Prangle D. Lazy ABC. ***Statistics and Computing*** 2016, 26(1), 171-185.

Lazy ABC

Dennis Prangle*

Abstract

Approximate Bayesian computation (ABC) performs statistical inference for otherwise intractable probability models by accepting parameter proposals when corresponding simulated datasets are sufficiently close to the observations. Producing the large quantity of simulations needed requires considerable computing time. However, it is often clear before a simulation ends that it is unpromising: it is likely to produce a poor match or require excessive time. This paper proposes lazy ABC, an ABC importance sampling algorithm which saves time by sometimes abandoning such simulations. This makes ABC more scalable to applications where simulation is expensive. By using a random stopping rule and appropriate reweighting step, the target distribution is unchanged from that of standard ABC. Theory and practical methods to tune lazy ABC are presented and illustrated on a simple epidemic model example. They are also demonstrated on the computationally demanding spatial extremes application of Erhardt and Smith (2012), producing efficiency gains, in terms of effective sample size per unit CPU time, of roughly 3 times for a 20 location dataset, and 8 times for 35 locations.

Keywords: importance sampling, ABC, unbiased likelihood estimators, epidemics, spatial extremes

*University of Reading. Email d.b.prangle@reading.ac.uk

1 Introduction

Approximate Bayesian computation (ABC) algorithms are a popular method of inference for a wide class of otherwise intractable probability models in applications such as population genetics, ecology, epidemiology and systems biology (Beaumont, 2010; Marin et al., 2012). They select parameter vectors θ for which datasets y simulated from the model of interest are sufficiently close to the observations. A bottleneck is the computational cost of producing the large quantity of model simulations needed, which becomes increasingly severe for more detailed models. However, it is often clear during a simulation that it is unpromising. For example it is likely to produce a poor match or to require excessive computation time. This paper presents *lazy ABC*, an importance sampling method which abandons some such simulations, a step referred to as *early stopping*, exploiting information from incomplete simulations to save time. The result is an ABC algorithm which is more scalable to applications where simulation is computationally demanding.

In more detail, standard ABC is based on a random likelihood estimator $\hat{L}_{\text{ABC}}$, which is 1 for a close match of simulated and observed data, and zero otherwise. The algorithm can be shown to target a distribution corresponding to the approximate likelihood $L_{\text{ABC}}(\theta) = \text{E}[\hat{L}_{\text{ABC}}|\theta]$. Lazy ABC is based on an alternative estimator $\hat{L}_{\text{lazy}}$. This equals zero with probability $1-\alpha$ – if early stopping is performed – and otherwise equals the $\hat{L}_{\text{ABC}}$ estimator multiplied by a weight. Letting the weight equal $1/\alpha$ makes $\hat{L}_{\text{lazy}}$ an unbiased estimator of $L_{\text{ABC}}(\theta)$. Results on random likelihood estimates show that importance sampling algorithms based on $\hat{L}_{\text{lazy}}(\theta)$ therefore target the same distribution as standard ABC. No further approximation has been introduced.

The lazy ABC estimator trades off an increase in variance for a reduction in computation time. It is shown that for this to be most advantageous α should be (1) larger when there is a high probability of the simulated dataset being a close match to the observations (2) smaller when the expected time to complete the simulation is large. To achieve this, α is based on X, a random variable encapsulating some preliminary information about the simulated dataset

Y, also a random variable. The final likelihood estimator is based on X and Y. However when early stopping occurs a realised value of zero is obtained without drawing a value of Y. This is an example of the computer science concept of *lazy evaluation* (Ralston et al., 2003, "functional programming" entry), which is the basis for the method's name.

The paper presents theoretical results on the optimal tuning of α in lazy ABC, making precise the two properties just outlined. This choice is asymptotically optimal in terms of maximising efficiency, which is defined as effective sample size (ESS) per unit CPU time. Based on this, a framework for tuning in practice is also presented. The main requirements are the estimation of the probability of close matches and of expected remaining computation times. Both of these are conditional on θ and x (realised values of X). As x is typically high dimensional this estimation is not feasible, so instead it is recommended to base the choice of α on a vector of low dimensional *decision statistics* $\phi(\theta, x)$. A computationally demanding example is presented based on the spatial extremes application of Erhardt and Smith (2012) where lazy ABC increases the efficiency by roughly 3 times for modestly sized data and 8 times for a larger example.

It should be emphasised that this approach to tuning is *optional*. In many applications a simple ad-hoc choice of α may produce significant efficiency gains, especially where application knowledge can be used to create heuristic criteria which quickly identify many unpromising simulations.

The focus of this paper is on importance sampling, which is widely used by ABC practitioners and particularly amenable to parallelisation. However the lazy ABC approach is also applicable to other algorithms, such as Markov chain Monte Carlo (MCMC) and sequential Monte Carlo (SMC), as discussed in the final section.

Several recent papers have proposed speeding up ABC by fitting a model to (θ, y) pairs, simulated either in a preliminary stage or in earlier ABC iterations (Buzbas and Rosenberg, 2013; Meeds and Welling, 2014; Moores et al., 2014; Wilkinson, 2014). This model is then sometimes or always used in place of the original model of interest in the inference algorithm.

A potential application of lazy ABC is to make use of such approximate models (their predictions given θ forming X in the notation above) to gain speed benefits without incurring additional approximation errors. More generally, there has been much interest over the past decade in Bayesian inference algorithms with random weights (e.g. Beaumont, 2003; Andrieu and Roberts, 2009; Fearnhead et al., 2010; Tran et al., 2014). A novelty of lazy ABC is that it introduces a random factor to the weights to reduce computation time, rather than to deal with intractability.

Finally, a byproduct of the paper's theory, Corollary 1, is of independent interest. This gives importance densities which optimise asymptotic ESS per unit CPU time for: (a) ABC importance sampling (b) importance sampling with random weights.

The remainder of the paper is structured as follows. Section 2 contains background material on ABC and importance sampling. Section 3 gives the lazy ABC algorithm and proves it targets the correct distribution. Section 4 presents theory and practical methods for tuning the algorithm, as well as the corollary mentioned above. Section 5 illustrates the method for a simple epidemic model. R code to implement this is available at `https://github.com/dennisprangle/lazyABCexample`. Section 6 contains a more challenging spatial extremes application and Section 7 is a discussion. Appendices contain proofs and discuss some extensions: lazy ABC with multiple stopping decisions and application of similar methods outside ABC.

2 Importance sampling

Consider analysing data y_{obs} under a probability model with density $\pi(y|\theta)$ and parameters θ. The likelihood is defined as $L(\theta) = \pi(y_{\text{obs}}|\theta)$. Bayesian inference introduces a prior distribution with density $\pi(\theta)$ and aims to find the posterior distribution $\pi(\theta|y_{\text{obs}}) = \pi(\theta)L(\theta)/\pi(y_{\text{obs}})$, where $\pi(y_{\text{obs}}) = \int \pi(\theta)L(\theta)d\theta$, or at least to estimate the posterior expectation $\mathrm{E}[h(\theta)|y_{\text{obs}}]$ of a generic function $h(\theta)$. Importance sampling is a simple method to do

this. Parameter values $\theta_{1:N}$ are simulated independently from an importance density $g(\theta)$ and given weights $w_i = L(\theta_i)\pi(\theta_i)/g(\theta_i)$ (n.b. $\theta_{1:N}$ represents the sequence $(\theta_i)_{1\leq i\leq N}$. Similar notation is used later.) It is assumed throughout that $g(\theta) > 0$ whenever $\pi(\theta) > 0$. Each of the (θ_i, w_i) pairs can be computed in parallel, allowing for efficient implementation.

A Monte Carlo estimate of $\mathrm{E}[h(\theta)|y_{\mathrm{obs}}]$ is $\mu_h = \frac{\sum_{i=1}^N h(\theta_i)w_i}{\sum_{i=1}^N w_i}$. Two properties of importance sampling estimates are

$$\mu_h \to \mathrm{E}[h(\theta)|y_{\mathrm{obs}}] \quad \text{almost surely as } N \to \infty, \tag{1}$$

$$E[N^{-1}\sum_{i=1}^{N} w_i] = \pi(y_{\mathrm{obs}}). \tag{2}$$

See Geweke (1989) for proof that (1) holds under weak conditions. To prove (2) note that each w_i is an unbiased estimator of $\pi(y_{\mathrm{obs}})$. Estimating this is of interest for model comparison.

2.1 Notation

The remainder of the paper is largely concerned with the distribution of random variables produced in an iteration of various importance sampling algorithms. Henceforth, expectations and probabilities that involve quantities produced by importance sampling should be read as being with respect to this distribution. In particular this means that below the marginal density of θ is taken to be $g(\theta)$. The preceding material in this section is the only time that a marginal density of $\pi(\theta)$ is used instead.

2.2 Random weights

Algorithm 1 describes random weight importance sampling (RW-IS), an importance sampling algorithm in which likelihood evaluations are replaced with random estimates of the likelihood. Under the condition that these estimates are non-negative and unbiased, the algorithm produces valid output, in the sense that (1) and (2) continue to hold. This can be seen by noting that Algorithm 1 is equivalent to a deterministic weight importance sampling

algorithm with augmented parameters (θ, ℓ), prior density $\pi(\theta)\pi(\ell|\theta)$, importance density $g(\theta)\pi(\ell|\theta)$ and likelihood ℓ. Here ℓ is the realisation of the likelihood estimator and $\pi(\ell|\theta)$ is the conditional density of this estimator. This algorithm gives the correct marginal posterior for θ. See Tran et al. (2014) for a more detailed proof and Fearnhead et al. (2010) for further background (including the observation that the non-negativity condition above can be removed.)

Input:

- Prior density $\pi(\theta)$ and importance density $g(\theta)$.
- Number of iterations to perform N.
- Likelihood estimator $\hat{L}$.

Repeat the following steps N times.

1 Simulate θ^* from $g(\theta)$.

2 Simulate ℓ^* from $\hat{L}|\theta$.

3 Set $w^* = \ell^*\pi(\theta^*)/g(\theta^*)$.

Output:
A set of N pairs of (θ^*, w^*) values.

Algorithm 1: Random weight importance sampling (RW-IS)

2.3 Approximate Bayesian computation

Many interesting models are sufficiently complicated that it is not feasible to calculate exact likelihoods or useful (i.e. reasonably low variance) unbiased estimators. ABC algorithms instead base inference on simulation from the model. Algorithm 2 (ABC-IS) is a standard importance sampling implementation of this idea.

For later theoretical work, it is helpful to interpret ABC-IS in the framework of RW-IS. In particular, Algorithm 2 can be obtained from Algorithm 1 by replacing the likelihood

Input:

- Prior density $\pi(\theta)$ and importance density $g(\theta)$.
- Number of iterations to perform N.
- Observed data y_{obs}.
- Summary statistics $s(\cdot)$, distance function $d(\cdot, \cdot)$ and threshold $\epsilon \geq 0$.

Algorithm:
Repeat the following steps N times.

1 Simulate θ^* from $g(\theta)$.

2 Simulate y^* from $Y|\theta^*$

3 Set $\ell^* = \mathbb{1}[d(s(y^*), s(y_{\text{obs}})) \leq \epsilon]$.

4 Set $w^* = \ell^* \pi(\theta^*)/g(\theta^*)$.

Output:
A set of N pairs of (θ^*, w^*) values.

Algorithm 2: ABC importance sampling (ABC-IS)

estimator $\hat{L}$ with a Bernoulli estimator

$$\hat{L}_{\text{ABC}} = \mathbb{1}[d(s(Y), s(y_{\text{obs}})) \leq \epsilon]. \tag{3}$$

This equals 1 if the distance between summary statistics of the simulated and observed datasets is less than or equal to a threshold ϵ. It is typically a biased estimate of the likelihood and ABC-IS targets an approximate likelihood $L_{\text{ABC}}(\theta) = E[\hat{L}_{\text{ABC}}|\theta]$.

A special case of ABC-IS is when $g(\theta) = \pi(\theta)$. The weights in this case equal zero or one, and it is often referred to as ABC rejection sampling. A generalisation of ABC-IS, considered in Section 7, is to use as a likelihood estimator $K(d(s(Y), s(y_{\text{obs}}))/\epsilon)$, where K is a density function known as the ABC kernel. Algorithm 2 uses a uniform kernel.

As $\epsilon \to 0$, the target distribution of ABC-IS converges to $\pi(\theta|s(y_{\text{obs}}))$. However, $\epsilon > 0$ is typically required to achieve a reasonable number of non-zero weights, so a trade-off must be made. The observed summary statistics $s(y_{\text{obs}})$ should ideally preserve most of the information on θ available from y_{obs}. However analysis of ABC algorithms shows that the quality of the approximation deteriorates with the dimension of $s(y)$. Therefore the choice of $s(\cdot)$ involves a trade-off between low dimension and informativeness. For further background details on all aspects of ABC see the review articles of Beaumont (2010) and Marin et al. (2012).

3 Lazy ABC

Lazy ABC is Algorithm 3. Given proposed parameters θ^*, step 2 performs an initial part of data simulation, with results x^*. A continuation probability $\alpha(\theta^*, x^*)$ is calculated. With this probability the simulation is completed, resulting in y^*, and a weight is calculated by steps 5 and 6. Otherwise the iteration is given weight zero. The desired behaviour is that simulating x^* is computationally cheap but can be used to quickly reject many unpromising importance sampling iterations.

Input:

- Prior density $\pi(\theta)$ and importance density $g(\theta)$.
- Number of iterations to perform N.
- Observed data y_{obs}.
- Summary statistics $s(\cdot)$, distance function $d(\cdot, \cdot)$ and threshold $\epsilon \geq 0$.
- Continuation probability function $\alpha(\theta, x)$ taking values in $[0, 1]$.
- Random variable X representing an initial stage of simulation.

Algorithm:
Repeat the following steps N times.

1 Simulate θ^* from $g(\theta)$.

2 Simulate x^* from $X|\theta^*$ and let $a^* = \alpha(\theta^*, x^*)$.

3 With probability a^* continue to step 4. Otherwise perform *early stopping*: let $\ell^* = 0$ and go to step 6.

4 Simulate y^* from $Y|\theta^*, x^*$.

5 Set $\ell^*_{\text{ABC}} = \mathbb{1}[d(s(y^*), s(y_{\text{obs}})) \leq \epsilon]$ and $\ell^* = \ell^*_{\text{ABC}}/a^*$.

6 Set $w^* = \ell^* \pi(\theta^*)/g(\theta^*)$.

Output:
A set of N pairs of (θ^*, w^*) values.

Algorithm 3: Lazy ABC

The following conditions are required for Algorithm 3 to be well defined:

C1 $\alpha(\theta, x) > 0$ whenever $\Pr(\hat{L}_{\text{ABC}} > 0|\theta, x) > 0$

(recall $\hat{L}_{\text{ABC}}$ is defined by (3))

C2 The random variable X is such that both $X|\theta$ and $Y|\theta, x$ can be simulated from.

Lazy ABC and ABC-IS both perform importance sampling inference for the same approximate likelihood $L_{\text{ABC}}(\theta)$. To see this first observe that Algorithm 3 can be interpreted as a RW-IS algorithm using a likelihood estimator of the form

$$\hat{L}_{\text{lazy}} = \begin{cases} \hat{L}_{\text{ABC}}/\alpha(\theta, X) & \text{with probability } \alpha(\theta, X) \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

By the arguments of Section 2, to show that lazy ABC and ABC-IS target the same likelihood it is sufficient to prove the following result.

Theorem 1. *Conditional on θ, $\hat{L}_{lazy}$ is a non-negative unbiased estimator of $L_{ABC}(\theta)$.*

Proof. Non-negativity is immediate. For unbiasedness first observe that $\text{E}(\hat{L}_{\text{lazy}}|\theta, x)$ equals zero when $\alpha(\theta, x) = 0$ and $E(\hat{L}_{\text{ABC}}|\theta, x)$ otherwise. By C1 if $\alpha(\theta, x) = 0$ then $\Pr(\hat{L}_{\text{ABC}} > 0|\theta, x) = 0$ and so $\text{E}(\hat{L}_{\text{ABC}}|\theta, x) = 0$. Hence $\text{E}(\hat{L}_{\text{lazy}}|\theta, X) = \text{E}(\hat{L}_{\text{ABC}}|\theta, X)$. Taking expectations over X gives the required result. □

3.1 Examples

The following are examples of situations in which lazy ABC can be useful. The first three form the main focus of the paper. Each assumes that Y is a deterministic function of a latent vector $X_{1:p}$ such that it is possible to simulate from $X_1|\theta$ and $X_i|\theta, x_{1:i-1}$ for all $1 < i \leq p$. Further examples are given in Appendix B which use the same framework to consider multiple stopping decisions.

Example 1: Partial simulation Let $X = X_{1:t}$ for some $t < p$.

Example 2: Partial calculation of s Assume that computing $s(Y)$ involves calculating variables $X'_{1:q}$ which are deterministic transformations of Y, and that this is the most expensive part of simulating $\hat{L}_{\text{ABC}}$. Let $X = (X_{1:p}, X'_{1:t})$ for some $t < q$. (This is applied in Section 6.2.)

Example 3: Random stopping times As for either previous example but with t replaced by a random stopping time variable T. This allows a stopping decision once a particular event has occurred.

Example 4: Deterministic stopping Suppose $s(Y) = (s_1(X_{1:t}), s_2(X_{t+1:p}))$. Let $X = X_{1:t}$ and $\alpha(\theta, x) = \mathbb{1}[\Pr(\hat{L}_{\text{ABC}} > 0|\theta, x) > 0]$. That is, early stopping occurs if and only if s_1 is too extreme for the ABC acceptance criterion to be met.

3.2 Terminology and notation

Simulation from $X|\theta$ is referred to as the *initial simulation stage* and from $s(Y)|\theta, x$ as the *continuation simulation stage*. It is often useful later to have $\alpha(\theta, x) = \alpha(\phi(\theta, x))$ where $\phi(\theta, x)$ is a vector of summaries referred to as the *decision statistics*.

Notation is now introduced for expected CPU times: $\bar{T}_1(\theta)$ is for steps 2 and 3 in Algorithm 3 conditional on θ, $\bar{T}_2(\theta, \phi)$ is for steps 4–6 conditional on (θ, ϕ) and $\bar{T}(\theta)$ is steps 2–4 of Algorithm 2 conditional on θ. The first two are roughly the times of the initial simulation and continuation stages, but also cover the other steps of Algorithm 3 given θ. The following definitions are also used later: $\bar{T}_1 = \mathrm{E}[\bar{T}_1(\theta)]$ and $\bar{T}_2(\phi) = \mathrm{E}[\bar{T}_2(\theta, \phi)|\phi]$. Note that $\bar{T}_1$ is a constant. The other terms above are deterministic functions of one or both of θ and ϕ and may therefore be random variables when their arguments are.

4 Tuning

There is considerable freedom to tune lazy ABC through the choice of X (when to consider stopping) and α (the function assigning continuation probabilities). Section 4.1 provides theory on the most efficient choice of α. This is used in Section 4.2 to motivate practical tuning methods.

4.1 Theory

A commonly used tool for the analysis of importance sampling algorithms is the effective sample size (ESS). Liu (1996) argued that typically the variance of the importance sampling estimator is roughly equal to that of N_{eff} independent samples where

$$N_{\text{eff}} = NE(W)^2/E(W^2),$$

N is the number of importance sampling iterations and the random variable W is the weight generated in an iteration of importance sampling (with zero representing rejection). The argument of Liu generalises immediately to RW-IS algorithms through the interpretation of them as importance sampling algorithms on an augmented parameter space given in Section 2.2.

Define efficiency as N_{eff}/T where T is the CPU time of the algorithm (i.e. ignoring any execution time savings due to parallelisation.) Various results on asymptotic efficiency for large N now follow. These are proved in Appendix C as corollaries of Theorem 2 which is stated there.

The choice of $\alpha(\theta, x)$ which maximises asymptotic efficiency is as follows for some $\lambda \geq 0$,

$$\alpha(\theta, x) = \min\left\{1, \lambda\frac{\pi(\theta)}{g(\theta)}\left[\frac{\gamma(\theta, x)}{\bar{T}_2(\theta, x)}\right]^{1/2}\right\}, \tag{5}$$

where $\gamma(\theta, x) = \Pr\left(d(s(Y), s(y_{\text{obs}})) \leq \epsilon | \theta, x\right)$ and $\bar{T}_2(\theta, x)$ is as defined in Section 3.2. Roughly,

$\gamma(\theta, x)$ is the probability that continuing the simulation will meet the ABC acceptance criterion and $\bar{T}_2(\theta, x)$ is the expected time it will take. The interpretation of λ is not clear, nor does a simple closed form expression for its optimal value seem possible.

This optimal choice of α is of limited practical use, as typically X is high dimensional and thus estimation of $\gamma(\theta, x)$ and $\bar{T}_2(\theta, x)$ is not feasible. Estimation is more feasible if (θ, x) is replaced by some low dimensional vector of summaries, $\phi(\theta, x)$, which will be referred to as the *decision statistics*. Consider decision statistics which satisfy the condition

C3 There is a function $u(\phi)$ such that $u(\phi(\theta, x)) = \frac{\pi(\theta)}{g(\theta)}$.

Then the optimal α amongst those of the form $\alpha(\theta, x) = \alpha(\phi(\theta, x))$ is as follows for some $\lambda \geq 0$,

$$\alpha(\phi) = \min\left\{1, \lambda u(\phi)\left[\frac{\gamma(\phi)}{\bar{T}_2(\phi)}\right]^{1/2}\right\}, \tag{6}$$

where $\gamma(\phi) = \Pr\left(d(s(Y), s(y_{\text{obs}})) \leq \epsilon | \phi\right)$ and $\bar{T}_2(\phi)$ is as defined in Section 3.2. See Figures 1D and 2D for example $\alpha(\phi)$ functions based on estimating (6).

Some cases in which condition C3 holds are: $\pi(\theta) = g(\theta)$ (ABC rejection sampling); $\phi(\theta, x) = (\theta, \ldots)$ (the decision statistics include θ); $\phi(\theta, x) = (\frac{\pi(\theta)}{g(\theta)}, \ldots)$. The optimal α when this condition does not hold is described by Theorem 2 in Appendix C.

4.1.1 Tuning g

It is not clear what the optimal choice of $g(\theta)$ is for lazy ABC. The examples later use typical choices from the ABC literature, but a better choice may improve performance further. This paper's theory gives the following corollary on the optimal choice of $g(\theta)$ for ABC-IS and RW-IS, which is of some general interest.

Corollary 1. *The asymptotic efficiency of ABC-IS is maximised by* $g(\theta) \propto \pi(\theta)\left[\frac{\gamma(\theta)}{\bar{T}(\theta)}\right]^{1/2}$, *where* $\gamma(\theta) = \mathrm{E}(\hat{L}_{ABC}|\theta)$. *This also holds for RW-IS with* $\gamma(\theta) = \mathrm{E}(\hat{L}^2|\theta)$.

Proof. See Appendix D. □

4.2 Methods

The theory above motivates choosing α by estimating (6). This section details a method to implement this approach. Its effectiveness is discussed in Section 7.

The method is outlined here and more detail is given in Sections 4.2.1 to 4.2.5. Tuning begins with a pilot run of N' iterations of ABC-IS, recording intermediate simulation states and timings of interest. This is used to estimate $\gamma(\phi)$ and $\bar{T}_2(\phi)$ for various choices of X and ϕ, considering only ϕ such that condition C3 holds. Under each of these choices, λ is found by numerically maximising an estimate of efficiency. The optimal choice of X and ϕ is then made. Following tuning, N iterations of lazy ABC are performed (unless the estimated efficiency gains are judged inadequate, in which case ABC-IS can be used).

A simpler variation on this method is possible if the set of possible $\phi(\theta, x)$ values is finite and small. A pilot run is performed as before. Then $\alpha(\phi)$ values are chosen by direct numerical optimisation of an estimate of the algorithm's efficiency.

4.2.1 Estimation of $\bar{T}_2(\phi)$

It may often suffice to treat $\bar{T}_2(\phi)$ as constant and estimate it as the mean CPU time of the continuation stage in the pilot run. This is the case if knowledge of the simulation process shows the number of computational operations required is unaffected by ϕ, or if the pilot run shows $\bar{T}_2(\phi)$ varies little relative to $\gamma(\phi)$. Alternatively, statistical methods such as regression can be used for estimation, which is straightforward when ϕ is low dimensional.

4.2.2 Estimation of $\gamma(\phi)$

Estimation of $\gamma(\phi)$ is more difficult. Two approaches are suggested: the "standard" approach, producing $\hat{\gamma}^{(1)}$, attempts accurate estimation but typically involves strong assumptions; the "conservative" approach, producing $\hat{\gamma}^{(2)}$, sacrifices accuracy to improve robustness. They are based on two equivalent expressions for $\gamma(\phi)$: $\Pr(d(s(Y), s(y_{\text{obs}})) \leq \epsilon | \phi)$ and $\mathrm{E}[\hat{L}_{\text{ABC}} | \phi]$. Examples of successful implementations of both approaches are given in Sections

5 and 6.

The standard approach is to model the relationship between ϕ and $d(s(Y), s(y_{\text{obs}}))$ and use this to estimate $\Pr(d(s(Y), s(y_{\text{obs}})) \leq \epsilon|\phi)$. However a difficulty is that for most ϕ values this involves extrapolating into the tails of the distribution of $d(s(Y), s(y_{\text{obs}}))|\phi$. See Figure 2A for example. This creates a danger of underestimating the optimal α values and potentially producing very large importance sampling weights. (This can be avoided for simple summary statistics by first modelling $s(Y)|\phi$; see Section 5 for example.)

The conservative approach is to select ϵ_1 following the pilot run such that a sufficiently large number of its simulations $y_{1:N'}$ satisfy $d(s(y_i), s(y_{\text{obs}})) \leq \epsilon_1$. Let z_i be indicator variables denoting meeting this condition and model the relationship between z_i and the simulated ϕ_i values. One method, used in the application later, is non-parametric logistic regression following Wood (2011). This approach is effectively tuning the method based on an ϵ value larger than that of interest. This is an inefficient way to sample from the target of interest. However, if tuning can be done well for the larger ϵ value then this method is safe from producing any dangerously large importance weights, as discussed further in Section 4.2.5. Nonetheless, for ϕ regions where there are no $z_i = 1$ values the conservative estimate is still based on extrapolation and unlikely to be accurate. Consequences of this are discussed in Section 7.

4.2.3 Estimating efficiency

The tuning method outlined above requires the use of N' pilot run iterations to estimate the efficiency of lazy ABC under various choices of tuning details (in particular X, ϕ and α). It is sufficient to estimate $[\mathrm{E}(W^2)\,\mathrm{E}(T)]^{-1}$, as this equals efficiency up to a constant of proportionality. This can be used to estimate efficiency relative to ABC-IS, which is a particularly interpretable form of the results as it shows the efficiency improvement of using lazy ABC.

Assume that for a particular choice of tuning details the following are available for $1 \leq$

$i \leq N'$: $t_i^{(1)}$ - initial simulation stage time; $t_i^{(2)}$ - continuation simulation stage time; α_i - continuation probability; $\hat{\gamma}_i$ - estimate of $\mathrm{E}(\hat{L}_{\mathrm{ABC}}|\phi_i)$; u_i - ratio $\pi(\theta)/g(\theta)$. An estimate up to proportionality of efficiency is then $[\widehat{W^2}\hat{T}]^{-1}$ where $\widehat{W^2} = N'^{-1}\sum_{i=1}^{N'} u_i^2\hat{\gamma}_i/\alpha_i$ and $\hat{T} = \sum_{i=1}^{N'} t_i^{(1)} + \sum_{i=1}^{N'} \alpha_i t_i^{(2)}$. An estimate of efficiency of ABC-IS is formed by taking $\alpha \equiv 1$. Note that this often overestimates $\mathrm{E}(T)$ as interrupting the simulation to record intermediate states and timings reduces algorithm efficiency. A more precise estimate would be possible using further pilot simulations without this interruption.

4.2.4 Combining pilot and main run output

To make efficient use of the pilot run, it can be used in the final output as well as for tuning. This is done by appending the pilot sequence of (θ, w) pairs to that from the main algorithm. Loosely speaking, since each individual sequence targets the same distribution, so does the combined sequence. More technically, it is straightforward to see that ABC versions of relations (1) and (2) are roughly true for the combined sequence when N and N' are large, and are exactly true as $N \to \infty$ regardless of N'. Also note that on appending the sequences, gains in efficiency are possible by multiplying the weights of one sequence by a constant, but this is not implemented here as little improvement was observed in the application later.

4.2.5 Choice of ϵ

In ABC-IS, an appropriate value ϵ is often unknown a priori and is instead chosen based on the simulated $d(s(Y), s(y_{\mathrm{obs}}))$ values. For lazy ABC in this situation one can use the pilot run to select a preliminary conservative choice of ϵ_1 as in Section 4.2.2 and perform lazy ABC with $\epsilon = \epsilon_1$. Alternative values of ϵ can then be investigated by updating the realised $\hat{L}_{\mathrm{ABC}}$ values in the weight calculations. For $\epsilon < \epsilon_1$ this simply reduces the number of non-zero weights. However $\epsilon \gg \epsilon_1$ is not recommended as this may introduce large weights and destabilise the importance sampling approximation.

5 Example: infectious diseases

This section illustrates a basic implementation of lazy ABC and the tuning approach of Section 4.2 on a simplified version of the standard Markovian SIR infectious disease model. While ABC is not necessary for inference under this model (see Andersson and Britton, 2000), it is used for more complex variants (McKinley et al., 2009; Brooks-Pollock et al., 2014).

Suppose the random variables $S(t), I(t), R(t)$ follow a discrete time Markov chain. These represent the population size which is susceptible, infectious or recovered after t transitions. Two transitions are possible. Let $M = S(0)+I(0)+R(0)$ be the total population size. With probability $k\frac{R_0}{M}S(t)I(t)$, an infection occurs giving $S(t+1) = S(t) - 1$, $I(t+1) = I(t) + 1$ and $R(t+1) = R(t)$. With probability $kI(t)$, a recovery occurs giving $S(t+1) = S(t)$, $I(t+1) = I(t)-1$ and $R(t+1) = R(t)+1$. The constant k is chosen so that the probabilities sum to 1. When $I(t) = 0$ the chain terminates and a simple random sample of size 100 is taken from the population. The observed data y_{obs} is the number of these which are recovered. The parameter of interest is the *basic reproduction number* R_0, which is given a prior of Gamma$(3, 1)$. It is assumed that $M = 10^5$, $I(0) = 10^3$ and $R(0) = 0$, giving $S(0) = 9.9 \times 10^4$.

Standard ABC can be implemented here simply by taking $s(y) = y$ and $d(s_1, s_2) = |s_1 - s_2|$. Standard ABC rejection sampling was performed on a simulated dataset with $y_{obs} = 73$ using $N = 10^4$ iterations and $\epsilon = 1$. These settings are also used for the three lazy ABC analyses that follow.

First a simple version of lazy ABC with ad-hoc tuning choices is presented. This considers stopping at $t = 1000$, so that $X = (S(1000), I(1000), R(1000))$. The decision statistic $\phi(\theta, X)$ is $I(1000)$. It seems reasonable to stop if there is evidence that the number of infectious is not growing. So let $\alpha(\phi) = 0.1$ if $\phi \leq 1000$, and $\alpha(\phi) = 1$ otherwise.

The second lazy ABC analysis uses the standard tuning method of Section 4.2, with the same choice of X and ϕ as above. A pilot run of 1000 standard ABC simulations was

performed. Figures 1A and B plot pilot run realisations of ϕ against y, the observation, and the continuation stage simulation time. From this $\bar{T}_2(\phi)$, expected remaining simulation time given ϕ, was estimated by fitting a non-parametric regression, using a log link to enforce positivity. Also $\hat{p}(\phi)$, the expectation given ϕ of the proportion of the population who are recovered when the Markov chain terminates, was estimated by fitting a non-parametric binomial GLM. This GLM model fits the pilot data well as illustrated by Figure 1A, which shows the resulting expected number recovered in the final subsample and a 95% confidence interval. All non-parametric regressions were fitted using the approach of Wood (2011). An estimate of the probability of ABC acceptance, $\hat{\gamma}(\phi)$ can be directly calculated from $\hat{p}(\phi)$ and is shown in Figure 1C. It is now possible to estimate $\alpha(\phi)$ for a given value of λ from (6). For each λ the efficiency estimate of Section 4.2.3 can be calculated. Numerical optimisation shows this is maximised for $\lambda = 3.45$. Figure 1D shows the estimated optimal α function.

Finally lazy ABC using the conservative tuning approach is considered. This proceeds as for the standard tuning approach just described, except for the estimation of $\gamma(\phi)$. A variable z is introduced to indicate whether the pilot run data has ABC distance less than or equal to $\epsilon_1 = 3$. This variable equals 1 for 50 cases and 0 for the remainder. Then $\gamma(\phi)$ is estimated by fitting a non-parametric logistic regression of z on ϕ, with the results are shown in Figure 1C. This estimate is more conservative than that above in that it always assigns a higher probability of ABC acceptance. Numerical optimisation chooses $\lambda = 2.54$ and the resulting $\alpha(\phi)$ function is shown in Figure 1D.

Table 1 shows the results of the analyses described above. Note that each analysis used the same sequence of random seeds so that their (θ, X, Y) simulations were the same. Each analysis accepted the same 194 parameter values, except for lazy ABC with conservative tuning which accepted a subset of 177 of these. Under lazy ABC with ad-hoc tuning all weights were 1. With standard tuning 189 weights equalled 1 with the others below 3. With conservative tuning 131 weights equalled 1, with the others below 4. Estimates of posterior quantities from all methods were very similar.

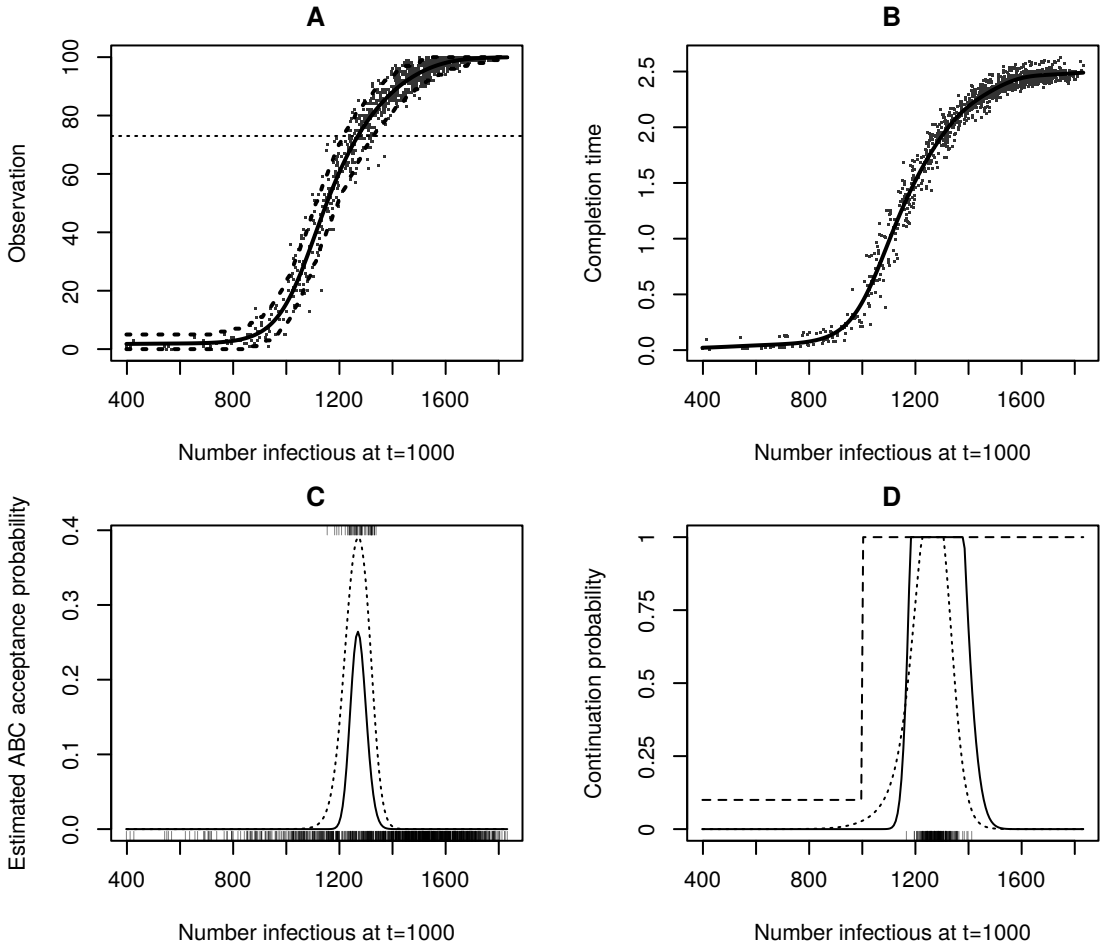

Figure 1: Tuning details of a simulation study applying lazy ABC to an SIR epidemic model. *Panel A* Pilot run realisation of $I(1000)$ (the decision statistic: number of infectious at $t = 1000$) and y (number recovered in subsample). The solid line is the prediction under a non-parametric logistic regression and the dashed lines are corresponding 0.025 and 0.975 quantiles. The dotted line shows the observed data. *Panel B* Pilot run realisations of $I(1000)$ and T_2 (continuation stage simulation time, in seconds). The solid line is a fitted non-parametric regression. *Panel C* Estimated $\gamma(\phi)$ functions – probability of eventual ABC acceptance – under standard (solid) and conservative (dotted) tuning. Standard tuning derives this curve from panel A. Conservative tuning fits a logistic regression to the indicator variable represented by vertical marks. This indicates which pilot runs result in an ABC distance of 3 or less. *Panel D* α functions (continuation probabilities) under ad-hoc (dashed), standard (solid) and conservative (dotted) tuning. The latter two cases are derived from the pilot run as described in Section 4.2. The marks on the horizontal axis indicate the ABC simulations which were accepted.

ABC method	Tuning	Time (s)	ESS	Relative efficiency	Estimated posterior	
					Mean	Standard deviation
Standard	-	18124	194	1	1.803	0.1267
Lazy	Ad-hoc	17848	194	1.02	1.803	0.1267
Lazy	Standard	5113	192	3.51	1.804	0.1276
Lazy	Conservative	3318	167	4.70	1.796	0.1212

Table 1: Simulation study on a SIR epidemic model. Four ABC analyses were performed on the same observations, sharing the same random seeds for their simulations. All used the same choice of $N = 10^4$ and $\epsilon = 1$. Iterations were run in parallel and computation times are summed over all cores used. Efficiency (ESS/CPU time) relative to standard ABC is shown. The pilot run required for tuning the last two rows tuning took a further 2016 seconds, with the remaining tuning calculations taking less than 2 seconds. For simplicity results are for lazy ABC output only, pilot run simulations have not been appended as in Section 4.2.4.

This example shows that it is straightforward to implement lazy ABC using ad-hoc tuning. The time savings here are small as the α function is quite conservative, only allowing early stopping when the epidemic is already dying out at time $t = 1000$. A more effective α could be sought using results on the typical behaviour of this model (see Andersson and Britton, 2000). The above instead considers the tuning method of Section 4.2 and shows it is simple to implement here and produces significant time savings. Conservative tuning does better, but this may reflect variability of results for a relatively small number of iterations.

6 Example: spatial extremes

This section uses lazy ABC in a computationally demanding application of ABC to spatial extremes introduced by Erhardt and Smith (2012). As well as illustrating the method for a complicated application, simulation studies are performed to investigate its efficiency.

6.1 Background

The observation $y_{t,d}$ represents the maximum measurement (e.g. of rainfall) during year t at location $x_d \in \mathbb{R}^2$. There are D locations and M years. The data are treated as M independent replications of a spatial distribution. Several models based on extreme value

theory have been proposed, and Erhardt and Smith concentrate on the *Schlather process* (Schlather, 2002).

The details of the Schlather process follow, together with the method Erhardt and Smith used to calculate ABC summary statistics. The full mathematical details are not essential to understanding the implementation of lazy ABC. Instead, what is most important is the order of operations required to simulate a dataset and summary statistics. This is summarised as Algorithm 4 for later reference.

The Schlather process is based on independent identically distributed mean zero stationary Gaussian processes $U_i(x)$ where $i = 1, 2, \ldots$ (indexing an infinite number of Gaussian processes) and $x \in \mathbb{R}^2$ (representing location).

The correlation between locations x and x' is given by the correlation function $\rho(h)$ where $h = ||x - x'||_2$. Let s_i be draws from a Poisson process on $s \in (0, \infty)$ with intensity $\mu^{-1}s^{-2}$, where $\mu = \mathrm{E}[\max(0, U(x))]$. Then the Schlather process is

$$Y(x) = \max_i s_i \max(0, U_i(x)). \tag{7}$$

Erhardt and Smith focus on the Whittle-Matérn correlation function with zero nugget

$$\rho(h; c, \nu) = \frac{2^{1-v}}{\Gamma(\nu)} \left(\frac{h}{c}\right)^{\nu} K_\nu \left(\frac{h}{c}\right), \tag{8}$$

where Γ is the gamma function and K_ν is the modified Bessel function of the third kind with order ν. This has two parameters: range $c > 0$ and smoothness $\nu > 0$.

A density function for the Schlather process is not available for $D > 2$, making inference difficult. Schlather (2002) provides a near-exact algorithm to simulate from the process based on only a finite number of copies of U_i, motivating the use of ABC by Erhardt and Smith. They applied ABC rejection and importance sampling with a uniform prior on $[0, 10]^2$ and investigated several choices of summary statistics. The analysis here focuses on the choice they find most successful, based on *tripletwise extremal coefficient estimators*. Given a triple

of 3 locations, i, j, k, this estimator is

$$\hat{\theta}_{ijk} = \frac{M}{\sum_{t=1}^{M} 1/\max(y_{t,i}, y_{t,j}, y_{t,k})}. \tag{9}$$

There are $O(D^3)$ such summaries, so Erhardt and Smith calculate a vector m of mean values within 100 clusters of triples, and use these as summary statistics. Their clustering process finds triples of similar shapes, ignoring differences of location and rotation. The ABC distance function between two vectors m_1 and m_2 of cluster means is

$$d(m_1, m_2) = \sum_{i=1}^{100} |m_{1i} - m_{2i}|. \tag{10}$$

Although applying dimension reduction techniques to such high dimensional summaries has been shown to often improve ABC results (Fearnhead and Prangle, 2012), this is not investigated here as the aim is to investigate the efficiency improvements of lazy ABC.

Implementing the analysis below used the R packages "SpatialExtremes" (Ribatet et al., 2013) to simulate from the Schlather process and "ABCExtremes" to implement some details of the approach of Erhardt and Smith.

6.2 Methods

Exploratory investigation of ABC code with $D = 20$ and $M = 100$ showed that the majority of time was spent simulating the data (7.1ms/iteration) and calculating extremal coefficient estimates (17.9ms/iteration), with the remaining steps being brief (3.1ms/iteration). The time costs of the first two of these scaled with D as roughly proportional to D and D^3 respectively, so the latter is expected to dominate for large D. Furthermore, interrupting and then resuming operations during the calculation of extremal coefficients is much simpler to implement than during simulation of data. Therefore the initial simulation stage of the lazy ABC analysis was chosen to be simulating the data at all locations, and extremal coefficient estimates at a subset of locations L. The continuation simulation stage was to

Input:

- Parameters: range $c > 0$ and smoothness $\nu > 0$.
- Locations $x_1, \ldots, x_D$.
- Number of years M.

Data simulation:

1 Calculate $D \times D$ covariance matrix $\Sigma(c, \nu)$ using (8).

2 Repeat the following steps for $t = 1, \ldots, M$.

  a Sample values $s_1, s_2, \ldots, s_q$ from a Poisson process (see text for intensity).

  b For each $1 \leq i \leq q$, sample $u_i(x_1), u_i(x_2), \ldots, u_i(x_D)$ from $N(0, \Sigma(c, \nu))$.

  c Calculate Schlather process realisations for year t by (7).

Summary statistic calculation:

1 Calculate extremal coefficient estimate $\hat{\theta}_{ijk}$ for all triples of locations ijk using (9).

2 Calculate mean $\hat{\theta}_{ijk}$ values for each cluster of triples.

Output:

- Summary statistic vector comprising the cluster means.

Algorithm 4: Overview of simulation of Schlather process data and summary statistics. Full details of the steps are given in the text.

calculate the remaining extremal coefficient estimates.

The decision statistic $\hat{d}$ was constructed as follows. Let m_{1i} be the ith cluster mean for the observed data. Let $\hat{m}_{2i}$ be the ith cluster mean for the simulated data using only extremal coefficient estimates available at the initial simulation stage, and B be the set of clusters for which any such estimates are available. Then define $\hat{d} = \sum_{i \in B} |m_{1i} - \hat{m}_{2i}|$. This is an estimate of the ABC distance d (10). It could be improved by estimating typical $\hat{m}_{2i}$ values for $i \notin B$ but including such constant terms has no effect on the analysis below.

It was assumed that $\bar{T}_2(\hat{d})$ is constant as, given D, L and M, the continuation stage always involves the same number of calculations. The value was estimated by the mean CPU time for this stage in the pilot run. Analyses were performed using both the standard and conservative γ estimators. To calculate $\hat{\gamma}_1(\hat{d})$ from pilot run output, the relationship between $\hat{d}$ and d was modelled statistically. Exploratory analysis showed that there was a roughly linear relationship, but for some choices of L this was heteroskedastic (see Figure 2A). Furthermore, for small $\hat{d}$ the distribution of $d|\hat{d}$ was skewed. So $\Pr(d \leq \epsilon|\hat{d})$ was estimated based on a linear regression of d on $\hat{d}$ with a Box-Cox transformation, using only simulations with nearby values of $\hat{d}$. This was done for several $\hat{d}$ values and interpolated estimates elsewhere formed $\hat{\gamma}_1(\hat{d})$. For the importance sampling case, $\log u$ was also included in each regression giving a number of functions mapping u to estimates of $\Pr(d \leq \epsilon|\hat{d}, u)$ for various $\hat{d}$ values, which were used for interpolation. Calculation of $\hat{\gamma}_2$ was as described in Section 4.2.2, taking ϵ_1 to give 100 acceptances in the pilot run. Given estimates of $\bar{T}_2$ and γ, tuning was performed as described in Section 4.2, with optimisation over possible choices of L by backwards selection.

Three simulation studies were performed. The first replicated the rejection sampling analysis of Erhardt and Smith on several simulated datasets. These used $D = 20, M = 100$ and true parameter values shown in Table 2. Each dataset used a different set of observation locations with integer coordinates sampled from $[0, 10]^2$. The first analysis was a replication of the standard ABC analysis, using ϵ values corresponding to 200 acceptances. Then lazy

ABC was performed on the same datasets under each method of estimating γ. To compare the methods fairly, lazy ABC used the same ϵ value as standard ABC and reused its random seeds so that the sequence of (θ, X, Y) realisations is also the same.

The second simulation study investigated rejection sampling for a single larger simulated dataset with $D = 35$, $c = 0.5$ and $\nu = 1$. Locations were chosen as before. As in a real application ϵ was not assumed to be known in advance and the approach of Section 4.2.5 was used to select this post-hoc. A complication for this dataset was that the simulation of Gaussian processes was difficult when both parameters were large: the default "direct method", based on Choleski decomposition, sometimes produced numerical errors. Simulation was possible via the turning bands method (TBM) but much slower (roughly 150 times the CPU time). A two stage simulation method was implemented. First the direct method was attempted and if this failed TBM was used. To save time lazy ABC was implemented with multiple stopping decisions, the first taking place after attempting the direct method. This has a binary decision statistic indicating success or failure. The second stopping decision is as described earlier. Tuning was performed as described in Appendix B.1.2, using $\hat{\gamma}_1$ fitted as described above by either the standard or conservative tuning method. The standard method used ϵ_1 to give 30 acceptances in the pilot run. As before all analyses reused the same random seeds.

Finally an importance sampling analysis was performed on the larger dataset. A sample of 10^4 log parameter values was taken from simulations of the preceding standard ABC analysis with distances below the 0.3 quantile. A Gaussian mixture distribution was constructed with locations given by this sample and variances equal to twice the empirical variance of the sample. After truncation to the prior support, this was used to give $g(\theta)$, where θ now represents the log parameters. This choice follows the suggestions of Beaumont et al. (2009), noting that using the log scale produced a better fit to the sample and that the subsample was used to avoid slow density calculations. The preceding $D = 35$ analysis was then repeated. As discussed in Section 4.2, $u = \pi(\theta)/g(\theta)$ was included as a decision statistic. Estimation of γ and $\bar{T}_2$ was performed as before with u included in the γ estimate as described earlier.

All ABC analyses performed 10^6 total iterations. For lazy ABC 10^4 of these comprised the pilot run.

6.3 Results

Figure 2 illustrates some details of tuning for one case of the $D = 20$ study. The results are shown in Table 2. For all datasets lazy ABC is roughly 4 times more efficient under the standard tuning method and 3 times under the conservative method. Efficiency gains for conservative tuning are slightly less than estimated. This is because the estimate is made for a choice of ϵ_1 larger than the final ϵ. The mean weights were also investigated, as these are useful in model selection as an estimate of $\pi(y_{\text{obs}})$. All lazy ABC estimates differed from the standard ABC estimate by no more than 4%.

		Standard	Lazy			Relative efficiency	
Range	Smooth	Time (10^3s)	Time (10^3s)	Sample size	ESS	Estimated	Actual
0.5	1	32.0	8.0 (11.6)	196 (199)	196.0 (198.7)	4.08 (3.28)	4.00 (2.79)
1	1	31.3	7.3 (9.8)	200 (200)	199.9 (200.0)	4.34 (4.31)	4.35 (3.25)
1	3	31.3	8.2 (11.2)	194 (198)	182.5 (196.5)	3.77 (3.43)	3.51 (2.79)
3	1	31.2	7.7 (11.1)	194 (200)	189.9 (200.0)	4.18 (3.56)	3.89 (2.86)
3	3	31.2	7.4 (11.0)	192 (199)	175.8 (199.0)	4.43 (3.65)	3.79 (2.87)
5	3	31.3	8.3 (11.1)	200 (200)	200.0 (200.0)	3.73 (3.49)	3.85 (2.87)

Table 2: Simulation study on spatial extremes replicating Erhardt and Smith (2012). Each row represents the analysis of a simulated dataset under the given values of range c and smoothness ν parameters (used in (8)). In each analysis a choice of ϵ was made under standard ABC so that the accepted sample size (and therefore ESS) was 200, and the same value was used for lazy ABC. Lazy ABC figures are shown for both the standard $\hat{\gamma}$ estimate and, in brackets, the conservative estimate. The lazy ABC output includes the pilot run as described in Section 4.2.4, and also includes the tuning time (roughly 120 seconds for the standard approach and 210 for the conservative). Iterations were run in parallel and computation times are summed over all cores used. For all datasets efficiency (ESS/time) to 1 significant figure was 0.006 for standard ABC and 0.02 or 0.03 under either approach to lazy ABC.

Table 3 shows results for the $D = 35$ dataset. In the initial rejection sampling analysis lazy ABC improved efficiency by roughly 8 times. For importance sampling the improvement factor is 2, showing that lazy ABC still improves efficiency, although this is harder when g concentrates on plausible choices of θ. For example, standard ABC now spends negligible

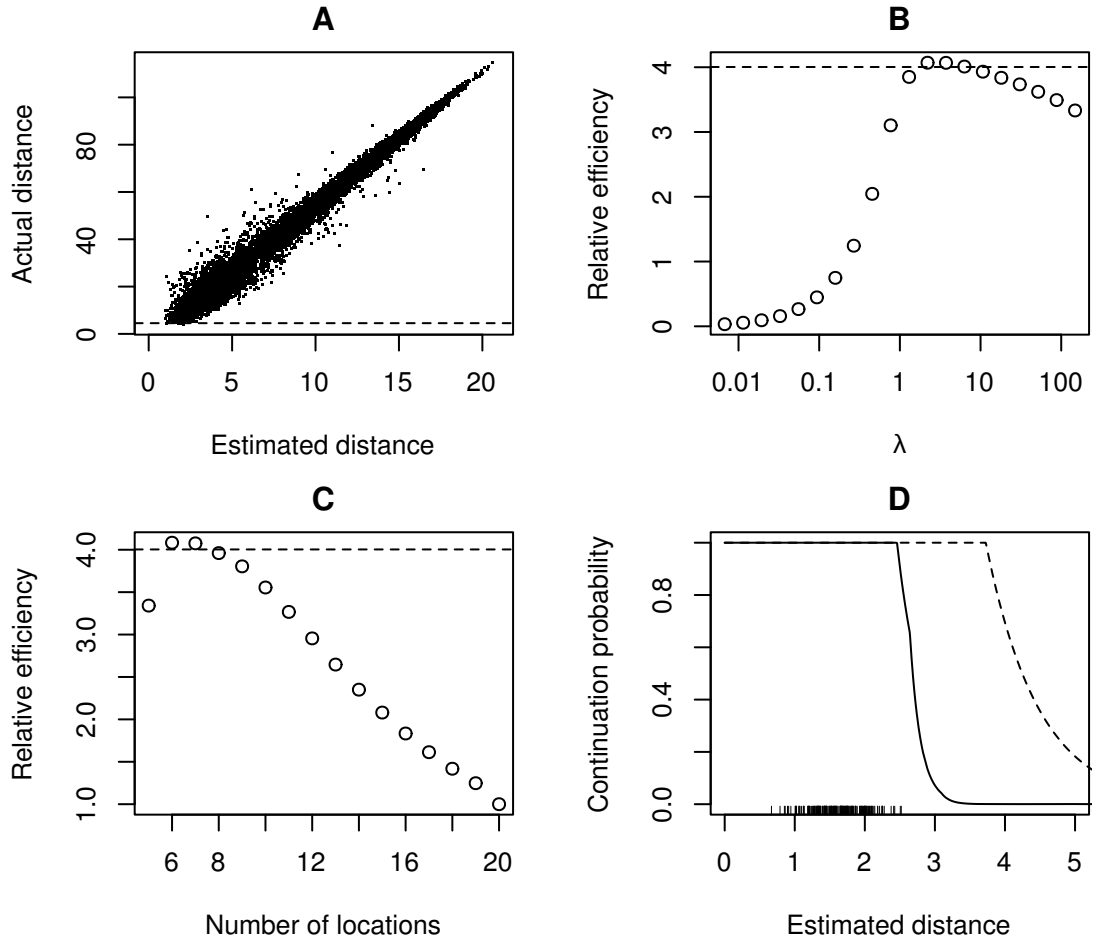

Figure 2: Details of tuning lazy ABC in a simulation study on spatial extremes corresponding to the first row of Table 2. Panels A-C concentrate on the standard tuning approach. *Panel A* Pilot run values of $\hat{d}$ (estimated distance) and d (actual distance). The dashed line shows the value of ϵ. *Panel B* Estimates of efficiency (ESS/time) for different values of λ (parameter of (6) which must be determined numerically). The dashed line shows the realised efficiency (ESS/time of the final lazy ABC algorithm). *Panel C* Estimated efficiency for the best choices of L (subset of locations from which early stopping is considered) of various lengths output by backwards selection. The dashed line shows the realised efficiency. *Panel D* Values of $\hat{d}$ and α (continuation probability) from non-pilot simulations under standard (solid line) and conservative (dashed line) tuning. The marks on the horizontal axis indicate the simulations which resulted in positive weights. (For this panel conservative tuning was performed using L as selected by standard tuning.)

time on TBM simulations. As before lazy ABC estimates of $\pi(y_{\text{obs}})$ differed from those of standard ABC by no more than 4%. In both cases a post-hoc selection of ϵ has been used successfully.

Table 3 shows that under rejection sampling the standard tuning ESS is considerably smaller than the sample size. This is due to two simulations which are given importance weights of 10. These have $\hat{\gamma}$ values of roughly 10^{-4} which appears to be an underestimate. Conservative tuning avoids large importance weights to give an ESS of roughly 200 and improves the relative efficiency for the same final choice of ϵ. For importance sampling conservative tuning also performs better. The reason here is not obvious but may be a better final selection of L.

		Standard			Lazy			Relative
	ϵ	Time (10^3s)	Sample size	ESS	Time (10^3s)	Sample size	ESS	efficiency
RS standard	2.61	241.4	210	210	22.8	200	139.6	7.0
RS conservative	2.61	241.4	207	207	25.5	200	197.7	9.0
IS standard	2.33	136.4	209	168	61.9	200	165	2.2
IS conservative	2.33	136.4	209	168	51.0	200	162	2.6

Table 3: Simulation study on a spatial extremes dataset with $D = 35$. Results are shown for rejection and importance sampling with standard and conservative tuning. The rejection sampling output was used to create the importance density. The final choice of ϵ is shown. For IS the two ϵ values are equal but there is a small difference for RS. The lazy ABC output includes the pilot run and the tuning time.

7 Discussion

This paper has introduced lazy ABC, a method to speed up inference by ABC importance sampling without introducing further approximations. The approach is to abandon some unpromising simulations before they are complete. By using a probabilistic stopping rule and weighting the accepted simulations accordingly, the algorithm targets exactly the same distribution as standard ABC, in the sense that Monte Carlo estimates of functions $h(\theta)$ and of the model evidence converge to unchanged values.

Results have been provided on the optimal tuning of the lazy ABC stopping rule and used to motivate a practical tuning method. This has been demonstrated for a simple epidemiological example and a computationally challenging application where it has produced improvements in efficiency (ESS/CPU time) over standard ABC of up to 8 times.

The tuning method is based on estimating the optimal choice of $\alpha(\phi)$, (6). The most difficult part was estimating $\gamma(\phi) = \Pr\left(d(s(Y), s(y_{\text{obs}})) \leq \epsilon | \phi\right)$ from pilot run data. Two approaches to this were described, a standard approach of direct estimation and a conservative approach of estimation using a larger ϵ value than is of interest for ABC. The latter approach improves robustness and make estimation simpler at the cost of some inefficiency. Both approaches performed well in the simulation studies but some improvements are desirable. Firstly, estimation of $\gamma(\phi)$ often involves extrapolation which may produce inaccurate results. Secondly, several choices by the user are required, especially for the standard approach. A more automated approach would be useful for lazy versions of ABC SMC algorithms, where a new choice of α would be needed for each ϵ value, or alternatively for lazy ABC algorithms which adapt α as more simulations become available. It would be of interest to find suboptimal but robust choices of α addressing these issues.

A related point is that lazy ABC can be generalised to allow a non-uniform ABC kernel. This may allow alternative approaches to tuning.

In some situations likelihood-based inference is possible, but calculating the likelihood or an unbiased estimate is expensive. Generalising lazy ABC ideas to this situation is possible. Appendix A describes this *lazy importance sampling* algorithm and shows that the theoretical results of the paper carry over to it. It also discusses why practical application seems more challenging than for lazy ABC. Nonetheless this is an interesting topic for future research.

Lazy ABC with multiple stopping decisions is another extension to the framework of the main paper and is described in Appendix B, with an example of implementation in Section 6. A tuning method is given when the decision statistics for all stopping decisions are discrete, and also some cases where one decision statistic is continuous. For more complex cases tuning

results are not available. For now it is recommended to discretise most decision statistics to avoid this difficulty.

This paper has concentrated on importance sampling, which is widely used by ABC practitioners, but the lazy ABC approach can be extended to ABC versions of MCMC and SMC, which are more efficient algorithms. The tuning results are applicable to SMC algorithms, but further practical methods are needed, as mentioned above. Further theory on optimal tuning is necessary for MCMC, although good performance may be possible with ad-hoc tuning. It would also be of interest to design algorithms in which the initial simulation stages can be resampled and continued many times.

Acknowledgements Thanks to Chris Sherlock, Richard Everitt, Scott Sisson, Christian Robert and two anonymous referees for helpful suggestions, and Robert Erhardt for advice on the spatial extremes example.

A Lazy importance sampling

The approach of lazy ABC can be generalised to non-ABC situations to give *lazy importance sampling* (LIS). This is Algorithm 1 using a likelihood estimator of the form:

$$\hat{L}_{\text{lazy}} = \begin{cases} \hat{L}/\alpha(\theta, X) & \text{with probability } \alpha(\theta, X) \\ 0 & \text{otherwise} \end{cases}$$

In addition to condition C1 from Section 3 assume:

C4 The distribution $(X, \hat{L})|\theta$ is such that $\hat{L}|\theta$ is a non-negative unbiased estimator of $L(\theta)$, and both $X|\theta$ and $\hat{L}|\theta, x$ can be simulated from.

This framework can be used when $\hat{L}$ is an expensive unbiased estimator. It also allows cases where either or both of X and $\hat{L}$ are non-random. For example, X may be a deterministic approximation of the likelihood and $\hat{L}|\theta$ may be a point mass at $L(\theta)$.

Close analogues of the lazy ABC results in this paper hold for LIS. Firstly, a variant of Theorem 1 shows that given conditions C1 and C4, $\hat{L}_{\text{lazy}}|\theta$ is a non-negative unbiased estimator of $L(\theta)$. The same proof can be used replacing $L_{\text{ABC}}(\theta)$ and $\hat{L}_{\text{ABC}}$ with $L(\theta)$ and $\hat{L}$. It follows that LIS targets the posterior distribution.

Secondly, modified versions of equations (5) and (6) on the optimal choice of α hold as before. In particular, condition C3 is still required for (6). The modification is that under LIS $\gamma(\phi) = \mathrm{E}[\hat{L}^2|\phi]$, whereas under lazy ABC $\gamma(\phi)$ can be written as $\mathrm{E}[\hat{L}_{\text{ABC}}|\phi]$. The lazy ABC and LIS results are both proved by Theorem 2 in Appendix C.

Exploratory investigation suggests that tuning LIS is harder in practice than lazy ABC. This is because $\mathrm{E}(\hat{L}^2|\phi)$ can be strongly influenced by the upper tail of $\hat{L}|\phi$ which is hard to estimate from pilot run output. Furthermore it is unclear whether the potential gains of LIS are comparable to lazy ABC. Under ABC as $\epsilon \to 0$ the acceptance rate typically converges to zero, so for small ϵ there is scope to save time by early stopping since most iterations do not contribute to the final sample. It is unclear whether this is the case under importance sampling with a good choice of $g(\theta)$.

Further work to investigate the usefulness of LIS as an inference method is required. LIS is also included because it is a useful device for simplifying the proofs in the remaining appendices.

B Multiple stopping decisions

The lazy ABC framework of Section 3 allows multiple stopping decisions, as follows. As in that section assume Y is a deterministic transformation of a latent vector $X_{1:p}$.

Example B1: Multiple stopping decisions Let $X = X_{1:p}$ and $\alpha(\theta, x) = \prod_{i=1}^{s} \alpha^{(i)}(\theta, x_{1:t_i})$. Thus, for each $1 \leq i \leq s$, once simulation of $X_{1:t_i}$ has been performed then $\hat{L}_{\text{lazy}}$ is set to zero with a certain probability, in which case no further simulation is necessary. It is often be useful to let $\alpha^{(i)}(\theta, x_{1:t_i}) = \alpha^{(i)}(\phi_i(\theta, x_{1:t_i}))$. That is, each stopping decision has

associated decision statistics ϕ_i.

Example B2: Multiple random stopping times As for Example B1 but with each t_i replaced with a random stopping time variable T_i. This permits stopping to be considered when various random events occur, without imposing a fixed order of occurrence.

The following alternative characterisation of these examples is useful below.

Lemma 1. *For any $1 \leq i \leq s$, Examples B1 and B2 can be represented as a lazy importance sampling algorithm with continuation probability $\alpha^{(i)}(\phi_i)$ and*

$$\hat{L} = \begin{cases} \hat{L}_{ABC}/\beta_i(\theta, X) & \textit{with probability } \beta_i(\theta, X) \\ 0 & \textit{otherwise,} \end{cases}$$

where $\beta_i(\theta, x) = \prod_{j \neq i} \alpha^{(j)}(\phi_j)$.

Proof. The likelihood estimator stated can easily be verified to have the same distribution as $\hat{L}_{\text{lazy}}$. □

It is also helpful to define $\bar{T}_{2i}(\theta, \phi_{1:s})$ as the expected time remaining from the calculation of ϕ_i until the likelihood estimate is computed conditional on θ and $\phi_{1:s}$, and $\bar{T}_{2i}(\phi_i) = \mathrm{E}[\bar{T}_{2i}(\theta, \phi_{1:s})|\phi_i]$.

B.1 Tuning

The efficiency estimate of Section 4.2.3 can be used in a multiple stopping decision setting given a choice of α. It is necessary to update the estimator $\hat{T}$ given there which is usually a straightforward task. Sections B.1.1 and B.1.2 describe situations of practical interest where the optimal form of α can be derived. However in general the problem is challenging, as illustrated by Section B.1.3.

B.1.1 Discrete decision statistics

Suppose $\alpha(\theta, x) = \prod_{i=1}^{s} \alpha^{(i)}(\phi_i)$ where $\phi_i(\theta, x)$ takes values in $\{1, 2, \ldots, d_i\}$ for d_i finite. Tuning requires selecting a finite number of $\alpha^{(i)}(\phi_i)$ values to optimise the efficiency estimate, which is possible by standard numerical optimisation methods. However note that producing an efficiency estimate as in Section 4.2.3 becomes difficult for large s.

B.1.2 One continuous decision statistic

Suppose $\alpha(\theta, x) = \prod_{i=1}^{s} \alpha^{(i)}(\phi_i)$ where $\phi_1(\theta, x)$ is continuous and $\phi_i(\theta, x)$ is as in Section B.1.1 for $i > 1$. Also suppose there exists $u_1(\phi_1) = \pi(\theta)/g(\theta)$, so that condition C3 holds. Applying Lemma 1 and (6) with $\gamma(\phi) = \mathrm{E}[\hat{L}^2|\phi]$, as justified in Appendix A, gives that efficiency is optimised by

$$\alpha^{(1)}(\phi_1) = \min\left\{1, \lambda u_1(\phi_1)\left[\frac{\gamma_1(\phi_1)}{\bar{T}_{21}(\phi_1)}\right]^{1/2}\right\}, \tag{11}$$

where $\gamma_i(\phi_1) = \mathrm{E}\left[\zeta(\beta_i(\theta, X))\mathbb{1}\left\{d(s(Y), s(y_{\mathrm{obs}})) \leq \epsilon\right\}|\phi_i\right]$, $\zeta(0) = 0$ and $\zeta(x) = x^{-1}$ for $x > 0$.

In general $\gamma_1(\phi_1)$ and $\bar{T}_{21}(\phi_1)$ depend on $\alpha^{(i)}(\phi_i)$ for $i > 2$ and so must be estimated several times during the tuning process which is costly. A special case where this can be avoided is when $\phi_{2:p}$ is fully determined by ϕ_1 (and so typically the decision associated with α_1 is guaranteed to occur last). For example this is the situation in the second simulation study of Section 6.2.

B.1.3 Multiple continuous decision statistics

Consider the setting of B.1.2 with the modification that every $\phi_i(\theta, x)$ is continuous and there exists a corresponding function $u_i(\phi_i) = \pi(\theta)/g(\theta)$. The same approach as above gives equations of the form

$$\alpha^{(i)}(\phi_i) = \min\left\{1, \lambda_i u_i(\phi_i)\left[\frac{\gamma_i(\phi_i)}{\bar{T}_{2i}(\phi_i)}\right]^{1/2}\right\},$$

for $i = 1, \ldots, s$. The definition of γ_i involves $\alpha^{(j)}$ for all $j \neq i$, and $\bar{T}_{2i}$ will also involve many of these terms. Thus deriving the optimal $\alpha^{(i)}$ functions involves solving a complicated system of non-linear implicit equations.

C Tuning α

This appendix concerns tuning α to optimise the asymptotic efficiency of lazy importance sampling. Theorem 2 on this topic is stated and several earlier results are shown to be corrolaries. The proof is given in Appendix C.1.

Recall from Section 4.1 that efficiency is defined here as N_{eff}/T where T is the CPU time of the algorithm and $N_{\text{eff}} = NE(W)^2/E(W^2)$. In the latter W represents importance sampling weight and N number of iterations. Assume that T follows a central limit theorem in N. Then the delta method gives that for large N efficiency asymptotically equals

$$\frac{\mathrm{E}(W)^2/\,\mathrm{E}(W^2)}{\mathrm{E}(T)/N}. \tag{12}$$

Theorem 2. *Fix some decision statistics $\phi(\theta, x)$. Amongst continuation probability functions of the form $\alpha(\theta, x) = \alpha(\phi(\theta, x))$, asymptotic efficiency* (12) *is maximised by the following expression for some $\lambda > 0$,*

$$\alpha(\phi) = \min\left\{1, \lambda\left[\frac{\mathrm{E}[\hat{L}^2 \frac{\pi(\theta)^2}{g(\theta)^2}|\phi]}{\bar{T}_2(\phi)}\right]^{1/2}\right\}. \tag{13}$$

Note that the expectation in the numerator is conditional on ϕ, with θ and x, and y in the lazy ABC case described below, treated as random.

If condition C3 holds then (13) simplifies to (6) with $\gamma(\theta) = \mathrm{E}(\hat{L}^2|\theta)$. If $\phi(\theta, x) = (\theta, x)$ it simplifies further to (5). This proves the results stated in Appendix A.

To apply Theorem 2 to lazy ABC let $\hat{L} = \hat{L}_{\text{ABC}}$ (as defined in (3)). Then LIS is equiv-

alent to the lazy ABC algorithm. Note that since $\hat{L}_{\text{ABC}}$ is a Bernoulli random variable, $\hat{L}^2 = \hat{L}_{\text{ABC}}$. If condition C3 holds then (13) simplifies to (6) with $\gamma(\theta) = \text{E}(\hat{L}_{\text{ABC}}|\theta) = \Pr\left(d(s(Y), s(y_{\text{obs}})) \leq \epsilon|\theta, x\right)$. If $\phi(\theta, x) = (\theta, x)$ it simplifies further to (5). This proves the results stated in Section 4.

C.1 Proof of Theorem 2

In LIS the importance sampling weight W equals $\frac{\hat{L}\pi(\theta)}{\alpha(\phi)g(\theta)}$ with probability $\alpha(\phi)$ and zero otherwise. Hence:

$$\text{E}(W^2) = \int \frac{\text{E}[\hat{L}^2|\theta, \phi, y]\pi(\theta)^2}{\alpha(\phi)g(\theta)^2}\pi(\phi, y|\theta)g(\theta)d\theta d\phi dy = \int \frac{\xi(\phi)}{\alpha(\phi)}g(\phi)d\phi, \tag{14}$$

where $\xi(\phi) = \text{E}\left[\hat{L}^2\left(\frac{\pi(\theta)}{g(\theta)}\right)^2\middle|\phi\right]$ and $g(\phi) = \int \pi(\phi|\theta)g(\theta)d\theta$.

The expected time of a single iteration of the LIS algorithm is

$$\text{E}(T)/N = \bar{T}_1 + \int \alpha(\phi)\bar{T}_2(\theta, \phi)\pi(\phi|\theta)g(\theta)d\theta d\phi = \bar{T}_1 + \int \alpha(\phi)\bar{T}_2(\phi)g(\phi)d\phi. \tag{15}$$

Note that $\text{E}(W)$ is a constant, so choosing $\alpha(\phi)$ to maximise expression (12) is equivalent to minimising $\text{E}(W^2)\,\text{E}(T)/N$. Call this problem P. Consider also the problems $P(\upsilon)$, minimising $\text{E}(W^2)$ under the constraint $\text{E}(T)/N = \upsilon$ and $P(\upsilon, \mu)$, minimising $\text{E}(W^2) + \mu[E(T)/N - \upsilon]$, or equivalently

$$\int \left[\frac{\xi(\phi)}{\alpha(\phi)} + \mu\alpha(\phi)\bar{T}_2(\phi)\right] g(\phi)d\phi. \tag{16}$$

Note that $P(\upsilon, \mu)$ is a Lagrange multiplier form of $P(\upsilon)$. Consider only $\mu > 0$. Also, let Υ be the set of $\text{E}(T)/N$ values attainable by some choice of α.

First consider minimising (16) subject to $0 \leq \alpha(\phi) \leq 1$. This can be done by pointwise

optimisation of the integrand. With α unconstrained the solution is

$$\alpha^*(\phi) = \lambda \left[\frac{\xi(\phi)}{\bar{T}_2(\phi)} \right]^{1/2}, \tag{17}$$

where $\lambda = \mu^{-1/2}$. Also note that $\alpha^*(\phi)$ may sometimes be infinite. The derivative of the integrand with respect to α is negative for $\alpha < \alpha^*$. Hence if $\alpha^*(\phi) > 1$, the constrained solution is $\alpha(\phi) = 1$, giving the global solution (13) from the theorem statement.

Substituting (13) into (14) and (15) shows that the resulting values of $\mathrm{E}(W^2)$ and $\mathrm{E}(T)/N$ are continuous in λ. Furthermore all $\mathrm{E}(T)/N$ values in Υ are attainable by (13) under some choice of λ. Hence given $\upsilon \in \Upsilon$ there is some μ^* for which the solution to $P(\upsilon, \mu^*)$ has $\mathrm{E}(T)/N = \upsilon$. This must also be a solution to $P(\upsilon)$ since otherwise a superior choice of α for $P(\upsilon)$ is also superior for $P(\upsilon, \mu^*)$. Now choose υ^* so that the solution to $P(\upsilon^*)$ minimises $\mathrm{E}(W^2)\,\mathrm{E}(T)/N$. This must be a solution to P since otherwise a superior choice of α for P is superior to the solution already found for some $P(\nu)$.

D Tuning g

This section proves Corollary 1. First the RW-IS case is considered. Recall that in this case $\gamma(\theta) = \mathrm{E}(\hat{L}^2|\theta)$.

RW-IS can be seen as a special case of LIS where $\phi = \theta$, $\bar{T}_1(\theta) = 0$ and $\bar{T}_2(\phi) = \bar{T}(\theta)$. Repeating the working of Appendix C.1 to optimise the choice of $\alpha(\theta)g(\theta)$ gives the unconstrained solution:

$$\alpha(\theta)g(\theta) = \lambda\pi(\theta) \left[\frac{\gamma(\theta)}{\bar{T}(\theta)} \right]^{1/2}. \tag{18}$$

Various choices of α, such as $\alpha \equiv 1$, give a solution which also meets the constraint on α. These all give algorithms which are equivalent to RW-IS with $g(\theta) \propto \pi(\theta) \left[\frac{\gamma(\theta)}{\bar{T}(\theta)} \right]^{1/2}$ as claimed.

Finally, to prove the ABC-IS case, note that this is a special case of RW-IS with $\hat{L} = \hat{L}_{\mathrm{ABC}}$

so the same result holds. Since $\hat{L}_{\text{ABC}}$ is Bernoulli, $\gamma(\theta) = \text{E}(\hat{L}^2|\theta) = \text{E}(\hat{L}_{\text{ABC}}|\theta)$ as claimed.

References

Andersson, H. and Britton, T. (2000). *Stochastic epidemic models and their statistical analysis*. Springer New York.

Andrieu, C. and Roberts, G. O. (2009). The pseudo-marginal approach for efficient Monte Carlo computations. *The Annals of Statistics*, pages 697–725.

Beaumont, M. A. (2003). Estimation of population growth or decline in genetically monitored populations. *Genetics*, 164(3):1139–1160.

Beaumont, M. A. (2010). Approximate Bayesian computation in evolution and ecology. *Annual Review of Ecology, Evolution and Systematics*, 41:379–406.

Beaumont, M. A., Cornuet, J. M., Marin, J.-M., and Robert, C. P. (2009). Adaptive approximate Bayesian computation. *Biometrika*, 96(4):983–990.

Brooks-Pollock, E., Roberts, G. O., and Keeling, M. J. (2014). A dynamic model of bovine tuberculosis spread and control in Great Britain. *Nature*, 511(7508):228–231.

Buzbas, E. O. and Rosenberg, N. A. (2013). AABC: approximate approximate Bayesian computation when simulating a large number of data sets is computationally infeasible. *Preprint*. Available at `http://www.arxiv.org/abs/1301.6282`.

Erhardt, R. J. and Smith, R. L. (2012). Approximate Bayesian computing for spatial extremes. *Computational Statistics & Data Analysis*, 56(6):1468–1481.

Fearnhead, P., Papaspiliopoulos, O., Roberts, G. O., and Stuart, A. (2010). Random-weight particle filtering of continuous time processes. *Journal of the Royal Statistical Society: Series B*, 72(4):497–512.

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

Geweke, J. (1989). Bayesian inference in econometric models using Monte Carlo integration. *Econometrica*, 57(6):1317–1339.

Liu, J. S. (1996). Metropolized independent sampling with comparisons to rejection sampling and importance sampling. *Statistics and Computing*, 6(2):113–119.

Marin, J.-M., Pudlo, P., Robert, C. P., and Ryder, R. J. (2012). Approximate Bayesian computational methods. *Statistics and Computing*, 22(6):1167–1180.

McKinley, T. J., Cook, A. R., and Deardon, R. (2009). Inference in epidemic models without likelihoods. *The International Journal of Biostatistics*, 5(1).

Meeds, E. and Welling, M. (2014). GPS-ABC: Gaussian process surrogate approximate Bayesian computation. In Zhang, N. L. and Tian, J., editors, *Uncertainty in Artificial Intelligence: Proceedings of the Thirtieth Conference*, pages 593–602.

Moores, M. T., Drovandi, C. C., Mengersen, K., and Robert, C. P. (2014). Pre-processing for approximate Bayesian computation in image analysis. *Preprint*. Available at `http://www.arxiv.org/abs/1403.4359`.

Ralston, A., Reilly, E. D., and Hemmendinger, D., editors (2003). *Encyclopedia of Computer Science*. John Wiley and Sons Ltd., Chichester, UK, 4th edition.

Ribatet, M., Singleton, R., and R Core team (2013). *SpatialExtremes: Modelling Spatial Extremes*. R package version 2.0-0.

Schlather, M. (2002). Models for stationary max-stable random fields. *Extremes*, 5(1):33–44.

Tran, M.-N., Scharth, M., Pitt, M. K., and Kohn, R. (2014). Importance sampling squared for Bayesian inference in latent variable models. *Preprint*. Available at `http://www.arxiv.org/abs/1309.3339`.

Wilkinson, R. D. (2014). Accelerating ABC methods using Gaussian processes. In *Proceedings of the 17th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 1015–1023.

Wood, S. (2011). Fast stable restricted maximum likelihood and marginal likelihood estimation of semiparametric generalized linear models. *Journal of the Royal Statistical Society: Series B*, 73(1):3–36.