mathbf{q}) + D\mathbf{s}_{i,j}(\mathbf{x} - \mathbf{y}), \tag{13}$$

where

$$
\begin{aligned}
\mathbf{y} &= (y_1, \ldots, y_{14}) \\
&= (\bar{a}_1, \bar{b}_1, \bar{c}_1, \bar{A}_1, \bar{B}_1, \bar{C}_1, \bar{S}_1, \bar{a}_2, \bar{b}_2, \bar{c}_2, \bar{A}_2, \bar{B}_2, \bar{C}_2, \bar{S}_2)
\end{aligned} \tag{14}
$$

is the time-varying vector that contains the model state variables,

$$
\begin{aligned}
\mathbf{q} &= (q_1, \ldots, q_9) \\
&= (\bar{Q}, \bar{\beta}_a, \bar{n}, \bar{\alpha}, \bar{\beta}_b, \bar{\kappa}, \bar{k}_{s0}, \bar{k}_{s1}, \bar{\eta})
\end{aligned} \tag{15}
$$

is the adjustable parameter vector, D is a coupling coefficient and $\mathbf{s}_{i,j} : \mathbb{R}^{14} \to \mathbb{R}^{14}$ is a vector function that selects the i-th and the j-th element of its argument, i.e.,

$$\mathbf{s}_{i,j}(\mathbf{x} - \mathbf{y}) = (0, \ldots, 0, x_i - y_i, 0, \ldots, 0, x_j - y_j, 0, \ldots, 0). \tag{16}$$

The definition of the latter function implies that coupling is performed only through two scalar time series, x_i and x_j, from the primary system. In particular, the coupling scheme we have chosen for the simulation setup in this paper is

$$\dot{\bar{a}}_i = -\bar{a}_i + \frac{\bar{\alpha}}{1 + \bar{C}_i^{\bar{n}}} + D(a_i - \bar{a}_i), \tag{17}$$

where $i = 1, 2$ denotes the cell number (i.e., the repressilator index). Thus, the coupling only appears in two of the fourteen differential equations we have in the secondary system.

Since we assume identical functional form for the primary and secondary systems, when the secondary parameter vector, $\mathbf{q}$, coincides with the primary parameter vector, $\mathbf{p}$, the state variables $\mathbf{y}$ synchronize with $\mathbf{x}$ for $D > D_c$, where D_c is a coupling threshold [29]. On the contrary, if $\mathbf{q} \neq \mathbf{p}$ then identical synchronization cannot occur. However, the difference $\|\mathbf{y} - \mathbf{x}\|$ is expected to be small whenever the difference of the two parameter vectors is small and D is sufficiently large. Therefore, the objective of a parameter estimation method based on synchronization is to compute a value $\hat{\mathbf{q}}$ such that $\|\mathbf{y} - \mathbf{x}\| \approx 0$ over time, since the latter implies $\hat{\mathbf{q}} \approx \mathbf{p}$.

Figure 2(left) shows how the synchronization error $|\bar{a}_1 - a_1|$ evolves in time when considering $D = 20$ and identical parameters for the primary and the secondary systems. We can see how this error can be very small, less that 10^{-10}. We are going to fix $D = 20$ in all simulations throughout the paper. In Fig. 2(right) we are representing one variable of the secondary system ($y_1 = \bar{a}_1$) versus the same variable of the primary system ($x_1 = a_1$) and observing the typical straight line, characteristic of the synchronization phenomenon.

It has been verified by means of numerical simulations that in order to obtain synchronization at least two coupling variables are needed (excluding S_1 or S_2), one from each repressilator in the primary system. Therefore, other combinations different from (a_1, a_2) are also possible, as for example (b_1, A_2). Coupling via a single variable is not sufficient to guarantee that the secondary system synchronizes with the primary one.

Results

Parameter Estimation Methods

Here we introduce three parameter estimation methods that combine the synchronization–based framework described above with state–of–the–art computational methods. In particular, we present results for the joint estimation of four parameters,

$$\mathbf{p} = (p_1, p_2, p_3, p_4) = (Q, \beta_a, n, \alpha), \tag{18}$$

in the coupled modified repressilator network. In all simulations shown in this manuscript we have assumed that just two scalar signals from the primary system are observed, namely the coupling variables (a_1, a_2), and we have numerically integrated the secondary system using a second order Runge-Kutta method with a time step $T_s = 0.005$ time units (t.u.).

Accelerated Random Search

We first focus on a parameter estimation method for chaotic intercellular networks that takes advantage of chaos synchronization and is based on an efficient Monte Carlo optimization procedure, known as accelerated random search (ARS) method [29,31]. In particular, the method we are going to describe consists in a variation of the technique proposed in [31]. Assuming that the variables (a_1, a_2) are observed during the time interval $(0, T_o)$, the value of the parameters in the secondary system can be calculated as the solution to the optimization problem

$$\hat{\mathbf{q}} = \arg \min_{\mathbf{q}} \{J(\mathbf{q})\}, \tag{19}$$

where $\mathbf{q} = (q_1, \ldots, q_4) = (\bar{Q}, \bar{\beta}_a, \bar{n}, \bar{\alpha})$ and the cost function

$$J(\mathbf{q}) = \frac{1}{T_o} \int_0^{T_o} \left(|\bar{a}_1 - a_1|^2 + |\bar{a}_2 - a_2|^2 \right) dt$$

$$\approx \frac{1}{N} \sum_{n=0}^{N-1} \left((\bar{a}_1(nT_s) - a_1(nT_s))^2 + (\bar{a}_2(nT_s) - a_2(nT_s))^2 \right) \tag{20}$$

is a quantitative representation of the synchronization error between the primary and the secondary systems. Notice that $N = T_o/T_s$, where $T_o = 10^5$ t.u.

The method can be outlined as follows:

1. Initialization. Choose:

(a) initial parameter values

$$\hat{\mathbf{q}}(0) = (\hat{q}_1(0), \hat{q}_2(0), ..., \hat{q}_m(0)), \tag{21}$$

where m represents the number of parameters we want to estimate;

(b) maximum and minumum "search variances" for each parameter, $\sigma_{j,max}^2$ and $\sigma_{j,min}^2$, respectively, where $j = 1, ..., m$;

(c) a "contraction factor" for each parameter, $c_j > 1$, $j = 1, ..., m$.

Set $\sigma_j^2(1) = \sigma_{j,max}^2$ for $j = 1, ..., m$ and, using the initial parameter vector, $\hat{\mathbf{q}}(0)$, evaluate the associated cost $J(\hat{\mathbf{q}}(0))$.

2. *Iterative step.* Let $N(x|\mu,\sigma^2)$ denote the Gaussian probability distribution of the variable x with mean μ and variance σ^2. Assume we have computed a vector of parameter estimates, $\hat{\mathbf{q}}(n-1) = (\hat{q}_1(n-1), ..., \hat{q}_m(n-1))$, with cost $J(\hat{\mathbf{q}}(n-1))$.

Choose an index $j \in \{1, ..., m\}$, define

$$J_{j,n-1}(\tilde{q}) = J(\hat{q}_1(n-1), ..., \hat{q}_{j-1}(n-1),$$
$$\tilde{q}, \hat{q}_{j+1}(n-1), ..., \hat{q}_m(n-1)) \tag{22}$$

and set $\sigma_j^2(0) = \sigma_{j,max}^2$ and $\tilde{q}_j(0) = \hat{q}_j(n-1)$. Choose an integer L (the number of iterations to be performed over each parameter) and carry out the following computations:

For $l = 1, ..., L$:

The method can be outlined as follows:

1. Initialization. Choose:

(a) initial parameter values

$$\hat{\mathbf{q}}(0) = (\hat{q}_1(0), \hat{q}_2(0), ..., \hat{q}_m(0)), \tag{21}$$

where m represents the number of parameters we want to estimate;

(b) maximum and minumum "search variances" for each parameter, $\sigma_{j,max}^2$ and $\sigma_{j,min}^2$, respectively, where $j = 1, ..., m$;

(c) a "contraction factor" for each parameter, $c_j > 1$, $j = 1, ..., m$.

Set $\sigma_j^2(1) = \sigma_{j,max}^2$ for $j = 1, ..., m$ and, using the initial parameter vector, $\hat{\mathbf{q}}(0)$, evaluate the associated cost $J(\hat{\mathbf{q}}(0))$.

2. *Iterative step.* Let $N(x|\mu,\sigma^2)$ denote the Gaussian probability distribution of the variable x with mean μ and variance σ^2. Assume we have computed a vector of parameter estimates, $\hat{\mathbf{q}}(n-1) = (\hat{q}_1(n-1), ..., \hat{q}_m(n-1))$, with cost $J(\hat{\mathbf{q}}(n-1))$.

2. Choose an index $j \in \{1, ..., m\}$, define

$$J_{j,n-1}(\tilde{q}) =$$
$$J(\hat{q}_1(n-1), ..., \hat{q}_{j-1}(n-1), \tilde{q}, \hat{q}_{j+1}(n-1), ..., \hat{q}_m(n-1)) \tag{22}$$

and set $\sigma_j^2(0) = \sigma_{j,max}^2$ and $\tilde{q}_j(0) = \hat{q}_j(n-1)$. Choose an integer L (the number of iterations to be performed over each parameter) and carry out the following computations:

For $l = 1, ..., L$:

(a) Draw $\bar{q} \sim N\left(q|\tilde{q}_j(l-1), \sigma_j^2(l-1)\right)$.

(b) Compute $J_{j,n-1}(\bar{q})$.

(c) If $J_{j,n-1}(\bar{q}) < J_{j,n-1}(\tilde{q}_j(l-1))$ then

$$\tilde{q}_j(l) = \bar{q},$$

$$\sigma_j^2(l) = \sigma_{j,max}^2 \tag{23}$$

else

$$\tilde{q}_j(l) = \hat{q}_j(l-1),$$

$$\sigma_j^2(l) = \sigma_j^2(l-1)/c_j \tag{24}$$

(d) If $\sigma_j^2(l) < \sigma_{j,min}^2$, then

$$\sigma_j^2(l) = \sigma_{j,max}^2 \tag{25}$$

Once the loop is completed, set $\hat{q}_j(n) = \tilde{q}_j(L)$ and $\hat{q}_k(n) = \tilde{q}_k(n-1)$ for every $k \neq j$.

Let $j = (j+1) \mod m$, set $\sigma_j^2(0) = \sigma_{j,max}^2$ and perform the loop over (a)-(d) again.

The algorithm can be stopped when the synchronization error $J(\hat{q}(n))$ reaches a certain threshold or after a prescribed number of steps n (e.g., when $n = n_o$ for some sufficiently large n_o). Notice that the total number of iterations is $n_o L$ and the time evolution of the secondary system state, $\mathbf{y}$, has to be integrated at each iteration, since a new candidate parameter vector is drawn each time.

We have carried out numerical simulations where this Monte Carlo optimization algorithm has been iterated a total of 8×10^4 times, with the number of consecutive iterations for each parameter (iterations per loop) set to $L = 100$, and the secondary system has been integrated for each iteration during $T_o = 1 \times 10^5$ time units.

The initial values of the parameters are sampled from a uniform distribution, namely

$$\hat{Q}(0), \hat{\beta}_a(0), \hat{n}(0) \sim U(0,2);$$

$$\hat{\alpha}(0) \sim U(100,500), \tag{26}$$

the maximum search variances for each parameter are

$$\sigma_{Q,max}^2 = \sigma_{\beta_a,max}^2 = \sigma_{n,max}^2 = 2,$$

$$\sigma_{\alpha,max}^2 = 500, \tag{27}$$

the minimum search variances are

$$\sigma_{Q,min}^2 = \sigma_{\beta_a,min}^2 = \sigma_{n,min}^2 = 0.0001,$$

$$\sigma_{\alpha,min}^2 = 0.01, \tag{28}$$

and the contraction factors are

$$c_Q = c_{\beta_a} = c_n = 2.5,$$

$$c_\alpha = 25. \tag{29}$$

The normalized quadratic error for each parameter q_i, $i = 1, \ldots, 4$, is defined as,

$$E_{p_i}(n) = ((p_i - \hat{q}_i(n))/p_i)^2. \tag{30}$$

Figure 3(A–D) shows the normalized quadratic errors for each parameter as a function of the number of iterations (nL). We can see how after 3×10^4 iterations all errors are very low for the estimated parameters. The values of the normalized quadratic errors in the 30,000-th iteration are:

$$E_Q = 2.7 \times 10^{-8}, E_{\beta_a} = 8.7 \times 10^{-7},$$

$$E_n = 1.6 \times 10^{-8}, E_\alpha = 1.1 \times 10^{-8}. \tag{31}$$

Approximate Bayesian Computation

In Bayesian estimation theory the parameters are modeled as random variables, rather than unknown but deterministic numbers. Consequently, ABC-based methods aim at approximating the probability distribution of the parameters, vector **q**, conditional on the observations from the primary system. The technique, to be described next, is nonparametric, i.e., the distribution of **q** is approximated by a set of random samples.

ABC methods have been conceived with the aim of inferring posterior distributions without having to compute likelihood functions [35–39]. The calculation of the likelihood is replaced by a comparison between the observed and the simulated data. In the setup of this paper, the comparison is carried out between the data from the primary system (the observations) and the data generated by the secondary system (the model with adjustable parameters). The comparison between these data represents, in our case, a measure of the synchronization error between these two systems.

Let π denote the a priori probability density function (pdf) of the random parameter vector **q**, let $f(\mathbf{y}|\mathbf{q})$ denote the probability distribution of the data **y** generated by the secondary system conditional on the realization of **q** and let $d(\mathbf{x},\mathbf{y})$ be a distance function that measures the synchronization error by comparing the observed time series **x** from the primary system and the generated time series **y** from the secondary one. Since the system of interest is deterministic, $f(\mathbf{y}|\mathbf{q})$ collapses into a Dirac delta measure when **q** is given. In the ABC framework, though, we are not interested in evaluating f but rather in generating **y** using the model conditional on **q**. This amounts to integrating the secondary system with a given realization of the adjustable parameters **q**.

The simplest ABC algorithm is the *ABC rejection sampler* [35]. For a given synchronization error threshold (often termed tolerance) > 0, the algorithm can be described as follows.
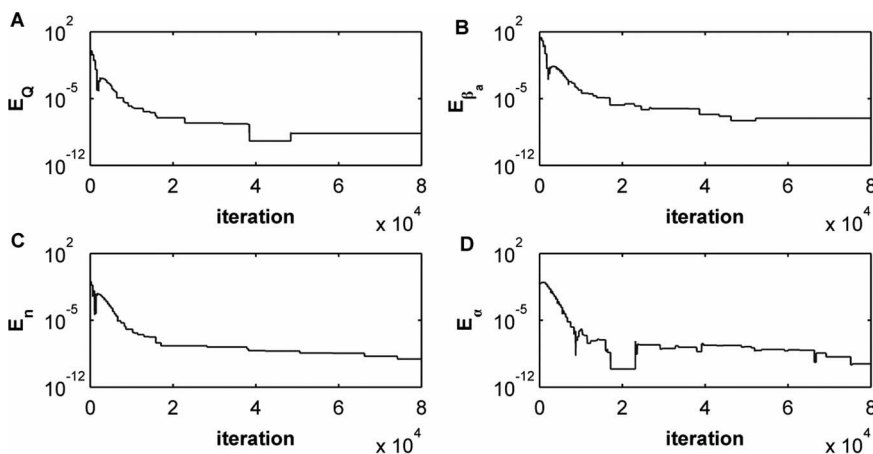

Figure 3. Normalized quadratic errors for each parameter as a function of the number of iterations for the ARS algorithm. (A) Q, (B) β_a, (C) n and (D) α.
doi:10.1371/journal.pone.0079892.g003

1. Sample $\tilde{\mathbf{q}}$ from $\pi(\mathbf{q})$.

2. Simulate a dataset $\tilde{\mathbf{y}}$ from the conditional probability distribution $f(\mathbf{y}|\tilde{\mathbf{q}})$.

3. If the distance function (synchronization error) is $d(\mathbf{x},\tilde{\mathbf{y}}) \leq \epsilon$, then accept $\tilde{\mathbf{q}}$, otherwise reject it.

4. Return to the first step.

The output of an ABC algorithm is a population of parameter values randomly drawn from the distribution $\pi(\mathbf{q}|d(\mathbf{x},\tilde{\mathbf{y}}) \leq)$, i.e., a distribution with density proportional to the prior but restricted to the set of values of $\mathbf{q}$ for which the synchronization error is at most .

The disadvantage of the ABC rejection sampler is that the acceptance rate is very low when the set of values of $\mathbf{q}$ for which the synchronization error is less than turns out to be relatively small. Thus, instead of implementing this method we have decided to implement two more sophisticated ABC algorithms. The first one is a Markov chain Monte Carlo (ABC MCMC) technique and the second one is a sequential Monte Carlo (ABC SMC) method.

Approximate Bayesian Computation Markov Chain Monte Carlo

The Approximate Bayesian Computation Markov Chain Monte Carlo (ABC MCMC) algorithm is a Metropolis-Hastings [37] MCMC method that incorporates one additional test to ensure that all parameters in the chain yield a synchronization error below the threshold . It can be outlined as follows.

1. Initialize $\mathbf{q}(i)$, $i=0$.

2. Generate $\tilde{\mathbf{q}}$ according to a proposal distribution $g(\mathbf{q}|\mathbf{q}(i))$

3. Simulate a dataset $\tilde{\mathbf{y}}$ from the conditional probability distribution $f(\mathbf{y}|\tilde{\mathbf{q}})$.

4. If the distance function (synchronization error) is $d(\mathbf{x},\tilde{\mathbf{y}}) \leq \epsilon$, go to the following step, otherwise set $\mathbf{q}(i+1)=\mathbf{q}(i)$ and go to step 6.

5. Set $\mathbf{q}(i+1)=\tilde{\mathbf{q}}$ with probability

$$\xi = \min\left(1, \frac{\pi(\tilde{\mathbf{q}})g(\mathbf{q}(i)|\tilde{\mathbf{q}})}{\pi(\mathbf{q}(i))g(\tilde{\mathbf{q}}|\mathbf{q}(i))}\right) \qquad (32)$$

and $\mathbf{q}(i+1)=\mathbf{q}(i)$ with probability $1-\xi$.

1. Set $i=i+1$, go to step 2.

The outcome of this algorithm is a Markov chain with the stationary distribution $\pi(\mathbf{q}|d(\mathbf{x},\tilde{\mathbf{y}}) \leq \epsilon)$ [37]. The parameters are assumed independent a priori, hence

$$\pi(\mathbf{q}) = \prod_{j=1,\ldots,m} \pi(q_i), \qquad (33)$$

and we also choose independent proposals for simplicity. In particular, the prior distributions for each parameter have been chosen as

$$\pi(Q)=\pi(\beta_a)=\pi(n)=U(0,5)$$

$$\pi(\alpha)=U(0,500), \qquad (34)$$

and the proposal distribution for each parameter is a Gaussian distribution centered at the previous value of the corresponding parameter and with a fixed standard deviation, different for each parameter. In particular,

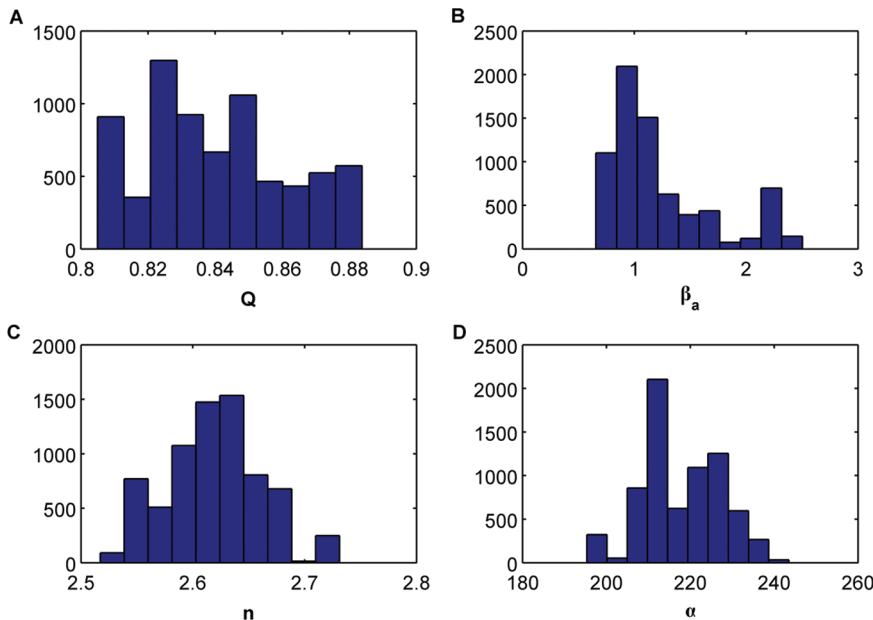$$\sigma(Q)=\sigma(\beta_a)=\sigma(n)=1$$

Figure 4. Histograms of the approximate marginal posterior distributions for each parameter for the ABC MCMC algorithm when considering $\epsilon = 0.001$. (A) Q, (B) β_a, (C) n and (D) α.
doi:10.1371/journal.pone.0079892.g004

$$\sigma(\alpha) = 100 \qquad (35)$$

and

$$g(\mathbf{q}|\tilde{\mathbf{q}}) = \prod_{j=1}^{m} N(q_j|q'_j, \sigma(q_j)). \qquad (36)$$

Since the proposal distribution is symmetric, $g(\mathbf{q}_i|\tilde{\mathbf{q}}) = g(\tilde{\mathbf{q}}|\mathbf{q}_i)$, and the prior is uniform, the acceptance probability in Eq.(32) is $\xi = 1$.

The distance function (synchronization error) has been chosen as

$$d(\mathbf{y},\mathbf{x}) = \frac{1}{T_o} \int_0^{T_o} \left(|\bar{a}_1 - a_1|^2 + |\bar{a}_2 - a_2|^2 \right) dt$$
$$(37)$$
$$\approx \frac{1}{N} \sum_{n=0}^{N-1} \left((\bar{a}_1(nT_s) - a_1(nT_s))^2 + (\bar{a}_2(nT_s) - a_2(nT_s))^2 \right)$$

where $T_o = 10^5$ t.u. and $NT_s = T_0$. This expression is equivalent to the cost function in the ARS method. The tolerance (threshold for the synchronization error) has been chosen as $\epsilon = 0.001$.

A chain of 5×10^5 samples has been generated, what implies that the secondary system has been integrated 5×10^5 times. The initial point of the chain is selected to ensure that the associated distance is less than 0.05. Figure 4(A–D) shows the histograms of the approximate marginal posterior distributions for each parameter. In order to reduce the strong correlation between consecutive samples in the Markov chain we have subsampled by a factor of 50. We have calculated the mean values of each histogram as well as the normalized quadratic errors according to the following expression

$$E_{p_i} = ((p_i - \bar{q}_i)/p_i)^2, \qquad (38)$$

where $\bar{q}_i$, for $i = 1, \ldots, 4$, represents the mean value of the histogram of the corresponding parameter. The values of the normalized quadratic errors for the estimated parameters are

$$E_Q = 1.3 \times 10^{-4}, E_{\beta_a} = 1.9 \times 10^{-1},$$

$$E_n = 5.9 \times 10^{-5}, E_\alpha = 5.7 \times 10^{-5}. \qquad (39)$$

We can see how three parameters are accurately estimated whereas for one of them, β_a the error is significantly higher compared to (31).

Approximate Bayesian Computation Sequential Monte Carlo

A more sophisticated application of the ABC methodology is the Approximate Bayesian Computation Sequential Monte Carlo algorithm [39,54,55]. In ABC SMC, a number of sampled parameter values (often termed particles), $\{\mathbf{q}^{(1)}, \ldots, \mathbf{q}^{(N)}\}$, drawn from the prior distribution $\pi(\mathbf{q})$, are propagated through a sequence of intermediate distributions $\pi(\mathbf{q}|d(\mathbf{x},\tilde{\mathbf{y}}) \leq_i)$, $i = 1, \ldots, T-1$, until they are converted into samples from the target distribution $\pi(\mathbf{q}|d(\mathbf{x},\tilde{\mathbf{y}}) \leq_T)$. The tolerances are chosen such

that $_1 > \ldots >_T > 0$, thus the empirical distributions gradually evolve towards the target posterior. The ABC SMC algorithm proceeds as follows [39].

1. Initialize $\epsilon_1 > \ldots > \epsilon_T$. Set the population indicator $t = 0$.
2. Set the particle indicator $i = 1$.
3. If $t = 0$, draw $\mathbf{q}^\star$ from the prior $\pi(\mathbf{q})$.

   Else, draw $\mathbf{q}^\star$ from

$$g_t(\mathbf{q}) = \sum_{i=1}^{N} w^{(i)}(t-1) K_t(\mathbf{q}|\mathbf{q}^{(i)}(t-1)) \qquad (40)$$

where $K_t(.|\mathbf{q}')$ is a symmetric kernel centred around $\mathbf{q}'$ and $w^{(1)}(t-1), \ldots, w^{(N)}(t-1)$ are importance weights such that $\sum_{i=1}^{N} w^{(i)}(t-1) = 1$.

4. If $\pi(\mathbf{q}^\star) = 0$, return to step 3.
5. Simulate a candidate dataset $\tilde{\mathbf{y}}\tilde{f}(\mathbf{y}|\mathbf{q}^\star)$.
6. If $d(\mathbf{x},\tilde{\mathbf{y}}) \geq \epsilon_t$ return to step 3.
7. Set $\mathbf{q}^{(i)}(t) = \mathbf{q}^\star$ and calculate the weight,

$$w_t^{(i)} = \begin{cases} 1, if & t = 0 \\ \dfrac{\pi\left(q_t^{(i)}\right)}{\sum\limits_{j=1}^{N} w^{(j)}(t-1) K_t\left(\mathbf{q}^{(i)}(t-1), \mathbf{q}^{(j)}(t)\right)} & t > 0 \end{cases} \qquad (41)$$

8. If $i < N$, set $i = i+1$ and go to step 3.
9. Normalize the weights.
10. If $t < T$, set $t = t+1$ and go to step 2. Otherwise stop.

The prior distributions we have considered for each parameter are $\pi(Q) = \pi(\beta_a) = \pi(n) = U(0,5)$ and $\pi(\alpha) = U(0,500)$, the same as for the ABC MCMC algorithm. The perturbation kernel K_t is Gaussian, namely

$$K_t(\mathbf{q}^\star|\mathbf{q}^{(i)}(t-1)) = \prod_{j=1}^{m} \mathcal{N}(q_j^\star|q_j^{(i)}(t-1), \sigma(q_j)), \qquad (42)$$

with standard deviations $\sigma(Q) = \sigma(\beta_a) = \sigma(n) = 0.1$ and $\sigma(\alpha) = 10$. The distance function (synchronization error) is the same as for the ABC MCMC and ARS algorithms with the same T_o value. To ensure the gradual transition between populations, the ABC SMC algorithm is run for $T = 12$ populations with $= \{1, 0.5, 0.1, 0.05, 0.025, 0.01, 0.005, 0.0025, 0.001, 0.0005, 0.00025, 0.0001\}$ and we have considered $N = 400$ particles per population.

Figure 5(A–D) shows the histograms of the approximate marginal posterior distributions for each parameter where the different weights of the different particles have been taken into account for the representation. We have calculated the mean values of each histogram (that match the actual values) as well as the normalized quadratic errors using Eq. (38), that is, the same expression as for the ABC MCMC algorithm. The values of the normalized quadratic errors for the estimated parameters with the ABC SMC method are
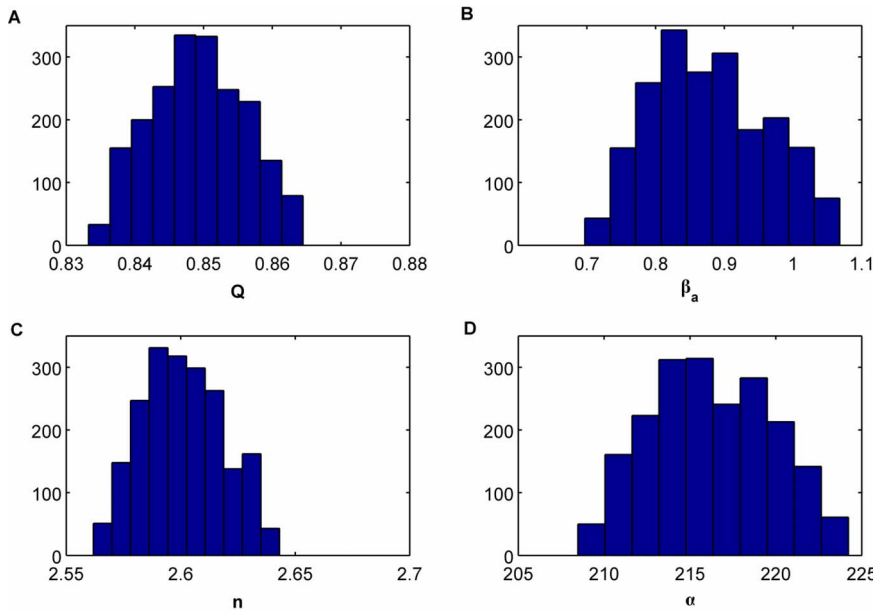
Figure 5. Histograms of the approximate marginal posterior distributions for each parameter for the ABC SMC algorithm when considering $\epsilon = 0.0001$**.** (A) Q, (B) β_a, (C) n and (D) α.
doi:10.1371/journal.pone.0079892.g005

$$E_Q = 5.1 \times 10^{-7}, E_{\beta_a} = 9.4 \times 10^{-4},$$
$$E_n = 2.5 \times 10^{-7}, E_\alpha = 3.2 \times 10^{-7}. \quad (43)$$

Figure 6(A–D) shows the output (i.e. the accepted particles) of the ABC SMC algorithm as scatterplots of some of the two-dimensional parameter combinations, where we have information of different populations in the same plot. As we iterate the algorithm we obtain populations that are more dense around the desired values, as shown in these plots, where the particles from the prior are represented in blue, particles from population 2 in green, particles from population 4 in light blue, population 6 in
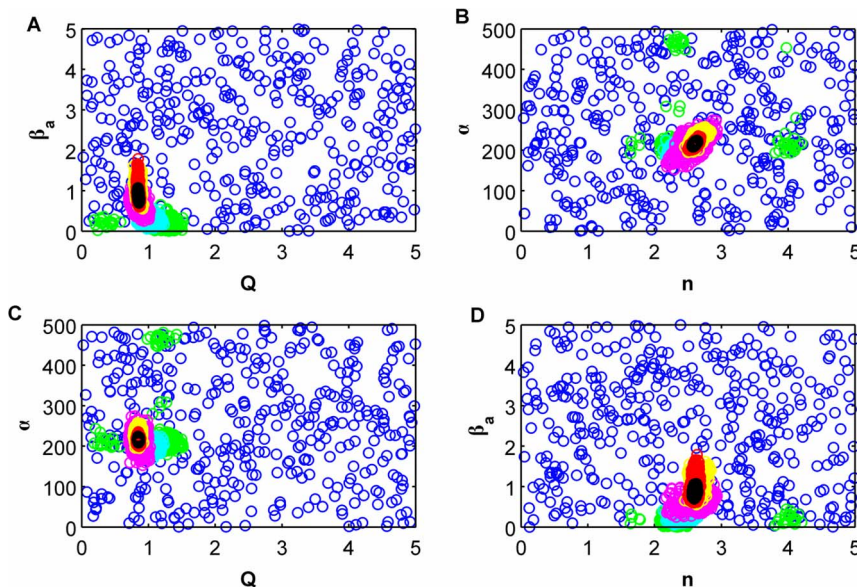

Figure 6. Two-dimensional scatterplots for the accepted particles of the ABC SMC of each population. The particles from the prior are represented in blue, particles from population 2 in green, particles from population 4 in cyan, population 6 in pink, 8 in yellow, 10 in red and particles from the posterior (population 12) in black color.
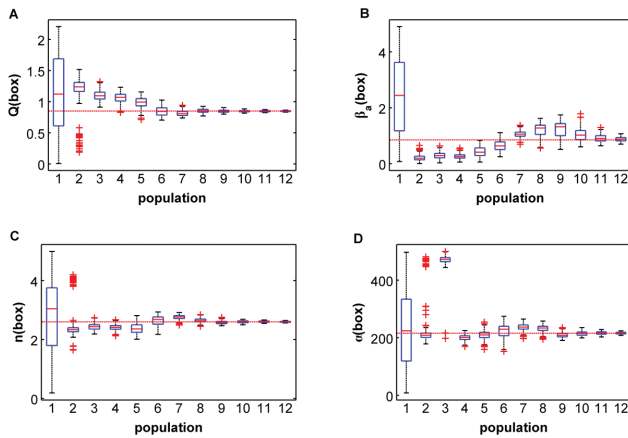doi:10.1371/journal.pone.0079892.g006

Figure 7. Box plots diagrams for the different populations for each parameter. (A) Q, (B) β_a, (C) n and (D) α.
doi:10.1371/journal.pone.0079892.g007

pink, 8 in yellow, 10 in red and particles from the posterior (population 12) in black color. We can see how for the last population the particles are tightly clustered around the desired value.

In order to gain insight of how the parameters are estimated during the evolution of the algorithm, we have represented some box–plot diagrams, one for each parameter to be estimated, as seen in fig. 7(A–D). In each diagram, we have information about the corresponding parameter as a function of the population index. In particular, the central mark of each box is the median of the population, the edges of the box are the 25-th and 75-th percentiles, the whiskers extend to the most extreme data points not considered as outliers, and the outliers are plotted individually using the plus symbols in red. The horizontal lines in red represent the actual values of the parameters we have used in our simulation. We can see from these plots how for a high enough population index the median values of the four parameters perfectly match the actual values.
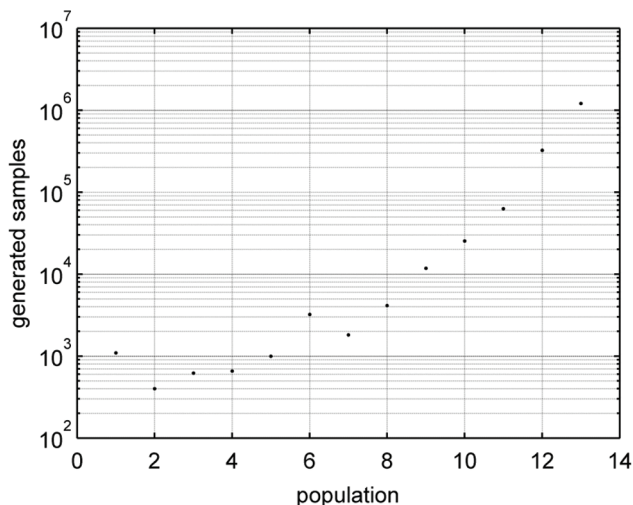

Figure 8. Computational cost for each population using the ABC SMC algorithm.
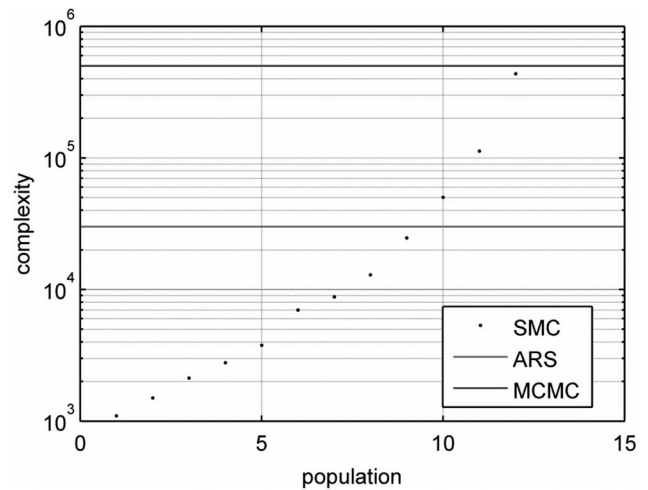doi:10.1371/journal.pone.0079892.g008

Figure 9. Computational complexity measured by the number of random samples generated by the algorithms. The solid blue line is the complexity of the ABC MCMC algorithm (with = 0.001). The solid red line is the complexity of the ARS method. The black dots indicate the complexity of the ABC SMC algorithm, for each population up to the $12-th$ one.
doi:10.1371/journal.pone.0079892.g009

We have also studied the computational cost of the algorithm. In fig. 8 we can see the number of samples or particles we have generated in order to have $N=400$ accepted particles for each population. We can see how the number of particles increases with the population index, being significantly high for the last population index, since it corresponds to a very small value of the synchronization error. Notice that the vertical axis of this figure is in a logarithmic scale.

Comparison of the Methods

Here we compare not only the accuracy but also the computational complexity of all three Monte Carlo methods for the joint estimation of the four parameters, Q, β_a, n and α, of the chaotic intercellular network. To do that, we have calculated for the ABC SMC algorithm the computational load up to each population. Specifically, the computational complexity of generating a sequence of m populations is given by the number of samples or particles that have to be generated before completing the $m-th$ population. Note that the computational load for the $m-th$ population also includes all samples needed to generate the $m-1$ previous populations.

Fig. 9 provides a graphical depiction of the complexity of the three methods, that we have investigated (ARS, ABC MCMC and ABC SMC algorithms). The line in red represents the complexity for the ARS method and the line in blue indicates the complexity for the ABC MCMC technique.

The parameter estimation errors attained with the ARS method are of the same order as the errors of the parameter estimates computed from the $12-th$ population of the ABC SMC algorithm. However, the number of samples generated to run the ARS procedure is $\approx 3\times 10^4$ while the ABC SMC technique demands the generation of $\approx 4\times 10^5$ random samples up to the $12-th$ population. The ABC MCMC method achieves the poorest performance, as it requires the generation of the highest number of samples ($\approx 5\times 10^5$) and produces the largest errors (up to three orders of magnitude worse than the ARS or ABC SMC estimates).

Discussion

We have investigated three computational inference techniques for parameter estimation in a chaotic intercellular network that consists of two coupled modified repressilators. The proposed methodology combines a synchronization–based framework for parameter estimation in coupled chaotic systems with some state–of–the–art computational inference methods borrowed from computational statistics. In particular, we have focussed on an accelerated random search algorithm and two approximate Bayesian computation schemes (ABC MCMC and ABC SMC). The three methods exploit the synchronization property of chaotic systems. Therefore, it is not necessary to estimate the initial conditions of the variables, which is an important advantage from a computational point of view.

We have carried out the numerical study in this paper assuming that only two variables from the primary system can be observed. This is the minimum number of observed variables in order to guarantee the synchronization of the secondary system. If additional variables can be observed it is possible to easily incorporate them into the proposal methodology. For example, if the variables $x_1, \ldots, x_8$ are observed (this is the full state of the first repressilator and the first variable, x_8, of the second repressilator) we can redefine the distance function of Eq. (38) as

$$d(\mathbf{y},\mathbf{x}) \approx \frac{1}{N}\sum_{n=0}^{N-1}\sum_{i=1}^{8}(x_i(nT_s)-y_i(nT_s))^2. \qquad (44)$$

It can be verified (numerically) that using the distance in (44) (which intuitively provides "more information" about the primary system) leads to more accurate parameter estimates (or, alternatively, a greater number of parameters can be estimated if necessary). Note, however, that this comes at the expense of an additional computational effort and, moreover, it is unclear that all these variables can be accurately measured in practice.

The proposed methods can be applied when the observed time series are contaminated with additive noise of moderate variance. For example, if the observations have the form

$\tilde{a}_1(nT_s)=a_1(nT_s)+u_1(n)$ and $\tilde{a}_2(nT_s)=a_2(nT_s)+u_2(n)$, where $u_1(n)$ and $u_2(n)$ are sequences of independent and identically distributed Gaussian noise variables with zero mean and variance σ_u^2, then the distance function of Eq. (30) is lower-bounded by the noise variance. Specifically, if $\hat{\mathbf{q}} \approx \mathbf{p}$ and, hence, we assume that $\bar{a}_i(nT_s) \approx a_i(nT_s)$, it turns out that the distance $d(\mathbf{y},\mathbf{x})$ is an estimator of (twice) the noise variance

$$d(\mathbf{y},\mathbf{x})$$
$$\approx \frac{1}{N}\sum_{n=0}^{N-1}\left((\bar{a}_1(nT_s)-\tilde{a}_1(nT_s))^2+(\bar{a}_2(nT_s)-\tilde{a}_2(nT_s))^2\right) \qquad (45)$$
$$\approx 2\sigma_u^2.$$

This indicates that the synchronization error cannot go below the (approximate) bound of $2\sigma_u^2$ and, therefore, the ABC-based methods can work as long as the tolerances ($_t$, $t=1,\ldots,T$, in the ABC SMC method, or ϵ in the ABC MCMC technique) are chosen to be greater than $2\sigma_u^2$. This means that the ABC SMC algorithm with $T=12$ populations and $_T=10^{-4}$ can still provide accurate parameter estimates when the observation noise variance is $\sigma_u^2 <_T/2$. In order to handle larger noise variances, one needs to relax the coupling (i.e., choose a smaller coupling factor D) and increase the observation period T_o. This makes the distance function more sensitive to the discrepancy between $\mathbf{q}$ and $\mathbf{p}$, which in practice means that we can choose a larger tolerance (e.g, $_T$ in the ABC SMC algorithm) and preserve the accuracy of the resulting estimate $\hat{\mathbf{q}}$.

Author Contributions

References

1. van Vreeswijk C, Sompolinsky H (1996) Chaos in Neuronal Networks with Balanced Excitatory and Inhibitory Activity. Science 274: 1724–1726.
2. Amit DJ, Brunel N (1997) Model of global spontaneous activity and local structured activity during delay periods in the cerebral cortex. Cereb. Cortex 7: 237–252.
3. Brunel N (2000) Dynamics of networks of randomly connected excitatory and inhibitory spiking neurons. J. Physiol Paris 94: 445–463.
4. Sompolinsky H, Crisanti A, Sommers HJ (1988) Chaos in random neural networks. Phys. Rev. Lett. 61: 259–262.
5. Sussillo D, Abbott LF (2009) Generating Coherent Patterns of Activity from Chaotic Neural Networks. Neuron 63: 544–557.
6. Ghosh A, Kumar VR, Kulkarni BD (2001) Parameter estimation in spatially extended systems: The Karhunen-Leve and Galerkin multiple shooting approach. Phys. Rev. E 64: 056222.
7. Baake E, Baake M, Bock HG, Briggs KM (1992) Fitting ordinary differential equations to chaotic data. Phys. Rev. A 45: 5524–5529.
8. Hatz K, Schloder JP, Bock HG (2012) Estimating Parameters in Optimal Control Problems. SIAM J. Sci. Comput. 34: A1707–A1728.
9. Petridis V, Paterakis E, Kehagias A (1998) A hybrid neural-genetic multimodel parameter estimation algorithm. IEEE Trans. Neural Networks 9: 862–876.
10. Timmer J (2000) Parameter estimation in nonlinear stochastic differential equations. Chaos Solitons and Fractals 11: 2571–2578.
11. Singer H (2002) Parameter Estimation of Nonlinear Stochastic Differential Equations: Simulated Maximum Likelihood versus Extended Kalman Filter and It-Taylor Expansion. Journal of Computational and Graphical Statistics 11: 972–995.

12. Sitz A, Schwarz U, Kurths J, Voss HU (2002) Estimation of parameters and unobserved components for nonlinear systems from noisy time series. Phys Rev. E 66: 016210.
13. Pisarenko VF, SornetteD (2004) Statistical methods of parameter estimation for deterministically chaotic time series. Phys. Rev. E 69: 036122.
14. Parlitz U, Junge L, Kocarev L (1996) Synchronization-based parameter estimation from time series. Phys. Rev. E 54: 6253–6259.
15. Parlitz U (1996) Estimating Model Parameters from Time Series by Autosynchronization. Phys. Rev. Lett. 76: 1232–1235.
16. Zhou C, Lai C-H (1999) Decoding information by following parameter modulation with parameter adaptive control. Phys. Rev. E 59: 6629–6636.
17. Maybhate A, Amritkar RE (1999) Use of synchronization and adaptive control in parameter estimation from a time series. Phys. Rev. E 59: 284–293.
18. d'Anjou A, Sarasola C, Torrealdea FJ, Orduna R, Grana M (2001) Parameter-adaptive identical synchronization disclosing Lorenz chaotic masking. Phys. Rev. E 63: 046213.
19. Konnur R (2003) Synchronization-based approach for estimating all model parameters of chaotic systems. Phys. Rev. E 67: 027204.
20. Huang D (2004) Synchronization-based estimation of all parameters of chaotic systems from time series. Phys. Rev. E 69: 067201.
21. Freitas US, Macau EEN, Grebogi C (2005) Using geometric control and chaotic synchronization to estimate an unknown model parameter. Phys. Rev. E 71: 047203.
22. Mariño IP, Míguez J (2005) Adaptive approximation method for joint parameter estimation and identical synchronization of chaotic systems. Phys. Rev. E 72: 057202.

23. Mariño IP, Míguez J (2006) An approximate gradient-descent method for joint parameter estimation and synchronization of coupled chaotic systems. Phys. Lett. A 351: 262–267.
24. Tao C, Zhang Y, Jiang JJ (2007) Estimating system parameters from chaotic time series with synchronization optimized by a genetic algorithm. Phys. Rev. E 76: 016209.
25. Yang X, Xu W, Sun Z (2007) Estimating model parameters in nonautonomous chaotic systems using synchronization. Phys. Lett. A 364: 378–388.
26. Yu D, Parlitz U (2008) Estimating parameters by autosynchronization with dynamics restrictions. Phys. Rev. E 77: 066221.
27. Ghosh D, Banerjee S (2008) Adaptive scheme for synchronization-based multiparameter estimation from a single chaotic time series and its applications. Phys. Rev. E 78: 056211.
28. Abarbanel HDI, Creveling DR, Farsian R, Kostuk M (2009) Dynamical State and Parameter Estimation. SIAM J. Appl. Dyn. Syst. 8: 1341–1381.
29. Sakaguchi H (2002) Parameter evaluation from time sequences using chaos synchronization. Phys. Rev. E 65: 027201.
30. Schumann-Bischoff J, Luther S, Parlitz U (2013) Nonlinear system identification employing automatic differentiation. Commun. Nonlinear Sci. Numer. Simulat. 18: 2733–2742.
31. Mariño IP, Míguez J (2007) Monte Carlo method for multiparameter estimation in coupled chaotic systems. Phys. Rev. E 76: 057203.
32. van Leeuwen PJ (2010) Nonlinear data assimilation in geosciences: an extremely efficient particle filter. Q.J.R. Meteorol. Soc 136: 1991–1999.
33. Mariño IP, Míguez J, Meucci R (2009) Monte Carlo method for adaptively estimating the unknown parameters and the dynamic state of chaotic systems. Phys. Rev. E 79: 056218.
34. Appel MJ, Labarre R, Radulovic D (2003) On Accelerated Random Search. SIAM J. Optim. 14: 708–731.
35. Pritchard J, Seielstad MT, Perez-Lezaun A, Feldman MW (1999) Population growth of human Y chromosomes: a study of Y chromosome microsatellites. Mol. Biol. Evol. 16: 1791–1798.
36. Beaumont MA, Zhang W, Balding DJ (2002) Approximate Bayesian Computation in Population Genetics. Genetics 162: 2025–2035.
37. Marjoram P, Molitor J, Plagnol V, Tavare S (2003) Markov chain Monte Carlo without likelihood. Proc. Natl. Acad. Sci. U.S.A. 100: 15324–15328.
38. Sisson SA, Fan Y, Tanaka MM (2007) Sequential Monte Carlo without likelihoods. Proc. Natl. Acad. Sci U.S.A. 104: 1760–1765.
39. Toni T, Welch D, Strelkowa N, Ipsen A, Stumpf MPH (2009) Approximate Bayesian computation scheme for parameter inference and model selection in dynamical systems. J. R. Soc. Interface 6: 187–202.
40. Elowitz MB, Leibler S (2000) A synthetic oscillatory network of transcriptional regulators. Nature 403: 335–338.
41. Ullner E, Koseka A, Kurths J, Volkov E, Kantz H, et al. (2008) Multistability of synthetic genetic networks with repressive cell-to-cell communication. Phys. Rev. E 78: 031904.
42. García-Ojalvo J, Elowitz B, Strogatz SH (2004) Modeling a synthetic multicellular clock: Repressilators coupled by quorum sensing. Proc. Natl. Acad. Sci. U.S.A. 101: 10955–10960.
43. McMillen D, Kopell N, Hasty J, Collins JJ (2002) Synchronizing genetic relaxation oscillators by intercell signaling. Proc. Natl. Acad. Sci. U.S.A. 99: 679–684.
44. You L, Cox III RS, Weiss R, Arnold FH (2004) Programmed population control by cellcell communication and regulated killing. Nature 428: 868–871.
45. Volkov EI, Stolyarov MN (1991) Birhythmicity in a system of two coupled identical oscillators. Phys. Lett. A 159, 61–66.
46. Han SK, Kurrer C, Kuramoto Y (1995) Dephasing and Bursting in Coupled Neural Oscillators. Phys. Rev. Lett. 75: 3190–3193.
47. Balázsi G, Cornell-Bell A, Neiman AB, Moss F (2001) Synchronization of hyperexcitable systems with phase-repulsive coupling. Phys. Rev. E 64: 041912.
48. Ullner E, Zaikin A, Volkov EI, García-Ojalvo J (2007) Multistability and Clustering in a Population of Synthetic Genetic Oscillators via Phase-Repulsive Cell-to-Cell Communication. Phys. Rev. Lett. 99: 148103.
49. Koseka A, Ullner E, Volkov E, Kurths J, García-Ojalvo J (2010), J. Theor. Biol. 263, 189.
50. Laje R, Mindlin GB (2002)., Phys. Rev. Lett. 89, 288102.
51. Koseska A, Volkov E, Zaikin A, Kurths J (2007) Inherent multistability in arrays of autoinducer coupled genetic oscillators. Phys. Rev. E 75: 031916.
52. Glass L, Mackey MC (1988) From Clocks to Chaos: The Rhythms of Life: Princeton University Press, Princeton, NJ. 248p.
53. Meinhardt H (1982) Models of Biological Pattern Formation: Academic Press, New York.
54. Del Moral P, Doucet A, Jasra A (2006) Sequential Monte Carlo samplers. J. R. Stat. Soc B 68: 411–436.
55. Del Moral P, Doucet A, Jasra A (2012) An adaptive sequential Monte Carlo method for approximate Bayesian computation. Stat Comput 22: 1009–1020.