Version dated: October 30, 2015

Fundamentals and Recent Developments in Approximate Bayesian Computation

JARNO LINTUSAARI[1*], MICHAEL U. GUTMANN[1,2*], RITABRATA DUTTA[1],

SAMUEL KASKI[1], AND JUKKA CORANDER[2]

[1]*Helsinki Institute for Information Technology HIIT,*

Department of Computer Science, Aalto University, Espoo, 00076, Finland

[2]*Helsinki Institute for Information Technology HIIT,*

Department of Mathematics and Statistics, University of Helsinki, Helsinki, 00014, Finland

[*]Contributed equally

Corresponding author: Jarno Lintusaari, Department of Computer Science, Aalto University, 00076 Espoo, Finland, E-mail: jarno.lintusaari@aalto.fi

Abstract.— Bayesian inference plays an important role in biology and in many more branches of science by providing a principled framework for dealing with uncertainty and quantifying how it changes in the light of new evidence. For many complex models and inference problems, however, only approximate quantitative answers are obtainable. Approximate Bayesian computation, or ABC in short, denotes a family of algorithms for approximate inference which makes a minimal set of assumptions by only requiring that sampling from a model is possible. We explain here the fundamentals of approximate Bayesian computation, review the classical algorithms, and highlight recent developments.

(Keywords: approximate Bayesian computation, Bayesian inference, likelihood-free inference, simulator-based models, stochastic simulation models)

Introduction

Complex biological systems are nowadays widely modeled by simulating them on computers. Many of the phenomena that are mimicked in this manner are inherently stochastic processes such as the demographic spread of a species (Currat and Excoffier 2004; Fagundes et al. 2007; Itan et al. 2009; Excoffier et al. 2013), the evolution of genomes (Marttinen et al. 2015) or the dynamics of gene regulation (Toni et al. 2009). Realistic stochastic simulation models almost inevitably have multiple parameters, and while it is relatively easy to run the simulator to generate data given any configuration of the parameters, the real interest is often focused on the inverse problem: the identification of parameter configurations which would plausibly lead to data that are sufficiently similar to the observed data. Solving such a nonlinear inverse problem is generally a very difficult task.

The Bayesian inference methodology provides a principled framework for solving the aforementioned inverse problem. A prior probability distribution on the model parameters is used to describe the initial beliefs about what values of the parameters could be plausible. To update the prior beliefs one uses the likelihood function, which is the probability to obtain the observed data or some small perturbation of them given the parameter values. The likelihood indicates which parameter values are congruent with the observed data. However, computing the likelihood function is mostly impossible for stochastic simulation models due to the unobservable (latent) random quantities included in the model. In some cases, Monte Carlo methods offer a way to handle the latent variables such that an approximate likelihood is obtained, but these methods have their limitations, and for large and complex models, they are "too inefficient by far" (Green et al. 2015, page 848). To deal with models where likelihood calculations fail, other techniques have been developed which are collectively referred to as likelihood-free inference or approximate Bayesian computation (ABC).

In a nutshell, ABC algorithms sample from the posterior distribution of the parameters by finding values which yield simulated data resembling the observed data to a sufficient degree. While widely applicable, ABC comes with its own set of difficulties, which are of both computational and statistical nature. The two main intrinsic difficulties are how to efficiently find plausible parameter values, and how to define what is similar to the observed data and what is not. All ABC algorithms have to deal with these two issues in some manner, and the different algorithms discussed here essentially differ in how they tackle the two problems.

The remainder of this article is structured as follows. We next discuss in more detail the characteristics of models which are defined in terms of a stochastic computer program, hereafter called "simulator-based models", and point out difficulties when performing inference for the unknown parameters of such models. The discussion leads to the basic rejection ABC algorithm which is presented in the subsequent section. This is followed by a presentation of popular ABC algorithms which were developed to increase the computational efficiency. We then consider several recent advances which aim to improve ABC both computationally and statistically. The final section provides conclusions and a discussion about likelihood-free inference methods related to ABC.

Simulator-based models

Definition

Simulator-based models are functions M which map the model parameters θ and some random variables V to data y. The functions M are generally implemented as computer programs where the parameter values are provided as input and where the random variables are drawn sequentially by making calls to a random number generator. The parameters θ govern the properties of interest of the generated data while the random variables V represent the stochastic variation inherent to the simulated process.

The mapping M may be as complex as needed, and this generality of simulator-based models allows researchers to implement hypotheses about how the data were generated without having to make excessive compromises motivated by mathematical simplicity, or other reasons not related to the scientific question being investigated.

Due to the presence of the random variables V, the outputs of the simulator fluctuate randomly even when using exactly the same values of the model parameters θ. This means that we can consider the simulator to define a random variable Y_θ whose distribution is implicitly determined by the distribution of V and the mapping M acting on V for a given θ (for this reason, simulator-based models are sometimes called implicit models, Diggle and Gratton 1984). Using the properties of transformation of random variables, it is possible to formally write down the distribution of Y_θ. For instance, for a fixed value of θ, the probability that Y_θ takes values in an ϵ neighborhood $B_\epsilon(y_0)$ around the observed data y_0 is equal to the probability to draw values of V which are mapped to that neighborhood,

$$\Pr\big(Y_\theta \in B_\epsilon(y_0)\big) = \Pr\big(M(\theta, V) \in B_\epsilon(y_0)\big), \tag{1}$$

as illustrated in Figure 1. Computing the probability analytically is, however, impossible for complex models. But it is possible to test empirically whether a particular outcome y_θ of the simulation ends up in the neighborhood of y_0 or not (see Figure 1). We will see that this property of stochastic simulation models plays a key role for performing inference about their parameters.

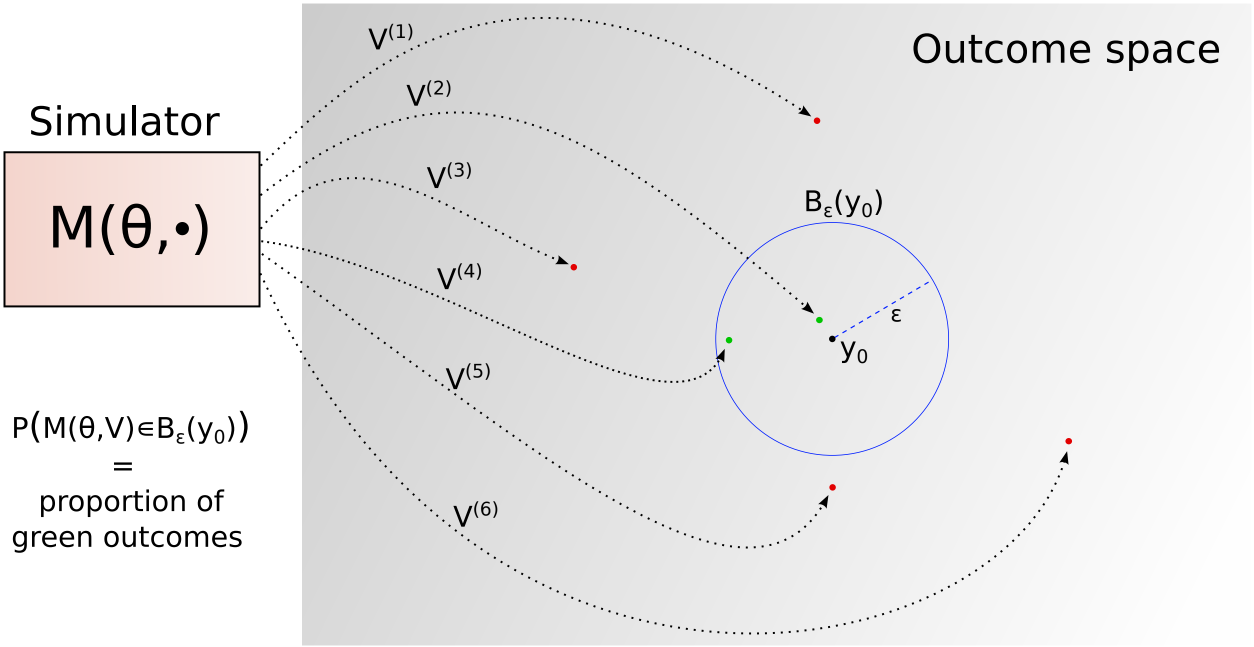


Figure 1: Illustration of the stochastic simulator M when run multiple times with a fixed value of θ. The black dot y_0 is the observed data and the arrows point to different simulated data sets. Two outcomes, marked in green, are less than ϵ away from y_0. The proportion of such outcomes provides an approximation of the likelihood.

Example

As an example of a simulator-based model, we here present the simple yet analytically intractable model by Tanaka et al. (2006) for the spread of tuberculosis.

The simulator begins with one infectious host and stops when a fixed number of infectious hosts m is exceeded. In the simulation, it is assumed that each infectious host randomly infects other individuals from an unlimited supply of hosts with the rate α, each time transmitting a specific strain of the communicable pathogen, characterized by its haplotype. It is thus effectively assumed that a strong transmission bottleneck occurs, such that only a single strain is passed forward in each transmission event, despite the eventual genetic variation persisting in the within-host pathogen population. Furthermore, the model states that the host stops being infectious, i.e. recovers or dies randomly with the rate δ, and that the pathogen mutates randomly within the host at the rate τ, thereby generating a novel haplotype under a single-locus infinite alleles model. The parameters of the model are thus $\theta = (\alpha, \delta, \tau)$. An output of the simulator is obtained by a random sample of size $n \ll m$ from the whole generated population of infected hosts, and this sample, represented by the vector y_θ, contains the sizes of the clusters of hosts carrying the same haplotype of the pathogen. For example, $y_\theta = (6, 3, 2, 2, 1, 1, 1, 1, 1, 1, 1)$ corresponds to a sampled population containing one cluster of 6 infected hosts, one cluster with three hosts, two clusters with two hosts each, as well as 7 singleton clusters. The simulation process is illustrated in Figure 2. Note that this model of pathogen spread is atypical in the sense that the observation times of the infections are all left implicit in the sampling process, in contrast to the standard likelihood formulation used for infectious disease epidemiological models (Anderson and May 1992).

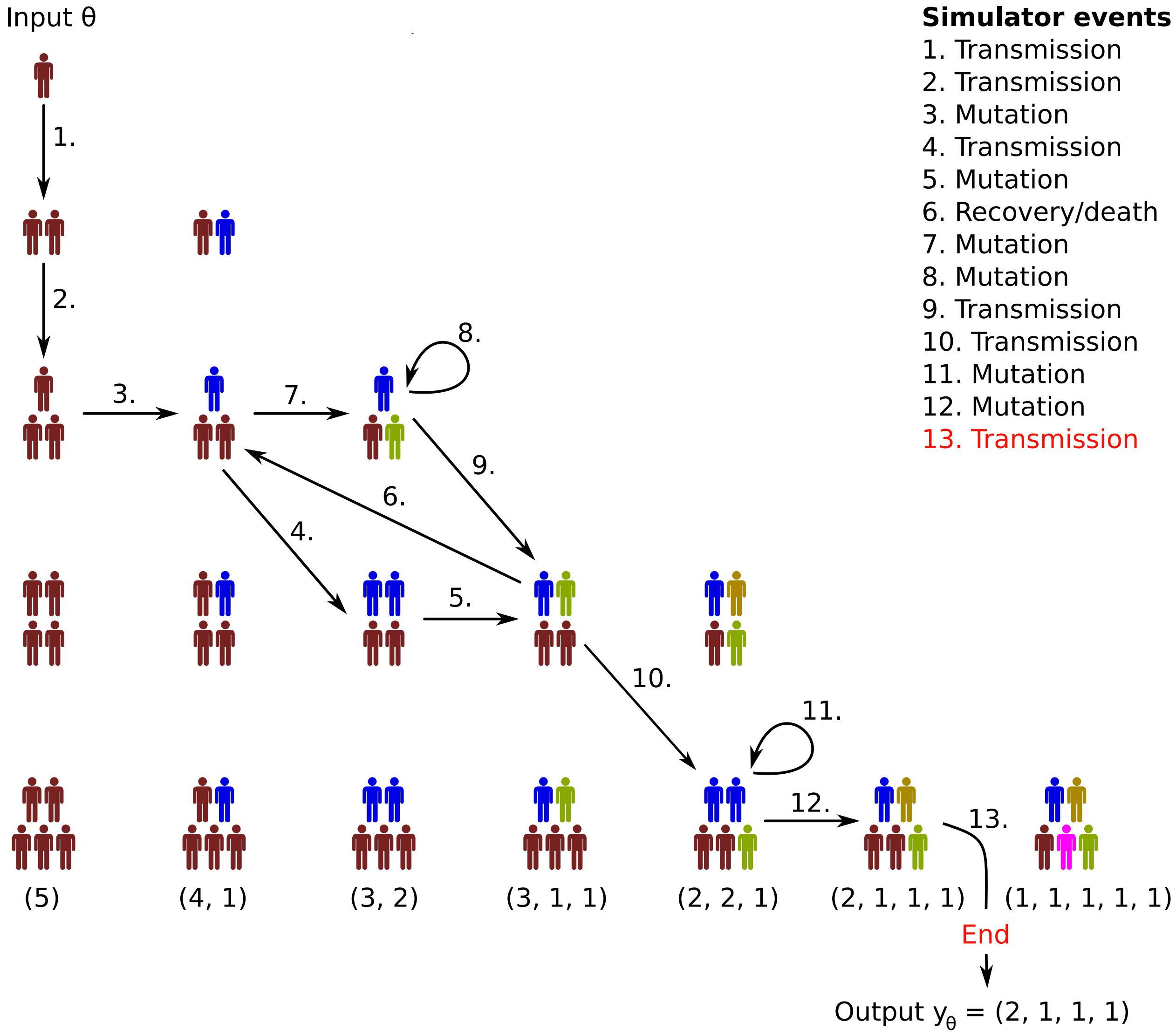


Figure 2: An example of a transmission process simulated under a parameter configuration θ without sub-sampling of the simulated infectious population. Arrows indicate the sequence of random events taking place in the simulation and different colors represent different haplotypes of the pathogen. The simulation starts with one infectious host who transmits the pathogen to another host. After one more transmission event, the pathogen undergoes mutation within one of the three hosts infected so far. As the sixth event in the simulation, one of the haplotypes is removed from the population due to the recovery/death of the corresponding host. The simulation stops when the infectious population size exceeds $m = 5$ and the simulator outputs the generated y_θ. Alternative infected host population configurations are shown as additional nodes not connected by the arrows and the bottom row lists host cluster sizes with the corresponding population configuration at the termination of the simulation.

Difficulties in performing statistical inference

Values of the parameters θ which are plausible in the light of the observations y_0 can be determined via statistical inference either by finding values which maximize the probability in Equation (1) for some sufficiently small ϵ, or by determining their posterior distribution. In more detail, in maximum likelihood estimation, the parameters are determined by maximizing the likelihood function $L(\theta)$,

$$L(\theta) = \lim_{\epsilon \to 0} c_\epsilon \Pr\left(Y_\theta \in B_\epsilon(y_0)\right), \tag{2}$$

where c_ϵ is a proportionality factor which may depend on ϵ, which is needed when $\Pr\left(Y_\theta \in B_\epsilon(y_0)\right)$ shrinks to zero as ϵ approaches zero. If the output of the simulator can only take a countable number of values, Y_θ is called a discrete random variable and the definition of the likelihood simplifies to $L(\theta) = \Pr\left(Y_\theta = y_0\right)$, which equals the probability to simulate data equal to the observed data. In Bayesian inference, the essential characterization of the uncertainty about the model parameters is defined by their conditional distribution given the data, i.e. the posterior distribution $p(\theta|y_0)$,

$$p(\theta|y_0) \propto L(\theta)p(\theta), \tag{3}$$

where $p(\theta)$ is the prior distribution of the parameters.

For complex models, however, neither the probability in Equation (1), nor the likelihood function $L(\theta)$ are available analytically in closed form as a function of θ, which is the reason why statistical inference is difficult for simulator-based models.

For the model of tuberculosis transmission presented in the previous section, computing the likelihood function becomes intractable if the infectious population size m is large, or if the death rate $\delta > 0$ (Stadler 2011). This is because for large m, the state space

Algorithm 1 Rejection sampling algorithm for simulator-based models. The algorithm produces N independent samples from the posterior distribution $p(\theta|y_0)$

```
for $i = 1$ to $N$ do
    repeat
        Generate $\theta$ from the prior $p(\cdot)$
        Generate $y_\theta$ from the simulator
    until $y_\theta = y_0$
    $\theta^{(i)} \leftarrow \theta$
end for
```

of the process, i.e. the number of different cluster vectors, grows very quickly which makes exact numerical calculation of the likelihood infeasible. Moreover, if the death rate δ is nonzero, the process is allowed to return to previous states which further complicates the computations. Finally, the assumption that not all infectious hosts are observed contributes additionally to the intractability of the likelihood.

Inference via rejection sampling

We present here an algorithm for exact posterior inference which can be applied when Y_θ can only take countably many values, that is, if Y_θ is a discrete random variable. As shown above, in this case $L(\theta) = \Pr(Y_\theta = y_0)$. The presented algorithm forms the basis of the algorithms for approximate Bayesian computation discussed in the later sections.

In general, samples from the prior distribution $p(\theta)$ of the parameters can be converted into samples from the posterior $p(\theta|y_0)$ by retaining each sampled value with a probability proportional to $L(\theta)$. This can be done sequentially by first sampling a parameter value from the prior, $\theta \sim p(\theta)$, and then accepting the obtained value with the probability $L(\theta)/(\max_\theta L(\theta))$. This procedure corresponds to rejection sampling (see, for example Robert and Casella 2004, Chapter 2). Now, with the likelihood $L(\theta)$ being equal to the probability that $Y_\theta = y_0$, the latter step can be implemented for simulator-based models even $L(\theta)$ is not available analytically: by running the simulator and checking

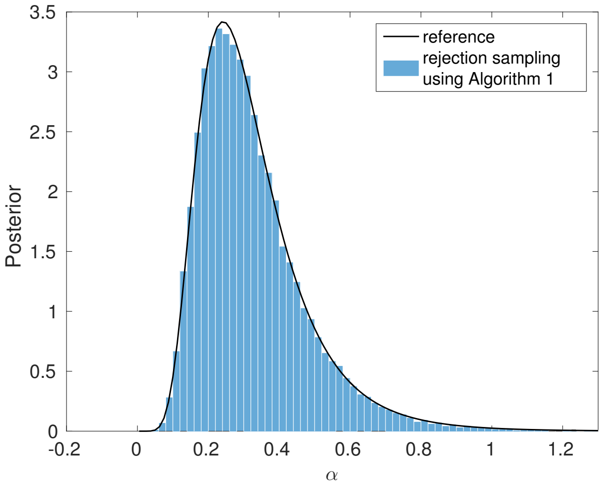


Figure 3: Exact inference for a simulator-based model of tuberculosis transmission. A very simple setting was chosen where the exact posterior can be numerically computed (black line), and where Algorithm 1 is applicable (blue bars).

whether the generated data equal the observed data. This gives the rejection algorithm for simulator-based models summarized as Algorithm 1. Rubin (1984) used it to provide intuition about how Bayesian inference about parameters works in general.

To obtain another interpretation of Algorithm 1, recall that for discrete random variables the posterior distribution $p(\theta|y_0)$ is, by definition, equal to the joint distribution of θ and Y_θ, normalized by the probability that $Y_\theta = y_0$. That is, the posterior is obtained by conditioning on the event $Y_\theta = y_0$. We can thus understand the test for equality of y_θ and y_0 on line 5 of the algorithm as an implementation of the conditioning operation.

To illustrate Algorithm 1, we generated a synthetic data set y_0 from the tuberculosis transmission model by running the simulator with the parameter values $\alpha = 0.2$, $\delta = 0$, $\tau = 0.198$, and setting the population size to $m = 20$. We further assumed that the whole population is observed, which yielded the observed data $y_0 = (6, 3, 2, 2, 1, 1, 1, 1, 1, 1, 1)$. The assumptions about the size of the population, and that the whole population has been observed, are unrealistic but they enable a comparison against the exact posterior distribution, which in this setting can be numerically computed using Theorem 1 of Stadler

(2011). Figure 3 shows a (rescaled) histogram of the samples obtained with Algorithm 1, and it can be seen that the histogram matches the posterior distribution very accurately. To obtain this result, we assumed that both of the parameters δ and τ were known, and assigned a uniform prior distribution in the interval $(0, 2)$ for the sole unknown parameter, the transmission rate α. A total of 20 million data sets y_θ were simulated, out of which 40 000 matched y_0 (acceptance rate of 0.2%).

Fundamentals of approximate Bayesian computation

The rejection ABC algorithm

While Algorithm 1 produces independent samples from the posterior, the probability that the simulated data equal the observed data is often negligibly small, which renders the algorithm impractical as virtually no simulated realizations of θ will be accepted. The same problem holds true if the generated data can take uncountably many values, i.e. when Y_θ is a continuous random variable.

To make inference feasible, the acceptance criterion $y_\theta = y_0$ in Algorithm 1 can be relaxed to

$$d(y_\theta, y_0) \leq \epsilon, \tag{4}$$

where $\epsilon > 0$ and $d(y_\theta, y_0) \geq 0$ is a “distance” function which measures the discrepancy between the two data sets, as considered relevant for the inference. With this modification, Algorithm 1 becomes the rejection ABC algorithm summarized as Algorithm 2. The first implementation of this algorithm appeared in the work by Pritchard et al. (1999).

Algorithm 2 does not produce samples from the posterior $p(\theta|y_0)$ in Equation (3) but samples from an approximation $p_{d,\epsilon}(\theta|y_0)$,

$$p_{d,\epsilon}(\theta|y_0) \propto \Pr\big(d(Y_\theta, y_0) \leq \epsilon\big) p(\theta), \tag{5}$$

Algorithm 2 Rejection ABC algorithm producing N independent samples from the approximate posterior distribution $p_{d,\epsilon}(\theta|y_0)$

```
for $i = 1$ to $N$ do
    repeat
        Generate $\theta$ from the prior $p(\cdot)$
        Generate $y_\theta$ from the simulator
    until $d(y_\theta, y_0) \leq \epsilon$
    $\theta^{(i)} \leftarrow \theta$
end for
```

which is the posterior distribution of θ conditional on the event $d(Y_\theta, y_0) \leq \epsilon$. Equation (5) is obtained by approximating the intractable likelihood function $L(\theta)$ in Equation (2) with $L_{d,\epsilon}(\theta)$,

$$L_{d,\epsilon}(\theta) \propto \Pr\big(d(Y_\theta, y_0) \leq \epsilon\big). \tag{6}$$

The approximation is two-fold. First, the distance function d is generally not a metric in the mathematical sense, at least because $d(y_\theta, y_0) = 0$ even if $y_\theta \neq y_0$. Second, ϵ is chosen large enough so that sufficiently many samples will be accepted. Intuitively, the likelihood of the data y_0 is replaced by the likelihood of the event $d(Y_\theta, y_0) \leq \epsilon$ consisting of outcomes near to y_0 as defined by the distance function d and the threshold value ϵ (see Figure 1).

The distance d is typically computed by first reducing the data to suitable summary statistics $t = T(y)$ and then computing the distance d_T between them, so that $d(y_\theta, y_0) = d_T(t, t_0)$, where d_T is often the Euclidean or some other metric for the summary statistics. When combining different summary statistics, they are usually re-scaled so that they contribute equally to the distance.

In addition to the accuracy of the approximation $p_{d,\epsilon}(\theta|y_0)$, the distance d and the threshold ϵ also influence the computing time required to obtain samples. For instance, if $\epsilon = 0$ and the distance d is such that $d(y, y_0) = 0 \iff y = y_0$, Algorithm 2 becomes Algorithm 1 and $p_{d,\epsilon}(\theta|y_0)$ becomes $p(\theta|y_0)$, but the computing time to obtain samples from $p_{d,\epsilon}(\theta|y_0)$ would then typically be impractically large. Hence, on a very fundamental level, there is a trade-off between statistical and computational performance in approximate Bayesian computation.

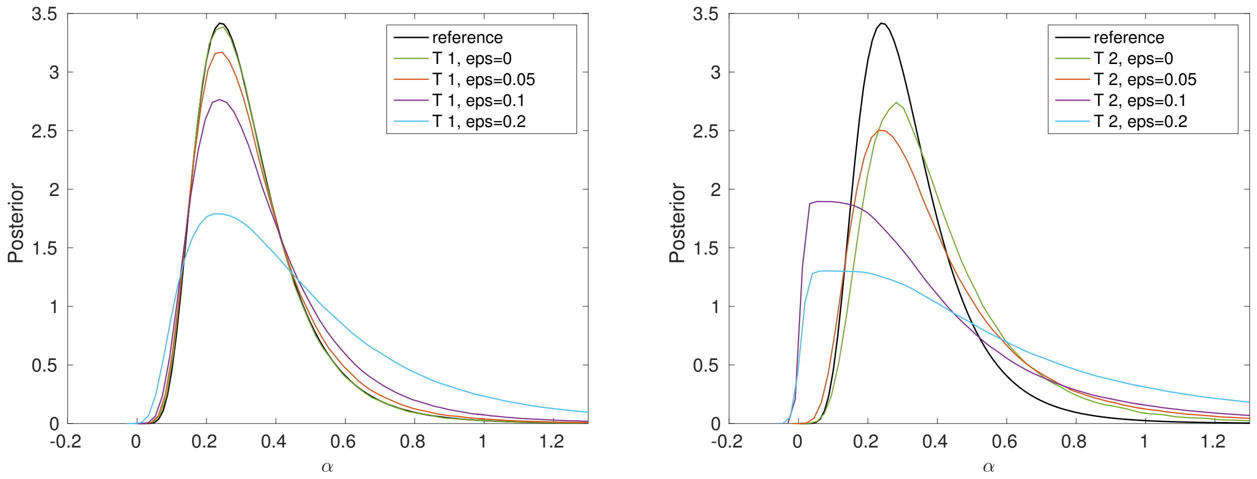
We next illustrate the trade-off and Algorithm 2 using the previous example about tuberculosis transmission. Two different distances d_1 and d_2 are considered,

$$d_1(y_\theta, y_0) = |T_1(y_\theta) - T_1(y_0)|, \qquad d_2(y_\theta, y_0) = |T_2(y_\theta) - T_2(y_0)|, \tag{7}$$

where T_1 is the number of clusters contained in the data divided by the sample size n and T_2 is a genetic diversity measure defined as $T_2(y) = 1 - \sum_i (n_i/n)^2$, where n_i is the size of the ith cluster. For $y_0 = (6, 3, 2, 2, 1, 1, 1, 1, 1, 1, 1)$, we have $T_1(y_0) = 11/20 = 0.55$ and $T_2(y_0) = 0.85$. For both d_1 and d_2, the absolute difference between the summary statistics is used as the metric d_T.

Figure 4(a) shows that using the summary statistic T_1 instead of the full data does not here lead to a visible deterioration of the inferred posterior when $\epsilon = 0$; the black and green curves in the figure match well. Figure 4(b) shows that for T_2, however, there is a clear difference as the posterior mode and mean are shifted to larger values of α, and the posterior variance is larger too. In both cases, increasing ϵ, that is accepting more parameters, leads to an approximate posterior distribution which is less concentrated than the true posterior.

Algorithm 1 and Algorithm 2 with T_1 produce here equivalent results but the number of simulations required to obtain the approximate posterior differs between the two algorithms. Figure 5(a) shows the posterior distribution obtained with the algorithms when the stopping criterion is changed to the total number of runs of the simulator instead of the number of accepted samples. It can be seen that for a computational budget of 100 000 simulations, the posterior obtained by Algorithm 1, shown in blue, differs substantially from the exact posterior (black curve), while the posterior from Algorithm 2



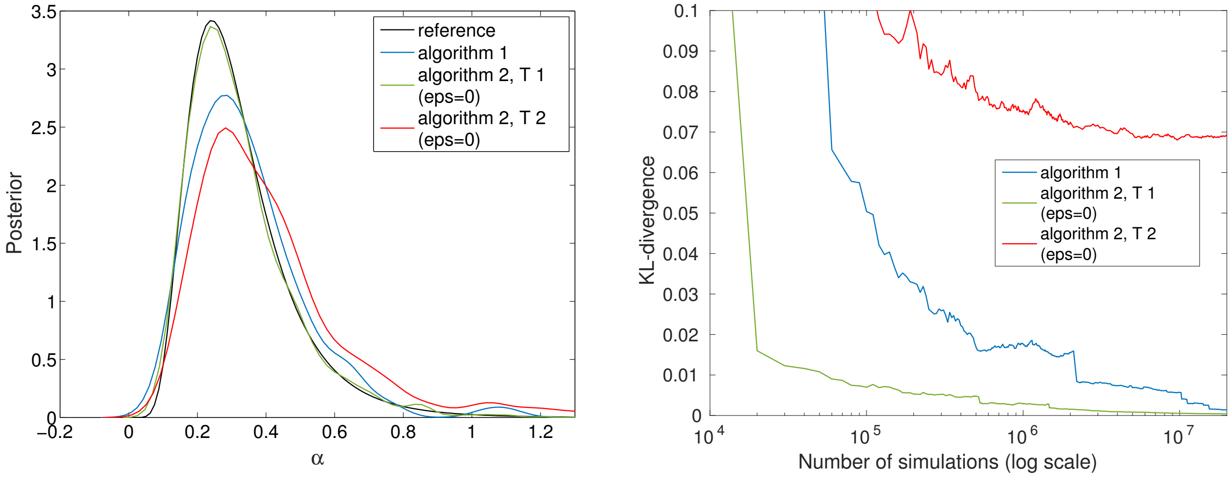
(a) Cluster frequency as summary statistic

(b) Genetic diversity as summary statistic

Figure 4: Inference results for the transmission rate α of tuberculosis. The plots show the posterior distributions obtained with Algorithm 2 and 20 million data set simulations (proposals).

with T_1 (green curve) is still matching it well. The relatively poor result with Algorithm 1 is due to its low acceptance rate (here 0.2%). While the accepted samples do follow the exact posterior $p(\theta|y_0)$, the algorithm did not manage to produce enough accepted realizations within the computational budget available, which implies that the Monte Carlo error of the posterior approximation remains non-negligible.

Figure 5(b) plots the number of data sets simulated versus the accuracy of the inferred posterior distribution, which is here measured by the Kullback-Leibler (KL) divergence between the exact and inferred posterior. Each curve visualizes the trade-off between statistical and computational efficiency of the algorithm used. Here, Algorithm 2 with summary statistic T_1 (green curve) features a better trade-off than Algorithm 1 (blue curve) which is in turn better than the trade-off of Algorithm 2 with summary statistic T_2 (red curve). The latter curve flattens out after approximately one million simulations which visualizes the approximation error introduced by using the summary statistic T_2. For Algorithm 1 (blue curve), nonzero values of the KL divergence are due to the Monte Carlo



(a) Results after 100 000 simulations

(b) Accuracy versus computational cost

Figure 5: Comparison of the efficiency of Algorithm 1 and 2. Smaller Kullback-Leibler divergence means more accurate inference of the posterior distribution.

error only, while for Algorithm 2 with summary statistic T_1 (green curve), they are due to both the Monte Carlo error and the use of summary statistics. For this particular example, the latter error is negligible.

Choice of the summary statistics

If the distance d is computed by projecting the data to summary statistics followed by their comparison using a metric in the summary statistics space (e.g. the Euclidean distance), the quality of the inference hinges on the summary statistics chosen as we have seen in Figures 4 and 5.

For consistent performance of ABC algorithm, the summary statistics should be sufficient for the parameters. But in practice, only exponential families enjoy sufficient statistics with finite dimensions. Hence, in all of the practical applications of ABC the chosen statistics are not sufficient for the parameters. Additionally, with the increase in the number of summary statistics used, more simulations tend to become rejected so that an increasing number of simulation runs is needed to obtain a satisfactory number of accepted parameter values, making the algorithm extremely inefficient. This is known as the curse of dimensionality for ABC.

One of the main remedies to the above issue is to efficiently choose informative summary statistics. Importantly, the summary statistics which are informative for the parameters in a neighborhood of the true parameter value, and the summary statistics most informative globally, are significantly different (Nunes and Balding 2010). General intuition says that the set of summary statistics which are locally sufficient would be a subset of the globally sufficient ones. So, for an efficient algorithm, a good strategy seems to first find a locality containing the true parameter with high enough probability and then choose informative statistics depending on that locality.

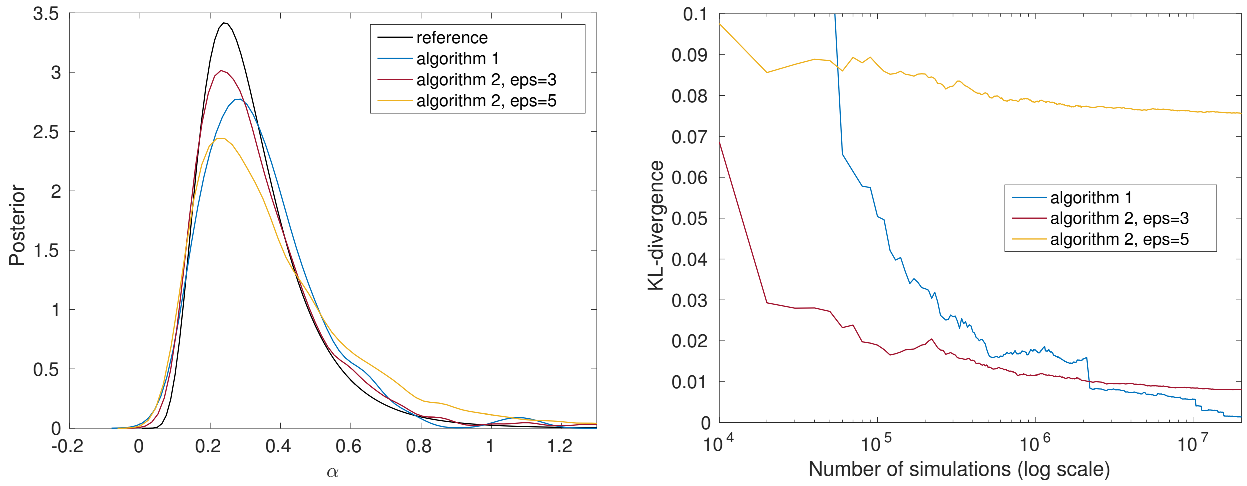
In line with the above, Nunes and Balding (2010), Fearnhead and Prangle (2012), and Aeschbacher et al. (2012) first defined "locality" through a pilot ABC run and then chose the statistics in that locality. Four methods for choosing the statistics are generally used: a) a sequential scheme based on the principle of approximate sufficiency (Joyce and Marjoram 2008); b) selection of a subset of the summary statistics maximizing prespecified criteria such as the Akaike information criterion (used by Blum et al. 2013) or the entropy of a distribution (used by Nunes and Balding 2010); c) partial least square regression which finds linear combinations of the original summary statistics which are maximally decorrelated and at the same time highly correlated with the parameters (Wegmann et al. 2009); d) assuming a statistical model between parameters and transformed statistics of simulated data, summary statistics are chosen by minimizing a loss function (Fearnhead and Prangle 2012; Aeschbacher et al. 2012). For comparison of the above methods in simulated and practical examples, we refer readers to the work by Blum and François (2010), Aeschbacher et al. (2012), and Blum et al. (2013).

Choice of the threshold

Having the distance function d specified (possibly using summary statistics), the remaining factor in the approximation of the posterior in Equation (5) is the specification of the threshold ϵ.

Larger values of ϵ result in biased approximations $p_{d,\epsilon}(\theta|y_0)$ (see e.g. Figure 4). The gain is a faster algorithm, meaning a reduced Monte Carlo error as one is able to produce more samples per unit of time. Therefore, when specifying the threshold the goal is to find a good balance between the bias and the Monte Carlo error. We illustrate this using Algorithm 2 with the full data without reduction to summary statistics (in other words, $T(y) = y$). In this case, Algorithm 2 with $\epsilon = 0$ is identical to Algorithm 1. Figure 6(a) shows that $\epsilon = 3$ yields a better posterior than $\epsilon = 0$ when using a maximal number of 100 000 simulations. This means that the gain from reduced Monte Carlo error is greater than the loss incurred by the bias. But this is no longer true for $\epsilon = 5$ where the bias dominates. Figure 6(b) shows that eventually the exact method will converge to the true posterior, while the other two will continue to suffer from the bias caused by the larger threshold. However, with smaller computational budgets (less than 2 million simulations), more accurate results are obtained with the nonzero threshold $\epsilon = 3$.

The choice of threshold is typically made by experimenting in a pilot setup using a precomputed pool of simulation-parameter pairs (y_θ, θ). The simulations can then be used to test different threshold values by using the simulated data sets in the role of the observed data and solving the inference problem where the true data-generating parameter is known. Different criteria, such as the mean squared error (MSE) between the mean of the approximation and the known generating parameters can be used as the criterion for the threshold choice (see e.g. Faisal et al. (2013), and the later section in this paper on validation of ABC). Prangle et al. (2014) discuss the use of a coverage property as the criterion to choose the threshold value ϵ. Intuitively, the coverage property tests if the



(a) Results after 100 000 simulations

(b) Accuracy given computational cost

Figure 6: Comparison of the trade-off between Monte Carlo error and the bias with different thresholds ϵ. Algorithm 1 is equivalent here to Algorithm 2 with $\epsilon = 0$. Smaller Kullback-Leibler divergences mean more accurate inference of the posterior distribution.

parameter values θ^* used to artificially generate a data set y_0^* are covered by the credible intervals constructed from the ABC output for y_0^* at correct rates (i.e. $\alpha\%$ credible interval should contain the true parameter in $\alpha\%$ of the tests).

Some of the theoretical convergence results can also be used to adjust the threshold value after a pilot run. Barber et al. (2015) provides convergence results for optimal ϵ with respect to the mean squared error of a posterior expectation (e.g. the posterior mean). The theoretically optimal sequence for the threshold ϵ is achieved by making it proportional to $N^{-1/4}$ as $N \to \infty$, where N is the number of accepted samples. If the constant in this relation is estimated in a pilot run, one can compute a new theoretically optimal threshold when N is increased. Blum (2010) derives corresponding results using an approach based on conditional density estimation, finding that ϵ should optimally be proportional to $N_s^{-1/(d+5)}$ as $N_s \to \infty$, where d is the dimension of the parameter space and N_s the total number of simulations performed. See also Fearnhead and Prangle (2012), Silk et al. (2013) and Biau et al. (2015) for similar results.

Beyond simple rejection sampling

The basic rejection ABC algorithm is essentially a trial and error scheme where the trial (proposal) values are sampled from the prior. We review here three popular algorithms which seek to improve upon the basic rejection approach. The first two aim at constructing proposal distributions which are closer to the posterior, whereas the third is a correction method which aims at adjusting samples obtained by ABC algorithms so that they are closer to the posterior.

Markov chain Monte Carlo ABC

The Markov chain Monte Carlo (MCMC) ABC algorithm is based on the Metropolis-Hastings MCMC algorithm which is often used in Bayesian statistics (Robert and Casella 2004, Chapter 7). In order to leverage that algorithm, we write $p_{d,\epsilon}(\theta|y_0)$ in Equation (5) as the marginal distribution of $p_{d,\epsilon}(\theta, y|y_0)$,

$$p_{d,\epsilon}(\theta, y|y_0) \propto p(\theta)p(y|\theta)\, \mathbb{1}[d(y, y_0) \leq \epsilon], \tag{8}$$

where $p(y|\theta)$ denotes the probability density (mass) function of Y_θ, and $\mathbb{1}[d(y, y_0) \leq \epsilon]$ equals one if $d(y, y_0) \leq \epsilon$ and zero otherwise. Importantly, while $p(y|\theta)$ is generally unknown for simulator-based models, it is still possible to use $p_{d,\epsilon}(\theta, y|y_0)$ as the target distribution in a Metropolis-Hastings MCMC algorithm by choosing the proposal distribution in the right way. The obtained (marginal) samples of θ then follow the approximate posterior $p_{d,\epsilon}(\theta|y_0)$.

Assuming that the Markov chain is at iteration i in state $x^{(i)} = (\theta^{(i)}, y^{(i)})$ where $d(y^{(i)}, y_0) \leq \epsilon$, the Metropolis-Hastings algorithm involves sampling candidate states $x = (\theta, y)$ from a proposal distribution $q(x|x^{(i)})$ and accepting the candidates with the probability $A(x|x^{(i)})$,

$$A(x|x^{(i)}) = \min\left(1, \frac{p_{d,\epsilon}(x|y_0)q(x^{(i)}|x)}{p_{d,\epsilon}(x^{(i)}|y_0)q(x|x^{(i)})}\right). \tag{9}$$

Choosing the proposal distribution such that the move from $x^{(i)} = (\theta^{(i)}, y^{(i)})$ to $x = (\theta, y)$ does not depend on the value of $y^{(i)}$, and that y is sampled from the simulator-based model with parameter value θ irrespective of $\theta^{(i)}$, we have

$$q(x|x^{(i)}) = q(\theta|\theta^{(i)})p(y|\theta), \tag{10}$$

where $q(\theta|\theta^{(i)})$ is a suitable proposal distribution for θ. As a result of this choice, the unknown quantities in Equation (9) cancel out,

$$\begin{aligned} A(x|x^{(i)}) &= \min\left(1, \frac{p(\theta)}{p(\theta^{(i)})}\frac{p(y|\theta)}{p(y^{(i)}|\theta^{(i)})}\frac{\mathbb{1}[d(y,y_0)\leq\epsilon]}{\mathbb{1}[d(y^{(i)},y_0)\leq\epsilon]}\frac{q(\theta^{(i)}|\theta)}{q(\theta|\theta^{(i)})}\frac{p(y^{(i)}|\theta^{(i)})}{p(y|\theta)}\right) \qquad (11)\\ &= \min\left(1, \frac{p(\theta)}{p(\theta^{(i)})}\frac{q(\theta^{(i)}|\theta)}{q(\theta|\theta^{(i)})}\frac{\mathbb{1}[d(y,y_0)\leq\epsilon]}{\mathbb{1}[d(y^{(i)},y_0)\leq\epsilon]}\right) \qquad (12)\\ &= \mathbb{1}[d(y,y_0)\leq\epsilon]\min\left(1, \frac{p(\theta)}{p(\theta^{(i)})}\frac{q(\theta^{(i)}|\theta)}{q(\theta|\theta^{(i)})}\right). \qquad (13) \end{aligned}$$

This means that the acceptance probability is only probabilistic in θ since a proposal (θ, y) is immediately rejected if the condition $d(y, y_0) \leq \epsilon$ is not met. While the Markov chain operates in the (θ, y) space, the choice of the proposal distribution decouples the acceptance criterion into an ordinary Metropolis-Hastings criterion for θ and the previously seen ABC rejection criterion for y. The resulting algorithm, shown in full in the appendix, is known as MCMC ABC algorithm and was introduced by Marjoram et al. (2003).

An advantage of the MCMC ABC algorithm is that the parameter values do not need to be drawn from the prior, which most often hampers the rejection sampler by incurring a high rejection rate of the proposals. As the Markov chain convergences, the proposed parameter values follow the posterior with some added noise. A potential disadvantage, however, is the continuing presence of the rejection condition $d(y, y_0) \leq \epsilon$ which dominates the acceptance rate of the algorithm. Parameters in the tails of the posteriors have, by definition, a small probability to generate data y_θ satisfying the rejection condition, which can lead to a "sticky" Markov chain where the state tends to remain constant for many iterations.

Sequential Monte Carlo ABC

The sequential Monte Carlo ABC algorithm can be considered as an adaptation of sequential importance sampling (see, for example, Robert and Casella 2004, Chapter 3). If one uses a general distribution $\phi(\theta)$ in place of the prior $p(\theta)$, Algorithm 2 produces samples which follow a distribution proportional to $\phi(\theta) \Pr(d(Y_\theta, y_0) \leq \epsilon)$. However, by weighting the accepted parameters $\theta^{(i)}$ with $w^{(i)}$,

$$w^{(i)} \propto \frac{p(\theta^{(i)})}{\phi(\theta^{(i)})}, \tag{14}$$

the resulting weighted samples follow $p_{d,\epsilon}(\theta|y_0)$. This kind of trick is used in importance sampling and can be employed in ABC to iteratively morph the prior into a posterior.

The basic idea is to use a sequence of shrinking thresholds ϵ_t and to define the proposal distribution ϕ_t at iteration t based on the weighted samples $\theta_{t-1}^{(i)}$ from the previous iteration. This is typically done by defining a mixture distribution,

$$\phi_t(\theta) = \frac{1}{N} \sum_{i=1}^{N} q_t(\theta|\theta_{t-1}^{(i)}) w_{t-1}^{(i)}, \tag{15}$$

where $q_t(\theta|\theta_{t-1}^{(i)})$ is often a Gaussian distribution with mean $\theta_{t-1}^{(i)}$ and a covariance matrix estimated from the samples. Sampling from ϕ_t can be done by choosing $\theta_{t-1}^{(i)}$ with probability $w_{t-1}^{(i)}$ and then perturbing the chosen parameter according to q_t. The proposed sample is then accepted or rejected as in Algorithm 2. Such iterative algorithms were proposed by Sisson et al. (2007); Beaumont et al. (2009); Toni et al. (2009) and are called Sequential Monte Carlo (SMC) ABC algorithms or Population Monte Carlo (PMC) ABC algorithms. The algorithm by Beaumont et al. (2009) is given in the appendix.

Similar to the MCMC ABC, after convergence the samples proposed by the SMC algorithm follow the posterior $p_{d,\epsilon}(\theta|y_0)$ with some added noise, thereby improving on the basic rejection ABC algorithm. For small values of ϵ, however, the probability to accept a parameter value becomes very small, even if the parameter value was sampled from the true posterior. This results in long computing times in the final iterations of the algorithm without notable improvements in the approximation of the posterior.

Post-sampling correction methods

We assume here that the distance $d(y_\theta, y_0)$ is specified in terms of summary statistics, that is, $d(y_\theta, y_0) = d_T(t_\theta, t_0)$, where $t_\theta = T(y_\theta)$ and $t_0 = T(y_0)$. When ϵ decreases to zero, the approximate posterior $p_{d,\epsilon}(\theta|y_0)$ in Equation (5) converges towards $p(\theta|t_0)$, where we use $p(\theta|t)$ to denote the conditional distribution of θ given a value of the summary statistics t.

We have seen that small values of ϵ are preferred in theory since the approximation has a small bias, but making them too small is not feasible in practice because of the correspondingly small acceptance rate and the resulting large Monte Carlo error. But if we could sample from $p(\theta|t)$, we could sample from the limiting approximate posterior by using $t = t_0$. The post-sampling correction methods presented here boil down to estimating $p(\theta|t)$ and using the estimated conditional distributions to sample from $p(\theta|t_0)$.

In order to facilitate sampling, $p(\theta|t)$ is expressed in terms of a generative (regression) model,

$$\theta = f(t, \xi), \tag{16}$$

where f is a vector valued function and ξ a vector of random variables for residuals. By suitably defining f, we can assume that the random variables of the vector ξ are independent, of zero mean and equal variance, and that their distribution p_ξ does not depend on t. Importantly, the model does not need to hold for all t because, ultimately, we would like to sample from it using $t = t_0$ only. Assuming that the model holds for $d_T(t, t_0) \leq \delta$ and that we have (weighted) samples $(t^{(i)}, \tilde{\theta}^{(i)}) = (T(y_\theta{}^{(i)}), \tilde{\theta}^{(i)})$ available from a previous iteration of an ABC algorithm with threshold $\delta > \epsilon$, the model can be estimated by regressing θ on the summary statistics t.

In order to sample θ using the estimated model $\hat{f}$, we need to know the distribution of ξ. For that, the residuals $\xi^{(i)}$ are determined by solving the regression equation,

$$\tilde{\theta}^{(i)} = \hat{f}(t^{(i)}, \xi^{(i)}). \tag{17}$$

The residuals $\xi^{(i)}$ can be used to estimate p_ξ, or as usually is the case in ABC, be directly employed in the sampling of the θ,

$$\theta^{(i)} = \hat{f}(t_0, \xi^{(i)}). \tag{18}$$

If the original samples $(t^{(i)}, \tilde{\theta}^{(i)})$ are weighted, both the $\xi^{(i)}$ and the new "adjusted" samples $\theta^{(i)}$ inherit the weights. By construction, if the relation between t and θ is estimated correctly, the (weighted) samples $\theta^{(i)}$ follow $p_{d,\epsilon}(\theta|y_0)$ with $\epsilon = 0$.

In most models f employed so far, the individual components of θ are treated separately, thus not accounting for possible correlations between them. For this paragraph we thus let θ be a scalar. The first regression model used was linear (Beaumont et al. 2002),

$$\theta = f_1(t, \xi), \qquad\qquad f_1(t, \xi) = \alpha + (t - t_0)^\top \beta + \xi, \tag{19}$$

which results in the adjustment $\theta^{(i)} = \tilde{\theta}^{(i)} - (t^{(i)} - t_0)^\top \hat{\beta}$ where $\hat{\beta}$ is the learned regression coefficient. Blum (2010) assumed a quadratic model,

$$\theta = f_2(t, \xi), \qquad f_2(t, \xi) = \alpha + (t - t_0)^\top \beta + \frac{1}{2}(t - t_0)^\top \gamma (t - t_0) + \xi, \tag{20}$$

where γ is a symmetric matrix, which adds a quadratic term to the linear adjustment. A more general nonlinear model was considered by Blum and François (2010),

$$\theta = f_4(t, \xi), \qquad f_4(t, \xi) = m(t) + \sigma(t)\xi, \tag{21}$$

where $m(t)$ models the conditional mean and $\sigma(t)$ the conditional standard deviation of θ. Both functions were fitted using a multi-layer neural network, and denoting the learned functions by $\hat{m}$ and $\hat{\sigma}$, the following adjustments were obtained

$$\theta^{(i)} = \hat{m}(t_0) + \hat{\sigma}(t_0)\hat{\sigma}(t^{(i)})^{-1}(\tilde{\theta}^{(i)} - \hat{m}(t^{(i)})). \tag{22}$$

The term $\hat{m}(t_0)$ is an estimate of the posterior mean of θ while $\hat{\sigma}(t_0)$ is an estimate of the posterior standard deviation of the parameter. They can both be used to succinctly summarize the posterior distribution of θ.

A more complicated model $f(t, \xi)$ is not necessarily better than a simpler one. It depends on the amount of the training data available to fit it, that is, the amount of original samples $(t^{(i)}, \tilde{\theta}^{(i)})$ which satisfy $d_T(t, t_0) \leq \delta$. The different models presented above were compared by Blum and François (2010) who also pointed out that techniques for model selection from the regression literature can be used to select among them.

Recent developments

We here present recent advances which aim to make approximate Bayesian computation both computationally and statistically more efficient. This presentation focuses on our own work (Gutmann et al. 2014; Gutmann and Corander 2015).

Computational efficiency

The computational cost of ABC can be attributed to two main factors:

1. While the distance between simulated and observed data is large for most parameters, it is unknown which parameter values tend to result in small distances.

2. Generating simulated data sets, that is, running the simulator, may be costly because of the detailed model.

MCMC ABC and SMC ABC were partly introduced to avoid proposing parameters in regions where the distance is large. Nonetheless, typically millions of simulations are needed to infer the posterior distribution of a handful of parameters only. A key obstacle to efficiency in these algorithms is the continued presence of the rejection mechanism $d(y_\theta, y_0) \leq \epsilon$, or more generally, the online decisions about the similarity between y_θ and y_0. In recent work, Gutmann and Corander (2015) proposed a framework called Bayesian optimization for likelihood-free inference (BOLFI) for performing approximate Bayesian computation which overcomes this obstacle by learning a probabilistic model about the stochastic relation between the parameter values and the distance $d(Y_\theta, y_0)$. After learning, the model can be used to approximate $L_{d,\epsilon}(\theta)$, and thus $p_{d,\epsilon}(\theta|y_0)$, for any ϵ without requiring further runs of the simulator.

Like the post-sampling rejection methods presented in the previous section, BOLFI relies on a probabilistic model to make ABC more efficient. However, the quantities modeled differ, since in the post-sampling rejection methods the relation between summary statistics and parameters is modeled, while BOLFI focuses on the relation between the parameters and the distance. A potential advantage of the latter approach is that the distance is a univariate quantity while the parameters may be multi-dimensional. Furthermore, it does not assume that the distance is defined via summary statistics.

For the learning of the model of $d(Y_\theta, y_0)$, data about the relation between θ and $d(Y_\theta, y_0)$ are needed. In BOLFI, the data are actively acquired focusing on regions of the parameter space where the distance tends to be small. This is achieved by leveraging techniques from Bayesian optimization (see for example Jones 2001; Brochu et al. 2010), hence its name. Ultimately, the framework provided by Gutmann and Corander (2015) reduces the computational cost of ABC by addressing both of the factors mentioned above. The first point is addressed by learning from data which parameter values tend to have small distances, while the second problem is resolved by focusing on areas where the distance tends to be small when learning the model and by not requiring further runs of the simulator once the model is learned.

While BOLFI is not restricted to a particular model for $d(Y_\theta, y_0)$, Gutmann and Corander (2015) used Gaussian processes in the applications in their paper. Gaussian processes have also been used in other work as surrogate models for quantities which are expensive to compute. Wilkinson (2014) used them to model the logarithm of $L_{d,\epsilon}(\theta)$, and the training data were constructed based on quasi-random numbers covering the parameter space. Making a Gaussianity assumption about the distribution of the summary statistics (Wood 2010; Leuenberger and Wegmann 2010), Meeds and Welling (2014) used Gaussian processes to model the empirical mean and covariances of the summary statistics, and the resulting likelihood approximation was used together with a Markov chain Monte Carlo algorithm. All these approaches have been demonstrated to assist in speeding up approximate Bayesian computation.

Statistical efficiency

We have seen that the statistical efficiency of ABC algorithms depends heavily on the summary statistics chosen, the distance between them, and the locality of the inference. In a recent work, Gutmann et al. (2014) formulated the problem of measuring the distance between simulated and observed data as a classification problem, using the maximal classification accuracy as an indicator of the similarity (50% classification accuracy indicating similarity, 100% accuracy dissimilarity).

The classification rule used to measure the distance was learned from the data, which simplifies the inference since only a function (hypotheses) space needs to be pre-specified by the user. In the process Gutmann et al. (2014) also chose a subset or weighted (nonlinear) combination of summary statistics, to achieve the best classification accuracy. This choice depended on the parameter values used to generate the simulated data. While computationally more expensive than the traditional approach, the classifier approach has the advantage of being a data-driven way to measure the distance between the simulated and observed data which respects the locality of the inference. From the theoretical perspective, the classifier-based approach provides additional insight to the inference problem, since the resulting parameter estimator is consistent under mild regularity conditions.

Validation of approximate Bayesian computation

Due to the several levels of approximation, it is generally a recommendable practice to perform validatory analysis of the ABC inferences. We here discuss some of the possibilities suggested in the literature.

The ability to generate data from simulator-based models enables basic sanity checks for the feasibility of the inference with a given setting and algorithm. The general approach is to perform inference where synthetic data sets y_0^* are generated with known parameter values θ^* to play the role of the observed data y_0. To assess whether the posterior distribution is concentrated around the right parameter values, one may then compute the average error between the posterior mean (mode) and θ^*, or the expected squared distance between the posterior samples and θ^* (Wegmann et al. 2009). To assess whether the spread of the posterior distribution is not overly large or small, one may compute confidence (credibility) intervals and check their coverage. When the nominal confidence levels are accurate, 95% confidence intervals, for example, should contain θ^* in 95% of the simulation experiments (Wegmann et al. 2009; Prangle et al. 2014). Such tests can be performed *a priori*, by sampling y_0^* from the prior before having seen the actual data to be analyzed, or also *a posteriori*, by sampling y_0^* from the inferred posterior or from the prior restricted to some area of interest (Prangle et al. 2014). Corresponding techniques have also been suggested for the purpose of specifying the threshold value ϵ as discussed earlier in this paper. As suggested there, it can be also beneficial here to store the generated data sets together with their parameter values so that the validations can be run without having to re-generate new data on every occasion.

The likelihood function indicates to which extent parameter values are congruent with the observed data. A strong curvature at its maximum indicates that the maximizing parameter value is clearly to be preferred while a minor curvature means that several other parameter values are nearly equally supported by the data. More generally, if the likelihood surface is mostly flat over the parameter space, the data are not providing sufficient information to identify the model parameters. While the likelihood function is generally not available for simulator-based models, the arguments provided do also hold for the approximate likelihood function $L_{d,\epsilon}(\theta)$ in Equation (6). On one hand, the approximate likelihood function can be used to investigate the identifiability of the simulator-based model. On the other hand, it allows one to assess the quality of the distance d or threshold

ϵ chosen. Flat approximate likelihood surfaces, for instance, indicate that ϵ could be too large or that the distance function d is not able to accurately measure differences between the data sets.

The approximate likelihood $L_{d,\epsilon}(\theta)$ can be obtained either by the method of Gutmann and Corander (2015) or also by any other ABC algorithm by assuming a uniform prior on a region of interest. Lintusaari et al. (2015) used such an approach to investigate the identifiability of the tuberculosis model considered as example in the previous sections, and to compare different distance functions. Further, one may (visually) compare the (marginal) prior and the inferred (marginal) posterior (e.g. Blum 2010). Both approaches are applicable not only to the real observed data y_0 but also to the synthetic data y_0^* for which the data-generating parameters θ^* are known. If the employed ABC algorithm is working appropriately, both $L_{d,\epsilon}(\theta)$ and the posteriors should clearly change when the characteristics of the observed data change markedly. In particular, if the number of observations is increased, the approximate likelihood and posterior should in general become more concentrated around the data-generating parameter values. While failure to pass such sanity checks may be an indicator that the choice of d and ϵ could be improved, it can also indicate that the model may not be fully identifiable.

Conclusions

With the use of stochastic simulation models, it is possible to simulate complex biological phenomena in a realistic manner. However the likelihood function for such models is usually intractable and serious methodological challenges are faced in performing statistical inference. Approximate Bayesian computation has become synonymous for approximate Bayesian inference for simulator-based models. We have here reviewed its foundations, the most widely considered inference algorithms, together with recent advances which increase its statistical and computational efficiency.

While the review is solely restricted to Bayesian methods, there exists a large body of literature on non-Bayesian approaches, for instance, the methods of simulated moments (Pakes and Pollard 1989; McFadden 1989) or indirect inference (Gouriéroux et al. 1993; Heggland and Frigessi 2004), both having their origin in econometrics.

We focused on the central topics related to parameter inference with ABC. Nevertheless, ABC is also applicable to model selection (see, for example, the review by Marin et al. 2012), and while we have reviewed methods making the basic ABC algorithms more efficient, we have not discussed the important topic of how to use ABC for high-dimensional inference. We point the interested readers to the work by Li et al. (2015) and also to the discussion by Gutmann and Corander (2015).

For practical purpose, there exist multiple software packages implementing the different ABC algorithms, summary statistic selection, validation methods, post processing, and ABC model selection methods. Nunes and Prangle (2015) provide a recent list of available packages with information about their implementation language, platform and targeted field of study. In summary, approximate Bayesian computation is currently a very active methodological research field, and this activity will likely result in several advances to improve its applicability to answering important biological research questions in the near future.

Acknowledgements

We acknowledge the computational resources provided by the Aalto Science-IT project.

Funding

This work was supported by the Academy of Finland (Finnish Centre of Excellence in Computational Inference Research COIN).

Author contributions: JL, MUG, RD and JC wrote the paper; JL performed the simulations; SK contributed to writing and planning of the paper.

*

References

Aeschbacher, S., M. Beaumont, and A. Futschik. 2012. A novel approach for choosing summary statistics in approximate Bayesian computation. Genetics 192:1027–1047.

Anderson, R. M. and R. M. May. 1992. Infectious Diseases of Humans: Dynamics and Control. Oxford University Press.

Barber, S., J. Voss, and M. Webster. 2015. The rate of convergence for approximate Bayesian computation. Electronic Journal of Statistics Pages 80–105.

Beaumont, M. A., J.-M. Cornuet, J.-M. Marin, and C. P. Robert. 2009. Adaptive approximate Bayesian computation. Biometrika 96:983–990.

Beaumont, M. A., W. Zhang, and D. J. Balding. 2002. Approximate Bayesian computation in population genetics. Genetics 162:2025–2035.

Biau, G., F. Cérou, and A. Guyader. 2015. New insights into approximate Bayesian computation. Annales de l'Institut Henri Poincaré (B) Probabilités et Statistiques 51:376–403.

Blum, M. and O. François. 2010. Non-linear regression models for approximate Bayesian computation. Statistics and Computing 20:63–73.

Blum, M. G. B. 2010. Approximate Bayesian computation: A nonparametric perspective. Journal of the American Statistical Association 105:1178–1187.

Blum, M. G. B., M. A. Nunes, D. Prangle, and S. A. Sisson. 2013. A comparative review of dimension reduction methods in approximate Bayesian computation. Statistical Science 28:189–208.

Brochu, E., V. Cora, and N. de Freitas. 2010. A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. arXiv:1012.2599 .

Currat, M. and L. Excoffier. 2004. Modern humans did not admix with neanderthals during their range expansion into europe. PLoS Biol 2:e421.

Diggle, P. J. and R. J. Gratton. 1984. Monte carlo methods of inference for implicit statistical models. Journal of the Royal Statistical Society. Series B (Methodological) 46:193–227.

Excoffier, L., I. Dupanloup, E. Huerta-Snchez, V. C. Sousa, and M. Foll. 2013. Robust demographic inference from genomic and snp data. PLoS Genet 9:e1003905.

Fagundes, N. J. R., N. Ray, M. Beaumont, S. Neuenschwander, F. M. Salzano, S. L. Bonatto, and L. Excoffier. 2007. Statistical evaluation of alternative models of human evolution. Proceedings of the National Academy of Sciences 104:17614–17619.

Faisal, M., A. Futschik, and I. Hussain. 2013. A new approach to choose acceptance cutoff for approximate Bayesian computation. Journal of Applied Statistics 40:862–869.

Fearnhead, P. and D. Prangle. 2012. Constructing summary statistics for approximate Bayesian computation: semi-automatic approximate Bayesian computation. Journal of the Royal Statistical Society: Series B (Statistical Methodology) 74:419–474.

Gouriéroux, C., A. Monfort, and E. Renault. 1993. Indirect inference. Journal of Applied Econometrics 8:S85–S118.

Green, P., K. Latuszynski, M. Pereyra, and C. P. Robert. 2015. Bayesian computation: a summary of the current state, and samples backwards and forwards. Statistics and Computing 25:835–862.

Gutmann, M. and J. Corander. 2015. Bayesian optimization for likelihood-free inference of simulator-based statistical models. Journal of Machine Learning Research in press.

Gutmann, M., R. Dutta, S. Kaski, and J. Corander. 2014. Statistical inference of intractable generative models via classification. arXiv:1407.4981 .

Heggland, K. and A. Frigessi. 2004. Estimating functions in indirect inference. Journal of the Royal Statistical Society: Series B (Statistical Methodology) 66:447–462.

Itan, Y., A. Powell, M. A. Beaumont, J. Burger, and M. G. Thomas. 2009. The origins of lactase persistence in europe. PLoS Comput Biol 5:e1000491.

Jones, D. 2001. A taxonomy of global optimization methods based on response surfaces. Journal of Global Optimization 21:345–383.

Joyce, P. and P. Marjoram. 2008. Approximately sufficient statistics and Bayesian computation. Statistical Applications in Genetics and Molecular Biology 26.

Leuenberger, C. and D. Wegmann. 2010. Bayesian computation and model selection without likelihoods. Genetics 184:243–252.

Li, J., D. J. Nott, and S. A. Sisson. 2015. Extending approximate Bayesian computation methods to high dimensions via Gaussian copula. arXiv:1504.04093 .

Lintusaari, J., M. U. Gutmann, S. Kaski, and J. Corander. 2015. On the identifiability of transmission dynamic models for infectious disease. bioRxiv doi:10.1101/021972 .

Marin, J.-M., P. Pudlo, C. Robert, and R. Ryder. 2012. Approximate Bayesian computational methods. Statistics and Computing 22:1167–1180.

Marjoram, P., J. Molitor, V. Plagnol, and S. Tavar. 2003. Markov chain Monte Carlo without likelihoods. Proceedings of the National Academy of Sciences 100:15324–15328.

Marttinen, P., N. J. Croucher, M. U. Gutmann, J. Corander, and W. P. Hanage. 2015. Recombination produces coherent bacterial species clusters in both core and accessory genomes. Microbial Genomics. Published ahead of print.

McFadden, D. 1989. A method of simulated moments for estimation of discrete response models without numerical integration. Econometrica 57:995–1026.

Meeds, E. and M. Welling. 2014. GPS-ABC: Gaussian process surrogate approximate Bayesian computation. *in* Proceedings of the 30th Conference on Uncertainty in Artificial Intelligence (UAI).

Nunes, M. A. and D. J. Balding. 2010. On optimal selection of summary statistics for approximate Bayesian computation. Statistical Applications in Genetics and Molecular Biology 9.

Nunes, M. A. and D. Prangle. 2015. abctools: An R package for tuning approximate Bayesian computation analyses. The R Journal. To appear.

Pakes, A. and D. Pollard. 1989. Simulation and the asymptotics of optimization estimators. Econometrica 57:1027–1057.

Prangle, D., M. G. B. Blum, G. Popovic, and S. A. Sisson. 2014. Diagnostic tools for approximate Bayesian computation using the coverage property. Australian & New Zealand Journal of Statistics 56:309–329.

Pritchard, J. K., M. T. Seielstad, A. Perez-Lezaun, and M. W. Feldman. 1999. Population growth of human Y chromosomes: a study of Y chromosome microsatellites. Molecular Biology and Evolution 16:1791–1798.

Robert, C. and G. Casella. 2004. Monte Carlo Statistical Methods. 2 ed. Springer.

Rubin, D. B. 1984. Bayesianly justifiable and relevant frequency calculations for the applied statistician. The Annals of Statistics 12:1151–1172.

Silk, D., S. Filippi, and M. P. Stumpf. 2013. Optimizing threshold-schedules for sequential approximate bayesian computation: applications to molecular systems. Statistical applications in genetics and molecular biology 12:603–618.

Sisson, S. A., Y. Fan, and M. M. Tanaka. 2007. Sequential Monte Carlo without likelihoods. Proceedings of the National Academy of Sciences 104:1760–1765.

Stadler, T. 2011. Inferring epidemiological parameters on the basis of allele frequencies. Genetics 188:663–672.

Tanaka, M. M., A. R. Francis, F. Luciani, and S. A. Sisson. 2006. Using approximate Bayesian computation to estimate tuberculosis transmission parameters from genotype data. Genetics 173:1511–1520.

Toni, T., D. Welch, N. Strelkowa, A. Ipsen, and M. P. Stumpf. 2009. Approximate Bayesian computation scheme for parameter inference and model selection in dynamical systems. Journal of The Royal Society Interface 6:187–202.

Wegmann, D., C. Leuenberger, and L. Excoffier. 2009. Efficient approximate Bayesian computation coupled with Markov chain Monte Carlo without likelihood. Genetics 182:129–141.

Wilkinson, R. 2014. Accelerating ABC methods using Gaussian processes. *in* Proceedings of the 17th International Conference on Artificial Intelligence and Statistics (AISTATS).

Wood, S. 2010. Statistical inference for noisy nonlinear ecological dynamic systems. Nature 466:1102–1104.

Appendix

Algorithm 3 MCMC-ABC algorithm producing N samples from the approximate posterior distribution $p_{d,\epsilon}(\theta|y_0)$

```
Require: Set the initial value $\theta^{(0)}$
for $i = 1$ to $N$ do
    Generate $\theta$ from a transition kernel $q(\cdot|\theta^{(i-1)})$
    Generate $y_\theta$ from the simulator
    if $d(y_\theta, y_0) \leq \epsilon$ then
        Calculate $A = A(\theta|\theta^{(i-1)}) = p(\theta)q(\theta^{(i-1)}|\theta)/(p(\theta^{(i-1)})q(\theta|\theta^{(i-1)}))$
        Generate $u$ from $\mathrm{Uni}(0, 1)$
        if $u < A$ then
            $\theta^{(i)} \leftarrow \theta$
            Continue to next iteration
        end if
    end if
    $\theta^{(i)} \leftarrow \theta^{(i-1)}$
end for
```

Algorithm 4 ABC-PMC algorithm producing N samples from the approximate posterior distribution $p_{d,\epsilon}(\theta|y_0)$. Here $q_t(\theta|\theta_{t-1}^{(i)}) = N(\theta|\theta_{t-1}^{(i)}, \Sigma_{t-1})$.

```
Require: Specify a decreasing sequence of thresholds $\epsilon_1 \geq \epsilon_2 \geq \cdots \geq \epsilon_T$ for $T$ iterations.
for $i = 1$ to $N$ do
    repeat
        Generate $\theta$ from the prior $p(\cdot)$
        Generate $y_\theta$ from the simulator
    until $d(y_\theta, y_0) \leq \epsilon_1$
    $\theta_1^{(i)} \leftarrow \theta$
    $\omega_1^{(i)} \leftarrow 1/N$
end for
$\Sigma_1 \leftarrow 2\,\mathrm{Cov}(\theta_1)$ {Twice the empirical variance}

for $t = 2$ to $T$ do
    for $i = 1$ to $N$ do
        repeat
            Draw $\theta^*$ from among $\theta_{t-1}$ with probabilities $\omega_{t-1}$
            Generate $\theta$ from $\mathcal{N}(\theta^*, \Sigma_{t-1})$
            Generate $y_\theta$ from the simulator
        until $d(y_\theta, y_0) \leq \epsilon_t$
        $\theta_t^{(i)} \leftarrow \theta$
        $\omega_t^{(i)} \leftarrow p(\theta)/(\sum_{k=1}^{N} \omega_{t-1}^{(k)} \mathcal{N}(\theta|\theta_{t-1}^{(k)}, \Sigma_{t-1}))$ {weights can be scaled with a constant}
    end for
    $\Sigma_t \leftarrow 2\,\mathrm{Cov}(\theta_t)$ {Twice the empirical variance}
end for
```