Statistical Applications in Genetics and Molecular Biology

Volume 10, Issue 1

2011

Article 45

Choice of Summary Statistic Weights in Approximate Bayesian Computation

Hsuan Jung, *University of Southern California*
Paul Marjoram, *University of Southern California*

Recommended Citation:

Jung, Hsuan and Marjoram, Paul (2011) "Choice of Summary Statistic Weights in Approximate Bayesian Computation," *Statistical Applications in Genetics and Molecular Biology*: Vol. 10: Iss. 1, Article 45.

DOI: 10.2202/1544-6115.1586

Available at: <http://www.bepress.com/sagmb/vol10/iss1/art45>

©2011 Berkeley Electronic Press. All rights reserved.

Choice of Summary Statistic Weights in Approximate Bayesian Computation

Hsuan Jung and Paul Marjoram

Abstract

In this paper, we develop a Genetic Algorithm that can address the fundamental problem of how one should weight the summary statistics included in an approximate Bayesian computation analysis built around an accept/reject algorithm, and how one might choose the tolerance for that analysis. We then demonstrate that using weighted statistics, and a well-chosen tolerance, in such an approximate Bayesian computation approach can result in improved performance, when compared to unweighted analyses, using one example drawn purely from statistics and two drawn from the estimation of population genetics parameters.

KEYWORDS: approximate Bayesian computation, genetic algorithms, summary statistics

Author Notes: Hsuan Jung, Keck School of Medicine, Preventive Medicine, University of Southern California. Paul Marjoram, Keck School of Medicine, Preventive Medicine, University of Southern California. The authors are grateful to two anonymous reviewers for comments that greatly strengthened this manuscript, and to the NIH and NSF for funding support through awards MH084678 and DMS1101060.

1 Introduction

Approximate Bayesian computation [ABC] is an analysis approach that has become increasingly popular over recent years, in part because as the complexity of modern data sets increases, the ability to perform exact computation, such as explicitly calculating posterior distributions, decreases. While the use of ABC methods is becoming more common, there is substantial scope for putting the approach on a more rigorous footing. In this paper we demonstrate that using weighted statistics in an ABC approach results in improved performance, when compared to unweighted analyses, over a range of example applications. In order to do this we develop a Genetic Algorithm [GA] that attempts to choose optimal weights for any given analysis. We then show a range of examples that illustrate that using weighted statistics in an ABC approach can result in improved performance, when compared to unweighted analyses.

We are in the midst of an era in which the complexity of datasets is growing extremely rapidly, particularly in the genetics community. This applies both in the sense that the amount of data appears to be growing exponentially and in that the level of detail of the data is much greater than before. In a loose sense, as (Bayesian) statisticians we spend a large part of our lives solving the following equation

$$f(\Theta | D) = f(D | \Theta)\pi(\Theta)/f(D). \quad (1)$$

Here, D represents a dataset of interest, while Θ denotes one or more parameters that are believed to influence features of the data. Thus $f(\Theta | D)$ is the posterior distribution for Θ , which is calculated as a function of the prior distribution π . In most of the examples in this paper D is a collection of genetic data, in the form of Single Nucleotide Polymorphisms [SNPs], while Θ denotes the mutation and/or recombination rates that underpin D . Our interest here is in analyses that rely upon a model, M , that is thought to capture key features of the relationship between the data and the parameters - indeed Θ denotes the parameters of that model. For the sake of notational convenience we suppress the dependency on M throughout. In an ideal world, we would calculate $f(\Theta | D)$ exactly. This becomes problematic for several reasons. On a trivial level, the term $f(D)$ is frequently intractable. However, since it represents a normalizing constant, we can rely on the fact that $\int_{\Theta} f(\Theta) d\Theta = 1$, to calculate the normalizing constant “after the fact”. A more fundamental problem arises when, as occurs increasingly in the modern era, we cannot calculate the likelihood term $f(D | \Theta)$ (and the same problem occurs for frequentists, of course). In this context, we resort to a more approximate analysis, where the approximation can be conducted in one of two places:

1. At the level of the model: We adopt a simpler model for which we are able to calculate the term $f(D | \Theta)$ explicitly. The price paid here is that the model now reflects reality less exactly. As the industrial statistician George Box noted: “All models are wrong; some are useful” (Box, 1979). In this case the model becomes more wrong, and often less useful as a result. An example of adopting a simpler model is the composite likelihood approach, such as Hudson’s estimate of recombination rate (Hudson, 2001) in which independence between non-independent pairs of loci is assumed in order to restore tractability.
2. At the level of the analysis: We continue to use a complex model, in that hope that it reflects reality more closely, but now calculate some $\phi(\Theta | D)$, where ϕ can be calculated in a (relatively) straightforward manner and is believed to closely approximate f . An example of this approach has recently acquired the label ABC (Beaumont, Zhang, and Balding, 2002, Marjoram and Tavaré, 2006).

Both schemes provide approximations to $f(\Theta | D)$. The difference is in where the approximation is made: by simplifying the model, so that you then get an exact answer for a less accurate model; or by introducing some tolerance in the degree of agreement between simulated and observed data, so that you can use a more realistic model but will estimate an approximation to the desired posterior.

Our purpose in this paper is not to debate the relative merits of these two choices - both are widely used - but instead to address the manner in which the approximate distribution ϕ is constructed when using ABC. This depends crucially upon statistics that are chosen to summarize the data. We begin by briefly introducing the details of the ABC approach used in our examples, an *accept/reject* algorithm, and then focus on how one can choose weights that can be applied to summary statistics in that method in order to improve the quality of the approximation ϕ . We introduce an approach that applies GAs to choose those weights and illustrate our method using a number of examples.

2 Methods

2.1 Approximate Bayesian Computation

ABC methods are a form of accept/reject algorithms. Accept/reject algorithms, for convenience abbreviated to rejection methods [RMs] throughout this manuscript, were first introduced by (von Neumann, 1951), and rely on the following simple intuition: two samples are most likely to have similar summary statistics if they

are generated by similar parameters (assuming that the statistics are informative for those parameters). More formally, suppose we are summarizing data using a set of n summary statistics $\mathbf{S} = \{S_1, \dots, S_n\}$, and have an underlying model M . We note that, for ease of interpretation, throughout this paper all statistics are normalized to have mean 0 and variance 1. We do this by simulating a large number of data-sets, sampling parameter values from the prior, and calculating the mean and variance of the statistic values on those data. Further, suppose we have a set of weights $\{w_1, \dots, w_n\}$ that reflect the *importance* of each statistic (where importance is used informally to denote the amount of information that statistic carries regarding the parameter(s) of interest). We repeat the following scheme for $j = 1, \dots, N$, for some large N :

1. Sample parameter(s) Θ_j from the prior $\pi(\Theta)$;
2. Simulate data D_j from M using parameter(s) Θ_j ;
3. Calculate $d_j = \sum_i w_i |S_{ij} - S_{iO}|$, where S_{ij} is the value of S_i on D_j , and S_{iO} is the value of S_i on the observed data of interest, D ;
4. If $d_j < \varepsilon$, where ε is a user defined tolerance (typically small), *accept* this iteration and store the value of Θ_j .

The set of accepted Θ_j form an empirical approximation, which we denote by ϕ_e , to the distribution $\phi = f(\Theta \mid D' \approx D)$, where the condition $D' \approx D$ is defined explicitly as $\sum_i |S_{iD'} - S_{iO}| < \varepsilon$, in an obvious extension of notation. This approach has been widely used in the literature (*e.g.*, Tavaré, Balding, Griffiths, and Donnelly (1997), Plagnol and Tavaré (2004), Innan, Zhang, Marjoram, Tavaré, and Rosenberg (2005)), although in none of these cases were weights employed in step 3.

Here there is an implicit dependence upon two things: first the choice of constant ε ; second the choice of distance metric $|\cdot|$. In particular, the choice of ε represents a trade-off between computational efficiency (which improves as ε increases), and accuracy of approximation (which generally improves as ε decreases). The accuracy of the empirical approximation can be made arbitrarily good (subject to computation considerations) by simply increasing N , the number of iterations performed. Choice of metric is somewhat arbitrary. Here, as is common, we use Euclidean distance.

2.2 Genetic algorithms

GAs are an approach in which a population of M computational algorithms, referred to here as *chromosomes*, is allowed to evolve over time in an attempt to produce an algorithm that efficiently addresses a problem of interest. Informally speaking, the

GA population evolves through a number of discrete generations, $G_1, \dots, G_k$. In each generation i , the set of chromosomes that exist in that generation, $C_{i,1}, \dots, C_{i,M}$, each have their *fitness* assessed, using some fitness function $F()$ which is a measure of how successfully each chromosome solves the problem of interest. The existing population of chromosomes then ‘reproduces’ to form a population of new individuals in the next generation, where the expected number of *offspring* produced by chromosome $C_{i,j}$ is an increasing function of $F(C_{i,j})$. A full discussion of GAs is outside the scope of this paper, but we refer the reader to Mitchell (1996) for a nice overview of the field at an introductory level.

In this paper we develop a GA that can address two fundamental problems:

1. How one should weight the summary statistics included in an ABC analysis, and
2. What threshold ε should be used in step 3. of the rejection method above?

This is one of a number of fundamentally important considerations that affect the closeness of the approximation ϕ to the posterior f .

Some progress has been made in the above areas. Beaumont et al. (2002) developed a method in which all iterations of a rejection algorithm are accepted but in which acceptances are weighted by their distance from the target, but did not address the issue of how to weight statistics. Joyce and Marjoram (2008) presented a method for deciding which subset of a potential collection of statistics might be used, but did not address whether those statistics should be weighted, or what tolerance ε should be used in the accept/reject step. Hamilton, Currat, Ray, Heckel, Beaumont, and Excoffier (2005) used weights defined by a local regression when performing an ABC estimation of migration rates. Wegmann, Leuenberger, and Excoffier (2009) and Bazin, Dawson, and Beaumont (2010) defined a reduced number of dimensions in which data varied in the most informative way, using partial least-squares and principal components approaches respectively, which is analogous to constructing a smaller number of meta-statistics. Returning to the issue of choice of threshold, ε , (Blum and Francois, 2010, Blum, 2010a,b) consider the issue from a non-parametric perspective. These ideas have been built upon by (Ratmann, Andrieu, Wiuf, and Richardson, 2009, Fearnhead and Prangle, 2010).

More generally, while theoretical progress in the issues related to ABC was initially relatively slow to occur, the volume of papers is now growing (*e.g.*, Sisson, Fan, and Tanaka (2007, 2009), Beaumont, Cornuet, Marin, and Robert (2009), Sisson et al. (2009)). An increasing number of ABC applications also now exist (*e.g.*, Estoup, Wilson, Sullivan, Cornuet, and Moritz (2002), Beaumont and Rannala (2004), Fagundes, Ray, Beaumont, Neuenschwander, Salzano, Bonatto, and Excoffier (2007), Bortot, Coles, and Sisson (2007), Jensen, Thornton, and Andolfatto (2008), Cornuet, Santos, Beaumont, Robert, Marin, Balding, Guillemaud, and Estoup (2008), Foll, Beaumont, and Gaggiotti (2008), Guillemaud, Beaumont, Ciosi,

Cornuet, and Estoup (2009), Lopes and Beaumont (2010)). An excellent overall review of the field can be found in Beaumont (2010).

The choice of summary statistics to use, or the weighting of those statistics, can be guided by intuition where good intuition is present, but is in general a non-trivial problem. While, in theory, adding extra statistics can only improve the degree of approximation between the two distributions ϕ and f , without weighting of statistics the effect can in fact be to make the approximation worse. This is because rather than construct ϕ directly, ABC constructs ϕ_e , an empirical approximation to ϕ , and the accuracy of this latter approximation decreases as the acceptance rate of the algorithm decreases. Adding uninformative statistics decreases the acceptance rate, and thereby lowers the accuracy of the analysis. For this reason it is key to choose to put high weight on a small but highly informative set of statistics. However, this is a non-trivial problem when good intuition is lacking. Similar considerations arise in the choice of threshold ε . A small ε is likely to result, in principle, in a closer level of agreement between ϕ and f , but also results in a lower acceptance rate, and a noisier empirical approximation ϕ_e to ϕ . This paper formalizes this intuition and presents a GA method for choosing the weights and tolerance to be used.

2.3 Applying Genetic Algorithms to ABC

We develop a GA to choose sensible weights for each of a set of summary stats relevant to the problem of interest, along with an optimal value for the threshold ε in step 3 of the rejection algorithm. The goal is to find the set of weights that optimizes the performance of the ABC algorithm. We begin by giving details of how each individual chromosome in the current population of GAs is parameterized, before going on to describe how fitness of those chromosomes is assessed and then used to construct the next generation.

In our application of the GA each chromosome consists of a set of weights, along with a threshold ε . Specifically, suppressing the dependency on generation for notational convenience, and assuming we are working with n possible summary statistics, chromosome i is denoted by $C_i = (W_{1i}, W_{2i}, \dots, W_{ni}, \varepsilon_i)$. Each generation consists of 100 such chromosomes (an arbitrarily chosen number), and for convenience we also restricted all weights (and ε) to range from 0 to 100. (Our results showed no evidence that any value should usefully be outside that range.)

We evaluate the performance of each chromosome using a set of 100 *training* data sets, $\mathcal{T}$, each of which is analyzed by a rejection analysis using the weights and ε -value carried by that chromosome. In order to reduce noise due to variation in the population of training data we use the same training data in each generation. We explore the effect of making changes to the training data later in this article.

The RMs used 500K data sets to perform the accept/reject step. We refer to these 500K iterations as the *Distribution* data set, $\mathcal{D}$, (see details below - they represent the data used in step 2. of the rejection algorithm). We used a separate, independent replicate of $\mathcal{D}$ for the analysis of each $t \in \mathcal{T}$. The generating parameter values for each element of $\mathcal{D}$ and $\mathcal{T}$ represent independent and identically distributed [i.i.d] draws from the prior parameter distributions. At the end of this procedure we record the posterior distribution for the parameter(s) of interest for each data set $t \in \mathcal{T}$, for each chromosome. From this we calculated the mean as a point estimate, $\hat{\theta}$ of the parameter(s). Our choice is arbitrary - one could of course use mode, median, or integrated squared error, for example - but the mean performed well in our experiments.

The fitness of the chromosome is then defined as the inverse of the sum of squared errors of parameter estimates for every sample in $\mathcal{T}$. To guard against making decisions based on analyses in which very few iterations were accepted, we substituted a penalty term, κ , if the estimation is made by fewer than $n_{\min}$ samples. That is, we define the fitness of chromosome C_i , denoted by $\text{Fit}(C_i)$, as

$$\frac{1}{\sum_{t \in \mathcal{T}} [(\hat{\theta}_t(C_i) - \theta_t)^2 I(N_t(C_i) \geq n_{\min}) + \kappa I(N_t(C_i) < n_{\min})]}, \quad (2)$$

where θ_t is the actual value of parameter θ in sample $t \in \mathcal{T}$; $\hat{\theta}_t(C_i)$ is the estimate of θ_t made by chromosome C_i ; and $N_t(C_i)$ is the number of accepted iterations that result when C_i is used to estimate θ_t via the accept/reject algorithm, *i.e.*,

$$N_t(C_i) = \sum_{d \in \mathcal{D}} I(\sum_j w_{ij} | S_j(t) - S_{ij} | < \epsilon_i),$$

where $S_j(t)$ is the value taken by summary statistic S_j on test data $t \in \mathcal{T}$. We use a penalty of $\kappa = 10^4$ and $n_{\min} = 100$. These numbers are arbitrarily chosen, but result in good performance. When estimating two parameters simultaneously, we extend (2) in the natural way, summing over the squared error of both statistics.

The other crucial component of a GA is the construction of new chromosomes to populate the next generation from those chromosomes present in the current generation. We chose to fix the number of chromosomes in each generation at 100. The chromosomes in the first generation were assigned weights that were independently sampled from a $\text{Unif}(0, 100)$ distribution. In subsequent generations we begin by constructing identical copies of the fittest 5 chromosomes from the previous generation. These are copied without subsequent mutation. Each remaining new chromosome is first generated by copying one of the chromosomes in the previous generation and then being modified. The probability for a chromosome in

the previous generation to be selected as such a ‘parent’ of each new chromosome is proportional to its fitness:

$$\Pr(\text{parent } i \text{ is chosen}) = \frac{\text{Fit}(C_i)}{\sum_j \text{Fit}(C_j)}. \quad (3)$$

After generating a set of offspring chromosomes for the next generation, these offspring are subject to recombination and mutation. First we randomly pair the offspring chromosomes. Each pair then has a probability of 0.5 of undergoing a recombination event. (Efficiency of GAs is typically improved by allowing a process analogous to recombination to occur (Mitchell, 1996).) Denote the two chromosomes in any given recombining pair by $\{w_{i1}, \dots, w_{in}, \epsilon_i\}$ and $\{w_{j1}, \dots, w_{jn}, \epsilon_j\}$. We generate a recombination breakpoint by sampling $B \sim \text{Unif}[2, n]$, an integer-valued Uniform distribution, and then recombine the two offspring, who then become $\{w_{i1}, \dots, w_{iB-1}, w_{jB}, \dots, w_{jn}, \epsilon_j\}$ and $\{w_{j1}, \dots, w_{jB-1}, w_{iB}, \dots, w_{in}, \epsilon_i\}$.

Regardless of how it is constructed, each offspring chromosome is then subject to mutation. We employed two types of mutation. The first type simply replaces the old weight, w , with a new weight, w' , chosen from a $\text{Unif}(0, 100)$ distribution; the second type makes a small change to the existing value by defining $w' = \delta w$, where $\delta \sim \text{Unif}(0.95, 1.05)$ (in the case where $w' > 100$, we set $w' = 200 - w'$). We arbitrarily set the rate of overall mutation to be 0.25. Given that a mutation occurs, we employed the former type of mutation with a probability of 0.4; otherwise we employed the latter type of mutation. Thus, our mutation operator is a mixture of two mechanisms. One in which the new weight is chosen uniformly across the entire range of possible weights, designed as a mechanism for escaping from local fitness maxima, and another in which we make a local perturbation to the weight, which encourages local learning. Mutation to the tolerance ϵ is handled in the same way.

For each analysis, the GA was evolved over 50 generations. We then evaluated the performance of the fittest chromosome in the final generation by analyzing 100 new, independent sets of 10000 data-sets, referred to as the *Validation* data $\mathcal{V}$, computing an m.s.e. for the posterior distribution resulting from each chromosome across each of the 10000 datasets. The generating parameter values for each dataset in $\mathcal{V}$ are again i.i.d. draws from the prior distribution. In an attempt to reduce noise within a single generation, we also assessed the performance of an algorithm constructed by taking the average of the weights and ϵ -values in the single chromosome that was fittest in each of the last five generations.

2.4 Improving Robustness

In order to keep the computational intensity manageable we used a training data set $\mathcal{T}$ consisting of 100 samples. Obviously such a small number of samples may not be completely representative, and there is therefore a danger that the GA will produce chromosomes that have evolved to analyze those particular data sets well, but may perform less well on other, as yet unseen data. To explore the extent to which this might be a problem we employed several methods to allow the training data $\mathcal{T}$ to change over time. Broadly speaking, as well as the results shown below, we found that allowing the training data to change also resulted in somewhat accelerated evolution of the GAs (results not shown). We tested 5 replacement schemes.

1. **Replacement Scheme [RS] 1:** Replace 4% (i.e. 4 samples) of the data sets in $\mathcal{T}$ appearing in the previous generation. Choose the data sets to replace uniformly at random from $\mathcal{T}$.
2. **RS 2:** Replace 4% of the data sets in $\mathcal{T}$ appearing in the previous generation. Choose those elements of $\mathcal{T}$ that are worst estimated by the fittest chromosome in that generation.
3. **RS 3:** Replace 4% of the data sets in $\mathcal{T}$ appearing in the previous generation. Choose those elements of $\mathcal{T}$ that are best estimated by the fittest chromosome in that generation.
4. **RS 4:** Replace 4% of the data sets in $\mathcal{T}$ appearing in the previous generation. Choose 2 data sets for which the fittest chromosome in that generation had the lowest acceptance rate, and 2 data sets for which that chromosome had the highest acceptance rate (that latter case implies that the data is very ‘easy’ to estimate, and might therefore not be particularly useful for assessing fitness of different chromosomes).
5. **RS 5:** Replace 2% of the data sets in $\mathcal{T}$ appearing in the previous generation. Choose the data sets for which the fittest chromosome in that generation had the lowest acceptance rate (implying, in some sense, that these data were hard to estimate).

2.5 Computational Considerations

Since the number of examples considered in this paper is somewhat extensive, and each scenario involves running 100 rejection algorithms 50 times each during the evolution of the GA, we chose to increase the efficiency of our simulations by re-using data. Specifically, for each test data set $t \in \mathcal{T}$ we pre-simulate a distribution data set $\mathcal{D}_t$ - over a wide range of parameters values drawn from the prior, π - and use this same set of 500K distribution data sets as the basis for each run of a

rejection algorithm on that test data t . Furthermore, as seems wise, each chromosome had its fitness assessed on the same *training* data. However, it is important to recall that the fittest chromosomes resulting from the GA have their performance tested on a completely independent set of *validation* data, thus resulting in unbiased estimates of performance. Run-times for the software for choice of weights by GA were around 2 hours on a standard PC per data-set, although we note that this does not include generation of the datasets $\mathcal{T}$, $\mathcal{D}$ and $\mathcal{V}$ - the computational requirements for which will be implementation dependent.

3 Results

We show results of applying our methods to a purely statistical problem and to the estimation of population genetics parameters. In the former, as a simple test of our method in a context in which highly informative summary statistics are known, we estimate parameters for data from Normal distributions. In the latter, we estimate mutation and recombination rates, both separately and together, for simulated genetic data. We present results for various analyses of the independent validation data $\mathcal{V}$.

For convenience, for each algorithm being considered we use the mean of the posterior distribution as a point estimate of the parameter. We then calculate the mean square error [MSE] of this point estimate across the 10K replicate datasets for any given $v \in \mathcal{V}$. We analyze the data in a paired way, so each $v \in \mathcal{V}$ is analyzed by each algorithm, and we then calculate the pairwise differences between the MSEs for each algorithm, for each $v \in \mathcal{V}$. We then present a histogram of the distribution of this difference in MSEs across the 100 sets of data in $\mathcal{V}$. In addition, we statistically test for a pairwise difference in performance between the MSEs resulting from methods using a Wilcoxon signed-rank test to ensure robustness to departures from normality.

We consider results from 3 methods. The label “GA” refers to the results obtained by the fittest chromosome in the final generation of the GA when applied to the new data in $\mathcal{V}$. In addition to this we present two benchmarks of performance. Results labelled “RM” are those obtained by analyzing the data in $\mathcal{V}$ with a rejection method that uses a pre-defined, informative, unweighted set of statistics. However, we note that rather than pick the tolerance ε in an arbitrary and possibly non-optimal way, we instead used the value of ε that was chosen to minimize the MSE of the estimates for parameter values in the *training data* $\mathcal{T}$. Finally, we consider a “Best RM”, labeled “BRM”, which shows results of analyzing the data in $\mathcal{V}$ using a RM with the same pre-defined, informative, unweighted set of statistics that

were used in the RM method, but for which we have chosen the value of ε that minimizes the MSE when analyzing the *Validation data* $\mathcal{V}$. While it is not possible to do the latter in real analyses, because it requires knowledge of the very parameters that are being estimated, it provides a very useful performance benchmark since the results of this BRM over-estimate the accuracy possible using a RM. This is because it will perform at least as well as any RM that might have been used with those same unweighted statistics. In other words, it is an (in reality unobtainable) ‘best case’ scenario.

In all examples shown below, the training data $\mathcal{T}$ consists of 100 data sets and the calibration data $\mathcal{D}$ consists of 500K data sets.

3.1 Application 1: Estimation of Normal Variance

We begin with a ‘proof of principle’ example in which we attempt to estimate the unknown variance of data sets consisting of collections of Normally distributed random variables (RVs). We show results for three cases, according to whether each element of $\mathcal{T}$, $\mathcal{D}$ and $\mathcal{V}$ consisted of 100, 500 or 1000 RVs sampled from a particular Normal distribution. Here, each element of the training data $\mathcal{T}$ consisted of 100 sets of RVs sampled from a $N(\mu, \sigma^2)$ distribution, where μ and σ were sampled independently from $\text{Unif}(10, 20)$ distributions for each $t \in \mathcal{T}$. $\mathcal{D}$ and $\mathcal{V}$ were simulated in the same way.

We used 4 statistics as the basis of our analysis: the first 4 un-centered moments of the data. We show a histogram of results for 100 such analyses in Figure 1. In each histogram we show results for the analysis defined by the best possible rejection method (BRM) and for that resulting from using the weights for statistics estimated by the GA. Specifically, for each element of $\mathcal{V}$ we record the difference between the MSE resulting from the analysis using the BRM, denoted by MSE_{BRM} , and the MSE resulting from the weighted analysis, denoted by MSE_W , using the weights estimated to be optimal after the training period. This is shown on the x-axis. Negative numbers correspond to situations in which the analysis that uses the weighted statistics out-performs the BRM. The y-axis shows the frequency of each range of MSE differences.

We see that, one or two outliers excepted, the analysis resulting from the set of weights and ε -value found by the GA substantially out-performs that resulting from the analysis in which equal weights are used for each statistic (p-value ~ 0 in each of the three comparisons). The absolute degree of improvement made by the GA increases as fewer Normal replicates are analyzed, showing that using well-weighted statistics increases robustness to stochastic variance in the values of those statistics in this example. It is also the case that the GA always results in

a RM that has a higher number of acceptances than the other two RMs [data not shown]. The outlier datasets in which the performance of the GA under-performs that of the BRM (which, we recall, is itself an unobtainable optimum when the data is analyzed in a fair manner), may reflect instances in which the GA has become over-adapted to the training data (i.e. over-fitting). It is likely that this would occur less often were the size of the training set increased. (We restricted ourselves to 100 training data sets because of the computational intensity of completing such a large simulation study.)

Figure 1: Results for analysis of Normal RVs. Panels show, from top to bottom, results for analysis of data sets consisting of 1000, 500, and 100 RVs.

3.2 Application 2: Estimation of Recombination Rates

We now consider the problem of estimating recombination rates from SNP data. There are a large number of existing algorithms for this problem, and we use one of the better of those, the likelihood-based package LDhat (McVean, Myers, Hunt, and Deloukas, 2004), to benchmark our results. The data was analyzed using the

default settings for LDHAT, with the single exception of changing the number of points being estimated from 101 to 200 in order to increase the accuracy of its results). We also specified the maximum recombination rate to match that of the prior we were using, in order to make the comparisons fair. In this example, and all those that follow, each element of $\mathcal{D}$, $\mathcal{T}$ or $\mathcal{V}$ consists of a single set of simulated SNP data, with mutation and recombination rates, θ and ρ , sampled independently from $\text{Unif}(15, 25)$ and $\text{Unif}(0, 10)$ distributions respectively. Furthermore, in all examples 50 haplotypes were simulated for each data set. Data sets were generated by the Fastcoal algorithm of Marjoram and Wall (2006).

The statistics that were used to summarize the data in this and all following applications were as follows

- S1 : the number of mutations.
- S2 : the mean number of pairwise differences.
- S3 : the mean pairwise LD (measured as r^2) for segregating sites. To reduce noise being introduced by the varying distance between pairs of loci, we arbitrarily restrict this statistic to pairs of loci within a distance of 0.5 to 0.6 of each other (where we scale haplotypes to have length 1).
- S4 : the mean value of the cross-ratio ad/bc in the set of 2×2 contingency tables showing counts of the four possible haplotypes at pairs of loci. This measure of association was widely used in linkage studies (Mather, 1951, Bailey, 1961), and was examined in Edwards (1963) and shown to have desirable properties. Here a and d are the counts for the two combinations of concordant pairs, while b and c are counts for the discordant pairs. Since we can arbitrarily designate which allele is annotated as being of type '1', we label the alleles in a way such that this statistic is always less than 1. Again, we restrict this statistic to pairs of loci within of distance of 0.5 to 0.6 of each other.
- S5 : the number of different haplotypes observed.
- S6 : the frequency of the most common haplotype.
- S7 : the number of singleton haplotypes.

This somewhat arbitrarily list was chosen to include statistics having varying degrees information about recombination and mutation rates in genetic data. As before, all statistics were normalized to have mean 0 and variance 1.

We show results for this scenario in Figure 2, where presents a comparison of the three analyses: RM, BRM and GA. Again, for each element of $\mathcal{V}$ we record the difference between the MSE resulting from the two analyses being compared (MSE_{RM} , MSE_{BRM} , and MSE_W). This is shown on the x-axis where, once again, negative numbers correspond to cases in which the first-listed analysis has smaller

MSE than that of the second-listed analysis; the y-axis shows the frequency of each range of MSE differences. Once again, in this more complex scenario the results from the weighted analysis out-perform both the standard RM and the BRM. While the degree of improvement is now smaller, it is significant (Wilcoxon p-values of $3.5e-7$ [W vs. BRM], $2.8e-11$ [W vs. RM]). We also note that, for reference we also show a comparison of performance between BRM and RM, where the BRM out-performs the latter (as it must, by construction - p-value $2e-16$). However, we again recall that the BRM is an unattainable optimum that represents a lower bound to the performance (in terms of MSE) obtainable by an unweighted rejection method.

To further assess performance we also benchmarked the GA analysis by comparing it to results from LDhat. For the data generated here, the GA outperforms LDhat (which gives an MSE of 5.46, compared to 4.78 from the GA - results not shown). However, we have not explored the relative performance of our method and LDhat across a fuller range scenarios, since our method is not intended exclusively for estimation of recombination rate.

We note that we ran replicate analyses in which identical GA schemes were run from different start-points. Results were indistinguishable, indicating that stochastic noise is not a factor here. We also note that the performance of the analysis using weights chosen by the GA was unaffected by the addition of a statistic that represents only noise (results not shown), indicating a degree of robustness to poorly chosen pools of statistics.

In an attempt to further improve performance we explored the effects of different schemes for replacement of the data in $\mathcal{T}$ from generation-to-generation. Results are shown in Table 1. The replacement schemes are shown in the same order as listed in Section 2.4. In each case we run two, paired analyses in which we use the GA to choose weights of summary statistics: one in which no replacement scheme is used; the other in which one of the replacement schemes is employed. We see that for three of replacement schemes no significant improvement in performance of the resulting RM is observed. However in cases in which we replace the two datasets in which the lowest acceptance rates are observed [RS5], or in which we replace data sets with both highest and lowest acceptance rates [RS4] a significant, but small improvement is observed.

When using a replacement scheme such as this, the training data changes from generation-to-generation. Consequently, in an attempt to improve robustness we also explored implementations in which we averaged the weight of each statistic over the fittest chromosome in the last 5 generations of the GA. In Table 2 we compare performance of the RM using weights generated by the GA, but without any replacement scheme to the analysis using the GA again, but in which both the replacement scheme and the averaging of weights across the last 5 generations was used. Overall we see a non-significant change in performance of the resulting RM

Figure 2: Results for estimation of recombination rate.

in most cases, but once again a small, and now more significant improvement when using replacement schemes RS4 and RS5.

Table 1: The effect of replacement schemes during the training of the GA on the estimation of ρ . See text for details. (The RS annotation refers to the description in section 2.4.)

Replacement scheme	p-value	median improvement in MSE
random [RS1]	9.8e-1	0.0001
high error [RS2]	6.0e-1	0.008
low error [RS3]	1.0e-1	-0.03
extreme acceptance [RS4]	1.4e-5	0.06
low acceptance [RS5]	3.5e-4	0.05

Table 2: The effect of averaging statistic weights over the last 5 generations of the GA. See text for details. (The RS annotation refers to the description in section 2.4.)

Replacement scheme	p-value	median improvement in MSE due to averaging
none	0.26	~ 0
random [RS1]	8.2e-2	0.02
high error [RS2]	1.3e-1	0.02
low error [RS3]	1.0e-1	0.02
extreme acceptance [RS4]	1.1e-7	0.07
low acceptance [RS5]	2.0e-7	0.06

It is of interest to observe not only that weighting statistics in a sensible manner improves performance of the resulting RM, as shown above, but also to observe which statistics are weighted most heavily. This is shown in Figure 3, where we show a box plot indicating the distribution of weights chosen by the GA (without use of any replacement scheme). We see that, as one might hope, the two statistics that directly measure quantities relating to linkage disequilibrium[LD], S_3 and S_4 are weighted most heavily. Further, the definition S_4 , which exploits 2x2 contingency tables reporting 2-locus allele frequencies, receives substantially greater weight than the more commonly-used mean pairwise LD, which underpins LDHat, for example. This measure was widely-used in linkage studies, but has, perhaps unfortunately, fallen out of favor more recently.

3.3 Application 2: Estimation of Mutation Rates

We also repeated the above analyses in a context in which we were estimating mutation rates rather the recombination rates. The choice of statistics was the same, since some of those statistics were chosen to carry good information regarding mutation rates. Results are shown in Figure 4, where again the x-axis is difference between the mean square error [MSE] of the estimates resulting from two methods, and the y-axis shows the frequency of each range of MSE differences. We once again see that the weighted analysis, on average, out-performs the BRM ($p = 7.7e - 5$), albeit by a relatively small amount. Again, we recall that the performance of the BRM is an unattainable lower bound (in terms of magnitude of the MSE) for the RM. Both the results for the weighted analysis and the BRM significantly out-perform the standard RM ($p \sim 0$ in both cases). The weighted analysis also out-performed the elegant benchmark estimator provided by (Watterson, 1975) [results not shown]. In Figure 5 we show the distribution of optimal weights for the statistics, with statistic

Figure 3: Box-plots showing distribution of weights of statistics for estimation of rho in Section 3.2.

labels corresponding to the definitions in Section 3.2. We see that most weight is placed upon statistic $S1$, the number of segregating sites.

3.4 Application 3: Joint estimation of recombination and mutation rates

Finally we attempt a more challenging problem: *joint* estimation of recombination and mutation rates. Application of RMs in which two parameters are estimated simultaneously are harder because of the impact on the acceptance rate, which can be expected to decrease sharply as the size of the parameters space increases, but examples do exist (Cornuet et al., 2008, Ratmann et al., 2009). Results for our analysis are shown in Figure 6. Here we see the results from the RM with weights chosen by the GA out-performs the BRM (but the results do not reach significance - $p \sim 0.09$). However, the results using weighted statistics do significantly outperform the standard RM ($p \sim 2e - 5$). It is also of interest to note that the average number of acceptances resulting from the BRM is of the order of 2K per analysis, whereas the average from the RM that results from the GA is 20K, an order of magnitude higher (although this number drops to 6K when using replacement scheme

Figure 4: Results for estimation of mutation rate.

RS4). This may be because of the *post facto* optimization of the choice of tolerance ε in the former. When compared to the performance when estimating each of these parameters separately, the MSE is relatively unchanged [results not shown], showing that in this context, at least, we are capable of estimating both parameters simultaneously. We note that, somewhat counter-intuitively, the MSE for ρ is lower than that for θ . The mutation rate is widely understood to be the easier of the two parameters to estimate, but this will also be a function of the range over which they vary. In our examples, ρ varies between 0 and 10, whereas θ varies between 15 and 25, and the two numbers are sampled independently within each data set. It follows that the degree of variation in properties of the data for that range of ρ is likely higher than that over the range of θ values. The ranges chosen were illustrative, but the relative accuracy of the estimates for ρ and θ is likely to change for different choices of parameter range.

Figure 5: Distribution of optimal weights when estimating theta.

4 Discussion

Over the last decade, as the complexity of data sets has grown, ABC has become an increasingly popular methodology, with a range of applications focusing mostly on genetics. However, a number of open questions remain. In this paper we focused on two specific issues. First, we demonstrated that, in a range of examples, an ABC analysis based on a rejection-algorithm will perform better if summary statistics are weighted when measuring similarity between observed and simulated data. This fits with the intuition that, while all statistics are equal, some are more equal than others in the sense that they carry more information about the parameter(s) of interest. While this intuition is straightforward, it is not commonly exploited in ABC analyses. Given the apparent desirability of weighting statistics, we are faced with the second issue: how should those weights be chosen? In some scenarios we can rely on intuition to guide our choice of weights. In others intuition will be lacking. In either case, it is appealing to have a general methodology to help guide our choices. We believe the methods presented in this paper provide the beginnings of such a framework.

In this paper we do not directly address the issue of which subset of a collection of statistics should be used in an ABC analysis. Our method leaves all statistics in the analysis, but by allowing the weights assigned to statistics to become arbitrarily small, statistics can be down-weighted to the extent that they to all intents

Figure 6: Joint estimation of recombination and mutation rates.

and purposes no longer influence the posterior distributions. In the form in which we implemented the GA here weights are never set identically equal to 0. However, if one did wish to explicitly exclude statistics from the analysis, it is easy to imagine variations on our approach in which weights sometimes mutate to take the value of 0 exactly. If statistics are unweighted, adding extra statistics that carry little information about the parameters of interest will hurt computational efficiency, in the sense that it will lower the acceptance rate. However, if statistics are weighted, adding extra stats will not impact efficiency in a substantial way if they are given low weight.

While we believe GAs might be applied to most any ABC approach based upon rejection algorithms (or other algorithms, for that matter), a number of non-straightforward issues remain. First, a number of the details of the implementation of the GA need to be chosen, and there is little guidance on this issue within the GA literature. For example, we somewhat arbitrarily chose the number of genera-

tions over which our algorithms evolved, as well as the details of the way in which chromosomes mutated. This is typical in GA applications, where one typically experiments somewhat to get populations that appear evolve well, and our choices are consistent with those that are commonly used in the field (Mitchell, 1996). However, we also note that the question of over how many generations one should allow the GA to evolve can largely be answered by observing the mean fitness of the population, and allowing it to evolve until that mean fitness appears to have stabilized.

It is clear that when running GAs there is a danger of adapting the population of algorithms to perform well just on the data sets on which their performance is being tested. Indeed, this is the goal of the algorithm. Dangers similar to that of over-fitting data in a regular statistical analysis exist. These will be particularly prominent if the number of data sets in the test data $\mathcal{T}$ is small. The analysis performed here was computationally intensive, and for that reason we used a reasonably small test data set consisting of 100 separate data sets. Consequently, we explored a variety of replacement schemes in which the exact make up of $\mathcal{T}$ varied over time. In general such schemes showed little improvement over the original implementation of our approach, indicating that 100 data sets is large enough to guarantee reasonable performance. However, we did find that schemes in which data sets with the most extreme acceptance rates were replaced showed some degree of improvement of performance, indicating that removing data sets that are hard to estimate, and replacing them with data that are more ‘typical’, will improve performance. In the present context at least, it seems to be better to train the GA on data that is typical, rather than on data that is unusual or hard to estimate.

In all our analyses the fittest RM resulting from our GA outperformed both rejection methods in which statistics were unweighted, but in which ϵ was chosen in a way to allow artificially good performance, as well as standard benchmark estimators of ρ and θ provided by McVean et al. (2004) and Watterson (1975). While we do believe that this indicates that our method is likely to be able to consistently produce more efficient versions of ABC rejection algorithms, by determining optimal sets of weights and algorithm tolerance, it is important to note that we are not claiming that the resulting ABC algorithms will out-perform (or otherwise) the methods of McVean et al. (2004) and Watterson (1975). Our method, by construction, can only predict parameters values that are within the range of the prior distribution, whereas those two methods are not similarly restrained. To aid a more meaningful comparison, we did restrict the range of estimates from the latter two methods to fall within the same range as that possible from the RM (by reducing the estimated value to the upper limit of our prior when it was predicted to be above that, and performing a similar check at the lower end of the range of the prior), but it remains the case that the relative efficiencies of those methods and our own is likely to be a function of the range of possible parameter values.

In conclusion, this paper demonstrates two things. First, the potential to improve performance in an ABC analysis by using well-chosen weights for the summary statistics upon which that analysis depends. Second, that GAs can be used to choose those weights, and at the same time choose the tolerance ϵ for the actual analysis. The details of the implementation of a GA, in terms of exactly how offspring should be constructed, for example, typically vary from implementation to implementation, preventing a general construction from being presented. Neither is one forced to employ GAs as a method for optimization, and we make no claim to optimality for that method of optimization here. However, the method we introduce here can be applied in any context in which one might use an analysis based upon rejection methods. User-friendly software that implements the methods discussed in this paper is available upon request from PM at pmarjora@usc.edu. The software is written in C++ and compiles under Windows and Mac OS X. As noted, run-times for choosing weights by GA were around 2 hours on a standard PC per data-set. Of course, the computational cost increases linearly with the cost of simulating any single dataset, so in situations in which this is very expensive our method might start to become less practical. In such contexts one might choose to cut computational corners by using the same Distribution data-set for the analysis of each member of the Training data, at some possible cost in terms of overall performance of the GA, (rather than generating a new Distribution dataset for each element of the training data, as we did here).

References

- Bailey, N. (1961): *Introduction to the Mathematics Theory of Genetic Linkage*, Oxford University Press.
- Bazin, E., K. Dawson, and M. Beaumont (2010): “Likelihood-free inference of population structure and local adaptation in a Bayesian hierarchical model,” *Genetics*, 185, 587–602.
- Beaumont, M., J. Cornuet, J. Marin, and C. Robert (2009): “Adaptive approximate Bayesian computation,” *Biometrika*, 96, 983–990.
- Beaumont, M., W. Zhang, and D. Balding (2002): “Approximate Bayesian computation in population genetics,” *Genetics*, 5, 2025–2035.
- Beaumont, M. A. (2010): “Approximate Bayesian computation in evolution and ecology,” *Annual Reviews of Ecology, Evolution and Systematics*, 41, 379–406.
- Beaumont, M. A. and B. Rannala (2004): “The Bayesian revolution in genetics,” *Nat Rev Genet*, 5, 251–261.
- Blum, M. and O. Francois (2010): “Non-linear regression models for approximate Bayesian computation,” *Statistics and Computing*, 20, 60–73.

- Blum, M. G. B. (2010a): “Approximate Bayesian Computation: A nonparametric perspective,” *JASA*, 105, 1178–1187.
- Blum, M. G. B. (2010b): “Choosing the summary statistics and the acceptance rate in Approximate Bayesian Computation,” in G. Saporta and Y. Lechevallier, eds., *COMPSTAT 2010 – Proceedings in Computational Statistics*, 47–56.
- Bortot, P., S. Coles, and S. Sisson (2007): “Inference for stereological extremes,” *JASA*, 102, 84–92.
- Box, G. E. P. (1979): “Robustness in the strategy of scientific model building,” in R. L. Launer and G. N. Wilkinson, eds., *Robustness in Statistics*, New York: Academic Press.
- Cornuet, J.-M., F. Santos, M. Beaumont, C. Robert, J.-M. Marin, D. Balding, T. Guillemaud, and A. Estoup (2008): “Inferring population history with DIY ABC: a user-friendly approach to approximate Bayesian computation,” *Bioinformatics*, 24, 2713–2719.
- Edwards, A. (1963): “The measure of association in a 2×2 table,” *Journal of the Royal Statistical Society. Series A*, 126, 109–114.
- Estoup, A., I. Wilson, C. Sullivan, J.-M. Cornuet, and C. Moritz (2002): “Inferring population history from microsatellite and enzyme data in serially introduced cane toads, *Bufo marinus*,” *Genetics*, 159, 1671–1687.
- Fagundes, N., N. Ray, M. Beaumont, S. Neuenschwander, F. Salzano, S. Bonatto, and L. Excoffier (2007): “Statistical evaluation of alternative models of human evolution,” *Proc. Natl. Acad. Sci.*, 104, 17614–17619.
- Fearnhead, P. and D. Prangle (2010): “Semi-automatic approximate Bayesian computation,” *Arxiv preprint*, arXiv:1004.1112v1.
- Foll, M., M. A. Beaumont, and O. Gaggiotti (2008): “An Approximate Bayesian Computation approach to overcome biases that arise when using amplified fragment length polymorphism markers to study population structure,” *Genetics*, 179, 927–939.
- Guillemaud, T., M. A. Beaumont, M. Ciosi, J. M. Cornuet, and A. Estoup (2009): “Inferring introduction routes of invasive species using approximate Bayesian computation on microsatellite data,” *Heredity*, 104, 88–99.
- Hamilton, G., M. Currat, N. Ray, G. Heckel, M. Beaumont, and L. Excoffier (2005): “Bayesian estimation of recent migration rates after a spatial expansion,” *Genetics*, 170, 409–417.
- Hudson, R. (2001): “Two-locus sampling distributions and their application,” *Genetics*, 159, 1805–17.
- Innan, H., K. Zhang, P. Marjoram, S. Tavaré, and N. Rosenberg (2005): “Statistical tests of the coalescent model based on the haplotype frequency distribution and the number of segregating sites,” *Genetics*, 169, 1763–1777.

- Jensen, J., K. Thornton, and P. Andolfatto (2008): “Bayesian estimator suggests strong, recurrent selective sweeps in *Drosophila*,” *PLoS Genetics*, 4.
- Joyce, P. and P. Marjoram (2008): “Approximately sufficient statistics and Bayesian computation,” *Statistical Applications in Genetics and Molecular Biology*, 7, Article 26.
- Lopes, J. S. and M. A. Beaumont (2010): “ABC: A useful Bayesian tool for the analysis of population data,” *Infect Genet Evol*, 10, 826–833.
- Marjoram, P. and S. Tavaré (2006): “Modern computational approaches for analysing molecular genetic variation data,” *Nat Rev Genet*, 7, 759–770.
- Marjoram, P. and J. D. Wall (2006): “Fast “coalescent” simulation,” *BMC Genetics*, 7:16.
- Mather, K. (1951): *The Measurement of Linkage in Heredity* (2nd ed.), London: Methuen.
- McVean, G., S. Myers, S. Hunt, and P. Deloukas (2004): “The fine-scale structure of recombination rate variation in the human genome,” *Science*, 304, 581–584.
- Mitchell, M. (1996): *An Introduction to Genetic Algorithms*, MIT Press.
- Plagnol, V. and S. Tavaré (2004): “Approximate Bayesian Computation and MCMC,” *Monte Carlo and Quasi-Monte Carlo Methods 2002: Proceedings*.
- Ratmann, O., C. Andrieu, C. Wiuf, and S. Richardson (2009): “Model criticism based on likelihood-free inference, with an application to protein network evolution,” *Proc Natl Acad Sci USA*, 106, 10576–10581.
- Sisson, S., Y. Fan, and M. Tanaka (2007): “Sequential Monte Carlo without likelihoods,” *Proc. Nat. Acad. Sci.*, 104, 1760–1765.
- Sisson, S., Y. Fan, and M. Tanaka (2009): “Correction for Sisson *et al.*, Sequential Monte Carlo without likelihoods,” *PNAS*, 106, 16889.
- Tavaré, S., D. Balding, R. Griffiths, and P. Donnelly (1997): “Inferring coalescence times for molecular sequence data,” *Genetics*, 145, 505–518.
- von Neumann, J. (1951): “Various techniques used in connection with random digits. Monte Carlo methods,” *Nat. Bureau Standards*, 12, 36–38.
- Watterson, G. A. (1975): “On the number of segregating sites in genetical models without recombination,” *Theor. Popn. Biol.*, 7, 256–276.
- Wegmann, D., C. Leuenberger, and L. Excoffier (2009): “Efficient approximate Bayesian computation coupled with Markov chain Monte Carlo without likelihoods,” *Genetics*, 182, 1207–1218.