BIOINFORMATICS

Vol. 00 no. 00 2012
Pages 1–2

Coalescent simulation in continuous space

Jerome Kelleher [1], Nicholas H. Barton[2] and Alison M. Etheridge[3]

[1] University of Edinburgh, King's Buildings, West Mains Road, EH9 3JT, UK
[2] Institute of Science and Technology, Am Campus I, A-3400 Klosterneuberg, Austria
[3] Department of Statistics, University of Oxford, 1 South Parks Road, Oxford OX1 3TG, UK

Received on XXXXX; revised on XXXXX; accepted on XXXXX

Associate Editor: XXXXXXX

ABSTRACT

Summary: Coalescent simulation has become an indispensable tool in population genetics and many complex evolutionary scenarios have been incorporated into the basic algorithm. Despite many years of intense interest in spatial structure, however, there are no available methods to simulate the ancestry of a sample of genes that occupy a spatial continuum. This is mainly due to the severe technical problems encountered by the classical model of isolation by distance. A recently introduced model solves these technical problems and provides a solid theoretical basis for the study of populations evolving in continuous space. We present a detailed algorithm to simulate the coalescent process in this model, and provide an efficient implementation of a generalised version of this algorithm as a freely available Python module.

Availability: Package available at http://pypi.python.org/pypi/ercs

Contact: jerome.kelleher@ed.ac.uk

Supplementary Information: Direct Python implementations of the simulation algorithms are available at Bioinformatics online.

1 INTRODUCTION

The extinction/recolonisation model (or spatial Λ-Fleming-Viot process) is a recently introduced model [Etheridge, 2008] that captures the dynamics of populations evolving in a spatial continuum [Barton et al., 2010a,b]. The model solves long-standing problems [Felsenstein, 1975] with the classical models of isolation by distance [Wright, 1943, Malécot, 1948] and has the potential to explain key biological facts that cannot be captured by simple diffusion [Barton et al., 2010b].

In this model each individual occupies a fixed location in continuous space and all movement and reproduction happen as the consequence of extinction/recolonisation events. Events fall randomly throughout space, independent (crucially) of the location of extant individuals. At an event, some fraction of the individuals nearby die, to be replaced by the offspring of a small number of parents chosen from the nearby population immediately before the event. Events at different scales model the effects of life-history and demography. For example, the regular reproduction of individuals may be modelled by means of very small and frequent events in which a few individuals in a local area die and are replaced by the progeny of nearby parents. On the other hand, broad-scale mortality events can affect a substantial fraction of the population over the entire species range. See Barton et al. [2013] for an extensive review of the model, its applications and background.

2 SINGLE LOCUS ALGORITHM

The coalescent process for the extinction/recolonisation model at a single locus is straightforward [Barton et al., 2010b], and we describe a detailed algorithm to simulate this process in this section. In the interest of simplicity here we restrict ourselves to a single class of event from the disc model [Barton et al., 2010a] in which events have a fixed radius r; within these discs, a single parent is chosen uniformly and individuals die with probability u.

The algorithm is described in terms of oriented trees [Knuth, 2011, p. 461]. In an oriented tree parent-child relationships are important but the order of children at a node is not. This information is encoded as a sequence $\pi_1 \dots \pi_n$, where π_j is the parent of node j and node j is a root if $\pi_j = 0$. Oriented trees have several advantages over more traditional approaches to encoding genealogies such as nested parentheses or linked structures. They provide a concise and elegant means of describing genealogies, are trivial to store and parse, and can be annotated with additional information without effort. Finally, since oriented trees are simple lists of integers, we can describe coalescent algorithms precisely and without ambiguity.

Let $\mathcal{R}_U(A)$ be an element of the set A chosen uniformly at random, and let $\mathcal{R}_E(\lambda)$ be an exponentially distributed random value with rate λ. Also, let $B(z, r, L)$ define a disc of radius r centred at z on a two-dimensional torus of diameter L and let $\varnothing$ denote the null point.

Algorithm S *(Single locus coalescent).* Simulate the ancestry (π, τ) of individuals sampled at locations $x_1 \dots x_n$ at time $t = 0$ under a model in which events with radius r and impact u occur at rate λ on a two-dimensional torus of diameter L.

S1 [Initialisation.] Set $\pi_j \leftarrow 0$, $\tau_j \leftarrow 0$ and $\chi_j \leftarrow \varnothing$ for $1 \leq j < 2n$. Then set $\chi_j \leftarrow x_j$ for $1 \leq j \leq n$ and set $S \leftarrow \{j \mid 1 \leq j \leq n\}$. Finally, set $\eta \leftarrow n + 1$ and $t \leftarrow 0$.

S2 [Event.] Set $t \leftarrow t + \mathcal{R}_E(\lambda)$ and $z \leftarrow \mathcal{R}_U([0, L)^2)$.

S3 [Birth.] Set $C \leftarrow \emptyset$. For each $j \in S$, if $\chi_j \in B(z, r, L)$ and $\mathcal{R}_U([0, 1)) < u$, set $C \leftarrow C \cup \{j\}$. Afterwards, if $|C| = 0$ go to **S2**; else, if $|C| = 1$ go to **S4**; otherwise, go to **S5**.

S4 [One lineage jumps.] Let $C = \{j\}$, set $\chi_j \leftarrow \mathcal{R}_U(B(z, r, L))$ and go back to **S2**.

S5 [Coalescence.] For each $j \in C$, set $\pi_j \leftarrow \eta$. Then, set $\tau_\eta \leftarrow t$, $\chi_\eta \leftarrow \mathcal{R}_U(B(z, r, L))$, $S \leftarrow (S \setminus C) \cup \{\eta\}$ and $\eta \leftarrow \eta + 1$. Finally, if $|S| > 1$ go back to **S2**. ▌

Algorithm **S** operates by mapping individuals in the sample to the integers $1 \leq j \leq n$ and those ancestral to the sample to integers greater than n. The set S contains the lineages ancestral to the sample at time t, and we proceed backwards in time event-by-event. Since most events will not hit any lineages, the majority of the time spent in the algorithm is looping quickly around steps **S2** and **S3**. Occasionally, a single lineage will be hit, we execute step **S4** and move the lineage to a new location. Very rarely, an event hits more than one lineage and C is the set of lineages born from a single parent. Thus, in step **S5** we set the parent of each lineage in C to be η and the location of η to a random point within the ball defining the event. We then record that lineage η entered the sample at time t by setting τ_η to t. Finally, we remove the children born in this event from the sample (as we are no longer interested in them) and insert η into the sample (as we are interested in its further history). This process continues until there is one individual in the sample, and (π, τ) describes the full history of the sample.

3 MULTILOCUS ALGORITHM

The single locus model is extended to incorporate recombination by letting each individual in the sample consist of multiple loci and allowing for multiple parents in events, so that a given individual may descend from different parents at different loci. Specifically, we consider a model in which each individual has m linearly arranged loci and there are ν parents at each event. (In a sexually reproducing species, $\nu = 2$ in small-scale reproduction events. We must consider the possibility of more than two parents in large-scale events, since several generations may elapse before the local area is repopulated.) For each child in an event, there is a probability ρ_ℓ that loci ℓ and $\ell + 1$ are inherited from different parents. See Etheridge and Véber [2013] for more details.

Simulating this multilocus coalescent follows the same pattern as Algorithm **S**: we begin with a sample of n lineages and proceed event-by-event until the ancestry of the sample is complete, and we have a pair (π, τ) describing the history at each locus. The sample S is most conveniently represented as a set of (location, ancestry) tuples $(x, a_1 \dots a_m)$. Each sequence $a_1 \dots a_m$ represents the ancestry of a lineage such that $a_\ell \neq 0$ if there is genetic material ancestral to the sample present in this lineage at locus ℓ and $a_\ell = 0$ otherwise [Hudson, 1983]. Termination of the algorithm is controlled by maintaining the invariant

$$\kappa = \sum_{(x,a)\in S} \sum_{1\leq \ell \leq m} [a_\ell \neq 0].$$

Thus $\kappa = nm$ initially and the algorithm terminates when $\kappa = m$, indicating that all loci have coalesced.

Suppose that C is the set of lineages born in an event. We must first decide which children descend from which parent at each locus. To do this, we first set $\delta_{k,\ell} \leftarrow \emptyset$ for $1 \leq k \leq \nu$ and $1 \leq \ell \leq m$. Then, for each child lineage $a_1 \dots a_m \in C$ we choose a parent for the first locus by setting $k \leftarrow \mathcal{R}_U(\{1, \dots, \nu\})$. We then iterate over each locus $1 \leq \ell \leq m$ and if $a_\ell \neq 0$ (that is, there is ancestral material at this locus) we set $\delta_{k,\ell} \leftarrow \delta_{k,\ell} \cup \{a_\ell\}$. Then, with probability ρ_ℓ a recombination event occurs and we choose a new parent for the next locus by setting $k \leftarrow \mathcal{R}_U(\{1, \dots, \nu\} \setminus \{k\})$.

Having completed the task of deciding which children have descended from which parents, we must generate the new parental ancestry sequences and update the genealogies and node times. This is accomplished by examining the sets of descendants $\delta_{k,\ell}$ for each parent k at locus ℓ and proceeding in the same manner as algorithm **S**, updating π, τ and η for each locus as appropriate.

This algorithm is far from optimal and can be improved in many ways to improve performance, particularly in special cases. It does provide a useful starting point for these special cases, however, and is easily adapted and analysed. A straightforward implementation of the algorithm in Python is included as Supplementary Material for reference.

4 IMPLEMENTATION

The multilocus coalescent algorithm described in the previous section is implemented as a Python module, `ercs`. The implementation supports an arbitrary number of event classes, including events from the Gaussian model [Barton et al., 2010b], and incorporates spatial indexing to improve performance over the basic algorithm outlined here. The module is written primarily in C in the interest of efficiency, and is distributed freely under the terms of the GNU General Public License.

ACKNOWLEDGEMENT

Funding: NHB is supported by European Research Council grant 250152; AME is supported in part by EPSRC grant EP/I01361X/1 and JK is supported by EPSRC grant EP/I013091/1.

REFERENCES

N. H. Barton, A. M. Etheridge, and A. Véber. A new model for evolution in a spatial continuum. *Electron. J. of Probab.*, 15:7, 2010a.

N. H. Barton, J. Kelleher, and A. M. Etheridge. A new model for extinction and recolonisation in two dimensions: quantifying phylogeography. *Evolution*, 64(9):2701–2715, 2010b.

N. H. Barton, A. M. Etheridge, and A. Véber. Modelling evolution in a spatial continuum. *J. Stat. Mech.*, doi:10.1088/1742-5468/2013/01/P01002, 2013.

A. M. Etheridge. Drift, draft and structure: some mathematical models of evolution. *Banach Center Publications*, 80:121–144, 2008.

A. M. Etheridge and A. Véber. The spatial Λ-Fleming-Viot process on a large torus: genealogies in the presence of recombination. *Ann. Appl. Probab.*, 22(6):2165–2209, 2013.

J. Felsenstein. A pain in the torus: some difficulties with the model of isolation by distance. *Amer. Nat.*, 109:359–368, 1975.

R. R. Hudson. Properties of a neutral allele model with intragenic recombination. *Theor. Pop. Biol.*, 23:183–201, 1983.

D. E. Knuth. *Combinatorial Algorithms*, volume 4A of *The Art of Computer Programming*. Addison Wesley, 2011.

G. Malécot. *Les mathématiques de l'hérédité*. Masson et Cie, Paris, 1948.

S. Wright. Isolation by distance. *Genetics*, 28:114–138, 1943.