Computational and Structural Biotechnology Journal 17 (2019) 1082–1090

COMPUTATIONAL AND STRUCTURAL BIOTECHNOLOGY JOURNAL

journal homepage: www.elsevier.com/locate/csbj

Comparison of Markov Chain Monte Carlo Software for the Evolutionary Analysis of Y-Chromosomal Microsatellite Data

Sven Gundlach [a], Olaf Junge [a], Lars Wienbrandt [b], Michael Krawczak [a], Amke Caliebe [a,*]

[a] Institute of Medical Informatics and Statistics, Kiel University, University Hospital Schleswig-Holstein, Brunswiker Strasse 10, 24105 Kiel, Germany
[b] Institute of Clinical Molecular Biology, Kiel University, University Hospital Schleswig-Holstein, Rosalind-Franklin-Strasse 12, 24105 Kiel, Germany

ARTICLE INFO

Article history:
Received 28 February 2019
Received in revised form 17 June 2019
Accepted 25 July 2019
Available online 29 July 2019

Keywords:
Coalescent
Population genetics
Bayesian inference
Convergence
Runtime

ABSTRACT

The evolutionary analysis of genetic data is an important subject of modern bioscience, with practical applications in diverse fields. Parameters of interest in this context include effective population sizes, mutation rates, population growth rates and the times to most recent common ancestors. Studying Y-chromosomal microsatellite data, in particular, has proven useful to unravel the recent patrilineal history of *Homo sapiens* populations. We compared the individual analysis options and technical details of four software tools that are widely used for this purpose, namely BATWING, BEAST, IMa2 and LAMARC, all of which use Bayesian coalescent-based Markov chain Monte Carlo (MCMC) methods for parameter estimation. More specifically, we simulated datasets for either eight or 20 hypothetical Y-chromosomal microsatellites, assuming a mutation rate of 0.0030 per generation and a constant or exponentially increasing population size, and used these data to evaluate the parameter estimation capacity of each tool. The datasets comprised between 100 and 1000 samples. In addition to runtime, the practical utility of the tools of interest can also be expected to depend critically upon the convergence behavior of the actual MCMC implementation. In fact, we found that runtime increased, and convergence rate decreased, with increasing sample size as expected. BATWING performed best with respect to runtime and convergence behavior, but only supports simple evolutionary models. As regards the spectrum of evolutionary models covered, and also in terms of cross-platform usability, BEAST provided the greatest flexibility. Finally, IMa2 and LAMARC turned out best to incorporate elaborate migration models in the analysis process.

1. Introduction

The human Y chromosome is of special interest for evolutionary analyses because it allows research to be focused upon male lineages, thereby complementing mitochondrial DNA-based research into the matrilineal history of populations. Notably, since the Y chromosome is only present in males, and in one copy only, the corresponding effective population size of the Y chromosome (Y-effective population size) equals 1/4 of that of autosomes. Furthermore, unlike for autosomes and the X chromosome, most parts of the Y chromosome are non-recombining [1]. These peculiarities motivated the frequent use of Y-chromosomal genetic data to study human history [2–4]. In contrast to single-nucleotide polymorphisms (SNPs), microsatellite DNA markers, or 'short tandem repeats' (STRs), have relatively high mutation rates facilitating the investigation of more recent population history. In addition, microsatellites provide a higher resolution of lineages than SNPs due to their higher allelic diversity. Taken together, Y-chromosomal microsatellites represent a class of DNA markers that logically have become intensively studied in human evolutionary genetics.

Various software exists to estimate population genetic parameters from genotypic data [5], including Y-effective population sizes, mutation rates, population growth rates and the times to most recent common ancestors, which are estimated under simple population genetic models. Many of the tools are implementations of the coalescent model, a mathematical concept that has become fundamental to population genetics and statistical genetics [6–13]. The standard coalescent model was first introduced by Kingman [14] but has undergone several refinements ever since, including the consideration of population dynamics and population structure as well as of recombination, selection and selfing. Most evolutionary software that is publicly available is Bayesian and coalescent-based. Since coalescent trees are simulated backwards in time and take only common ancestors into account (Fig. 1), they are simpler to generate than forward-simulated genealogies. The true coalescent tree of a given set of Y chromosomal haplotypes is of course unknown, so that integration over all possible trees would be required for the exact estimation of an evolutionary

* Corresponding author.
E-mail addresses: gundlach@medinfo.uni-kiel.de (S. Gundlach), junge@medinfo.uni-kiel.de (O. Junge), l.wienbrandt@ikmb.uni-kiel.de (L. Wienbrandt), krawczak@medinfo.uni-kiel.de (M. Krawczak), caliebe@medinfo.uni-kiel.de (A. Caliebe).

https://doi.org/10.1016/j.csbj.2019.07.014

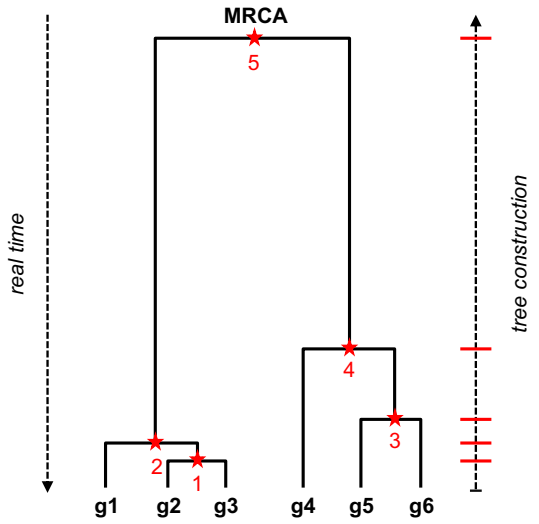

Fig. 1. Exemplary coalescent tree. The coalescent tree featured connects six Y-chromosomal genetic profiles (g1, …, g6) and is constructed backward in time. Coalescent events are depicted as red stars, numbered in the order of their occurrence during the tree construction process. Small red horizontal lines mark the corresponding coalescent times, i.e. the times until two profiles, or groups of profiles, shared a most recent common ancestor. The earliest coalescent time is the time to the most recent common ancestor (TMRCA) of all profiles in the tree.

parameter. In most cases, however, this is computationally impossible. Consequently, the Markov chain Monte Carlo (MCMC) approach has become widely used in statistical genetics because it circumvents the need for formal integration. Instead, MCMC-based software simulates large numbers of trees, each time yielding tree-specific values of the parameters of interest. The trees are simulated so as to properly reflect the probability distribution underlying all possible trees. Consequently, the entirety of the simulation-derived parameter values is taken to reflect the posterior distribution of the parameters of interest.

MCMC has become widely popular for work on both Bayesian and non-Bayesian problems in statistics, particularly biostatistics. One of the hallmarks of MCMC is its applicability to high-dimensional complex data and models [15]. MCMC comes in different guises (including the well-known Metropolis-Hastings algorithm) that differ by the transition probabilities of their respective Markov chains, with Bayesian MCMC underlying most tools for evolutionary analyses.

Inference about evolutionary processes using coalescent-based computational methods is not only of scientific interest in its own right, but has become a crucial basis of many practical applications, for example, in developmental biology, conservation genetics, genetic epidemiology and ancient DNA analysis [16,17]. In forensics, Y-chromosomal microsatellite profiles are particularly important because they allow pinpointing of male trace donors. Y-chromosomal microsatellites are mainly used for disentangling mixed traces with a strong female component, such as in sexual assault cases. One well established application of the coalescent in forensics is the simulation-based estimation of Y-profile frequencies with BATWING [18].

In the present study, we compared four open source software tools for the evolutionary analysis of Y-chromosomal microsatellite data. All tools use MCMC-based simulation of coalescent trees for named purpose. BATWING [16], which was one of the pioneer tools, is an implementation of the Metropolis-Hastings algorithm of MCMC. Another frequently used tool is BEAST, together with related software BEAST 2 [12,13]. Both aim at a combination of user-friendliness and applicability to a variety of research questions. The IM software is an implementation of so-called 'Isolation with Migration (IM)' models. IMa2 [19] is a widely used tool in evolutionary biology that allows for population divergence, which is not the case for BATWING and BEAST. Finally, LAMARC and its fork MIGRATE [20,21] focus upon migration between populations and support an extensive variety of such models. In our study, we characterized BATWING, BEAST, IMa2 and LAMARC with regard to their type of implementation, modelling options, estimation accuracy, runtime behavior and conceptual limitations.

2. Materials and Methods

2.1. Evaluated Software

The goal of the present study was to compare software tools that use an MCMC coalescent framework for the evolutionary analysis of Y-chromosomal microsatellite data. Suitable tools were initially identified through a search on Google Scholar and PubMed. We used combinations of various search terms, including 'Bayesian', 'Markov chain Monte Carlo', 'evolutionary', 'genealogical inference', 'coalescent', 'microsatellite', 'STR' and 'Y chromosome'. The results were then used to guide additional searches in online forums (e.g. Google Groups) and tool websites as well as the consultation of referenced publications.

The criteria for selecting a given tool for further analysis comprised (i) an MCMC coalescent framework with Bayesian analysis of single chains in single populations, (ii) the ability to handle Y-chromosomal microsatellite data, and (iii) open source license. Four tools suitable for evaluation were identified, namely BATWING, BEAST, IMa2 and LAMARC.

- BATWING ('Bayesian Analysis of Trees With Internal Node Generation') was written in C by Wilson and Balding [16] and was intended for the analysis of within-species data.
- BEAST ('Bayesian Evolutionary Analysis Sampling Trees') is available in two major versions, BEAST and BEAST 2 [12,13,22], both of which are continually developed further. At the time of evaluation, however, only BEAST fully supported microsatellite data. Although BEAST 2 (v2.4.5) should allow microsatellite data analysis as well, the respective package BEASTvntr was found not to be fully functional. BEAST was written in Java and is applicable to both within- and between-species data.
- IMa ('Isolation with Migration – analytic') draws upon the Isolation with Migration model by Nielsen and Wakeley [23], later implemented in the IMa software in C++ by Hey and Nielsen [24]. IMa is especially suited to the analysis of data from closely related species, and its successor IMa2 can even handle multiple populations. There is also a separate fork with parallelization through OpenMP, called IMa2p [25]. However, since IMa2p involves multiple Markov chains, it was not considered here. A successor version of IMa2 was released recently (IMa3 [26]) but could not be considered because our study started considerably earlier.
- LAMARC is written in C++ and focuses upon migration between populations [21]. It combines the functionalities of the older tools Coalesce, Fluctuate and Recombine, which were merged into LAMARC. LAMARC also has a fork, MIGRATE-N, with parallel computation capabilities. Since MIGRATE-N is only applicable to multiple populations, however, it was not evaluated in our study.

2.2. Simulations

Data for use in our comparative software evaluation were simulated with SAMPLE, a tool that is distributed alongside BATWING. SAMPLE reads model specifications and parameter settings in the same format as BATWING and uses a coalescent model as well. Three different population growth models are available, namely (i) a constant population size model, (ii) a purely exponential model, and (iii) a combined model with an initial period of constant Y-effective population size followed by a period of exponential growth for a specified number of generations. SAMPLE uses a symmetric one-step mutation model with a scaled mutation rate defined as $\theta = 2N_e\mu$. Here, μ denotes the mutation rate per microsatellite locus per generation and N_e is the Y-

effective population size. Population substructure can be allowed for in SAMPLE as well, although at a rather simple level. The output of the tool consists of genetic profiles, stored in one file, and the coalescent tree and some additional information about model parameters in a second file. Only the simulated profiles were used in the present study.

Our software evaluation was carried out using 100 datasets of size 100, 500 and 1000 each. Data were simulated for a single population under either a constant or a purely exponential growth model. Due to limitations of IMa2 in terms of the maximum line size allowed in its input files, the number of microsatellite loci had to be limited to eight. All eight loci were treated as completely linked (i.e. no recombination was allowed for during the coalescent process), and random seeds were used. With a constant population size model, only parameter θ has to be specified in the input file of SAMPLE. We chose $\theta = 60$, which corresponds to a Y-effective population size N_e of 10,000 [27,28] and a mutation rate μ of 0.0030 per generation per microsatellite locus (typical value in the Y Chromosome Haplotype Reference Database [29], Release 59). For the exponential growth model, we chose a mutation rate of $\mu = 0.0030$, a growth rate of $\alpha = 0.0050$ per generation and a final Y-effective population size of $N_e = 10{,}000$ [28,30]. These parameters correspond to realistic scenarios and still allowed the analyses to be carried out in reasonable time. For an additional feasibility analysis considering 20 linked loci at a time, we simulated 100 datasets of size 100, adopting the constant population size model.

2.3. Analysis of Simulated Data

The simulated data were analyzed with BATWING v1.0, BEAST v1.8.4, IMa2 v8.27.12 and LAMARC v2.1.10, using two different sets of prior distributions for the evolutionary parameters θ (scaled mutation rate), N_e (Y-effective population size), μ (mutation rate) and α (growth rate). The sets of priors comprised different distribution types and different variance values (for details, see below), and the set with large variance values will henceforth be referred to as 'less informative' whereas the set with small variance values will be termed 'more informative'. BEAST and BATWING offer a variety of prior distribution types whilst IMa2 and LAMARC only support uniform distributions. Thus, we chose a log-normal distribution for the more informative set and a gamma distribution for the less informative set for BEAST and BATWING. The log-normal distribution was specified by its mean $m = \exp(\mu + (\sigma^2/2))$ and standard deviation $s = (\exp(\sigma^2)-1)\exp(2\mu + \sigma^2)$, where μ and σ denote the mean and standard deviation of the corresponding (non-logarithmic) normal distribution. The gamma distribution was specified either by its shape and scale parameter k_g and θ_g, respectively (BEAST), or by its shape and rate parameter α_g and β_g, respectively (BATWING). We set $\alpha_g = 1$. Note that $m = k_g\theta_g$ and $\theta_g = \alpha_g/\beta_g$ for a gamma distribution so that setting $k_g = \alpha_g = 1$ implies that the remaining parameters θ_g and β_g result directly from a given value of m. For IMa2 and LAMARC, which only support uniform distributions, the 2.5% and 97.5% quantiles of the respective prior distributions of BATWING and BEAST were used as boundaries for the uniform distributions. Priors were centered on the correct value, i.e. m was equated to the value used in the simulations whereas s was chosen so as to ensure sufficient difference in information content between the two sets of prior distributions. For further details on the prior distributions used in our study, see Supplementary Table S1. To investigate the robustness of the estimates against misspecification of the prior distribution, we performed additional analyses setting the prior means to 1/2 or 1/10 of the correct values.

A symmetric single-step mutation model was used in all analyses. Scaled mutation rate $\theta = kN_e\mu$ (not to be confused with scale parameter θ_g of the gamma distribution) is the central parameter characterizing the genetic heterogeneity at the leaves of the coalescent tree. Unfortunately, the four tools studied use different values of k in their definition of θ for Y-chromosomal markers, and these settings had to be adopted because they cannot be altered by the user (see Supplementary Table S1). Prior distributions were placed on θ, N_e or μ, or combinations thereof. The exponential growth model additionally requires specification of a prior distribution for growth rate α. With BEAST, IMa2 and LAMARC, the analysis had to be restricted to completely linked loci whereas BATWING is designed for haplotypes anyway. We used the default 'classic operator mix' for BEAST which models transition of the MCMC invoking the four procedures 'subtreeSlide', 'narrowExchange', 'wideExchange' and 'wilsonBalding'. IMa2 uses an inheritance scalar $c = 0.25$ for Y-chromosomal data. Note that all analyses were carried out adopting the correct population growth model as employed in the simulations of the respective datasets.

The total length, l_{MCMC}, of the Markov chain was chosen equal to 22,000,000 and the output number was set equal to 100,000. In BATWING, this was achieved by setting Nbetsamp = 10 and treebetN = 22. Here, Nbetsamp is the number of attempts by the Markov chain, between output, to change the model parameters whereas treebetN is the number of attempted changes of the tree architecture between attempted changes of the model parameters. This implies l_{MCMC}=Nbetsamp*treebetN*(number of output). For the other three tools, the interval between MCMC output was chosen equal to 220 steps accordingly. In LAMARC, the number of attempted changes of model parameters and of tree architecture was set similar to BATWING. Running LAMARC turned out to be time-consuming. Therefore, l_{MCMC} had to be reduced to 220,000 for this tool and, hence, the output number to 1000. A burn-in of 10% initiated all chain runs to reduce the impact of the start values which were chosen equal to the correct values.

The four tools evaluated in our study are open source and could therefore be optimized, for example, in terms of their runtime behavior. The default configurations were modified so as to achieve maximum performance, but with 'correct' behavior. Additional compiler flags were set to fit the executables to the hardware used. Optimization flags were left unchanged when pre-assigned by the respective tool. BATWING was compiled with the following settings: 64 bits, high level optimization, aggressive optimization of arithmetic calculations, link time optimization, loop unrolling and native architecture optimization. BEAST was started with an initial memory allocation of 4 GB and a maximum memory allocation of 16 GB, using BEAGLE library v3.1.2 [31]. IMa2 was compiled with 64 bits, moderate level optimization (pre-assigned by the software), link time optimization, loop unrolling and native architecture optimization. The configuration was rebuilt for the native architecture with GNU Autoconf v2.69 and automake v1.15.1. IMa2 has a lower threshold of three for the number of microsatellite repeats. To avoid falling below this limit during the mutation process, the simulated repeat numbers were increased by 100. For LAMARC, input-output (IO) of special-case divergence code was disabled because this IO was unnecessary and would have greatly increased runtime. For further performance improvement, LAMARC was compiled with 64 bits, high level optimization (pre-assigned by the software), link time optimization, loop unrolling (pre-assigned by the software) and native architecture optimization. Like with IMa2, the configuration was rebuilt for the native architecture with GNU Autoconf v2.69 and automake v1.15.1. Except for BEAST, all tools were compiled with gcc v7.3.0 for a 16-cores (2x Intel(R) Xeon(R) E5-2620 v4/2.1 GHz) system with 64 GB DRAM and Ubuntu 18.04.1 LTS (Linux), which was also used for analysis. BEAST was run with Java RE 1.6.

All analyses were carried out using custom batch scripts in a round robin distribution. The number of threads was set to automatic in the parallel version of BEAST. Output was processed, summarized and analyzed with custom scripts using statistics software R v3.5.1 [32] and exported with R package xlsx [33]. Except for IMa2, which provides graphical and other analysis options itself, output was evaluated by Tracer [34] to test parameter settings and visualize convergence behavior. BATWING output had to be converted into Tracer readable format first, using C programs.

To analyze the impact of l_{MCMC} on parameter estimation accuracy, we also re-ran each tool on 10 randomly chosen datasets, but with a 10-fold increased l_{MCMC} value (i.e. 220 million instead of 22 million), adopting a

constant population size and less informative priors. Owing to the poor runtime behavior of LAMARC, we had to limit l_{MCMC} by 4.4 million for this tool (instead of 220,000 as in the former evaluation of LAMARC). For each dataset and each analysis, parameter estimates were derived at different points of the chain, namely after 110,000, 220,000, 880,000, 2.2 million, 4.4 million, 8.8 million, 22 million, 44 million, 88 million, 110 million, 154 million and 220 million iterations. Since no intermediate parameter values were provided by IMa2 during chain runs, multiple runs of the Markov chain had to be carried out for this tool.

3. Results

3.1. Population Genetic Models and Other Functionalities

The four tools evaluated in our study support different population genetics models and provide different data analysis options (Table 1). BEAST offers the greatest variety in terms of evolutionary models, especially with respect to population growth and mutation. The standard constant population size model is implemented in all tools and, with the exception of IMa2, the tools also support exponential population growth. Some additional growth models are implemented in BEAST. The standard single-step mutation model is available in all tools. BATWING provides only one additional mutation model, namely the K-allele model, whereas LAMARC and IMa2 offer a multitude of alternative mutation models. BEAST even allows customization of the mutation model by XML specification. The tools vary markedly regarding the handling of multiple populations and migration. IMa2 and LAMARC support the most flexible migration models through the consideration of unconstrained migration parameters. BEAST and BATWING, in contrast, allow no migration after splitting events. Only basic evolutionary models are implemented in BATWING, which is also the only tool limited to haplotypes and incapable of handling DNA sequence data. Some tools have specialized evolutionary models incorporated, such as molecular clocks (BEAST) or meiotic recombination (LAMARC).

All tools use a Bayesian MCMC method, and LAMARC additionally includes a maximum likelihood approach based upon importance sampling. LAMARC and IMa2 also implemented more advanced MCMC models by including heating (also called 'Metropolis-coupled' MCMC). BEAST allows modification of the transition probabilities of the Markov chain by the use of so-called 'MCMC operators'. IMa2 and LAMARC support only a uniform prior distribution for the evolutionary parameters, which may be too simplistic a choice if external information on parameters is available. BATWING and BEAST, in turn, are more flexible in terms of prior distributions. The standard, constant population size coalescent model has only one parameter, namely the scaled mutation rate $\theta = kN_e\mu$. Here, μ denotes the mutation rate per microsatellite locus per generation and N_e is the Y-effective population size. LAMARC is the only tool for which θ is the sole parameter and estimation is also restricted to this parameter. The other three tools allow input of two of the three parameters θ, N_e and μ, and the third parameter has to be calculated from the other two, if required. Only θ is directly estimated from the data. For the estimation of N_e and μ, the prior distributions are taken into account.

3.2. Technical Characteristics

All tools are open source under GNU Public License or compatible, but are implemented using different programming languages. Thus BEAST was written in Java whereas the other tools were written in C or C++. Further technical characteristics include (i) the support of parallel computation, (ii) the user interface and (iii) the availability of special options such as batch mode or GPU usage (Table 2).

- Only BEAST supports parallel mode. A fork and a successor with parallel mode are available for IMa2, and a fork is available for LAMARC, but these versions require either multiple populations or multiple chains.
- GPU support was announced possible for BEAST alone, but such support was not available for microsatellite data in the BEAST version tested.
- Batch scripts are explicitly incorporated by LAMARC only whereas, for the other tools, they may be used but have to be adapted manually.
- As regards user interface, BATWING and IMa2 are command-line only while BEAST is operated via a graphical user interface (GUI) but also offers a command-line mode. Input files are most easily created in the GUI using the BEAUTI software, which is part of BEAST. Additional technical parameters, such as maximum heap size, can be customized in the executable scripts of BEAST. LAMARC offers a GUI via compiler option but can also be used in command-line mode and through a text-based user interface (TUI).
- The quality of the documentation of the tools varies considerably. The best documentation regarding the theoretical basis of the software, the definition of input parameters and the interpretation of output is provided by the BATWING manual, which is available online. The most important documentation for BEAST is available on its website and its well-supported online user forum. Moreover, BEAST comes with a manual and a book, even though the latter is mainly on BEAST 2. Despite the comprehensive documentation, however, it was difficult

Table 1
Functionalities of investigated tools.

Functionality	Tool			
	BATWING	BEAST	IMa2	LAMARC
Data type	STR, SNP, K-Allele	DNA, STR, SNP	DNA, STR, SNP	DNA, STR, SNP, K-Allele
Linkage	Completely linked (haplotypes)	Mix of linked and unlinked markers	Mix of linked and unlinked markers	Mix of linked and unlinked markers
Population size model	Constant, exponential	Coalescent: constant, expansion, exponential, logistic; other: Yule, birth and death	Constant	Constant, exponential
Mutation model	Stepwise, K-allele	Various standard substitution models, custom XML specification	Infinite sites, HKY, stepwise, compound locus	Stepwise, K-allele, F84, GTR, brownian motion
Migration model	No migration after splitting events	No migration after splitting events	Gene flow: unconstrained	Gen flow: constant, symmetric, unconstrained
Inference framework	MCMC	MCMC	MCMC	MCMC, MLE
Prior distributions	Constant, uniform, normal, lognormal, gamma	Constant, CTMC, (infinite) uniform, normal, lognormal, (inverse) gamma, exponential, 1/x	Uniform	Uniform, original or logarithmic values
Output parameters	μ, N_e, α, β, TMRCA, likelihoods	N_e, α, μ, TMRCA, branch rates, likelihoods	θ, μ, migration rates	θ, α, migration rates, recombination rates, likelihoods
Special features		Molecular clock models, MCMC operators	Heating	Recombination models, multiple chains, heating

STR, short tandem repeat or microsatellite; SNP, single-nucleotide polymorphism; XML, extensible markup language; HKY, Hasegawa-Kishino-Yano; F84, Felsenstein 84 nucleotide model; GTR, general time-reversible model; MCMC, Markov chain Monte Carlo; MLE, maximum likelihood estimation; CTMC, continuous time Markov chain; TMRCA, time to the most recent common ancestor.

Table 2
Technical details of investigated tools.

Specific	Tool			
	BATWING	BEAST	IMa2	LAMARC
Evaluated version	V1.0	V1.8.4	V8.27.12	V2.1.10
End of development	Yes	No	No	No
Fork	R package rforensicbatwing	BEAST 2	IMa2p	MIGRATE-N
Open version control system	No	Git	No	No
GUI	No	Yes	No	(Yes)
TUI	No	No	No	Yes
Batch mode	No	No	No	Yes
Parallel mode	No	Yes	No	No
GPU	No	(Yes)	No	No
Programming language	C	Java	C++	C++
Compiled binary for OS	Win, Unix, Mac	Java 1.6	Win, Unix, Mac	Win, Unix, Mac
Source code available	Yes	Yes	Yes	Yes
Documentation	Manual	Manual, website, tutorial, forum, (book), workshops	Manual, how-to	Website, tutorials
License	GPLv2	LGPL	GPLv2	Apache license, version 2.0
Special features	Simulation program SAMPLE	Supports Tracer	Data summary and visualization, live analysis updates, AC	Supports Tracer, run report in XML, input data converter

GUI, graphical user interface; TUI, text-based user interface; GPU, graphics processing unit; OS, operating system; Win, Microsoft Windows; Mac, Macintosh; GPLv2, GNU General Public License Version 2; LGPL, Lesser GNU Public License.2; AC, autocorrelation; XML, extensible markup language.

for us to find the correct settings for BEAST, perhaps partly due to the complexity of the implemented population genetic models. Even dedicated workshops are being organized for potential BEAST users. The documentation of IMa2 comprises both an extensive manual and a short introduction that covers general principles of, and assumptions underlying, the tool. LAMARC provides HTML documentation with tutorials on its website, which can also be downloaded.

BEAST and LAMARC yield output that can be uploaded to the Tracer tool for data visualization and convergence diagnostics. IMa2 provides direct options for data summary and visualization, live analysis updates and the calculation of autocorrelation as part of the tool. A special feature of BEAST is its recommended supplementation with the external BEAGLE library for better performance. This library is independent of BEAST and has to be installed separately. For version management, BEAST uses a GitHub repository and thereby offers simple access to all versions and forks as well as support of all other Git features.

All tools evaluated except BEAST are restricted in terms of their parameterization. BATWING can only handle 255 input parameters which limits, for example, the number of loci. IMa2 has a range of hard-coded limits for several parameters. Limits are in effect for the number of loci (≤400), the number of chains (≤1000), the number of genes (≤1000), the number of populations (≤10), the number of linked loci (≤15) and the number of characters per line (≤300). The latter restriction limits the number of linked loci even further because these loci are required to be entered in the same line. Note that unlinked loci are specified in different lines. LAMARC has hard-coded limits for starting values e.g. for θ (≤100), for the scaled migration rate M (≤10,000, where $M = m/\mu$ and m denotes the probability for a lineage to immigrate in a given generation) and for the scaled growth rate A (≤15,000, where $A = \alpha/\mu$).

3.3. Accuracy and Convergence

To compare the parameter estimation accuracy of the four tools, we simulated 100 datasets of eight loci of size 100, 500 and 1000 each, assuming either a constant or an exponentially growing population size. Under a constant population size model, a sample size of 100 yielded estimates very close to the parameter values used in the simulations, i.e. $\theta = 60$ and $\mu = 0.0030$, with all tools (Table 3). Increasing the sample size to

Table 3
Parameter estimation from simulated data (8 loci, constant population size).

Sample size	Prior	Parameter	BATWING	BEAST	IMa2	LAMARC
100	More inf	θ	60 (45–75)	62 (45–77)	61 (45–77)	60 (47–75)
		μ	0.0031 (0.0025–0.0035)	0.0031 (0.0025–0.0036)	0.0031 (0.0031–0.0032)	0.0030 (0.0024–0.0038)
	Less inf	θ	61 (44–77)	62 (44–80)	61 (45–78)	61 (45–76)
		μ	0.0037 (0.0033–0.0041)	0.0036 (0.0032–0.0041)	0.0031 (0.0031–0.0032)	0.0031 (0.0023–0.0038)
500	More inf	θ	60 (53–69)	n.c.	60 (53–69)	n.c.
		μ	0.0031 (0.0028–0.0033)	n.c.	0.0030 (0.0030–0.0031)	n.c.
	Less inf	θ	60 (53–69)	n.c.	60 (54–70)	n.c.
		μ	0.0037 (0.0031–0.0043)	n.c.	0.0030 (0.0030–0.0031)	n.c.
1000	More inf	θ	60 (55–65)	n.c.	62 (57–68)	n.c.
		μ	0.0030 (0.0029–0.0032)	n.c.	0.0030 (0.0030–0.0030)	n.c.
	Less inf	θ	60 (54–65)	n.c.	62 (57–68)	n.c.
		μ	0.0037 (0.0031–0.0043)	n.c.	0.0030 (0.0030–0.0030)	n.c.

θ, scaled mutation rate; μ, mutation rate per generation and microsatellite locus; more inf, set of more informative priors; less inf, set of less informative priors, n.c., no convergence achieved when stopping the MCMC. For each of the simulated 100 datasets per combination of sample size and prior distribution set, the means of the posterior distributions of θ and μ were obtained. Given are the mean and 0.025 and 0.975 quantiles (in brackets) of these 100 means. The values used for simulation were $\theta = 60$ and $\mu = 0.0030$. Note that the MCMC chain length was reduced by a factor of 100 for LAMARC due to poor runtime behavior. For BATWING and BEAST, the original output was Y-effective population size N_e and μ; θ was calculated as $\theta = 2N_e\mu$. IMa2 outputs θ and μ directly whilst for LAMARC only output θ is provided, so that μ was calculated from the correct value $N_e = 10{,}000$.

Table 4
Parameter estimation from simulated data (8 loci, exponential population growth).

Sample size	Prior	Parameter	BATWING	BEAST	LAMARC
100	More inf	θ	60 (45–78)	57 (43–74)	67 (50–81)
		μ	0.0030 (0.0026–0.0034)	0.0029 (0.0026–0.0033)	0.0034 (0.0025–0.0040)
		α	0.0050 (0.0037–0.0065)	0.0050 (0.0038–0.0067)	0.0055 (0.0037–0.0076)
	Less inf	θ	63 (41–97)	59 (38–88)	69 (49–88)
		μ	0.0034 (0.0029–0.0038)	0.0035 (0.0031–0.0040)	0.0034 (0.0025–0.0044)
		α	0.0056 (0.0039–0.0080)	0.0062 (0.0040–0.0090)	0.0056 (0.0039–0.0082)
500	More inf	θ	60 (52–70)	n.c.	n.c.
		μ	0.0030 (0.0028–0.0032)	n.c.	n.c.
		α	0.0050 (0.0038–0.0066)	n.c.	n.c.
	Less inf	θ	60 (51–72)	n.c.	n.c.
		μ	0.0033 (0.0028–0.0038)	n.c.	n.c.
		α	0.0056 (0.0041–0.0072)	n.c.	n.c.
1000	More inf	θ	60 (53–66)	n.c.	n.c.
		μ	0.0030 (0.0028–0.0032)	n.c.	n.c.
		α	0.0050 (0.0039–0.0059)	n.c.	n.c.
	Less inf	θ	60 (52–68)	n.c.	n.c.
		μ	0.0033 (0.0029–0.0038)	n.c.	n.c.
		α	0.0056 (0.0042–0.0069)	n.c.	n.c.

α, growth rate per generation; for further details, see legend to Table 3.

500, accurate estimates were still obtained with BATWING and IMa2, but LAMARC and especially BEAST did not converge. A similar behavior was observed for a sample size of 1000. Again, BATWING and IMa2 gave surprisingly good estimates whereas LAMARC and BEAST were far from converging. In theory, accuracy and precision should be higher for more informative priors but no big difference was noted between the two types of distribution in our simulations (Table 3). IMa2 could not be studied under an exponential growth model because such a model was not supported by the software. For the remaining tools, the results were found to be similar to those obtained with a constant population size model (Table 4). To investigate the tool performance for larger numbers of loci, we simulated and analyzed datasets of size 100 for 20 loci. IMa2 was excluded since it could not handle 20 linked loci. The other three tools yielded similar results to those obtained for the eight loci (Table 5).

The above analyses were carried out with chain lengths of 22 million for BATWING, BEAST and IMa2. For LAMARC, the chain lengths had to be reduced to 220,000 because of its poor runtime behavior. To investigate the convergence of the MCMC implementations more generally, we randomly selected 10 datasets and re-analysed them under a constant population size model, but with a 10-fold increased chain length of 220 million (4.4 million for LAMARC). The chains were then evaluated at different iteration steps of the Markov chain. With a dataset size of 1000 (see Supplementary Tables S2 and S3 for smaller sample sizes), where convergence was an issue, BATWING was found to converge fastest (Fig. 2) and the correct value $\theta = 60$ was reached at a chain length of ~8.8 million (for details see Supplementary Table S4). IMa2 reached $\hat{\theta} = 62$ after ~22 million steps. For LAMARC, $\hat{\theta} = 78$ at the chain length limit of 4.4 million, a value notably higher than the former estimates but possibly acceptable for certain practical applications. Finally, BEAST showed very slow convergence but reached $\hat{\theta} = 61$ after 154 million steps.

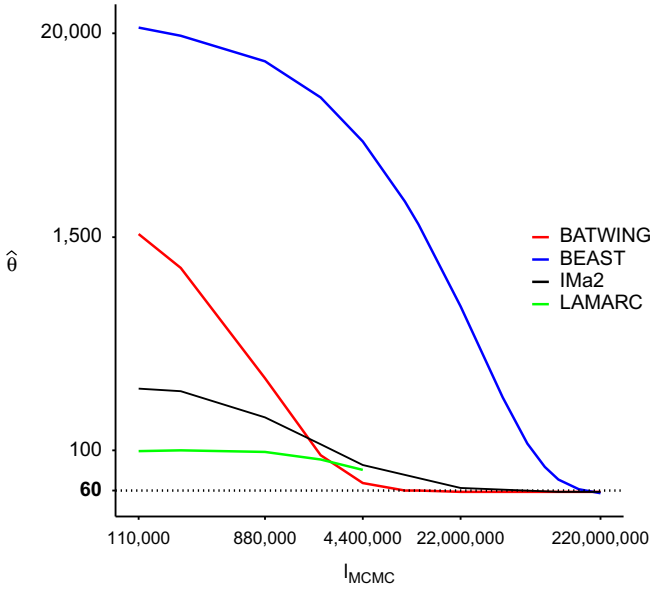

Fig. 2. Convergence behavior for $\hat{\theta}$. $\hat{\theta}$, estimated scaled mutation rate; l_{MCMC}, length of the evaluated Markov chain. Ten randomly chosen datasets with a constant population size of 1000 were analyzed for a chain length of 220 million and the set of less informative priors. For each analysis, parameter estimates of θ were derived after a different number of iterations, l_{MCMC}, of the Markov chain, e.g. after l_{MCMC} = 110,000, 220,000, 2.2 million or 220 million iterations of the Markov chain. Since no intermediate parameter values were provided by IMa2 during chain runs, multiple runs of the Markov chain had to be carried out for this tool. For LAMARC the maximal chain length was limited by 4.4 million due to poor runtime behavior. For each of the 10 datasets, the means of the posterior distribution of θ were obtained after different numbers l_{MCMC} of iterations. Shown are the means of these 10 means for each tool. The correct value, i.e. the value used for simulation, was θ = 60 (dashed horizontal line). For BATWING and BEAST, $\hat{\theta}$ was calculated from $\theta = 2N_e\mu$ whereas, for IMa2 and LAMARC, $\hat{\theta}$ was obtained directly. Note that both axes have logarithmic scale. For exact values of the means and maximal and minimal values of $\hat{\theta}$ and for estimates for N_e and μ, see Supplementary Table S4.

3.4. Estimates for Incorrect Priors

In the preceding analyses, the prior distributions used by the tools were centered on the correct parameter values. To investigate the

Table 5
Parameter estimation from simulated data (20 loci, constant population size).

Sample size	Prior	Parameter	BATWING	BEAST	LAMARC
100	Less inf	θ	61 (50–74)	63 (51–75)	61 (50–73)
		μ	0.0037 (0.0032–0.0043)	0.0036 (0.0032–0.0040)	0.0031 (0.0025–0.0037)

For details, see legend to Table 3.

Table 6
Parameter estimation from simulated data, using incorrect priors (8 loci, constant population size).

Parameter (correct value)	MF		BATWING	BEAST	IMa2	LAMARC
θ (60)	1/2	Prior	–	–	U(0,111)	U(0,111)
		Estimate	60 (44–76)	62 (44–79)	61 (45–79)	60 (45–75)
	1/10	Prior	–	–	U(0,22)	U(0,22)
		Estimate	59 (43–74)	60 (43–76)	22 (22—22)	22 (22–22)
μ (0.003)	1/2	Prior	Γ(1,0.003)	Γ(1,0.003)	–	–
		Estimate	0.0049 (0.0043–0.0054)	0.0049 (0.0042–0.0055)	–	–
	1/10	Prior	Γ(1,0.003)	Γ(1,0.003)	–	–
		Estimate	0.0101 (0.0087–0.0112)	0.0101 (0.0086–0.0113)	–	–
N_e (10,000)	1/2	Prior	Γ(1,5000)	Γ(1,5000)	–	–
		Estimate	8213 (7283–9261)	8414 (7281–9312)	–	–
	1/10	Prior	Γ(1,1000)	Γ(1,1000)	–	–
		Estimate	3356 (2923–3733)	3409 (2955–3801)	–	–

θ, scaled mutation rate; μ, mutation rate per generation per microsatellite locus; N_e, Y-effective population size; correct value: parameter value used in the underlying simulations; Γ(a,b), gamma distribution with shape a and scale b (the mean is for a = 1 equal to the scale); U(a,b), uniform distribution with boundaries a and b; MF, misspecification factor. Simulations were performed for a sample size of 100. For each of the simulated 100 datasets per prior distribution, the means of the posterior distributions of θ, μ and N_e were obtained. Given are the mean and 0.025 and 0.975 quantiles (in brackets) of these 100 means. MF values of 1/2 and 1/10 correspond to means of 5000 and 1000, respectively, of the priors for N_e (instead of the correct value of 10,000) and to means of 30 and 6, respectively, of the priors for θ (instead of the correct value of 60). Note that the MCMC chain length was reduced by a factor of 100 for LAMARC due to poor runtime behavior. For BATWING and BEAST, the original output comprised N_e and μ; θ was calculated as $\theta = 2N_e\mu$. IMa2 and LAMARC output θ directly.

possible effects of a misspecification, we performed additional analyses for a sample size of 100 using 'incorrect' prior distributions. The mean of the priors for θ was reduced to 30 and 6, i.e. 1/2 and 1/10 of the correct value of 60. Since BEAST and BATWING do not allow the priors for θ to be specified by the user, we accordingly set the mean of the priors for the Y-effective population size N_e to 5000 and 1000 (the correct value being 10,000). If and when available, the priors for μ were left unchanged. For moderate misspecifications of the prior means of θ and N_e by a factor of 1/2, the estimates of θ were still found to approximate the true value of 60 very well for all four tools (Table 6). For BEAST and BATWING, where N_e is included in the output as well, the reduced prior means for N_e yielded lower estimates of this parameter. However, this reduction was compensated by higher estimates of μ, which led to fairly accurate estimates of θ. A similar effect became apparent for BEAST and BATWING when more drastic misspecification of the priors by a factor of 1/10 was assumed (Table 6). Since IMa2 and LAMARC only allow for uniform prior distributions, the support of these prior distributions did not contain the correct parameter value under the drastic misspecification scenario. Consequently, the two tools yielded estimates coinciding with the upper boundary of the uniform distribution in this case.

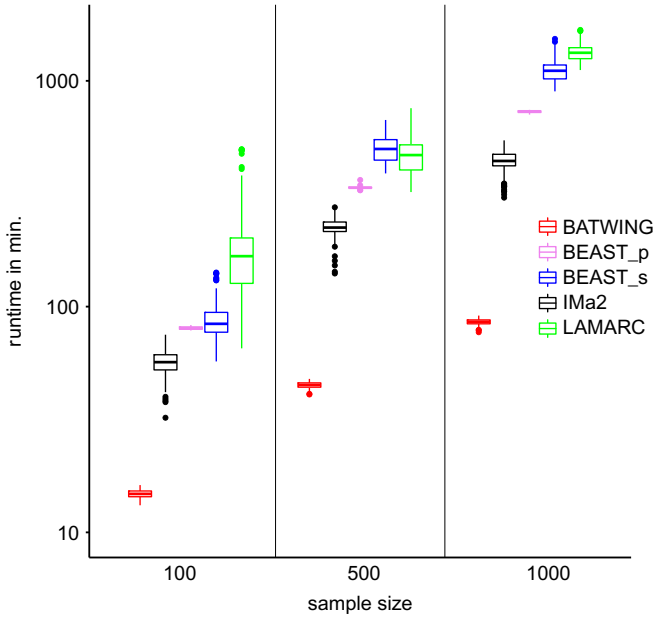

Fig. 3. Runtime under a constant population size model. BEAST_s, sequential version of BEAST; BEAST_p, parallel version of BEAST. The analyses were performed on a 16-core computer (2x Xeon(R) E5-2620 v4/2.1 GHz) with 64 GB DRAM running Linux OS. For LAMARC, the chain length was reduced by a factor of 100 due to poor runtime behavior. Runtimes obtained for the less and the more informative sets of prior distributions were merged in the figure because they were barely distinguishable. Box plots are based upon the results obtained from 200 data sets, except for BEAST_p where only 10 datasets underlie each box plot. Note that the Y-axis has a logarithmic scale.

3.5. Runtime

The runtime varied significantly between tools and sample sizes (Fig. 3, Supplementary Fig. S1, Supplementary Table S5). BATWING was fastest in all instances, with runtimes between 15 min (sample size n = 100) to 1.5 h (n = 1000) for eight loci and a constant population size model. IMa2 performed second best, with runtimes between one hour (n = 100) to 7.5 h (n = 1000). BEAST had a comparatively long runtime between 1.5 h (n = 100) and 19 h (n = 1000). The parallel version of BEAST achieved no noticeable runtime reduction for n = 100, but did so for n = 1000 (12 h). With runtimes between 3 h (n = 100) and 22.5 h (n = 1000), LAMARC was slowest even though the chain length had already been reduced by a factor of 100. BATWING was slightly slower with an exponentially growing population size than with a constant size (Supplementary Table S5). In contrast, BEAST and LAMARC were faster under the exponential population model, although still slower than BATWING. For 20 loci, the runtimes of BATWING and BEAST were a little longer than for eight loci, while the runtime for LAMARC was roughly doubled.

4. Discussion

In the present study, we compared four software tools for the coalescent-based analysis of genetic data using MCMC methods, namely BATWING, BEAST, LAMARC and IMa2. The tools were found to differ markedly in terms of their functionality and runtime behavior which implies that none of them would qualify as a uniquely optimal choice for practical application. Whether a given tool is useful or not in a given scientific project depends critically upon the actual requirements of the project.

As one of the pilots of MCMC-based evolutionary data analysis, BATWING not surprisingly provides only limited functionality and has reached a stage of 'final version', without further development. Unfortunately, these realities are somewhat typical and reflect a recurrent phenomenon in academia-driven software production. Scientists who developed a piece of software often turn towards new challenges once a project is finished, leaving their tool behind without sustainable support. On the other hand, the simplicity of BATWING also meant that selecting the parameters for data analysis was easiest for this tool. BATWING also reached estimates of given accuracy most rapidly and

clearly outperformed the other three tools with respect to runtime behavior. Consequently, BATWING rightly seems the best choice under the proviso that the evolutionary model of interest is implemented in the tool.

One of the advantages of BEAST is its ubiquitous applicability on a variety of platforms and the possibility for users to adapt the tool to their specific needs by adding plugins. BEAST is under constant development, with frequent changes, additions or removals of functionalities, which may render keeping track of the modelling specifications difficult for users. Nevertheless, means of interaction with users is provided by an online user forum, and all versions can be accessed through Git. High computing performance is achieved mainly by the use of external library BEAGLE, which may not be required for all scenarios but which provides features such as GPU support. Note that related version BEAST 2 has an even stronger focus on plugins. Another benefit of BEAST is its great flexibility of population genetics modelling which clearly exceeded that of the other three tools, even although migration cannot yet be taken into account. The cross-platform focus of BEAST also has one important disadvantage, namely that the programming language Java chosen for BEAST runs in a virtual machine, which may partially explain why BEAST is significantly slower than BATWING and IMa2. In fact, our simulations revealed that BEAST has runtimes that are one order of magnitude longer than those of BATWING or IMa2 to achieve similar accuracy. Even parallel mode could not significantly alleviate this problem. This may be a serious drawback in practice, especially when large numbers of samples are involved, and careful review of the output is necessary to ensure that the algorithm has converged.

IMa2 includes all functionalities necessary for a coalescent-based analysis of microsatellite data, from parameterization via MCMC to additional features such as autocorrelation analysis or summaries and plots of the results. The runtime behavior of IMa2 is only surpassed by that of BATWING, and the tool showed good convergence. The possibility to model migration is another asset of IMa2 whereas its limitation to a constant population size model and a uniform prior distribution represent disadvantages. Recently, a new version, IMa3, has become available that also offers a parallel mode similar to IMa2p [26], the parallelized version of IMa2.

With its focus on migration models, LAMARC resembles IMa2 and provides a large variety of models to such effect. The tool also has the same disadvantage as IMa2 in that it only supports uniform prior distributions. The most severe problem with LAMARC, however, is its poor runtime behavior, which was the worst of all four tools. In this regard, LAMARC appears almost impracticable for applications involving large datasets. Owing to its poor performance, we even had to reduce the chain lengths of LAMARC to 220,000 (instead of 22 million) in our analyses. Note that LAMARC was written in a C dialect, like BATWING and IMa2, and does not use a virtual machine so that its performance is not explicable by programming language. This notwithstanding, the convergence behavior of LAMARC was still found to be slightly better than that of BEAST.

Our simulations revealed that practical application to large samples may be problematic for some of the software. While a sample size of 100 still allowed all four tools to yield fairly accurate results in reasonable time, this turned out to become increasingly difficult for BEAST and LAMARC when sample sizes increased to 500 and 1000. Undoubtedly, this phenomenon is a direct consequence of the MCMC approach common to all tools because more samples mean more and larger possible coalescent trees and, consequently, a less comprehensive search of the corresponding state space at given chain length. Another important problem posed by large sample size is the slowdown of convergence. Therefore, it would be important in practice to ensure that the MCMC analysis has actually converged for the dataset under study. Multiple diagnostic instruments are available for this purpose. The easiest one would be to perform several runs with different seeds and different chain lengths, followed by an assessment of whether a consistent (asymptotic) estimate has been reached. In our study, we observed that parameter θ, in particular, was liable to overestimation with non-convergent chains. Other suitable diagnostic procedures include trace plots, autocorrelation and the Gelman-Rubin and Geweke diagnostics [35,36]. Many of these approaches are implemented in the Tracer software, which is supported by BEAST and LAMARC.

In summary, current runtimes for chain lengths necessary to yield sufficient accuracy for large sample sizes might be prohibitive, at least for LAMARC and BEAST. For better comparability between the implemented MCMC approaches, accuracy and runtime behavior were evaluated using single chains only. It is obvious, however, that single chains require long runtimes to search a tree space comprehensively, leading to impractical performance with large datasets. One possible solution to this problem would be the implementation of multiple chains. A more advanced version of this approach is Metropolis-coupled MCMC where several Markov chains are run in parallel, but with different heating parameters. The overall chain comprises two steps per iteration. First, all individual chains are updated, using a typical proposal distribution. Then, state spaces are swapped between chains according to a second proposal distribution. Metropolis-coupled MCMC is implemented in LAMARC and IMa2 whilst BEAST offers so-called 'MCMC operators' that allow modification of the proposal distribution of the Markov chain. Evaluation of advanced MCMC methods as implemented in one or the other of the four tools was outside the scope of this study, but would be worthwhile future research. Suchlike studies would reveal to what extent optimization of MCMC operators or the parameters of Metropolis-coupled MCMC alleviates the convergence and runtime problems of LAMARC and BEAST.

The accuracy and convergence behavior of individual tools were assessed in our study in simulated datasets that were generated with SAMPLE, software distributed alongside BATWING and employed here merely for efficiency reasons. This notwithstanding, the use of SAMPLE might raise concerns whether our results were biased towards BATWING. However, coalescent simulation under constant or exponentially growing population models is straightforward and its outcome should not depend upon the software used to create the data. Moreover, the MCMC analysis implemented in BATWING employs an approach different from the simulation with SAMPLE so that said bias is indeed quite unlikely.

Most of our analyses employed prior distributions that were centered on the correct values. Since, in reality, the true value of a parameter is typically unknown, we also performed simulations with misspecified prior means. These additional analyses still resulted in sufficiently accurate estimates of θ for all four tools, which indicates that the likelihoods dominated the incorrect priors and thus ensured adequate posterior distributions. However, it must be remembered that our study employed only two simple demographic models (i.e. constant and exponentially growing population size). Hence, it would be an interesting subject of future comparative studies of the four tools to assess the impact of misspecification under more complex models involving, for example, population bottlenecks.

We simulated Y-chromosomal microsatellite data and, hence, our results regarding accuracy and runtime are confined to this type of genetic markers. Clearly, an extension of the tool comparison to autosomal markers, DNA sequences and single-nucleotide polymorphisms (SNPs) would be warranted. Another possible augmentation would be the replacement of the single-step symmetric mutation model, which is known to be an oversimplification, by more realistic models that have been proposed in the past [37–39].

The scaled mutation rate θ is the critical parameter of a coalescent with mutations. It is defined as $\theta = kN_e\mu$, where usually $k = 4$ or $k = 2$. The rationale behind θ is to combine the evolutionary effects of mutation and Y-effective population size in one parameter. In fact, under certain assumptions, the genetic variation in a population only depends upon the product of N_e and μ, rather than each parameter individually [40]. The term 'scaled mutation rate' refers to the fact that time is scaled by N_e in the coalescent approximation. Since θ also equals the expected

number of mutations separating two genetic profiles, it determines the genetic distribution at the leaves of the coalescent tree. Parameter θ can be estimated directly by coalescent-based MCMC methods, whereas N_e and μ have to be derived (indirectly) from θ using additional information included in the priors. Estimates of θ are therefore generally more accurate than those of N_e and μ, although both parameters may often be scientifically interesting in their own right. This phenomenon also became apparent in our analyses with incorrect priors, where N_e was estimated too low with BEAST and BATWING, but the estimates of θ were fairly accurate. The same analyses also showed that, adopting uniform prior distributions for θ (the sole options for IMa2 and LAMARC), the boundaries of these distributions are sometimes returned as estimates, indicating that either the model or the priors were inadequate for the data analyzed. Notably, LAMARC avoids estimation of N_e and μ altogether and accounts for θ alone in its MCMC implementation. Consequently, LAMARC provides no opportunity to estimate Y-effective population sizes or mutation rates directly.

When drawing practical conclusions from our comparison of the four MCMC-based tools, one has to bear in mind that they were built for specific purposes. While LAMARC and IMa2 put a focus on migration, BATWING was developed mainly as a proof of principle, without including extensive population genetics models at all. In turn, BEAST concentrates on cross-platform applicability and a large variability of implemented models. Another factor determining the function and format of each tool is the data types to be analyzed. For example, different requirements result when studying within-species or between-species data, data of closely related species or data from populations with geographic structure. Therefore, the results of our comparative study should also be gauged in relation to the purpose and provenance of the respective tool.

Acknowledgements

The work of Sven Gundlach was partly financed by the German Federal Ministry of Education and Research (BMBF) through its e:Med framework (01ZX1510).

Declaration of Competing Interest

None.

Appendix A. Supplementary data

Supplementary data to this article can be found online at https://doi.org/10.1016/j.csbj.2019.07.014.

References

[1] Jobling MA, Tyler-Smith C. The human Y chromosome: an evolutionary marker comes of age. Nat Rev Genet 2003;4:598–612.
[2] Wei W, Ayub Q, Xue Y, Tyler-Smith C. A comparison of Y-chromosomal lineage dating using either resequencing or Y-SNP plus Y-STR genotyping. Forensic Sci Int Genet 2013;7:568–72.
[3] Nagle N, Ballantyne KN, van Oven M, Tyler-Smith C, Xue Y, Taylor D, et al. Antiquity and diversity of aboriginal Australian Y-chromosomes. Am J Phys Anthropol 2016; 159:367–81.
[4] Bajic V, Barbieri C, Hubner A, Guldemann T, Naumann C, Gerlach L, et al. Genetic structure and sex-biased gene flow in the history of southern African populations. Am J Phys Anthropol 2018;167:656–71.
[5] Excoffier L, Heckel G. Computer programs for population genetics data analysis: a survival guide. Nat Rev Genet 2006;7:745–58.
[6] Anderson CN, Ramakrishnan U, Chan YL, Hadly EA. Serial SimCoal: a population genetics model for data from multiple populations and points in time. Bioinformatics 2005;21:1733–4.
[7] Chen GK, Marjoram P, Wall JD. Fast and flexible simulation of DNA sequence data. Genome Res 2009;19:136–42.
[8] Excoffier L, Dupanloup I, Huerta-Sanchez E, Sousa VC, Foll M. Robust demographic inference from genomic and SNP data. PLoS Genet 2013;9:e1003905.
[9] De Mita S, Siol M. EggLib: processing, analysis and simulation tools for population genetics and genomics. BMC Genet 2012;13:27.
[10] Liang L, Zollner S, Abecasis GR. GENOME: a rapid coalescent-based whole genome simulator. Bioinformatics 2007;23:1565–7.
[11] Mailund T, Schierup MH, Pedersen CN, Mechlenborg PJ, Madsen JN, Schauser L. CoaSim: a flexible environment for simulating genetic data under coalescent models. BMC Bioinform 2005;6:252.
[12] Bouckaert R, Heled J, Kuhnert D, Vaughan T, Wu CH, Xie D, et al. BEAST 2: a software platform for Bayesian evolutionary analysis. PLoS Comput Biol 2014;10:e1003537.
[13] Drummond AJ, Suchard MA, Xie D, Rambaut A. Bayesian phylogenetics with BEAUTI and the BEAST 1.7. Mol Biol Evol 2012;29:1969–73.
[14] Kingman JFC. The coalescent. Stoch Process Appl 1982;13:235–48.
[15] Sorensen D, Gianola D. Likelihood, Bayesian, and MCMC methods in quantitative genetics. Berlin: Springer; 2007.
[16] Wilson IJ, Weale ME, Balding DJ. Inferences from DNA data: population histories, evolutionary processes and forensic match probabilities. J R Stat Soc Stat 2003; 166:155–88.
[17] Barido-Sottani J, Boskova V, Plessis LD, Kuhnert D, Magnus C, Mitov V, et al. Taming the BEAST-A community teaching material resource for BEAST 2. Syst Biol 2018;67: 170–4.
[18] Andersen MM, Caliebe A, Jochens A, Willuweit S, Krawczak M. Estimating trace-suspect match probabilities for singleton Y-STR haplotypes using coalescent theory. Forensic Sci Int Genet 2013;7:264–71.
[19] Hey J. Isolation with migration models for more than two populations. Mol Biol Evol 2010;27:905–20.
[20] Beerli P, Palczewski M. Unified framework to evaluate panmixia and migration direction among multiple sampling locations. Genetics 2010;185:313–26.
[21] Kuhner MK. LAMARC 2.0: maximum likelihood and Bayesian estimation of population parameters. Bioinformatics 2006;22:768–70.
[22] Drummond AJ, Rambaut A. BEAST: Bayesian evolutionary analysis by sampling trees. BMC Evol Biol 2007;7:214.
[23] Nielsen R, Wakeley J. Distinguishing migration from isolation: a Markov chain Monte Carlo approach. Genetics 2001;158:885–96.
[24] Hey J, Nielsen R. Integration within the felsenstein equation for improved Markov chain Monte Carlo methods in population genetics. Proc Natl Acad Sci U S A 2007; 104:2785–90.
[25] Sethuraman A, Hey J. IMa2p—parallel MCMC and inference of ancient demography under the isolation with migration (IM) model. Mol Ecol Resour 2016;16:206–15.
[26] Hey J, Chung Y, Sethuraman A, Tishkoff S, Lachance J, Sousa VC, et al. Phylogeny estimation by integration over isolation with migration models. Mol Biol Evol 2018; 35:2805–18.
[27] Tenesa A, Navarro P, Hayes BJ, Duffy DL, Clarke GM, Goddard ME, et al. Recent human effective population size estimated from linkage disequilibrium. Genome Res 2007;17:520–6.
[28] Macpherson JM, Ramachandran S, Diamond L, Feldman MW. Demographic estimates from Y chromosome microsatellite polymorphisms: analysis of a worldwide sample. Hum Genomics 2004;1:345–54.
[29] Willuweit S, Roewer L. The new Y chromosome haplotype reference database. Forensic Sci Int Genet 2015;15:43–8.
[30] Shi W, Ayub Q, Vermeulen M, Shao RG, Zuniga S, van der Gaag K, et al. A worldwide survey of human male demographic history based on Y-SNP and Y-STR data from the HGDP-CEPH populations. Mol Biol Evol 2010;27:385–93.
[31] Ayres DL, Darling A, Zwickl DJ, Beerli P, Holder MT, Lewis PO, et al. BEAGLE: an application programming interface and high-performance computing library for statistical phylogenetics. Syst Biol 2012;61:170–3.
[32] R Core Team. R: A language and environment for statistical computing. Vienna, Austria: R Foundation for Statistical Computing; 2018.
[33] Dragulescu AA, Arendt C. XLSX: read, write, format excel 2007 and excel 97/2000/XP/2003 files. R package version 061. https://CRANR-projectorg/package=xlsx; 2018.
[34] Rambaut A, Drummond AJ, Xie D, Baele G, Suchard MA. Posterior summarization in Bayesian phylogenetics using tracer 1.7. Syst Biol 2018;67:901–4.
[35] Geweke J. Evaluating the accuracy of sampling-based approaches to calculating posterior moments. In: Bernardo J, Berger J, Dawid A, Smith A, editors. Bayesian statistics 4. New York: Oxford University Press; 1992. p. 169–93.
[36] Gelman A, Rubin DB. Inference from iterative simulation using multiple sequences. Stat Sci 1992;7:457–72.
[37] Sainudiin R, Durrett RT, Aquadro CF, Nielsen R. Microsatellite mutation models: insights from a comparison of humans and chimpanzees. Genetics 2004;168:383–95.
[38] Jochens A, Caliebe A, Roesler U, Krawczak M. Empirical evaluation reveals best fit of a logistic mutation model for human Y-chromosomal microsatellites. Genetics 2011; 189:1403–11.
[39] Simonsson I, Mostad P. Stationary mutation models. Forensic Sci Int Genet 2016;23: 217–25.
[40] Hein J, Schierup M, Gene Genealogies Wiuf C. Variation and evolution: a primer in coalescent theory. New York: Oxford University Press; 2004.