Supplement to "Population dynamics inferred from genetic data via sequential Monte Carlo"

Alex Smith[1,*], Edward L. Ionides[2], Aaron A. King[3,4,5]

1 Department of Bioinformatics, University of Michigan, Ann Arbor, Michigan, USA
2 Department of Statistics, University of Michigan, Ann Arbor, Michigan, USA
3 Department of Ecology & Evolutionary Biology, University of Michigan, Ann Arbor, Michigan, USA
4 Center for the Study of Complex Systems, University of Michigan, Ann Arbor, Michigan, USA
5 Department of Mathematics, University of Michigan, Ann Arbor, Michigan, USA
∗ E-mail: alxsmth@umich.edu

Supplementary Content

S1 The GenPOMP model: linking infectious disease dynamics with genetic data **S-2**

S2 A GenSMC algorithm for filtering the GenPOMP model **S-7**

S2.1 The implementation of GenSMC in the `genPomp` program S-13

S2.2 Extending GenSMC to infer unknown parameters: The GenIF algorithm S-13

S2.3 Scalability of GenSMC S-14

S3 A theoretical derivation of the GenSMC algorithm **S-15**

S3.1 A basic SMC algorithm S-15

S3.2 A targeted SMC approach with a partial plug-and-play property S-17

S3.3 SMC with hierarchical sampling S-19

S3.4 Just-in-time evaluation of some state variable components S-20

S3.5 Moving from discrete time to continuous time S-21

S4 Details of the HIV model used in the main text **S-21**

S4.1 Initial values for the HIV model S-23

S4.2 Algorithmic parameters used for the numerical results S-25

S1 The GenPOMP model: linking infectious disease dynamics with genetic data

We define a class of models that describes an environment within which our general software implementation can be described. We aim at sufficient generality to represent the breadth of applicability of our methodology and the key methodological innovations, yet including enough details to describe the specific data analysis in the main text.

Data consist of n^* genetic sequences of a pathogen. We use a convention that $j{:}k$ denotes the arithmetic sequence $(j, j+1, \ldots, k)$, so that the entire collection of genetic sequence data can be written as

$$(g_1^*, g_2^*, \ldots, g_{n^*}^*) = g_{1:n^*}^*.$$

We use asterisks to denote data, to distinguish these from quantities arising in the model. The times at which the sequenced samples are collected are also data, and the total number of sequences, n^*, will be modeled as the outcome of a random process rather than some fixed quantity. We write the genetic sequence times as

$$(t_1^*, t_2^*, \ldots, t_{n^*}^*) = t_{1:n^*}^*.$$

We suppose that the data are collected in a time interval

$$\mathbb{T} = [t_o, t_{\text{end}}],$$

with $t_0 \le t_1^* < t_2^* < \cdots < t_{n^*}^* \le t_{\text{end}}$. Note that we allow multiple observations at the same time: such ties can be resolved arbitrarily in the ordering of the t_n^*. For simplicity, we exclude the possibility of such simultaneous observations in the following. If no sequence is available for the diagnosis at some time t_n^*, we set $g_n^* = \text{NA}$. Otherwise, we suppose the collection of sequences $g_{1:n^*}^*$ consist of aligned sequences of length L, i.e., $g_n^* \in \{A, C, T, G\}^L$.

Here, we do not include the possibility of additional clinical or epidemiological measurements available at diagnosis, though an extension to allow this is fairly straightforward. Further, we consider that only a consensus pathogen sequence is available from each host, so we ignore the possibility of extracting information from data on pathogen genetic diversity within hosts. Nevertheless, our framework can account for sequencing error and differences between observed and transmitted pathogen populations.

The partially observed Markov process (POMP) model consists of a latent, unobservable, Markov process $\{X(t), t \in \mathbb{T}\}$ and an observable process $\{Y(t), t \in \mathbb{T}\}$. $X(t)$ takes values in a set $\mathbb{X}$ and $Y(t)$ takes values in a set $\mathbb{Y}$. A POMP model for genetic data, which we call a GenPOMP, is required to have the following structure. $\{Y(t)\}$ consists of a collection of random number N of diagnosis times, denoted $T_{1:N}$, and corresponding sequences $G_{1:N}$. The observed outcomes are $N = n^*$ and $(T_n, G_n) = (t_n^*, g_n^*)$ for $n \in 1{:}n^*$. We adhere to a convention that random variables are denoted by upper case letters; the corresponding lower case letters are used for possible values of the random variable, and asterisks denote the actual data for observable variables; blackboard bold typeface is used for sets.

Recall that, in the main text, we wrote $X(t) = \big(\mathcal{T}(t), \mathcal{P}(t), \mathcal{U}(t)\big)$ where $\mathcal{T}(t)$ is a *transmission forest* and $\mathcal{P}(t)$ is a *phylogeny*. Here, it is convenient to take a different, but functionally equivalent, approach. We do not require that $X(t)$ itself contains $\mathcal{T}(t)$ and $\mathcal{P}(t)$, but we do require that $\{X(u), t_0 \le u \le t\}$ is sufficient to construct $\mathcal{T}(t)$ and $\mathcal{P}(t)$. This additional layer of abstraction lets us define the GenPOMP model without having to explicitly construct the processes $\{\mathcal{T}(t), t \in \mathbb{T}\}$ and $\{\mathcal{P}(t), t \in \mathbb{T}\}$.

The set $\mathbb{X}$ should describe the state of each individual in a study population. The study population is supposed to contain a finite number of individuals drawn from a countable collection of individuals who could potentially enter the study population. We suppose these potential individuals are labeled with values in the natural numbers, $\mathbb{N} = \{1, 2, 3, \dots\}$, and so collections of individuals in the study population take values in the set $\mathbb{H}$ consisting of all finite subsets of $\mathbb{N}$. We suppose there is a random process $\{H(t), t \in \mathbb{T}\}$, with $H(t)$ taking a value in $\mathbb{H}$ corresponding to the identities of all individuals in the study population at time t. Formally, we suppose that $H(t)$ is constructed from $X(t)$ via a suitable function mapping $\mathbb{X}$ to $\mathbb{H}$. We suppose that each individual in the study population has a state in a set $\mathbb{S}$. For a simple compartment model, $\mathbb{S}$ could be finite or countable, however, we also allow for the possibility of continuous real-valued state variables. In particular, we will later define a random clock process governing the rate of pathogen evolution within each individual infected host. To keep track of the state of each member of the study population, we suppose that the state of any individual i in the study population at time t is given by a random variable $X_i(t)$, constructed from $X(t)$ via a suitable function mapping $\mathbb{X}$ to $\mathbb{S}$. A canonical way to do this is to take

$$\mathbb{X} = \bigcup_{h \in \mathbb{H}} \mathbb{S}^h, \tag{S1}$$

for which an element $\big(s_{i_1}, s_{i_2}, \dots, s_{i_k}\big) \in \mathbb{X}$ is interpreted to mean that the study population is $\{i_1, i_2, \dots, i_k\} \in \mathbb{H}$ and individual i_j is in state $s_j \in \mathbb{S}$. Our definition of the study population is the collection of individuals being modeled, and so the state of individuals outside the study population is necessarily undefined. In order to define $\{X_i(t), t \in \mathbb{T}\}$ as a stochastic process, one can formally define an additional state $\odot$ and set $X_i(t) = \odot$ when $i \notin H(t)$. Note that, in general, $\{X_i(t), t \in \mathbb{T}\}$ does not inherit the Markov property from $\{X(t), t \in \mathbb{T}\}$. If individual state transitions occur as an independent Markovian process once that individual is infected (as is the case in our HIV example) then $\{X_i(t), t \in [t_i, t_{\text{end}}]\}$ has a conditional Markov property given $i \in H(t_i)$.

The state process may, in general, need to include other components in addition to $\{X_i(t), i \in H(t)\}$. For example, $X(t)$ may include dynamic variables affecting the entire population, such as environmental or sociological processes. For the remainder of this article, the specific construction in equation (S1) suffices, but that is not essential to our approach. If $\mathbb{S}$ is countable then $\mathbb{X}$, given by (S1), is also countable and $\{X(t)\}$ is a Markov chain. Otherwise, $\{X(t)\}$ is a more general Markov process.

Some basic properties of individuals characterize the model as a disease transmission system, and these are required to construct the evolutionary process model for the pathogen. This leads us to define functions that return properties about the state of an individual, and we call these *query functions*. This notation differs from usual compartment models, where each individual is modeled as residing in a single compartment. We write properties as functions of $X(t)$, rather

than components of $X(t)$, to keep applicability to a broad class of population models. As long as the required query functions can be defined for a population model, the statistical methodology developed will apply, giving the scientist considerable flexibility in the specification of the model.

We require that an individual's state, i.e., its value in $\mathbb{S}$, can describe whether that individual is infected and infectious. We represent this requirement by supposing that there is a query function

$$Q_I : \mathbb{S} \to \{0, 1\}$$

defined as,

$$Q_I(s) = \begin{cases} 1 & \text{if } s \text{ is an infected state,} \\ 0 & \text{if } s \text{ is an uninfected state.} \end{cases}$$

To link the model to diagnosis data, we require that a state in $\mathbb{S}$ determines whether an individual is diagnosed while part of the study population. Specifically, we suppose there is a query function

$$Q_D : \mathbb{S} \to \{0, 1\}$$

such that

$$Q_D(s) = \begin{cases} 1 & \text{if } s \text{ is a state for individuals diagnosed as infected while in the study population,} \\ 0 & \text{otherwise.} \end{cases}$$

We then define

$$D(t) = \sum_{i \in H(t)} Q_D\big(X_i(t)\big),$$

which counts the number of individuals diagnosed while in the study population, by time t. This counting process (i.e., a non-decreasing integer-valued process) is relevant for relating the model to the data on the study population. Note that $D(t)$ does not count the number of clinically diagnosed individuals in the study population at time t, which would require a different accounting for the possibility of immigration and emigration of diagnosed individuals.

Now, we define the set of infected states to be

$$\mathbb{I} = \big\{ s \in \mathbb{S} : Q_I(s) = 1 \big\}.$$

We suppose that the state contains information about the identify of the infector, and we do this by requiring the existence of a query function

$$Q_L : \mathbb{I} \to \mathbb{N} \cup \{0\}$$

defined such that

$$Q_L(s) = \begin{cases} j & \text{if } s \text{ is infected by individual } j \text{ within the study population,} \\ 0 & \text{if } s \text{ is infected by an infector outside the study population.} \end{cases}$$

The capability to construct the query function $Q_L(s)$ requires that the identity of the infector is stored in the state variable at the point of infection, so it is available later as part of the state of the infectee. Information on the identity of the infector is not usually required for a compartment model, but is useful when working with genetic data in order to track lineages of the pathogen.

The evolutionary process of the pathogen genome within an individual in the host populations is modeled using a relaxed molecular clock, meaning that standard molecular models for evolution are applied on a stochastically perturbed timescale. It has become established that the usual models for molecular evolution fit sequence data better if one allows such fluctuations in the rate of evolution (Drummond et al., 2006). To implement a relaxed clock, we construct a random process on each edge of the transmission tree. This process scales calendar time to evolutionary time, the latter meaning a modified timescale on which the evolutionary rate is constant. We therefore require the existence of a query function

$$Q_\Gamma : \mathbb{I} \to \mathbb{R}$$

returning the relaxed evolutionary clock time corresponding to evolution of a transmissible pathogen population within an infected individual. Specifically, $Q_\Gamma(s)$ represents the random, individual-specific, clock time for the evolutionary process that separates the host's transmissible pathogen population from the rest of the pathogen community when the host is in state $s \in \mathbb{I}$. For an individual based model in which an individual is infected within the study population, this corresponds to the evolutionary process within the host subsequent to infection. Immigrant pathogens require additional assumptions on how they relate genetically to pathogens already circulating in the study population. Conditional on the randomly perturbed molecular clock, pathogen evolution is usually specified by a general time-reversible Markov model.

We also suppose the existence of a query function

$$Q_\Delta : \mathbb{I} \to \mathbb{R}$$

which returns the relaxed evolutionary clock time separating the measurable pathogen population from the transmissible host population within an infected individual. If and when an individual gives rise to a pathogen genetic sequence within the dataset, this clock time adds to the clock time $Q_\Gamma(s)$ in determining the probability distribution of the measured sequence.

The separation of the pathogen evolutionary process into transmitted and untransmitted mutations has multiple interpretations. The choice of primary interpretation has consequences for the appropriate model specification of the branch separating the measurement node v from the transmission tree. The plausibility of these different interpretations will depend on the biological system under investigation.

(B1) Measurement error. Sequencing error could be modeled by an arbitrary evolution-like process on the branch separating the measured sequence from the transmissible sequence.

(B2) Transmissible versus measurable strains. The measured sequence may reflect the dominant strain reproducing most competitively within the host. It is conceivable that much of the diversity resulting from within-host evolution may lead to pathogens which are non-viable or non-competitive for between-host transmission. The evolutionary branch corresponding to the measurement event could represent this dead-end evolution, leaving the main body of the transmission tree to represent evolution of a transmissible strain.

(B3) Within-host diversity. A strain transmitted subsequent to sequencing could be more similar to an ancestral strain than to the sequenced strain by chance, due to within-host genetic variation, even without appealing to a phenomenon such as (B2). The measurement branch

permits such behavior, so may help to adjust for unmodeled within-host pathogen genetic diversity.

Other model-specific quantities can be defined by additional query functions, but are not essential components of a GenPOMP model. For example, epidemiological models commonly consider the number of susceptible or removed individuals. Also, having defined an appropriate query function for a category of individuals, one can define a process counting such individuals. For example, to complement the query function Q_I for infected individuals, we can define a process

$$I(t) = \sum_{i \in H(t)} Q_I\big(X_i(t)\big)$$

counting the number of infected individuals in the study population. We can also write the size of the study population at time t as,

$$P(t) = |H(t)| = \sum_{i \in H(t)} 1.$$

Our framework therefore incorporates the structure of arbitrary compartment models (Bretó et al., 2009) represented at the level of compartment membership for each individual.

The history of the query functions for infected individuals,

$$\big\{Q_I\big(X_i(u)\big), Q_D\big(X_i(u)\big), Q_L\big(X_i(u)\big), Q_\Gamma\big(X_i(u)\big), Q_\Delta\big(X_i(u)\big) : t_0 \leq u \leq t\big\}, \tag{S2}$$

is sufficient to construct the transmission forest, $\mathcal{T}(t)$, and phylogeny, $\mathcal{P}(t)$, described in the New Approaches section of the main text. Formally, for (S2) and elsewhere, we extend the query functions to take an undefined value, denoted by $\odot$, when the argument is outside the defined domain. To specify the measurement process model, recall that the measurement process $\{Y(t)\}$ consists of an increasing sequence of diagnosis times $\{T_n\}$ associated with the diagnosis counting process $\{D(t)\}$, together with a collection of genetic sequences $\{G_n\}$. We suppose that the sequences $\{G_n\}$ are modeled as a continuous-time Markov chain on $\mathcal{P}(t)$. The probability distribution of the genetic sequence G_n at time T_n, conditional on $\{X(t), t \leq t_n\}$ and $G_{1:n-1}$, therefore depends on $\mathcal{P}(t)$ and $G_{1:n-1}$. If a genetic sequence for the diagnosis at time T_n is not available, we assign G_n the value NA. We suppose this occurs with probability $1 - p_G$, independently of $\{X(t)\}$.

We have defined the GenPOMP model so that the pathogen genetic sequence arises only in the measurement model. No genetic sequences are included in the state process, or its particle representation. Our approach is consistent with viewing the genetic evolutionary model as a principled way to define and evaluate a statistical metric between genetic sequences that respects the tree structure of the evolutionary process and has the property that similar sequences are more likely to come from closely related pathogens. A measurement model satisfying these criteria and providing a statistical fit to the data need not be judged on the details of its biological strengths and weaknesses if the microevolutionary processes are not the focus of the investigation. The individual, stochastic molecular clocks determining the rate of evolution within each host are included in the latent process component of the GenPOMP model to facilitate Monte Carlo integration over the distribution of these clocks, as described in Section S2.

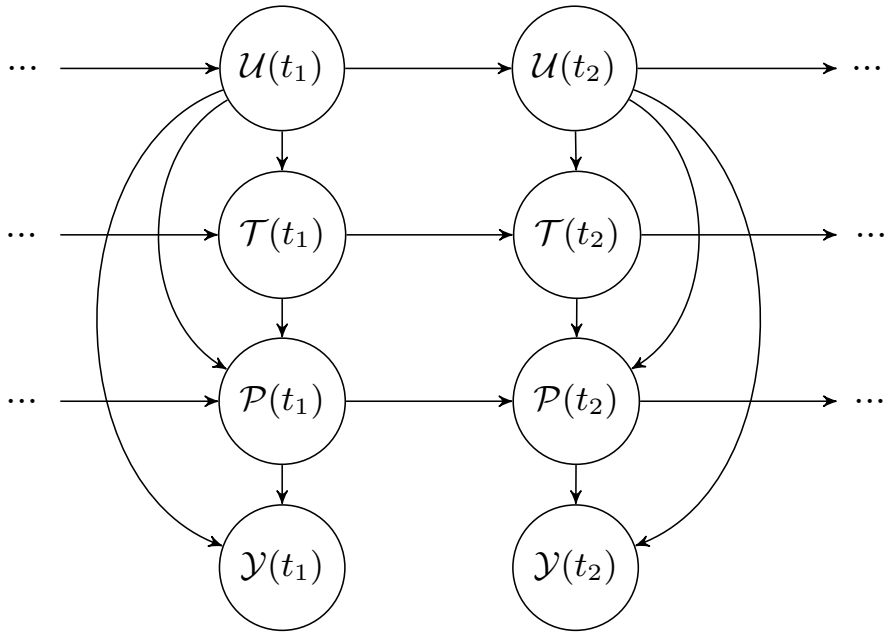

Figure S-1: A directed acyclical graph representation of dependencies among GenPOMP model components.

The definition of a GenPOMP model given here is general and abstract. The population model $\{X(t)\}$ corresponds to an arbitrary individual-based Markovian model constrained to include concepts of transmission of a pathogen and measurement of pathogen genetic sequences. The measurement model is constrained to be based on a Markovian evolutionary process, but this is standard in current models used for phylodynamic inference. Our methodological approach applies to this general GenPOMP model class, subject to being able to simulate from the individual-based model and compute the rate at which individual hosts provide a pathogen sequence. The Markovian assumption is convenient algorithmically. In one sense, it is not fundamentally a limitation since non-Markovian models may be approximated by Markovian models with additional state variables. In another sense, it is a practical limitation since increasing the size of the state space increased the computational effort required.

S2 A GenSMC algorithm for filtering the GenPOMP model

We develop a sequential Monte Carlo (SMC) approach for the framework of Section S1. We will use the name GenSMC to describe an SMC algorithm for GenPOMP models. As an instance of SMC, the basic principles and theoretical foundation for GenSMC follows from the general theory of SMC (Liu, 2001). However, GenPOMP models have a particular structure that places particular

demands on a GenSMC algorithm. Many variations are possible on our GenSMC algorithm, but demonstration of one successful GenSMC algorithm provides a foundation and motivation for future improvements. Our GenSMC approach is presented as pseudocode in Algorithm S-1, which is an expanded version of Algorithm 1 in the main text. We proceed to define the notation that will be required.

To construct our algorithm, we specify a concrete class of GenPOMP models. Let $\{X(t), t \in \mathbb{T}\}$ be a latent GenPOMP process with the form

$$X(t) = \Big(\Phi(t), \Psi(t), \Gamma(t), \Delta(t), D(t)\Big), \tag{S3}$$

having components $\Phi(t)$, $\Psi(t)$, $\Gamma(t)$, $\Delta(t)$ and $D(t)$ defined as follows:

$\{D(t)\}$ records diagnosis events within the study population, as defined in Section S1. We suppose that no diagnoses occur simultaneously, so $\{D(t)\}$ is a *simple* counting process. Therefore, we can model $\{D(t)\}$ via a conditional intensity process $\rho\big(\Phi(t), \Psi(t)\big)$ such that

$$\begin{aligned}
\mathbb{P}\big[D(t+\delta) - D(t) = 0 \,|\, \Phi(t), \Psi(t)\big] &= 1 - \delta\rho\big(\Phi(t), \Psi(t)\big) + o(\delta), \\
\mathbb{P}\big[D(t+\delta) - D(t) = 1 \,|\, \Phi(t), \Psi(t)\big] &= \delta\rho\big(\Phi(t), \Psi(t)\big) + o(\delta), \\
\mathbb{P}\big[D(t+\delta) - D(t) > 1 \,|\, \Phi(t), \Psi(t)\big] &= o(\delta),
\end{aligned}$$

where $o(\delta)$ denotes a function $f : [0, \infty) \to \mathbb{R}$ satisfying $\lim_{\delta \to 0} f(\delta)/\delta = 0$. Here, $\rho(X(t))$ is called the diagnosis rate.

$\{\Psi(t)\}$ is a piecewise constant process which records a list of the identity labels of individuals diagnosed by time t.

$\{\Phi(t)\}$ contains everything else in the GenPOMP model, so is essentially arbitrary within the general requirements of a GenPOMP model. We suppose that observation events are also recorded in the state process; specifically, the observation counting process $\{D(t)\}$ is a function of $\{\Phi(t)\}$ which gives rise to observation times $\{T_1, T_2, \dots\}$ at which the genetic measurements $\{G_1, G_2, \dots\}$ are made.

$\{\Gamma(t)\}$ is a list of the relaxed clock process for all the interior edges of the transmission tree, i.e., $\Gamma(t) = \{Q_\Gamma(X_i(t)), i \in \mathbb{N}\}$ where Q_Γ is defined in Section S1.

$\{\Delta(t)\}$ is a list of the relaxed clock process for the terminal branches of the transmission tree, i.e., $\Delta(t) = \{Q_\Delta(X_i(t)), i \in \mathbb{N}\}$ where Q_Δ is defined in Section S1.

The relaxed clock processes affect the micro-evolution of the pathogen, but in our model the genetic process has no consequence for the transmission dynamics: the genetic sequence is simply a marker, and the genetic models we use are models for neutral evolution. A consequence of this is that the relaxed clock processes only have to be evaluated when needed to compute the conditional probability mass function for attaching a new genetic sequence to the genetic tree. If these components of the latent process can be computed when needed, there is no need to continually update them. Our computational strategy to take advantage of this is called a just-in-time representation and is formally described in Section S3.4. Informally, the just-in-time representation is the tool that

lets us define the latent GenPOMP model as a continuous-time Markov process while updating the relaxed clock processes at diagnosis times, when needed. To simulate the GenPOMP model forward in time using a just-in-time representation, we need to be able to evaluate the relaxed clock process over arbitrary time intervals, and also split the evolutionary time over a branch of the transmission tree if a new measurement divides this branch. An example of a Markovian clock with these properties is the Gamma process.

We will show that the relaxed clock processes $\{\Gamma(t)\}$ and $\{\Delta(t)\}$ can be represented by two processes $\{U(t)\}$ and $\{V(t)\}$ which generate the evolutionary clocks that are necessary to evaluate the likelihood of the sequences. The processes $\{U(t)\}$ and $\{V(t)\}$ are constant except at diagnosis times, and so are fully specified by the discrete processes $U_{0:N}$ and $V_{0:N}$, with $U_n = U(T_n)$ and $V_n = V(T_n)$. The construction of $\{U(t), V(t)\}$ is an instance of just-in-time variables, as discussed further in Section S3.4. Therefore, for the purposes of Algorithm S-1, it is convenient to replace the representation in equation (S3) with an equivalent representation,

$$X(t) = \big(\Phi(t), \Psi(t), U(t), V(t), D(t)\big), \tag{S4}$$

The construction of $\{U(t)\}$ and $\{V(t)\}$ is described in Figure S-2

Algorithm S-1 is written using discrete time steps corresponding to the sequence of observation times, together with the start and end times of the interval $\mathbb{T}$. It convenient to define

$$t_0^*{=}t_0, \qquad t_{n^*+1}^* = t_{\text{end}},$$

so that $\mathbb{T} = [t_0^*, t_{n^*+1}^*]$. $\{\Psi(t)\}$ is fully specified by its values at the discrete set of observation times, and so we define a process $\{\Psi_n\}$ with

$$\Psi_n = \Psi(t_n^*).$$

To provide a discrete time representation of $\{\Phi(t)\}$, we write

$$\Phi_n = \{\Phi(t), t_{n-1}^* \le t \le t_n^*\},$$

for $n = 1, \ldots, n^* + 1$, with $\Phi_0 = \Phi(t_0^*)$. Similarly, we write

$$D_n = \{D(t), t_{n-1}^* \le t \le t_n^*\}.$$

Diagnosis events are modeled as perfectly observed, almost tautologically. We write $d^*(t)$ for the observed value of $D(t)$, defined as

$$d^*(t) = \sup\{n : t_n^* \le t\}.$$

Also, we write d_n^* for the observed value of D_n. Perfectly observed components of the latent process of a POMP model require special attention in sequential Monte Carlo algorithms, and so Algorithm S-1 uses the targeted proposal developed in Section S3.2 to handle the diagnosis process.

Hierarchical sampling (described in Section S3.3) is carried out in Algorithm S-1 over the components $\Phi(t)$ and $\Psi(t)$ in (S3), as well as over the components U_n and V_n in the just-in-time representation of $\{\Gamma(t)\}$ and $\{\Delta(t)\}$.

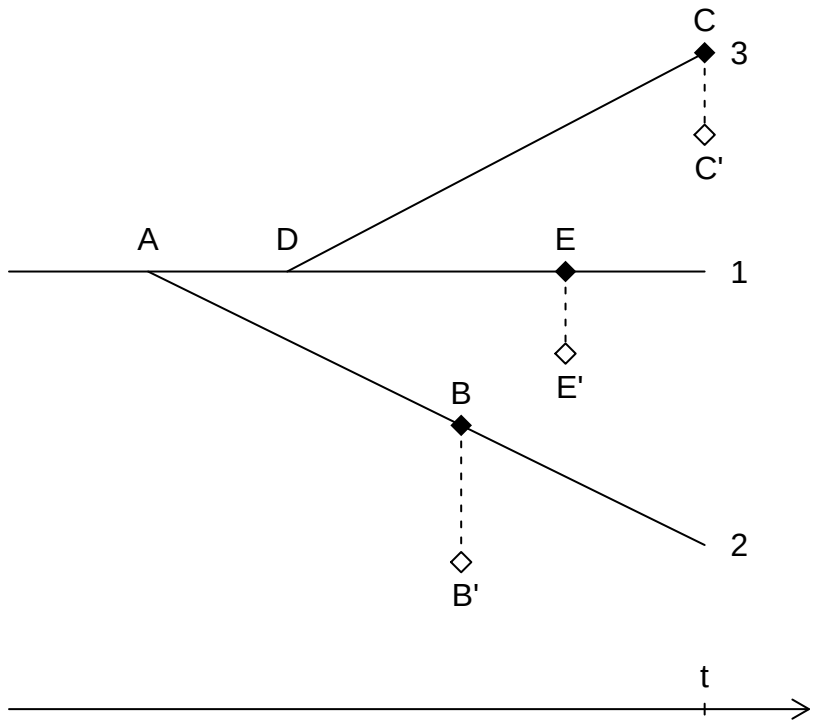

Figure S-2: The diagram represents the transmission tree for a particle where individual 1 infected individual 2 at time $A < t$ and individual 3 at time $D < t$. Sequences are collected at times B, C and E. Measured but untransmitted sequence mutations occur along BB', CC' and EE'. For this particle, we know that the sequence at time B corresponds to individual 2, and the sequence at time E belongs to individual 1. Suppose we then wish to evaluate the probability of the new sequence at time t conditional on it belonging to individual 3, as shown on the diagram. From the previous observed sequences, assigned to B' and E', this particle has already been assigned evolutionary clock times for the segments AB' and AE'. To place the new sequence at C', we first generate a new clock process for the segment DC', which is represented by the variable $U^P_{n,jkl}$ in step 8 of Algorithm S-1. Then, we split the evolutionary clock time for AE' into AD and DE', in a way that is consistent with the corresponding calendar times and the stochastic evolutionary clock process. This computation is represented by the variable $V^P_{n,jklm}$ in step 10 of Algorithm S-1.

The pseudocode for Algorithm S-1 adopts a space-saving convention that index j always ranges over $1:J$, index k ranges over $1:K$, index l ranges over $1:L$, and index m ranges over $1:M$. Thus, for example, line 6 of Algorithm S-1 has an implicit loop over $j \in 1:J$ and $k \in 1:K$.

If $g_n^* = \text{NA}$ then $w_2(n, j, k, l, m)$ is defined to be the probability of not recording a genetic sequence at diagnosis. In this case, steps 7 to 16 are not necessary: it suffices to take $K = 1$, with U_n and V_n being undefined. This special case is omitted from Algorithm S-1 for simplicity.

To implement Algorithm S-1, we require code to generate initial values, and to simulate the dynamic model for all the hierarchical layers conditional on the diagnosis events. Specifically, we require simulators for

$$f_{\Phi_0,\Psi_0}(\phi_0, \psi_0), \tag{S5}$$
$$f_{\Phi_n|\Phi_{n-1},\Psi_{n-1},D_n}\big(\phi_n|\phi_{n-1}, \psi_{n-1}, d_n^*\big), \tag{S6}$$
$$f_{\Psi_n|\Phi_n,\Psi_{n-1}}\big(\psi_n|\phi_n, \psi_{n-1}\big), \tag{S7}$$
$$f_{U_n|U_{n-1},V_{n-1},\Phi_{0:n},\Psi_{0:n}}\big(u_n|u_{n-1}, v_{n-1}, \phi_{0:n}, \psi_{0:n}\big), \tag{S8}$$
$$f_{V_n|V_{n-1},U_n,\Phi_{0:n},\Psi_{0:n}}\big(v_n|v_{n-1}, u_n, \phi_{0:n}, \psi_{0:n}\big). \tag{S9}$$

We then require code to evaluate the diagnosis rate,

$$\rho\big(\phi(t), \psi(t)\big) \tag{S10}$$

as well as the genetic measurement model,

$$f_{G_n|G_{1:n-1},\Phi_{0:n},\Psi_{0:n},U_{0:n},V_{0:n}}\big(g_n^* \,|\, g_{1:n-1}^*, \phi_{0:n}, \psi_{0:n}, u_{0:n}, v_{0:n}\big). \tag{S11}$$

All the densities in (S5–S11) may additionally depend on a parameter vector θ.

Algorithm S-1: GenSMC

input: dynamic model simulators listed in (S5–S9) and observation model evaluators (S10, S11); sequences, $g^*_{1:n^*}$; observation times $t^*_{1:n^*}$; initial time, t^*_0; terminal time, $t^*_{n^*+1} = t_{\text{end}}$; number of particles, J; number of hierarchical samples, K, L, M.

simulate $\big(\Phi^F_{0,j}, \Psi^F_{0,j}\big) \sim f_{\Phi_0,\Psi_0}(\phi_0, \psi_0)$ and set $U^F_{0,j} = V^F_{0,j} = 0$

for n in $1:n^*$ **do**

  propose transmission process: $\Phi^P_{n,j}(t) \sim f_{\Phi_n|\Phi_{n-1},\Psi_{n-1},D_n}\big(\phi_n|\Phi^F_{n-1,j}, \Psi^F_{n-1,j}, d^*_n\big)$

  set $w_1(n,j) = \exp\big\{-\int_{t^*_{n-1}}^{t^*_n} \rho\big(\Phi^P_{n,j}(t), \Psi^P_{n-1,j}(t)\big)\, dt\big\}\, \rho\big(\Phi^P_{n,j}(t^*_n), \Psi^P_{n-1,j}(t^*_n)\big)$

  set $\Phi^P_{0:n,j} = (\Phi^F_{0:n-1,j}, \Phi^P_{n,j})$

  propose attachment site: $\Psi^P_{n,jk} \sim f_{\Psi_n|\Phi_n,\Psi_{n-1}}\big(\psi_n|\Phi^P_{n,j}, \Psi^F_{n-1,j}\big)$

  set $\Psi^P_{0:n,jk} = (\Psi^F_{0:n-1,j}, \Psi^P_{n,jk})$

  evolution on the new branch: $U^P_{n,jkl} \sim f_{U_n|U_{n-1},V_{n-1},\Phi_{0:n},\Psi_{0:n}}\big(u_n|U^F_{n-1,j}, V^F_{n-1,j}, \Phi^P_{0:n,j}, \Psi^P_{0:n,jk}\big)$

  set $U^P_{0:n,jkl} = (U^F_{0:n-1,j}, U^P_{n,jkl})$

  evolution on the split branch: $V^P_{n,jklm} \sim f_{V_n|V_{n-1},U_n,\Phi_{0:n},\Psi_{0:n}}\big(v_n|V^F_{n-1,j}, U^P_{n,j}, \Phi^P_{0:n,j}, \Psi^P_{0:n,jk}\big)$

  set $V^P_{0:n,jklm} = (V^F_{0:n-1,j}, V^P_{n,jklm})$

  set $w_2(n,j,k,l,m) = f_{G_n|G_{1:n-1},\Phi_{0:n},\Psi_{0:n},U_{0:n},V_{0:n}}\big(g^*_n \,|\, g^*_{1:n-1}, \Phi^P_{0:n,j}, \Psi^P_{0:n,jk}, U^P_{0:n,jkl}, V^P_{0:n,jklm}\big)$

  weights: $w(n,j,k,l,m) = w_1(n,j)\, w_2(n,j,k,l,m)$

  set $w(n,j,k) = (1/LM) \sum_{l=1}^{L} \sum_{m=1}^{M} w(n,j,k,l,m)$

  resample: select index $(l',m')(j,k)$ with probability $\frac{w(n,j,k,l,m)}{w(n,j,k)}$

  set $w(n,j) = (1/K) \sum_{k=1}^{K} w(n,j,k)$

  resample: select index $k'(j)$ with probability $\frac{w(n,j,k)}{w(n,j)}$

  set $w(n) = (1/J) \sum_{j=1}^{J} w(n,j)$

  resample: select indices $j'(j)$ with probability $\frac{w(n,j)}{w(n)}$

  set $\Phi^F_{0:n,j} = \Phi^P_{0:n,j'(j)}$ and $\Psi^F_{0:n,j} = \Psi^P_{0:n,j'(j)\,k'(j')}$

  set $U^F_{0:n,j} = U^P_{0:n,j'(j)\,k'(j')\,l'(j',k')\,m'(j',k')}$ and $V^F_{0:n,j} = V^P_{0:n,j'(j)\,k'(j')\,l'(j',k')\,m'(j',k')}$

  conditional log likelihood estimate: $\hat{\ell}_{n|1:n-1} = \log w(n)$

end

simulate $\Phi^P_{n^*+1,j}(t) \sim f_{\Phi_{n^*+1}|\Phi_{n^*},\Psi_{n^*},D_{n^*+1}}\big(\phi(t)|\Phi^F_{n^*,j}, \Psi^F_{n^*,j}, d^*_{n^*+1}\big)$

set $w(n^*{+}1, j) = \exp\big\{-\int_{t^*_{n^*}}^{t_{\text{end}}} \rho\big(\Phi^P_{n^*+1,j}(t)\big)\, dt\big\}$

conditional log likelihood: $\hat{\ell}_{n^*+1|1:n^*} = \log\big\{(1/J) \sum_{j=1}^{J} w(n^*{+}1, j)\big\}$

output: log likelihood estimate: $\hat{\ell} = \sum_{n=1}^{n^*+1} \hat{\ell}_{n|1:n-1}$, and filtered state estimates

complexity: $\mathcal{O}(J\,K\,L\,M\,n\,\log n)$, assuming the transmission forest is balanced

S2.1 The implementation of GenSMC in the `genPomp` program

Many computational issues arise in effective implementation of a GenSMC method such as Algorithm S-1. Data structures are needed to keep track of the individuals in the study population, and the genetic relationships between pathogens in different hosts. Efficient implementation of all these computations, including use of a multi-processor computing environment, is necessary to work on problems of a practical scientific scale. The record of our implementation is the open-souce code for the `genPomp` program that we developed to carry out inference for GenPOMP models, available at `https://github.com/kingaa/genpomp`. The accuracy of `genPomp` has been successfully tested against exact analytic calculations for some very small scale situations, and against the `pomp` package (King et al., 2016) for situations where no diagnoses lead to genetic sequences.

There is a substantial difference in the level of abstraction between the formal mathematical representation of a GenPOMP model in Algorithm S-1 and the practical implementation in `genPomp`. One could write more pseudocode to bridge this gap, but that is beyond the scope of this article. We have focused instead on the foundational task of understanding how Algorithm S-1 fits in with the theory and practice of SMC.

S2.2 Extending GenSMC to infer unknown parameters: The GenIF algorithm

Sequential Monte Carlo (SMC) algorithms such as Algorithm S-1 produce a Monte Carlo approximation to the likelihood of the model, but do not directly provide estimates of unknown parameters. A substantial literature has emerged on using SMC as a basis for statistical inference (Kantas et al., 2015). Iterated filtering (Ionides et al., 2006, 2015) uses SMC, together with parameter perturbations, to maximize the likelihood function. Iterated filtering has demonstrated effectiveness on various nonlinear models arising in infectious disease transmission studies (Ionides et al., 2015, and references therein). We developed an adaptation of Algorithm IF2 of Ionides et al. (2015), which we call GenIF as an abbreviation of *iterated filtering for GenPOMP models*. Our implementation of this GenIF algorithm is included within the `genPomp` program, as described fully in the source code. Conceptually, and computationally, GenIF is a simple extension to GenSMC. GenIF carries out multiple iterations of Algorithm S-1 (GenSMC) adding perturbations to the candidate values of unknown parameters. GenSMC selects particles consistent with the data, and so allowing particles to have diversity in their parameters values naturally selects for parameter values consistent with the data. The theory and practice of iterated filtering focuses on using this phenomenon, with multiple SMC iterations having perturbations of decreasing magnitude, to maximize the likelihood. Previous iterated filtering theory does not encompass the just-in-time variables employed by GenSMC. In the context of GenIF, this means that the current theoretical justification of IF2 (Ionides et al., 2015) does not perfectly apply when we carry out inference for the molecular evolution parameters. Heuristically, however, the principle of iterated filtering still applies, and we rely on empirical results to confirm that maximization performance is satisfactory.

Algorithms that permit numerically satisfactory likelihood maximization and likelihood evaluation provide a foundation for carrying out likelihood-based statistical inference. Profile likelihood meth-

ods can be used for obtaining confidence intervals, and likelihood ratio tests or Akaike's information criterion can be used for model selection.

S2.3 Scalability of GenSMC

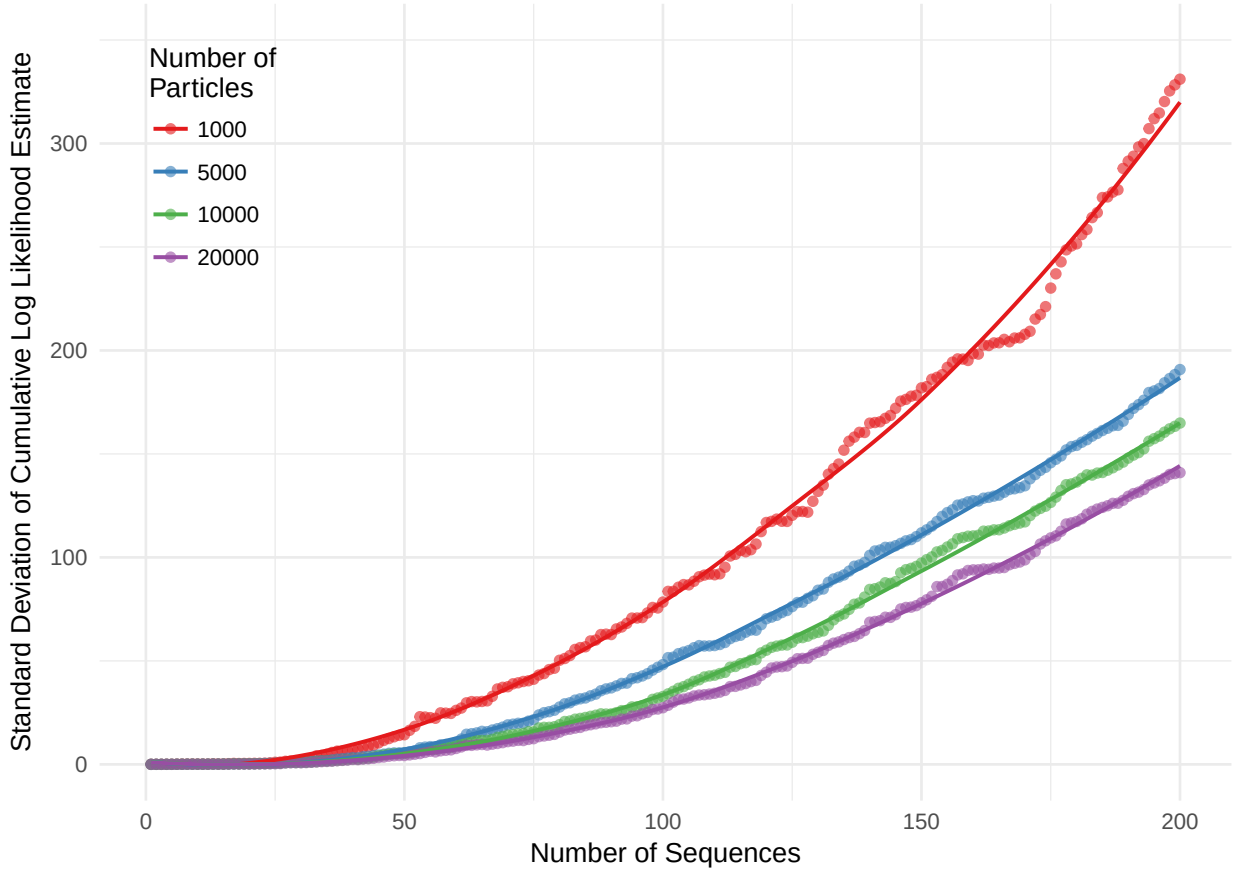

Figure S-3: Results from an experiment exploring how the standard deviation of the log likelihood estimate scales with both the number of sequences fit and the number of particles used. We ran the particle filter at the truth 80 times for each number of particles (1000, 5000, 10,000 and 20,000) on a simulated dataset of 200 sequences.

To explore the scalability of our GenSMC implementation, we performed the following experiments using simulated data. We first simulated an epidemic conditional on observing 200 sequences. We then ran the particle filter at the truth using 1000, 5000, 10,000, and 20,000 particles. For each number of particles we used we ran 80 particle filters. Finally, for each sequence, we computed the standard deviation of the cumulative log likelihood estimate across the 80 filtering evaluations. This computation yields a measure of the variability in the log likelihood estimate if one were to

stop filtering at each sequence. The results from this experiment provide a controlled assessment of how Monte Carlo variance scales as the number of sequences grows. The standard deviation of the log likelihood estimate remains relatively small up to around 25 sequences (Figure S-3). An interpretation of this is that placing early sequences on the growing phylogenetic tree is relatively easy. It can become harder to find trees with appropriate places to attach later sequences, leading to increasing Monte Carlo variance. Monte Carlo variance is expected to grow as the size of a computational problem increases, but we did not find a rapid exponential growth. The peeling algorithm for computing the likelihood of the genetic sequences conditional on the phylogeny was typically the largest computational component, though not for all regions of parameter space.

S3 A theoretical derivation of the GenSMC algorithm

To derive and justify GenSMC (Algorithm S-1) for the GenPOMP model, we work up in stages from a simple and standard SMC algorithm. Initially working in discrete time, we start in Section S3.1 by writing an SMC algorithm that allows for general dependence between the latent process and the observation process. Then, we consider a useful class of targeted proposal distributions in Section S3.2. We add hierarchical layers of resampling in Section S3.3. In Section S3.4, we consider a *just-in-time* approach to construction of state variables which can have their creation postponed until necessary. In Section S3.5, we move these developments into the context of continuous time models. Putting these components together, we obtain Algorithm S-1.

S3.1 A basic SMC algorithm

Consider a model consisting of a latent stochastic process $X_{0:N} = (X_0, X_1, \ldots, X_N)$ and an observable process $Y_{1:N} = (Y_1, Y_2, \ldots, Y_N)$. In this setting, N corresponds to the number of discrete time points, differing from the notation of Section S1. Data consist of a sequence $y^*_{1:N} \in \mathbb{Y}^N$, modeled as a realization of $Y_{1:N}$. We suppose X_n and Y_n take values in measurable spaces $\mathbb{X}$ and $\mathbb{Y}$, and we require the existence of a joint density $f_{X_{0:N},Y_{1:N}}$ on $\mathbb{X}^{N+1} \times \mathbb{Y}^N$. Conditional densities are denoted using subscripts, for example, the density of Y_n given $Y_{1:n-1}$ and $X_{0:n}$ is written as

$$f_{Y_n|X_{0:n},Y_{1:n-1}}(y_n \,|\, x_{0:n}, y_{1:n-1}). \tag{S12}$$

In a standard POMP model, $\{X_n\}$ is a latent Markov process and the conditional distribution of Y_n depends only on X_n (Bretó et al., 2009). In the context of GenPOMP, we require the marginal Markov property for the latent process,

$$f_{X_n|X_{0:n-1}}(x_n \,|\, x_{0:n-1}) = f_{X_n|X_{n-1}}(x_n \,|\, x_{n-1}). \tag{S13}$$

but we allow a general form for the measurement model in equation (S12), where the conditional distribution of the nth observation can depend on the entire histories of the latent process and the observation process. SMC techniques for POMP models can be extended to this more general dependence structure (Liu, 2001). A basic SMC algorithm is outlined in Algorithm S-2. This is essentially the basic bootstrap filter algorithm of Gordon et al. (1993), generalized to allow for

the dependence on the history of the process in (S12). Notationally, for Algorithm S-2 we set $\mathscr{X}_n = X_{0:n}$ and use superscripts F and P to denote particles representing the filtering and prediction distributions respectively. We use systematic resampling in place of multinomial resampling (Arulampalam et al., 2002; Douc et al., 2005).

Algorithm S-2: A basic Sequential Monte Carlo (SMC) algorithm:

input: simulator for $f_{X_n|X_{n-1}}(x_n \,|\, x_{n-1})$; simulator for $f_{X_0}(x_0)$; evaluator for $f_{Y_n|X_{1:n},Y_{1:n-1}}(y^*_n \,|\, x_{1:n}, y^*_{1:n-1})$; data, $y^*_{1:N}$; number of particles, J.

initialize filter particles: simulate $X^F_{0,j} \sim f_{X_0}(x_0)$ for j in $1:J$

initialize particle filter history: $\mathscr{X}^F_{0,j} = X^F_{0,j}$

for n in $1:N$ **do**

  prediction simulation: $X^P_{n,j} \sim f_{X_n|X_{n-1}}\big(x_n|X^F_{n-1,j}\big)$ for j in $1:J$.

  history of the prediction: $\mathscr{X}^P_{n,j} = \big(\mathscr{X}^F_{n-1,j}, X^P_{n,j}\big)$

  evaluate weights: $w(n,j) = f_{Y_n|X_{0:n},Y_{1:n-1}}(y^*_n|\mathscr{X}^P_{n,j}, y^*_{1:n-1})$ for j in $1:J$

  normalize weights: $\tilde{w}(n,j) = w(n,j)/\sum_{m=1}^J w(n,m)$

  apply systematic sampling to select indices $k_{1:J}$ with $\mathbb{P}\{k_j = m\} = \tilde{w}(n,m)$.

  resample: set $X^F_{n,j} = X^P_{n,k_j}$ and $\mathscr{X}^F_{n,j} = \mathscr{X}^P_{n,k_j}$ for j in $1:J$

  estimate conditional log likelihood: $\hat{\ell}_{n|1:n-1} = \log\big(J^{-1}\sum_{m=1}^J w(n,m)\big)$

end

output: log likelihood estimate, $\hat{\ell} = \sum_{n=1}^N \hat{\ell}_{n|1:n-1}$; filter sample, $\mathscr{X}^F_{n,1:J}$, for n in $0:N$.

Computational resources are an issue for GenPOMP models, since the spaces $\mathbb{X}$ and $\mathbb{Y}$ are both large. Furthermore, the dependence on the history in (S12) leads to additional computational requirements for both memory and numerical operations. Careful implementation of SMC is therefore necessary to make the approach practical. We therefore proceed to develop extensions of Algorithm S-2 that are necessary to improve numerical tractability for GenPOMP models.

To understand Algorithm S-2, and subsequently extend it, we write out an algebraic justification of the prediction and filtering steps. For a general latent process $X_{0:N}$ and observable process $Y_{1:N}$ modeling data $y^*_{1:N}$ collected at times $t_{1:N}$, assuming (S12) and (S13), the *prediction identity* is

$$f_{X_{0:n}|Y_{1:n-1}}(x_{0:n} \,|\, y^*_{1:n-1}) = f_{X_n|X_{n-1}}(x_n \,|\, x_{n-1}) f_{X_{0:n-1}|Y_{1:n-1}}(x_{0:n-1} \,|\, y^*_{1:n-1}) \tag{S14}$$

The SMC interpretation of (S14) is that $f_{X_{0:n-1}|Y_{1:n-1}}(x_{0:n-1} \,|\, y^*_{1:n-1})$ is represented by a collection of J filter particles $\mathscr{X}^{F,j}_{n-1}$, $j = 1, \ldots, J$. Algorithm S-2 corresponds to a basic version of SMC in which particle j has a time t_n value generated from $f_{X_n|X_{n-1}}(x_n \,|\, X^F_{n-1,j})$ to give rise to a time t_n prediction particle $X^P_{n,j}$. $X^P_{n,j}$ inherits its history from $X^F_{n-1,j}$ and so $\mathscr{X}^P_{n,j} = \big(\mathscr{X}^F_{n-1,j}, X^P_{n,j}\big)$.

A general *filtering identity* is

$$f_{X_{0:n}|Y_{1:n}}(x_{0:n} \,|\, y^*_{1:n}) = \left[\frac{f_{Y_n|X_{0:n},Y_{1:n-1}}(y^*_n \,|\, x_{0:n}, y^*_{1:n-1})}{f_{Y_n|Y_{1:n-1}}(y^*_n \,|\, y^*_{1:n-1})}\right] f_{X_{0:n}|Y_{1:n-1}}(x_{0:n} \,|\, y^*_{1:n-1}). \tag{S15}$$

The SMC interpretation of (S15) is that observation y^*_n requires the prediction particle $\mathscr{X}^P_{n,j}$ representing $f_{X_{0:n}|Y_{1:n-1}}(x_{0:n} \,|\, y^*_{1:n-1})$ to be given a weight proportional to $f_{Y_n|X_{0:n},Y_{1:n-1}}(y^*_n \,|\, \mathscr{X}^P_{n,j}, y^*_{1:n-1})$.

The denominator on the right hand side of (S15) is an irrelevant constant for computing the normalized weights. However, this denominator is approximated in Algorithm S-2 as the normalizing constant, giving a Monte Carlo estimate of the nth term in a factorization of the likelihood of the data,

$$f_{Y_{1:N}}(y^*_{1:N}) = f_{Y_0}(y^*_0) \prod_{n=1}^{N} f_{Y_n|Y_{1:n-1}}(y^*_n \,|\, y^*_{1:n-1}). \tag{S16}$$

For a discrete time representation of a simple GenPOMP model, Algorithm S-2 might be directly applicable. For example one can take X_n to correspond to all the information about individuals in the population at time n, so that $X_{0:n}$ includes the transmission tree. We could also suppose that $X_{0:n}$ includes information on who would get sequenced if there are observed sequences—but not how many sequences were observed, which is part of the measurement. For example, at each time point t_n, the state could contain a permutation listing the order in which eligible individuals are sequenced. This construction may appear somewhat contrived, and we proceed to relax it by allowing part of the latent process to be fully observed and therefore also be part of the measurement process. Regardless of that issue, evaluation of $f_{Y_n|X_{0:n},Y_{1:n-1}}(y^*_n|x_{0:n}, y^*_{1:n-1})$ involves evaluating the likelihood of a phylogeny, which can be computed efficiently by a peeling algorithm, together with term for the probability of the sequence being collected.

S3.2 A targeted SMC approach with a partial plug-and-play property

Some models of interest may have the feature that the event of obtaining a measurement has an appreciable consequence for the latent dynamics. HIV, for example, has the features that sequencing of the pathogen typically occurs at diagnosis. The fraction of infections which are sequenced is high, and diagnosis plays an important role in transmission dynamics both through changes in sexual contact behavior and reduced infectivity due to antiviral drugs. For HIV, it is therefore natural to consider models where sequencing events correspond to transitions of an individual between states and therefore correspond to a perfectly observed component of the latent process. This kind of situation needs some extra care, since $f_{Y_n|X_{0:n},Y_{1:n-1}}(y^*_n|\mathscr{X}^P_{n,j}, y^*_{1:n-1})$ in Algorithm S-2 becomes zero for every draw of $\mathscr{X}^P_{n,j}$ which is not consistent with y^*_n. The standard SMC approach to this is to allow for the possibility of targeted SMC proposal distributions, not necessarily the "vanilla" choice $f_{X_n|X_{n-1}}$. Suppose the proposal distribution for the SMC algorithm is $q_n(x_n|x_{n-1}, y^*_n)$, which is permissible since the proposal distribution is in general allowed to depend on any past, current or future observations. This corresponds to rewriting (S14) as

$$f_{X_{0:n}|Y_{1:n-1}}(x_{0:n} \,|\, y^*_{1:n-1}) = \left[\frac{f_{X_n|X_{n-1}}(x_n \,|\, x_{n-1})}{q_n(x_n|x_{n-1}, y^*_n)}\right] q_n(x_n|x_{n-1}, y^*_n) f_{X_{0:n-1}|Y_{1:n-1}}(x_{0:n-1} \,|\, y^*_{1:n-1}), \tag{S17}$$

which is interpreted to mean that the targeted SMC proposal particle $\mathscr{X}^P_{n,j}$, with $X^P_{n,j}$ drawn from $q_n(x_n|X^F_{n-1,j}, y^*_n)$, must be given a weight $f_{X_n|X_{n-1}}(X^P_{n,j} \,|\, X^F_{n-1,j})\{q_n(X^P_{n,j} \,|\, X^F_{n-1,j}, y^*_n)\}^{-1}$ corresponding to the ratio in square brackets in (S17).

A special case of a targeted proposal arises in the situation where part of the state variable is perfectly observed. To describe this situation, suppose we can partition the latent and observable

processes as,

$$X_n = (A_n, B_n), \tag{S18}$$
$$Y_n = (B_n, C_n), \tag{S19}$$

with the data being $(b^*_{1:N}, c^*_{1:N})$. The prediction identity in (S17) can then be written as

$$\begin{aligned} &f_{A_n,B_n,X_{0:n-1}|Y_{1:n-1}}(a_n, b^*_n, x_{0:n-1} \,|\, y^*_{1:n-1}) \\ &= \left[\frac{f_{A_n,B_n|X_{n-1}}(a_n, b^*_n \,|\, x_{n-1})}{q_n(a_n|x_{n-1}, y^*_n)}\right] q_n(a_n|x_{n-1}, y^*_n) f_{X_{0:n-1}|Y_{1:n-1}}(x_{0:n-1} \,|\, y^*_{1:n-1}). \end{aligned} \tag{S20}$$

Then, to obtain the filtering distribution $f_{A_n,X_{1:n-1}|B_n,Y_{1:n-1}}(a_n, x_{0:n-1} \,|\, b^*_n, y^*_{1:n-1})$ one normalizes the weighted particle representation of $f_{A_n,B_n,X_{0:n-1}|Y_{1:n-1}}(a_n, b^*_n, x_{1:n-1}|y^*_{1:n-1})$ in (S20), with the normalizing constant being the conditional likelihood, $f_{B_n|Y_{1:n-1}}(b^*_n|y^*_{1:n-1})$. A particular target choice of interest in (S20) is

$$q_n(a_n|x_{n-1}, y^*_n) = f_{A_n|X_{n-1}}(a_n \,|\, x_{n-1}). \tag{S21}$$

(S20) becomes

$$\begin{aligned} &f_{A_n,B_n,X_{0:n-1}|Y_{1:n-1}}(a_n, b^*_n, x_{0:n-1} \,|\, y^*_{1:n-1}) \\ &= \left[f_{B_n|A_n,X_{n-1}}(b^*_n|a_n, x_{n-1})\right] f_{A_n|X_{n-1}}(a_n \,|\, x_{n-1}) f_{X_{0:n-1}|Y_{1:n-1}}(x_{0:n-1} \,|\, y^*_{1:n-1}). \end{aligned} \tag{S22}$$

On the component of the state space that is not perfectly observed, the proposal in (S21) is *plug-and-play* (Bretó et al., 2009; He et al., 2010) meaning that the algorithm needs only a simulation from $f_{A_n|X_{n-1}}(a_n \,|\, x_{n-1})$. However, we require numerically tractable evaluation of the importance sampling weight

$$f_{B_n|A_n,X_{n-1}}(b^*_n|a_n, x_{n-1}),$$

arising from the identity (S22), and so we describe the algorithm as *partially plug and play*.

Using a targeted proposal typically leads to algorithms without the plug-and-play property. Here, we work with situations where $f_{B_n|A_n,X_{n-1}}(b^*_n|a_n, x_{n-1})$ is tractable, even if the complete transition density of (A_n, B_n) is intractable. Thus, $f_{A_n|X_{n-1}}(a_n \,|\, x_{n-1})$ can be specified in a fairly arbitrary way.

Example 1. B_n might be the number of diagnoses at time n, which might have a Poisson or negative binomial distribution conditional on A_n.

Example 2. Writing the number of sequenced diagnoses at time n by D^S_n, unsequenced diagnoses by D^U_n, and infected individuals by I_n, we might have $B_n = (D^S_n, D^U)$ and $A_n = I_n$. The joint distribution of D^S_n, D^U_n and $I_n - D^S_n - D^U_n$ might be multinomial given I_n.

Example 3. B_n might describe the race or age group of diagnosed individuals as well as whether they were sequenced.

S3.3 SMC with hierarchical sampling

For computational considerations, it may be preferable to maintain J filtering particles and generate K prediction particles from each, rather than maintaining JK filtering particles. Computation of the K prediction particles can be localized on a single core of multi-processor hardware, and the memory usage may scale with J rather than JK.

In the context of Algorithm S-2, extended to include the general proposal distribution of Section S3.2, we write $\{X^P_{n,jk}, k \in 1\!:\!K\}$ for K draws from $q_n(x_n|X^F_{n-1,j}, y^*_n)$ for each value of j. We compute the weights in the second layer of the hierarchy by

$$w_{n,jk} = f_{X_n|X_{n-1}}\big(X^P_{n,jk} \,|\, X^F_{n-1,j}\big)\Big[q_n\big(X^P_{n,j} \,|\, X^F_{n-1,j}, y^*_n\big)\Big]^{-1}.$$

We then define $X^F_{n,j}$ to be a draw from $\{X^P_{n,jk}, k \in 1\!:\!K\}$ with probability proportional to $w_{n,jk}$, with the history $\mathscr{X}^F_{n,j}$ being constructed accordingly. We then assign $\mathscr{X}^F_{n,j}$ a weight

$$w_{n,j} = \frac{1}{K}\sum_{k=1}^{K} w_{n,jk}. \tag{S23}$$

The filter particles $\{\mathscr{X}^F_{n,j}, j = 1, \ldots, J\}$ can be again resampled with weight proportional to $w_{n,j}$ if so desired. Resampling each layer of the hierarchy one at a time gives an approach that we call *staggered resampling*. It might sometimes be preferable to resample J particles from all JK particles $\{X^P_{n,jk}, j = 1\!:\!J, k = 1\!:\!K\}$ with weights $w_{n,jk}$. This process, resampling two or more layers of the hierarchy at the same time, we call *simultaneous resampling*. The staggered resampling in (S23) can have computational advantages in terms of memory: one never needs to keep all JK particles in memory simultaneously. Also, staggered resampling is convenient in a multi-processor computational environment, where the computations for the first layer of the hierarchy can be split across processors and the second layer can be computed without any need for communication between processors.

Another motivation for hierarchical sampling arises when one can separate the generation of the prediction particle into a computationally expensive step followed by a cheap step. Heuristically, if the particles are large and computationally expensive, one wants to ensure that a particle does not get culled due to a single unfortunate draw from a proposal distribution. A component of the proposal distribution that is computationally expensive but not critical for the particle weight should be carried out relatively few times. By contrast, a component of the proposal distribution that is computationally cheap but critical for the particle weight, and hence for the survival of the particle, should be carried out relatively many times. For this motivation, there may be no compelling reason to carry out staggered resampling, in which case simultaneous resampling should be preferred. Both hierarchical sampling possibilities can arise in different parts of a single algorithm, potentially giving rise to several layers of sampling and resampling.

SMC with hierarchical sampling fits within the general theory of SMC (Naesseth et al., 2015), and theory exists to guide a good sampling structure (Skinner et al., 1989). In practice, however, preliminary experimentation is a good guide. Hierarchical resampling receives diminishing returns

for increasing values of K, since since J is the basic Monte Carlo sample size which asymptotically justifies the Monte Carlo approach. Moderate values of $K > 1$ can have compelling practical advantages, which can be quantified by evaluating the variance of the Monte Carlo likelihood estimate.

S3.4 Just-in-time evaluation of some state variable components

In equation S3, our GenPOMP model included state processes $\{\Gamma(t)\}$ and $\{\Delta(t)\}$ which have no role in the dynamics, meaning that they do not affect the infinitesimal transition probabilities for $\{\Phi(t)\}$ and $\{\Psi(t)\}$ but do affect the measurements. If the measurements depend only on some subset or combination of these state variables, it is computationally desirable to generate the required subsets or combinations only when needed. Carrying out this computational shortcut, which we call just-in-time generation, does not change the model under consideration so long as the required variables are properly constructed at the time they become necessary. Two advantages to just-in-time state variable generation are

1. There may be state variables which, on some event of positive probability, have no effect on the measured components of the system. These state variables can be omitted when carrying out inferences on the rest of the system.
2. The sampling of these variables, and consequent resampling of particles, occurs only when information on the just-in-time variables arrives. In combination with hierarchical sampling (Section S3.3), trying multiple copies of the just-in-time variables for each particle can help to prevent particles being lost due to a single unfortunate draw of a random variable.

To formalize the definition of just-in-time variables, we suppose that X_n can be split into two parts, written as

$$X_n = (\Phi_n, \Upsilon_n).$$

We say that $\Xi_n = h_n(X_{0:n})$ gives a just-in-time representation of Υ_n if

$$f_{Y_n|Y_{1:n-1},X_{0:n}}(y_n \,|\, y_{1:n-1}, x_{0:n}) = f_{Y_n|Y_{1:n-1},\Phi_{0:n},\Xi_{0:n}}\big(y_n \,|\, y_{1:n-1}, \phi_{0:n}, \xi_{0:n}\big), \tag{S24}$$

where $\xi_n = h_n(x_{0:n})$. If we can evaluate (S24) and simulate draws from $f_{\Phi_n,\Xi_n|\Phi_{0:n-1},\Xi_{n-1}}$, then we can effectively replace Υ_n by Ξ_n in an SMC method such as Algorithm S-2. A particular case, arising in the just-in-time replacement of $\big(\Gamma(t), \Delta(t)\big)$ by $\big(U(t), V(t)\big)$ in Algorithm S-1, occurs when the dynamics of $\{\Phi_n\}$ do not depend on $\{\Upsilon_n\}$, i.e.,

$$f_{\Phi_n|X_{0:n-1}}(\phi_n \,|\, x_{0:n-1}) = f_{\Phi_n|\Phi_{0:n-1}}(\phi_n \,|\, \phi_{0:n-1}). \tag{S25}$$

In this case, implementing a just-in-time scheme requires that we can draw from $f_{\Xi_n|\Phi_{1:n},\Xi_{n-1}}$ and we can evaluate the density in equation (S24). In practice, Ξ_0 may be a trivial random variable, since there is no observation at t_0, but this is not necessary for the just-in-time construction.

The utility of just-in-time evaluation depends in part on the reduction of dimension in replacing Ξ_n by Υ_n. For example, nothing is gained by the just-in-time representation $\Xi_n = \Upsilon_n$.

S3.5 Moving from discrete time to continuous time

Continuous time Markov population models can be approximated in discrete time by a Markov chain (Bretó et al., 2009) using a stochastic Euler method. A continuous time measurement model can similarly be discretized to match the time steps of the Euler approximation. For a continuous time latent process model, suppose that $\{X(t), t \in \mathbb{T}\}$ is a right continuous stochastic process taking values in $\mathbb{X}$. We suppose that the continuous-time measurement process $\{Y(t)\}$ consists of a counting process, $\{D(t)\}$, together with a sequence of measurements $\{G_1, G_2, \dots\}$ where G_n occurs at time $T_n = \inf\{t : D(t) \geq n\}$. This notational setup is based on Section S1, but we no not require any of the additional structure of a GenPOMP model at this point. We write $t_1^* < t_2^* < \cdots < t_{n^*}^*$ for the observation times of the data, $g_{1:n^*}^*$. Here, we suppose that $D(t)$ is part of $X(t)$ and, specifically, is represented by the observed component $B(t)$ in the decomposition

$$X(t) = \big(A(t), B(t)\big)$$

corresponding to a continuous-time version of equation (S18). This situation arises in GenPOMP models when $\{D(t)\}$ counts diagnosis events for a disease transmission model $\{X(t)\}$, as in Section S1. Suppose that the rate of observation events at time t does not depend on the measurement process $\{Y_n : t_n \leq t\}$ given the current state process $X(t)$, i.e.,

$$\begin{aligned}
\mathbb{P}[D(t+\delta) - D(t) = 0 \,|\, X(t), \{Y_s, s \leq t\}] &= 1 - \rho(A(t))\,\delta + o(\delta), &\text{(S26)}\\
\mathbb{P}[D(t+\delta) - D(t) = 1 \,|\, X(t), \{Y_s, s \leq t\}] &= \rho(A(t))\,\delta + o(\delta). &\text{(S27)}
\end{aligned}$$

Then, dividing the interval $(t_{n-1}^*, t_n^*]$ into subintervals of width δ and taking $\delta \to 0$, the limit of discrete approximations using (S26) and (S27) corresponds to a combined weight from evaluating (S23) in each of the $1/\delta$ subintervals with no measurement followed by one subinterval with a measurement, i.e.,

$$\begin{aligned}
&\lim_{\delta \to 0} \left\{ \prod_{m=1}^{(t_n^* - t_{n-1}^*)/\delta} \Big(1 - \rho\big(A(t_{n-1}^* + m\delta)\big)\Big) \right\} \times \rho\big(A(t_k^*)\big) f_{G_k|G_{1:n-1},T_{1:n},X_{0:n}}(g_n^* \,|\, g_{1:n-1}^*, t_{1:n}^*, x_{0:n})\\
&\quad = \exp\left\{ -\int_{t_{n-1}^*}^{t_n^*} \rho\big(A(s)\big)\, ds \right\} \times \rho\big(A(t_n^*)\big) \times f_{G_k|G_{1:n-1},T_{1:n},X_{0:n}}(g_n^* \,|\, g_{1:n-1}^*, t_{1:n}^*, x_{0:n}). \qquad \text{(S28)}
\end{aligned}$$

Note that one can view the first two terms of the product in equation (S28) as a density with respect to Poisson counting measure.

S4 Details of the HIV model used in the main text

In this section we provide additional details that describe the HIV model used in the main text. As the system is Markovian, we can fully specify the model by defining probabilities of each possible change to the state of the system given the current state over an interval of time δ. There are three types of events that change the state of system, each in a fundamentally different way:

1. An individual changes class. This event modifies an existing lineage on a transmission tree.
2. An individual in the study population infects a new individual. This event adds a new lineage to an existing transmission tree.
3. An individual outside the study population infects a new individual. This event seeds a new transmission tree consisting of a single individual. The genetic tree associated with with this new transmission tree joins all other genetic trees at the polytomy.

We define probabilities for the first two types of events from an individual-based perspective. Recall that the state of any individual i at time t is given by a random process $\{X_i(t)\}$. The probabilities of class changes for each individual over an interval of time δ are given by

$$\begin{aligned}
\mathbb{P}\big[X_i(t+\delta) = I_1 \mid X_i(t) = I_0\big] &= \delta\gamma_{I_0} + o(\delta),\\
\mathbb{P}\big[X_i(t+\delta) = J_0 \mid X_i(t) = I_0\big] &= \delta\rho_{_0} + o(\delta),\\
\mathbb{P}\big[X_i(t+\delta) = I_2 \mid X_i(t) = I_1\big] &= \delta\gamma_{I_1} + o(\delta),\\
\mathbb{P}\big[X_i(t+\delta) = J_1 \mid X_i(t) = I_1\big] &= \delta\rho_{_1} + o(\delta),\\
\mathbb{P}\big[X_i(t+\delta) = J_2 \mid X_i(t) = I_2\big] &= \delta\rho_{_2} + o(\delta),\\
\mathbb{P}\big[X_i(t+\delta) = J_1 \mid X_i(t) = J_0\big] &= \delta\gamma_{J_0} + o(\delta),\\
\mathbb{P}\big[X_i(t+\delta) = J_2 \mid X_i(t) = J_1\big] &= \delta\gamma_{J_1} + o(\delta),\\
\mathbb{P}\big[X_i(t+\delta) = \odot \mid X_i(t) = s\big] &= \delta(\mu_s + \phi) + o(\delta).
\end{aligned} \tag{S29}$$

Above, μ_s is a state-dependent death rate for an individual in state $s \in \mathbb{S} = \{I_0, I_1, I_2, J_0, J_1, J_2\}$, $X_i(t) = \odot$ if individual i is not in the study population at time t, and ϕ is a constant rate of emigration from the study population. The probability that an infected individual from inside the population gives rise to a new infection is,

$$\mathbb{P}\big[\text{the i}^{th} \text{ individual infects a new individual in } [t, t+\delta] |\ X_i(t) = s\big] = \delta\varepsilon_s + o(\delta),$$

where ε_s is the infectiousness of an individual in state s. The probability that an infected individual from outside the population gives rise to a new infection is,

$$\mathbb{P}\big[\text{an infection occurs from outside the study population in } [t, t+\delta]\big] = \delta\psi + o(\delta).$$

Note that this last probability, in contrast to those before, is not defined on a per capita basis. Also note that all new infections start in class I_0; this model does not allow immigration of later stage infected (or diagnosed) individuals into the population.

This model closely resembles a model from a recent phylodynamic analysis of the Detroit HIV epidemic Volz et al. (2013), but differs in key ways. First, whereas Volz et al. (2013) modeled incidence as a smooth, deterministic function, we model incidence mechanistically as a function of the states of individuals in the system. Second, instead of using a system of deterministic ordinary differential equations to model counts of individuals in each state, our model incorporates stochasticity into the process of state transitions.

S4.1 Initial values for the HIV model

The *initial value* for a GenPOMP model is $X(t_0)$. In general, the initial value can be treated as an unknown parameter vector which can be estimated using our GenIF methodology. There may be only limited information about these parameters in the data, but that is not a major problem for constructing profile likelihood estimates on other parameters of interest. However, a more parsimonious modeling approach is to set $X(t_0)$ to be a suitable function of the values of the dynamic parameters. For example, under a stationarity assumption for the dynamic system, one might set $X(t_0)$ to be a random draw from the stationary distribution or some mean value approximation to this. Our HIV model is not stationary, since we follow an age-cohort, but nevertheless we decided to initialize at plausible values given the dynamic parameters rather than estimate additional parameters. Further investigation could relax this assumption.

Part of the specification of $X(t_0)$ involves determining the genetic relationship assumed between infections that do not occur in the study population during the modeled period. The time t_0 at which we start modeling the population does not have to match the time at which we start to observe it. We could, for example, have zero sequencing probability before some time point. However, for our HIV model, these two times coincide. In the context of this HIV model, this component of the initial value involves determining the depth of the assumed polytomy, quantified by the time $t_{\text{root}} < t_0$ at which all trees in the transmission forest are modeled as meeting in the phylogenetic tree.

We carried out the following construction of the initial values of the membership of each compartment. We first note that the total number of diagnosed individuals is a perfectly observed quantity. By selecting a cohort, we have the advantage of working with a well-defined subpopulation. Over the time period from 2000 to 2012 we know exactly how many individuals were diagnosed. The MDCH dataset only has gene sequences between 2004 and 2012, so we decided to set $t_0 = 2004$. By 2004, the cohort grew to have 42 diagnosed individuals. Our aim in specifying initial counts is to apportion these 42 individuals to the three different classes of diagnosed individuals and populate the three unobserved states (the undiagnosed individuals) with counts. We assume no deaths over this period of four years. We constructed initial counts for each class by calculating under some additional assumptions under which these values become numerically tractable. First, we made the approximation that all rates of flow, with the exception of $h(t)$, are fixed at a current parameter estimate. Further, we suppose that $h(t)$ is constant at some fixed value,

$$h(t) = h_0,$$

ignoring the dependence of $h(t)$ on the state of the system. We then approximate the initial state by setting up and solving differential equations representing a deterministic solution to the model equation, formally equivalent to requiring the system of equations (S29) to hold in expectation. We fixed all rates of flow except $h(t_0)$ as described in the main text. Then, if the study cohort begins with all counts at zero in 2000, there is only one possible h_0 for which the total number of diagnoses in this approximating model matches the observed total number of diagnoses. We then solve for this value of h_0 and in doing so we obtain the counts in each compartment. Trajectories for the six states and their final values after four years are shown in Figure S4.1. This approach to setting initial counts is not self-consistent with the model, as the model assumes that the rate of

new infections is dependent on the state of the system, or with the timing of diagnoses observed in the four years leading up to the start of filtering. This simple way of setting the initial conditions is a starting point. Exploring the effect of initial conditions on model fits could be an area of future work.

We treated the time of the polytomy as an initial value parameter, with each particle starting with its own polytomy time. In this way, the polytomy time fits naturally into the iterated filtering maximization routines.

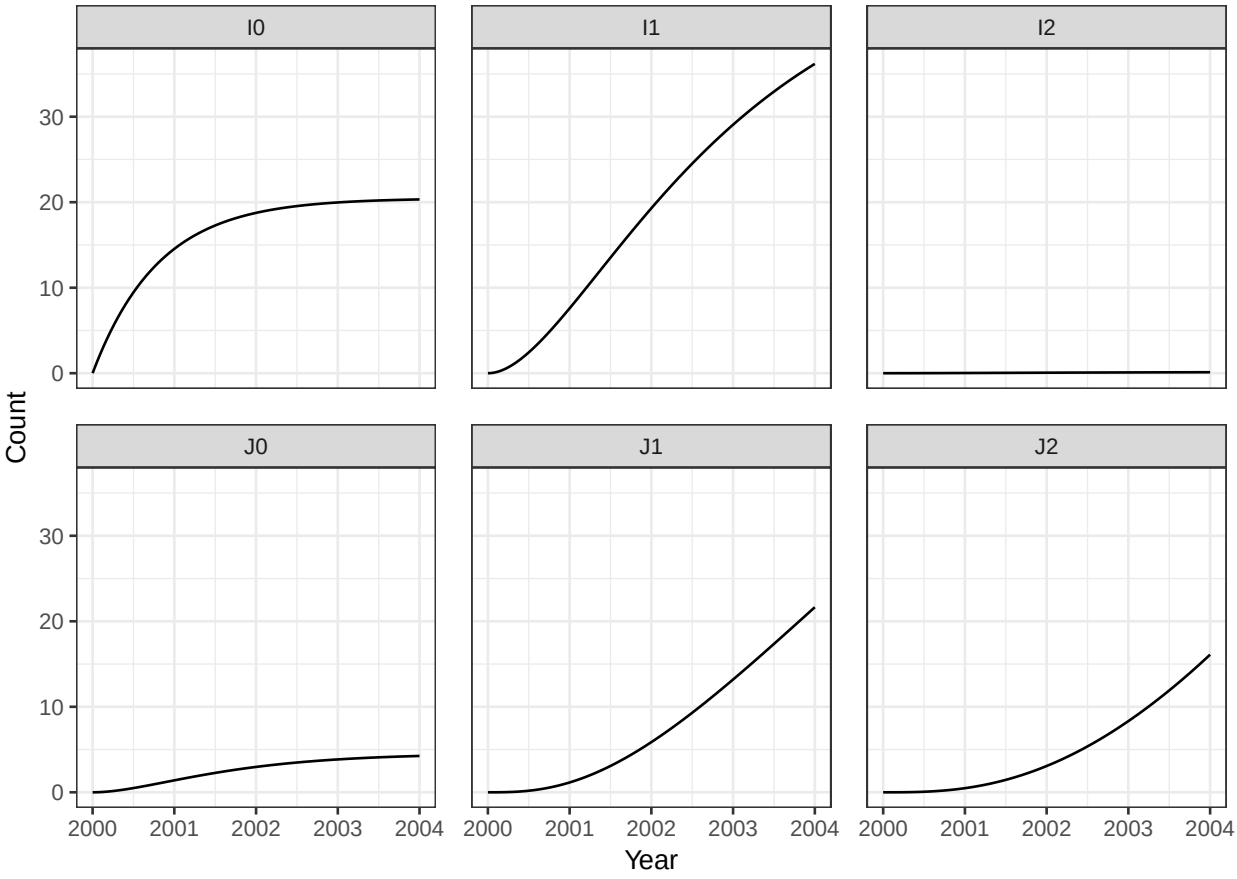

Figure S-4: Trajectories of counts of each class of infected individuals over four years prior to $t_0 = 2004$ when assuming a constant rate of new infections, all flows between and out of compartments as specified in the main text, and zero individuals initially in the cohort. We used the resulting counts in 2004 as the initial values for the data analysis.

S4.2 Algorithmic parameters used for the numerical results

The choice of algorithmic parameters can affect the numerical efficiency of the GenSMC and GenIF algorithms. For large computations, when Monte Carlo variability is an appreciable component of parameter uncertainty, this can have an effect on the quality of the resulting statistical inferences. In Table S-1 we supply the algorithmic parameters that we used in the simulation study (for GenSMC) and in the data analysis (for both GenIF and GenSMC). We selected J, K, L and M such that Monte Carlo uncertainty on parameter estimates and confidence intervals was tolerable (Ionides et al., 2016) and such that runtimes were not prohibitively long.

Three of the algorithmic parameters are only used in GenIF: the random walk standard deviation, σ_{rw}, the cooling factor, α_c, and the number of GenIF iterations, I. Together, these parameters determine the extent to which GenIF shrinks the diameter of the parameter swarm. In the GenIF algorithm, perturbation of parameters over which we are maximizing occurs for each particle just before the proposal step. We perturb the parameters by multiplying each by a random deviate from a log normal distribution with mean one and standard deviation $\sigma_{rw}\alpha_c^i$, where $i \in \{0, 1, ..., I-1\}$ is the iteration of GenIF. This choice of perturbation is appropriate for nonnegative parameters. Although our framework allows for a different random walk standard deviation for each parameter, in this case we found that the same random walk standard deviation for all parameters was effective, and we report this value in Table S-1.

The algorithmic parameters in Table S-1 together with the source code at `https://github.com/kingaa/genpomp` are sufficient to reproduce the methodology we apply in our analysis. The HIV sequence data we analyzed are not publicly available, in accordance with our data use agreement with Michigan Department of Community Health.

Table S-1: Algorithmic parameters used in the simulation study and the data analysis.

Algorithmic parameter	**Description**	**Simulation Study**		**Data Analysis**			
		GenSMC		**GenIF**		**GenSMC**	
		Diagnosis data only	**Diagnosis data and genetic sequences**	**Diagnosis data only**	**Diagnosis data and genetic sequences**	**Diagnosis data only**	**Diagnosis data and genetic sequences**
J	Number of particles	10000	60000	10000	10000	10000	10000
K	Number of attachment sites per sequence	-	5	-	10	-	10
L	Number of relaxed clock gamma samples per attachment site	-	10	-	10	-	10
M	Number of relaxed clock beta samples per gamma	-	1	-	1	-	1
α_c	Cooling factor	-	-	0.95	0.95	-	-
σ_{rw}	Random walk standard deviation	-	-	0.01	0.01	-	-
I	Number of GenIF iterations	-	-	50	30	-	-

Supplementary References

Arulampalam, M. S., S. Maskell, N. Gordon, and T. Clapp. 2002. A tutorial on particle filters for online nonlinear, non-Gaussian Bayesian tracking. IEEE Transactions on Signal Processing, **50**:174 – 188.

Bretó, C., D. He, E. L. Ionides, and A. A. King. 2009. Time series analysis via mechanistic models. Annals of Applied Statistics, **3**:319–348.

Douc, R., O. Cappé, and E. Moulines. 2005. Comparison of resampling schemes for particle filtering. In Proceedings of the 4th International Symposium on Image and Signal Processing and Analysis, 2005, pages 64–69. IEEE.

Drummond, A. J., S. Y. W. Ho, M. J. Phillips, and A. Rambaut. 2006. Relaxed phylogenetics and dating with confidence. PLoS Biology, **4**:e88.

Gordon, N., D. J. Salmond, and A. F. M. Smith. 1993. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. IEE Proceedings–F, **140**:107–113.

He, D., E. L. Ionides, and A. A. King. 2010. Plug-and-play inference for disease dynamics: Measles in large and small towns as a case study. Journal of the Royal Society Interface, **7**:271–283.

Ionides, E. L., C. Bretó, and A. A. King. 2006. Inference for nonlinear dynamical systems. Proceedings of the National Academy of Sciences of the USA, **103**:18438–18443.

Ionides, E. L., C. Breto, J. Park, R. A. Smith, and A. A. King. 2016. Monte Carlo profile confidence intervals. ArXiv:1612.02710.

Ionides, E. L., D. Nguyen, Y. Atchadé, S. Stoev, and A. A. King. 2015. Inference for dynamic and latent variable models via iterated, perturbed Bayes maps. Proceedings of the National Academy of Sciences of the USA, **112**:719–724.

Kantas, N., A. Doucet, S. S. Singh, J. Maciejowski, N. Chopin, et al. 2015. On particle methods for parameter estimation in state-space models. Statistical Science, **30**:328–351.

King, A. A., D. Nguyen, and E. L. Ionides. 2016. Statistical inference for partially observed Markov processes via the R package pomp. Journal of Statistical Software, **69**:1–43.

Liu, J. S. 2001. Monte Carlo Strategies in Scientific Computing. Springer, New York.

Naesseth, C. A., F. Lindsten, and T. Schön. 2015. Nested sequential Monte Carlo methods. In Proceedings of the 32nd International Conference on Machine Learning, Lille, France, 6-11 July, 2015, volume 37, pages 1292–1301. Journal of Machine Learning Research.

Skinner, C. J., D. Holt, and T. F. Smith. 1989. Analysis of complex surveys. John Wiley & Sons.

Volz, E. M., E. L. Ionides, E. Romero Severson, M. Brandt, E. Mokotoff, and J. S. Koopman. 2013. HIV-1 transmission during early infection in men who have sex with men: A phylodynamic analysis. PLoS Medicine, **10**:e1001568.