An R Package for Nonparametric Inference on Dynamic Populations with Infinitely Many Types

An R package for nonparametric inference on dynamic populations with infinitely many types

Filippo Ascolani, *Duke University*

Stefano Damato, *USI-SUPSI, Switzerland*

Matteo Ruggiero, *University of Torino and Collegio Carlo Alberto*

September 25, 2024

Fleming–Viot diffusions are widely used stochastic models for population dynamics which extend the celebrated Wright–Fisher diffusions. They describe the temporal evolution of the relative frequencies of the allelic types in an ideally infinite panmictic population, whose individuals undergo random genetic drift and at birth can mutate to a new allelic type drawn from a possibly infinite potential pool, independently of their parent. Recently, Bayesian nonparametric inference has been considered for this model when a finite sample of individuals is drawn from the population at several discrete time points. Previous works have fully described the relevant estimators for this problem, but current software is available only for the Wright–Fisher finite-dimensional case. Here we provide software for the general case, overcoming some non trivial computational challenges posed by this setting.

The R package FVDDPpkg efficiently approximates the filtering and smoothing distribution for Fleming–Viot diffusions, given finite samples of individuals collected at different times. A suitable Monte Carlo approximation is also introduced in order to reduce the computational cost.

Software available at https://github.com/StefanoDamato/FVDDPpkg.

Keywords: population genetics; Bayesian inference; time series data; hidden Markov models; Monte Carlo.

1 Introduction

Fleming–Viot (FV) diffusions are among the most widely used models in population genetics and stochastic population dynamics. See Ethier and Kurtz (1993) for an extensive review. They take values in the space of purely atomic probability measures defined on an arbitrary complete and separable metric space $\mathbb{Y}$, which in our application denotes the set of all possible *allelic types*. The process, denoted $\{X_t, t \geq 0\}$, describes the temporal evolution of the relative frequencies of types in an ideally infinite population, when the individuals that compose the population undergo random genetic drift, due to panmictic random mating, and at birth can mutate to a yet unseen type, drawn from a possibly infinite potential pool. Fleming–Viot diffusion can also accommodate mathematical descriptions of other evolutionary forces like natural selection or recombination, but in this paper we do not consider these cases. The time-homogeneous transition function, found by Ethier and Griffiths (1993), can be written

$$P_t(x, \mathrm{d}x') = \sum_{m=0}^{\infty} d_m^{\theta}(t) \int_{\mathbb{Y}^m} \Pi_{\alpha+\sum_{k=1}^m \delta_{y_k}}(\mathrm{d}x') x(\mathrm{d}y_1) \cdots x(\mathrm{d}y_m). \tag{1}$$

See eq. (2) in the Supplementary Material for the diffusion infinitesimal generator. Here Π_α is the reversible measure of the FV diffusion and denotes the law of a Dirichlet process (Ferguson, 1973) with baseline measure $\alpha = \theta P_0$, with $\theta > 0$ being the overall mutation rate and P_0 a probability measure on $\mathbb{Y}$ representing the potential pool of allelic types from which mutants are drawn. Furthermore, $d_m^{\theta}(t)$ is the transition probability of a pure-death process on $\mathbb{Z}_+$ which starts at infinity almost surely and jumps from m to $m-1$ at rate $\lambda_m = m(\theta + m - 1)/2$. These were computed in Griffiths (1980) and Tavaré (1984) and are in fact the transition probabilities of the block-counting process in Kingman's coalescent with mutation, given by

$$d_m^{\theta}(t) = \begin{cases} \sum_{k \geq m} e^{-\lambda_k t} (-1)^{k-m} \binom{k}{m} \frac{(\theta+2k-1)(\theta+m)_{(k-1)}}{k!}, & m \geq 1, \\ 1 - \sum_{k \geq 1} e^{-\lambda_k t} (-1)^{k-1} \frac{(\theta+2k-1)(\theta)_{(k-1)}}{k!}, & m = 0, \end{cases} \tag{2}$$

where $(a)_{(b)} = a(a+1)\ldots(a+b-1)$ is the ascending factorial or Pochhammer symbol. A different representation of (1) highlights the role of the death process as latent variable which determines the correlation between X_s and X_{s+t} (here s can be set equal to 0 as the process is time-homogeneous), and reads

$$X_t \,|(D_t = m, Y_{0,1}, \ldots, Y_{0,m}) \sim \Pi_{\alpha+\sum_{i=1}^m \delta_{Y_{0,i}}} \qquad Y_{0,i} \mid X_0 \overset{\text{iid}}{\sim} X_0. \tag{3}$$

Here D_t is the above death process with transition probabilities $d_m^\theta(t)$. The realisation of D_t determines the number $m \in \mathbb{Z}_+$ of individuals whose type is drawn from the initial state X_0 and used in the distribution of the arrival state X_t. The intuition is that for small t a large m is more likely, so that X_0 and X_t will tend to be similar, whereas at $t \to \infty$, m will be small or zero with high probability, and X_0 and X_t will be essentially uncorrelated. Cf. also Walker et al. (2007).

A projection of the above FV diffusion onto a measurable partition $A_1, \dots, A_K$ of the type space $\mathbb{Y}$ yields a *neutral* Wright–Fisher (WF) diffusion with K types, cf. Etheridge (2000) and Feng (2010). Recently, Jenkins and Spanò (2017) and Sant et al. (2023) addressed the problem of generating exact draws from the transition functions of this and other WF diffusions, in particular addressing the intractability of (2), which also appears in the associated finite-dimensional special case of (1).

Given the increasing availability of population-level allelic data, it is often of interest to study the temporal evolution of the associated frequencies, which can be modelled through a FV or a WF diffusion. See, e.g., Tataru et al. (2017),Gory et al. (2018), and Paris et al. (2019) for recent works in this direction. From a statistical perspective, the problem can be formulated as a hidden Markov model, a popular inferential framework for temporally evolving quantities (Cappé et al., 2005). In such setting, the FV process which models the evolving allelic frequencies is taken as the unobserved target of inference, and finitely-many biological samples from the population are assumed to be collected at discrete times. Denote by $Y_{t_k,i}$ the type of the ith individual observed at time t_k, for $i = 1, \dots, n_k$ and times $0 = t_0 < t_1 < \cdots < t_p = T$. Given a dataset $\mathbf{Y}_{0:T} := (\mathbf{Y}_0, \dots, \mathbf{Y}_T)$, where $\mathbf{Y}_k := (Y_{t_k,1}, \dots, Y_{t_k,n_k})$ is a row vector, the task is then to make inference on the process states, hence on the configuration of frequencies in the underlying population. Mathematically this amounts to fully describe the law of $X_t \mid \mathbf{Y}_{0:T}$, which is typically called *filtering distribution* when $t = T$, or *smoothing distribution* when $0 \leq t < T$. See Section 2 for a more detailed formalization of the setting. These inferential problems were fully solved and implemented by Papaspiliopoulos and Ruggiero (2014) and Kon Kam King et al. (2021; 2024) for the finite-dimensional WF special case. In the above infinite-dimensional framework, Papaspiliopoulos et al. (2016), Ascolani et al. (2021) and Ascolani et al. (2023) provided recursions that in principle allow to fully evaluate such distributions, showing that they are qualitatively similar to the formulation (3), with the important difference that the latent variable D_t is substituted by another variable with *finite state space*, yielding in fact finite mixtures of laws of Dirichlet processes structurally similar to $\sum_{i \in I} v_i \Pi_{\alpha_i}$, where I has finite cardinality and $\sum_{i \in I} v_i = 1$ (see Theorem 1 in the Supplementary Material). These results surprisingly overcome the intractability of the death process D_t in (3),

and in particular the need to deal with the infinite series expansion for its transition probabilities, making the evaluation of these distributions possible with a finite computational effort, without the need for stochastic truncation or approximation strategies. However, sampling in practice from these distributions leads to non trivial computational challenges that are not present in the finite-dimensional subcase, mainly related to the growth in the number of observed types as more data are collected. This aspect reflects the growth of the cardinality of I in the above expression, making exact computations quickly infeasible. Another aspect which is peculiar of this setting is how the weights v_i in some of the distributions of interest depend on the support of the offspring distribution P_0, which from a modelling perspective can be taken as atomic or nonatomic.

Here we present a simple and efficient Monte Carlo procedure to approximate the filtering and smoothing distributions of a FV process under the assumed data collection framework. These are implemented in the publicly available R package *FVDDPpkg*. The acronym for the package relates to another line of research on *dependent Dirichlet processes* (DDPs), which in Bayesian nonparametric statistics are considered as collections of correlated random probability measures with Dirichlet process marginals. See Quintana et al. (2022) for a recent review.

2 Model

We assume a hidden Markov model formulation, whereby the unobserved process (sometimes called *signal*) X_t is a FV diffusion with dynamics as in (1), denoted $X \sim \text{FV}(\alpha)$, and the observations, which provide the allelic types of the individuals, are conditionally *iid* given the current FV state, namely $Y_{t_k,i}|X_{t_k} = x \overset{\text{iid}}{\sim} x$. That is, when the current configuration of the type frequencies is the atomic probability measure x on $\mathbb{Y}$, so that this admits representation $x = \sum_{i=1}^{\infty} v_i \delta_{z_i}$, with $\sum_{i\geq 1} v_i = 1$ and $z_i \in \mathbb{Y}$, then $Y_{t_k,i} = z_i$ with probability v_i. This essentially amounts to sampling without replacement when the population from which we draw is infinite.

Let now $X_0 \sim \Pi_\alpha$ be the initial state of the process. Recall that $\alpha = \theta P_0$, with $\theta > 0$ a fixed constant and P_0 a probability measure on $\mathbb{Y}$. Denote as above by $\mathbf{Y}_{0:T} := (\mathbf{Y}_0, \ldots, \mathbf{Y}_T)$ the overall dataset, where $\mathbf{Y}_k := (Y_{t_k,1}, \ldots, Y_{t_k,n_k})$ for times $0 = t_0 < t_1 < \cdots < t_p = T$, and let K be the number of unique values (allelic types) in $\mathbf{Y}_{0:T}$. We are interested in the *filtering* and *smoothing distribution* for X_t, which can be written (Papaspiliopoulos et al., 2016) and Ascolani et al. (2023) as

$$X_t|(\mathbf{Y}_{0:T}, \mathbf{M}_t = \mathbf{m}) \sim \Pi_{\alpha+\sum_{k=1}^{K} m_k \delta_{y_k^*}}, \qquad p(\mathbf{M}_t = \mathbf{m}) = w_{\mathbf{m}}(t). \tag{4}$$

Here $\mathbf{M}_t$ is a certain death process on a finite state space $\mathcal{M} \subset \mathbb{Z}_+^K$, where $\mathbf{m} = (m_1, \ldots, m_K) \in \mathcal{M}$ describes the multiplicities of $(y_1^*, \ldots, y_K^*)$, the unique values (types) in $\mathbf{Y}_{0:T}$. These elements input information in the mixture components through the empirical measure $\sum_{k=1}^K m_k \delta_{y_k^*}$. Notice that (4) is structurally similar to representation (3), with the key difference that the death process $\mathbf{M}_t$ has a *finite* state space. Further details on this representation, including $\mathcal{M}$ and $w_{\mathbf{m}}(t)$, can be found in the Supplementary Material.

These distributions allow one to make inference on the evolution of the frequencies in the population, given the data, at any time of interest. Moreover, they can be used to evaluate the probability of observing again the already recorded allelic types in new individuals drawn from the population; see Proposition 2 in Ascolani et al. (2021) and Section 1 in the Supplementary Material.

3 Methods

The task of evaluating the right hand side of (4) poses significant challenges. The probability weights $w_{\mathbf{m}}(t)$ depend on the transition probabilities of a certain death process (informally, a projection of D_t in (1)), whose computation can be numerically unstable and requires some care (Griffiths (1984)). Even assuming regularly spaced data collection times at intervals of length Δ, the cardinality of $\mathcal{M}$ grows polynomially with the number of observed datapoints (cf. Section 3.2 in Ascolani et al. (2023)), thus storing all the $w_{\mathbf{m}}(\Delta)$ becomes demanding even with moderately large datasets. Finally, when $t < T$ as in (4), the set of nodes $\mathbf{m}$ with strictly positive probability depends on the nature of baseline distribution P_0 (Proposition 3.6 in Ascolani et al. (2021)). Specifically, if P_0 is nonatomic, $\mathcal{M}$ depends on the unique values shared across different data collection times (cf. point C of Proposition 3.6 in Ascolani et al. (2021)).

An accurate approximation can nevertheless be obtained by exploiting different factors. First, the weights can be computed recursively. If $t \in (t_j, t_{j+1})$, (4) can be computed starting by the distributions of the signal given past and future observations used separately, i.e.

$$
\begin{aligned}
\mathcal{L}\left(X_{t_j} | \mathbf{Y}_{0:j}\right) &= \sum_{\mathbf{m}_1 \in \mathcal{M}_1} u_{\mathbf{m}_1} \Pi_{\alpha + \sum_{k=1}^K m_{1,k} \delta_{y_k^*}}, \\
\mathcal{L}\left(X_{t_{j+1}} | \mathbf{Y}_{j+1:T}\right) &= \sum_{\mathbf{m}_2 \in \mathcal{M}_2} v_{\mathbf{m}_2} \Pi_{\alpha + \sum_{k=1}^K m_{2,k} \delta_{y_k^*}},
\end{aligned}
\tag{5}
$$

where $\mathcal{M}_1 \subset \mathbb{N}^K$ and $\mathcal{M}_2 \subset \mathbb{N}^K$ are suitable sets (see Sections 2 and 3 of the Supplementary

material). Below we focus on the (arguably harder) case of approximating the smoothing distribution, while the procedure for filtering is discussed in Section 4 of the Supplementary material. Each weight $w_{\mathbf{m}}$ in (4) for the smoothing distribution turns out to be

$$w_{\mathbf{m}} = \sum_{\mathbf{m}_1 \in \mathcal{M}_1} \sum_{\mathbf{m}_2 \in \mathcal{M}_2} u_{\mathbf{m}_1} v_{\mathbf{m}_2} \sum_{\mathbf{k}_1, \mathbf{k}_2 \,:\, \mathbf{k}_1 + \mathbf{n} + \mathbf{k}_2 = \mathbf{m}} \tilde{w}^{\mathbf{m}_1, \mathbf{m}_2}_{\mathbf{k}_1, \mathbf{n}, \mathbf{k}_2}, \tag{6}$$

with $u_{\mathbf{m}_1}, v_{\mathbf{m}_2}$ taken from (5) and where

$$\tilde{w}^{\mathbf{m}_1, \mathbf{m}_2}_{\mathbf{k}_1, \mathbf{n}, \mathbf{k}_2} \propto p_{\mathbf{m}_1, \mathbf{k}_1}(t - t_i) p_{\mathbf{m}_2, \mathbf{k}_2}(t_{i+1} - t) q(\mathbf{k}_1, \mathbf{n}, \mathbf{k}_2).$$

Here in turn the vector indices $\mathbf{n}$ are the multiplicities of the allelic types observed at the time t of interest (one can simply set $\mathbf{n}$ to be a vector of zeros if at time t no data are available), $p_{\mathbf{n},\mathbf{k}}(t)$ are the transition probabilities of a death process on $\mathbb{Z}_+^K$ which decreases from $\mathbf{m} \in \mathbb{Z}_+^K$ to $\mathbf{m} - \mathbf{e}_j$ at rate $\lambda_{\mathbf{m},j} := m_j(|\mathbf{m}| + \theta - 1)$, where $\mathbf{e}_j$ denotes the canonical vector in the direction j, and q is a suitable function which depends on P_0. Weights in (6) represent the probability that, starting two independent death processes at the points of $\mathcal{M}_1$ and $\mathcal{M}_2$, the arrival points $\mathbf{k}_1$ and $\mathbf{k}_2$ sum up to $\mathbf{m} - \mathbf{n}$. This is then reweighted through the function q, which formalizes how the information contained in the data collected prior to and after t is combined. Further details are given in Section 3 of the Supplementary Material (cf. Proposition 5).

Computing the transition probabilities $p_{\mathbf{n},\mathbf{k}}(t)$ is the hardest task. Indeed, their exact value depends on a sum with alternating signs (see formula (B40) in Kon Kam King et al. (2021)) and they assign most of the probability mass to few nodes, hence a large majority of elements in $\mathcal{M}$ retains a low cumulative probability (cf. Figure 4 in Ascolani et al. (2021)). Kon Kam King et al. (2021) (cf. Section 4.2) exploit this principle for Wright–Fisher diffusions through a pruning strategy. Here we instead consider a Monte Carlo approximation of (6), leading to the following key part of the algorithm for approximating the terms $p_{\mathbf{n},\mathbf{k}}(t)$. Given (5):

- draw $\mathbf{m}_1$ and $\mathbf{m}_2$ with probability $u_{\mathbf{m}_1}$ and $v_{\mathbf{m}_2}$ respectively;
- simulate two independent death processes with rates $\lambda_{\mathbf{m}_1}$ and $\lambda_{\mathbf{m}_2}$ over the time intervals $t - t_j$ and $t_{j+1} - t$ respectively, where $\lambda_{\mathbf{m}} := (\lambda_{\mathbf{m},1}, \ldots, \lambda_{\mathbf{m},K})$ and $\lambda_{\mathbf{m},j}$ is as above;
- return the two arrival nodes $\mathbf{k}_1$ and $\mathbf{k}_2$.

The death process simulation can be performed by means of the Gillespie algorithm (Gillespie, 2007), as described in the Supplementary Material. Once the above procedure is repeated for the desired number of particles, for a fixed $\mathbf{m}$ the unnormalized version of $w_{\mathbf{m}}$ is approximated

by the empirical average of $q(\mathbf{k}_1, \mathbf{n}, \mathbf{k}_2)$, for all the sampled $(\mathbf{k}_1, \mathbf{k}_2)$ such that $\mathbf{m} = \mathbf{k}_1 + \mathbf{n} + \mathbf{k}_2$. Normalizing the weights is straightforward as their cardinality is finite, and after normalization we can in addition apply a suitable *pruning*, that is all the weights below a certain threshold (which we chose no larger than 10^{-9}) are eliminated and a new normalization is performed. Further details can be found in Section 4 of the Supplementary Material.

The above strategy typically leads to mixtures with a relatively small number of components which collectively provide a good approximation to the true distribution. Figure 1 shows that the algorithm is able to reproduce efficiently large mixtures. The exact smoothing distribution in the left panel is composed of more than 30.000 components, many of which have an almost null weight: indeed 90%, 95%, and 99% of the mass is given by respectively 237, 432, and 1170 nodes with the largest weight. The approximate distribution whose nodes are represented in the right panel captures the same information using 12.500 components, with an average absolute error of order $5 \cdot 10^{-6}$. In the figure, the x-axis depicts the cardinality of the vectors $\mathbf{m}$, showing that the vast majority of the probability mass is retained by nodes with few elements: this is coherent with the fact that the death process $\mathbf{M}_t$ decreases very fast (cf. Figure 4 in Ascolani et al. (2021)). See Section 6 of the Supplementary Material for further details on the simulation setting and the associated computational cost: in particular, Figure 3 in the Supplementary Material depicts the growth of the (time and memory) cost as a function of the Monte Carlo samples. Clearly, one could choose a higher threshold value for pruning the mixture components, leading to more parsimonious mixtures and higher computational efficiency, at the cost of a larger approximation error. The feather-like shape in the figure shows that nodes corresponding to higher cardinalities have typically smaller weights. This behaviour is mainly a product of the fact that the death process propagates the probability mass towards the empty node (the origin of the graph in Figure 1 in the Supplementary Material). Hence the interplay between updates with new data, which moves the mass upwards (cf. Figure 1 in the Supplementary Material), with the temporal propagation which operates for both distributions in (5), which become part of the smoothing distribution, ends up keeping most of the probability mass in nodes that are relatively near the origin.

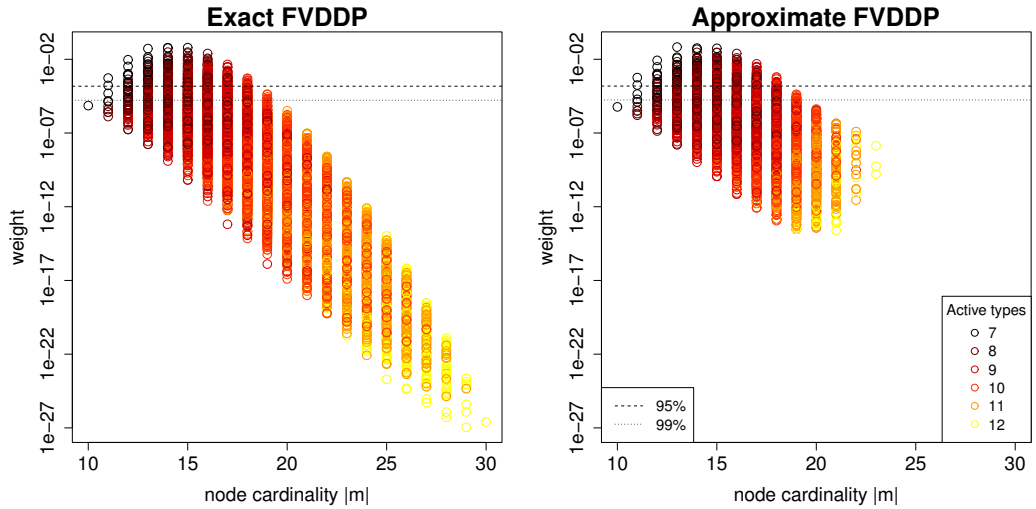

Figure 1: Comparison between probability mass assigned to mixture components in the exact (left) and approximate (right) smoothing distribution. The simulation is based on synthetic data from 3 different times, each with 10 observations, and 10^6 particles for the death process simulations. The cardinality $|\mathbf{m}| = \sum_{j=1}^{K} m_j$ associated to each node is depicted on the x-axis, with the corresponding probability weight (in log scale) on the y-axis. Observations are color-coded based on the amount of active types ($\text{card}\{j : m_j \neq 0\}$), darker colors indicating lower values. Above the dashed lines lie the 95% and 99% of the total mass, respectively. The figure shows that the exact distribution can be efficiently approximated through simulation of the death processes by mixture with a significant lower number of components having non negligible probability mass.

4 Discussion

The *R* package *FVDDPpkg* allows one to compute the filtering and smoothing distributions for the above FV-driven hidden Markov model, with an arbitrary set of observation times. It also allows one to evaluate the probability of observing the already recorded allelic type in new individuals to be drawn from the population, through the corresponding predictive distributions (see Section 1 of the the Supplementary Material for more details). The package efficiently implements the Monte Carlo approximation discussed above by exploiting the dynamic programming paradigm and optimized routines in the C++ language (via the Rcpp package, Eddelbuettel and François (2011)). The recursive structure of the problem is exploited and a suitable pruning strategy allows to save both computational time and memory requirements. Extensive documentation and

a vignette are also available to increase the accessibility.

5 Acknowledgements

MR was partially supported by the European Union – Next Generation EU, PRIN- PNRR 2022 (P2022H5WZ9).

References

Ascolani, F., Lijoi, A., and Ruggiero, M. Predictive inference with Fleming–Viot-driven dependent Dirichlet processes. *Bayesian Analysis*, 16(2):371 – 395, 2021.

Ascolani, F., Lijoi, A., and Ruggiero, M. Smoothing distributions for conditional Fleming–Viot and Dawson–Watanabe diffusions. *Bernoulli*, 29(2):1410 – 1434, 2023.

Cappé, O., Moulines, E., and Ryden, T. *Inference in Hidden Markov Models*. Springer Series in Statistics. Springer-Verlag, Berlin, Heidelberg, 2005.

Eddelbuettel, D. and François, R. Rcpp: Seamless R and C++ Integration. *Journal of Statistical Software*, 40(8):1–18, 2011.

Etheridge, A. M. *An introduction to superprocesses*. Number 20 in University Lecture Series. American Mathematical Society, Providence, RI, 2000.

Ethier, S. N. and Griffiths, R. C. The Transition Function of a Fleming-Viot Process. *The Annals of Probability*, 21(3):1571 – 1590, 1993.

Ethier, S. N. and Kurtz, T. G. Fleming–Viot Processes in Population Genetics. *SIAM Journal on Control and Optimization*, 31(2):345–386, 1993.

Feng, S. *The Poisson–Dirichlet Distribution and Related Topics*. Probability and Its Applications. Springer, Berlin, Heidelberg, 2010.

Ferguson, T. S. A Bayesian Analysis of Some Nonparametric Problems. *The Annals of Statistics*, 1(2):209 – 230, 1973.

Gillespie, D. T. Stochastic simulation of chemical kinetics. *Annu. Rev. Phys. Chem.*, 58:35–55, 2007.

Gory, J. J., Herbei, R., and Kubatko, L. S. Bayesian inference of selection in the Wright-Fisher diffusion model. *Statistical Applications in Genetics and Molecular Biology*, 17(3):20170046, 2018.

Griffiths, R. Lines of descent in the diffusion approximation of neutral wright-fisher models. *Theoretical Population Biology*, 17(1):37–50, 1980.

Griffiths, R. C. Asymptotic line-of-descent distributions. *Journal of Mathematical Biology*, 21: 67–75, 1984.

Jenkins, P. A. and Spanò, D. Exact simulation of the Wright–Fisher diffusion. *The Annals of Applied Probability*, 27(3), jun 2017.

Kon Kam King, G., Papaspiliopoulos, O., and Ruggiero, M. Exact inference for a class of hidden Markov models on general state spaces. *Electronic Journal of Statistics*, 15:2832–2875, 2021.

Kon Kam King, G., Pandolfi, A., Piretto, M., and Ruggiero, M. Approximate filtering via discrete dual processes. *Stochastic Processes and Their Applications*, 168:104268, 2024.

Papaspiliopoulos, O. and Ruggiero, M. Optimal filtering and the dual process. *Bernoulli*, 20(4), nov 2014.

Papaspiliopoulos, O., Ruggiero, M., and Spanò, D. Conjugacy properties of time-evolving Dirichlet and gamma random measures. *Electronic Journal of Statistics*, 10(2):3452 – 3489, 2016.

Paris, C., Servin, B., and Boitard, S. Inference of selection from genetic time series using various parametric approximations to the Wright–Fisher model. *G3: Genes, Genomes, Genetics*, 9(12): 4073–4086, 2019.

Quintana, F. A., Müller, P., Jara, A., and MacEachern, S. N. The Dependent Dirichlet Process and Related Models. *Statistical Science*, 37(1):24 – 41, 2022.

Sant, J., Jenkins, P. A., Koskela, J., and Spanó, D. EWF: simulating exact paths of the Wright-Fisher diffusion. *Bioinformatics*, 39(1), 2023.

Tataru, P., Simonsen, M., Bataillon, T., and Hobolth, A. Statistical inference in the Wright–Fisher model using allele frequency data. *Systematic biology*, 66(1):e30–e46, 2017.

Tavaré, S. Lines-of-descent and genealogical processes, and their applications in population genetics models. *Advances in Applied Probability*, 26(2):119–64, 1984.

Walker, S. G., Hatjispyros, S. J., and Nicoleris, T. A Fleming-Viot process and Bayesian nonparametrics. *Annals of Applied Probability*, 17(1):67–80, 2007.

Supplementary material

1 Introduction

The Hidden Markov model described in the paper can be described as

$$Y_{t,i}|X_t = x \overset{\text{iid}}{\sim} x, \quad X \sim \text{FV}(\alpha), \quad X_0 \sim \Pi_\alpha, \tag{1}$$

where $X := \{X_t, t \geq 0\}$ is a FV process with parameter α, reversible with respect to the law Π_α of a Dirichlet process (Ferguson, 1973). This FV diffusion has transition function as in (1) of the main document, and its infinitesimal generator can be written

$$\mathbb{A}\phi^m(\mu) := \sum_{i=1}^{m} \langle A_i f, \mu^m \rangle + \frac{1}{2} \sum_{1 \leq k \neq i \leq m} \langle \Phi_{ki} f - f, \mu^m \rangle, \tag{2}$$

acting on functions $\phi^m(\mu) = \langle f, \mu^m \rangle = \int_{\mathbb{Y}^m} f d\mu^m$, for $f \in B(\mathbb{Y}^m)$ and μ^m being a m-fold product measure. Furthermore, A_i is the mutation operator

$$Af(x) = \frac{1}{2}\theta \int [f(z) - f(x)]\nu_0(dz),$$

which is the generator of a Feller process on $\mathbb{Y}$, operating on f as a function of x_j alone, and $\Phi_{ij} f$ is the function of $m-1$ variables obtained by setting the ith and the jth variables in f equal. Cf. Ethier and Kurtz (1993), Section 3.

Assume data $Y_{t_k,i} \in \mathbb{Y}$ are received at times $0 = t_0 < t_1 < \cdots < t_p = T$ and denote by $\mathbf{Y}_k = (Y_{t_k,1}, \ldots, Y_{t_k,n_k})$ the vector of all observations collected at times t_k. Let also $\mathbf{Y}_{0:T} = (\mathbf{Y}_0, \ldots, \mathbf{Y}_T)$ denote the entire data set. In biological applications the values contained in $\mathbf{Y}_{0:T}$ are allelic types, but the theory we develop holds in general for locations in a Polish space. Denote by $(y_1^*, \ldots, y_K^*)$ the vector of unique values in $\mathbf{Y}_{0:T}$, with $K \in \mathbb{N}$ being their total number. Finally, we denote vectors in $\mathbb{Z}_+^K$ as $\mathbf{n} = (n_1, \ldots, n_K)$, where $\mathbf{m} \leq \mathbf{n}$ if $m_i \leq n_i$ for $i = 1, \ldots, K$, and let $\mathbf{n}_j \in \mathbb{Z}_+^K$ be the vector of multiplicities of observations in $\mathbf{Y}_j$. Finally, set $|\mathbf{n}| := \sum_{k=1}^K n_k$. We collect in Table (1) a summary of the used notation.

The main result, which motivates the introduction of the present package, is obtained combining Proposition 3.1 in Papaspiliopoulos et al. (2016) and Theorem 1.1 in Ascolani et al. (2023) and is summarized next. It states that the distribution of X_t, given $\mathbf{Y}_{0:T}$, is always given by a finite mixture of Dirichlet processes.

$\mathbb{Y}$	Set of all possible types
$Y_{t,i}$	Type of individual i at time t
$\mathbf{Y}_k$	Vector with all observations collected at time t_k
$\mathbf{Y}_{0:T}$	Vector containing all the collected observations
K	Number of unique types in $\mathbf{Y}_{[0,T]}$
y_i^*	i-th unique type in $\mathbf{Y}_{[0,T]}$, with $i = 1, \dots, K$

Table 1: List of useful notations

Theorem 1.1. *Consider model (1). Then for every t there exists a set $\mathcal{M} \subset \mathbb{Z}_+^K$ of finite cardinality and weights $\{w_{\mathbf{m}}, \mathbf{m} \in \mathcal{M}\}$ summing up to one such that*

$$\mathcal{L}(X_t|\mathbf{Y}_{0:T}) = \sum_{\mathbf{m}\in\mathcal{M}} w_{\mathbf{m}} \Pi_{\alpha+\sum_{k=1}^K m_k \delta_{y_k^*}}, \tag{3}$$

where Π_α denotes the law of a Dirichlet process with baseline measure α.

Notice that the representation (3) allows also to derive the predictive distribution of new observations at time t, given all the available information. This follows by the nice properties of mixtures of Dirichlet processes, cf. Corollary 3.8 in Ascolani et al. (2023), whereby from (3) one yields the mixture of Pólya urn schemes

$$Y_{t,1}|\mathbf{Y}_{0:T}, \mathbf{m} \sim \frac{\alpha + \sum_{k=1}^K m_k \delta_{y_k^*}}{\theta + \sum_{k=1}^K m_k}, \qquad \mathbf{m}|\mathbf{Y}_{0:T} \sim w_{\mathbf{m}}.$$

In the following we discuss how the weights $\{w_{\mathbf{m}}, \mathbf{m} \in \mathcal{M}\}$ can be recursively computed. We consider two distinct cases:

1. $t \geq T$; in this case X_t is conditioned only to past observations, and the left-hand side of (3) is the *filtering distribution*;

2. $t = t_j$, for $j \in 0, \dots, p-1$; ; in this case X_t is conditioned to both data collected prior to t and that become available at a later time, and the left-hand side of (3) is the *smoothing distribution* (cf. Section 3).

Notice that the case $t < T$ with $t \neq t_j$ for every j can be incorporated in case 2 above without loss of generality, by adding a new collection time where no datapoints are observed.

We discuss the details about the above two cases in Sections 2 and 3 respectively. Section 4 discusses how to approximate the filtering distribution with a Monte Carlo procedure implemented in the package, while Sections 5 and 6 provide additional details on some aspects of the code and on the setting underlying Figure 1 in the main document.

2 Filtering distributions

Let $t \geq T$ in (3). The law $\mathcal{L}(X_t|\mathbf{Y}_{0:T})$ is obtained by recursively computing $\mathcal{L}(X_{t_j}|\mathbf{Y}_{0:j})$, where $\mathbf{Y}_{i:j}$ denotes the set of observations collected in the time instances $(t_i, \ldots, t_j)$. We unpack this computation into more elementary operations, stated in the next three propositions.

Proposition 2.1. *Consider model (1). Then*

$$\mathcal{L}\left(X_{t_0}|\mathbf{Y}_0\right) = \Pi_{\alpha+\sum_{k=1}^{K} n_{0,k}\delta_{y_k^*}}.$$

Proposition 2.1 follows immediately by the conjugacy of Dirichlet processes (Ferguson, 1973), and amounts to conditioning the stationary distribution to the first available data. Here t_0 is taken to be 0 without loss of generality, as by stationarity if $X_0 \sim \Pi_\alpha$ then $X_s \sim \Pi_\alpha$ for any $s > 0$ until the first available data point. Next we consider prediction of the signal, given past data. In relation to Proposition 2.1 this would mean the law of X_{t_0+s} given $\mathbf{Y}_0$. More generally, we condition on the previous j samples to elicit the recursive nature of the computation, and show how to evaluate the law of the unobserved signal at time t_j, given information collected up to time t_{j-1}.

Proposition 2.2. *Consider model (1) and assume*

$$\mathcal{L}\left(X_{t_{j-1}}|\mathbf{Y}_{0:j-1}\right) = \sum_{\mathbf{m}\in\mathcal{M}} w_{\mathbf{m}}\Pi_{\alpha+\sum_{k=1}^{K} m_k\delta_{y_k^*}},$$

for some $\mathcal{M} \subset \mathbb{Z}_+^K$ where $\mathcal{M}$ has finite cardinality. Then

$$\mathcal{L}\left(X_{t_j}|\mathbf{Y}_{0:j-1}\right) = \sum_{\mathbf{n}\in L(\mathcal{M})} \tilde{w}_{\mathbf{n}}\Pi_{\alpha+\sum_{k=1}^{K} m_k\delta_{y_k^*}},$$

where

$$L(\mathcal{M}) = \{\mathbf{n} \in \mathbb{Z}_+^K : \exists\ \mathbf{m} \in \mathcal{M} : \mathbf{n} \leq \mathbf{m}\} \tag{4}$$

and

$$\tilde{w}_{\mathbf{n}} = \sum_{\mathbf{m}\in\mathcal{M}\,:\,\mathbf{n}\leq\mathbf{m}} w_{\mathbf{m}} p_{\mathbf{m},\mathbf{n}}(t_j - t_{j-1}),$$

with $p_{\mathbf{m},\mathbf{n}}(s)$ *described below.*

Proposition 2.2 amounts to Proposition 3.1 in Papaspiliopoulos et al. (2016). The weights $p_{\mathbf{m},\mathbf{n}}(s)$ are the transition probabilities of a death process that jumps from $\mathbf{m}$ to $\mathbf{m} - e_i$ at rate $m_j(\theta + |\mathbf{m}| - 1)/2$, with e_i being the canonical vector in the i-th direction, given by

$$p_{\mathbf{m},\mathbf{n}}(s) = \begin{cases} e^{-\lambda_{|\mathbf{m}|}s} & \text{if } \mathbf{n} = \mathbf{m} \\ C_{|\mathbf{m}|,|\mathbf{n}|}(s)\text{MVH}\left(\mathbf{n}; |\mathbf{n}|, \mathbf{m}\right) & \text{if } \mathbf{n} < \mathbf{m} \end{cases} \tag{5}$$

where $\lambda_{|\mathbf{m}|} = |\mathbf{m}|(\theta + |\mathbf{m}| - 1)/2$,

$$\text{MVH}\left(\mathbf{n}; |\mathbf{n}|, \mathbf{m}\right) = \binom{|\mathbf{m}|}{|\mathbf{n}|}^{-1} \prod_{j\geq 1} \binom{m_j}{n_j} \tag{6}$$

is the probability mass function of a multivariate hypergeometric random variable and

$$C_{|\mathbf{m}|,|\mathbf{n}|}(s) = \left(\prod_{h=|\mathbf{n}|+1}^{|\mathbf{m}|} \lambda_h\right) (-1)^{|\mathbf{m}|-|\mathbf{n}|} \sum_{k=|\mathbf{n}|}^{|\mathbf{m}|} \frac{e^{-\lambda_k s}}{\prod_{|\mathbf{n}|\leq h\leq|\mathbf{m}|, h\neq k}(\lambda_k - \lambda_h)}.$$

Finally, we show how to update the distribution obtained through Proposition 2.2, once further data become available at time t_j.

Proposition 2.3. *Consider model* (1) *and let*

$$\mathcal{L}\left(X_{t_j}|\mathbf{Y}_{0:j-1}\right) = \sum_{\mathbf{m}\in\mathcal{M}} w_{\mathbf{m}} \Pi_{\alpha+\sum_{k=1}^K m_k \delta_{y_k^*}},$$

for some $\mathcal{M} \subset \mathbb{Z}_+^K$. *Then*

$$\mathcal{L}\left(X_{t_j}|\mathbf{Y}_{0:j}\right) = \sum_{\mathbf{n}\in\mathcal{M}+\mathbf{n}_j} \tilde{w}_{\mathbf{n}} \Pi_{\alpha+\sum_{k=1}^K n_k \delta_{y_k^*}},$$

where $\mathcal{M} + \mathbf{n}_j = \{\mathbf{m} + \mathbf{n}_j : \mathbf{m} \in \mathcal{M}\}$ *and*

$$\tilde{w}_{\mathbf{m}+\mathbf{n}_j} \propto w_{\mathbf{m}} \text{PU}(\mathbf{n}_j|\mathbf{m}),$$

with $\text{PU}(\mathbf{n}|\mathbf{m})$ *the probability of drawing a vector* $\mathbf{n}$ *according to the predictive scheme*

$$\mathbf{Y}_{i+1}|\mathbf{Y}_{1:i} \sim \frac{\theta P_0 + \sum_{k=1}^K m_k \delta_{y_k^*} + \sum_{i=1}^i \delta_{\mathbf{Y}_i}}{\theta + |\mathbf{m}| + i}, \quad i = 0, \ldots, |\mathbf{n}| - 1.$$

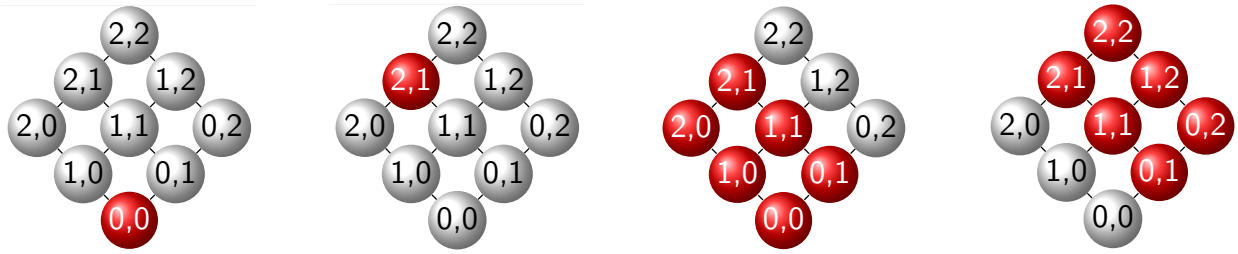

Figure 2: Graphs in $\mathbb{Z}_+^2$ representing the mixture components with positive weight in the current mixture of interest. From left: single mixture component when no observations are available; after observing multiplicities 2 and 1 for the two available types respectively; after propagating the signal and before further data collections; at the further data collection time after observing multiplicities 0 and 1 for the two available types.

Proposition 2.3 is given by Lemma 3.1 in Papaspiliopoulos et al. (2016), where $\text{PU}(\mathbf{n}|\mathbf{m})$ denotes the probability of drawing through Pólya urn sampling a vector of multiplicities $\mathbf{n}$ from an urn with initial composition given by $\mathbf{m}$. The above three Propositions show how the filtering distribution can be computed for every data collection time t_j.

Note that each of the above distributions is indexed by a finite set $\mathcal{M} \subset \mathbb{Z}_+^K$, and can thus be associated to a finite graph on $\mathbb{Z}_+^K$, where each node corresponds to an element $\mathbf{m} \in \mathcal{M}$ and identifies a mixture component. We clarify this intuition with a simple example in the special case of $K = 2$ possible observed types. In this case, the Dirichlet process Π_α reduces to a Beta distribution $\pi_{\alpha_1,\alpha_2} = \text{Beta}(\alpha_1, \alpha_2)$. In absence of data, the prior/initial distribution π_{α_1,α_2} does not carry any multiplicities of types, and it is then associated to the graph origin (Figure 2, leftmost panel). Suppose now the data at time t_0 carry multiplicities $\mathbf{n}_0 = (2, 1)$, i.e., we have collected two data points of type 1 and one of type 2. By Proposition (2.1), the graph is now given by the single node $\mathbf{n}_0$, since the initial distribution has been updated to $\pi_{\alpha_1+2,\alpha_2+1}$ (Figure 2, second panel). As described by Proposition 2.2, when the signal is propagated from t_0 to t_1, the probability mass initially concentrated on $\mathbf{n}_0$ is spread onto lower nodes, i.e., nodes in $L(\mathbf{n}_0)$, including the starting node (Figure 2, third panel). How this mass is spread is governed by the death process on $\mathbb{Z}_+^K$ through its transition probabilities. This operation thus leads to an increase in the cardinality of the mixture components, which does not depend on the time interval considered. The mixture components are now $\pi_{\alpha_1+i,\alpha_2+j}$ with $(0, 0) \leq (i, j) \leq (2, 1)$. A further update with $\mathbf{n}_1 = (0, 1)$, as described by Proposition 2.3, shifts the probability mass upwards by exactly $\mathbf{n}_1$, since each component is now $\pi_{\alpha_1+i+0,\alpha_2+j+1}$ (Figure 2, rightmost panel).

3 Smoothing distributions

Let now $t = t_j$, with $j \leq p-1$, so that the conditioning is on data collected both before and after t_j. From Section 2 we know that the law of $X_{t_{j-1}}$ given $\mathbf{Y}_{0:j-1}$ can be written as

$$\mathcal{L}\left(X_{t_{j-1}}|\mathbf{Y}_{0:j-1}\right) = \sum_{\mathbf{m}_{j-1}\in\mathcal{M}_{j-1}} u_{\mathbf{m}_{j-1}}\Pi_{\alpha+\sum_{k=1}^{K} m_{j-1,k}\delta_{y_k^*}}, \tag{7}$$

for some $\mathcal{M}_{j-1} \subset \mathbb{Z}_+^K$ of finite cardinality. By Lemma 3.2 in Ascolani et al. (2023) we can compute the law $\mathcal{L}\left(X_{t_{j+1}}|\mathbf{Y}_{j+1:p}\right)$ in the same way, i.e., propagating backwards is the same as propagating forward. Thus, by analogy with (7), we can write

$$\mathcal{L}\left(X_{t_{j+1}}|\mathbf{Y}_{j+1:p}\right) = \sum_{\mathbf{m}_{j+1}\in\mathcal{M}_{j+1}} v_{\mathbf{m}_{j+1}}\Pi_{\alpha+\sum_{k=1}^{K} m_{j+1,k}\delta_{y_k^*}}, \tag{8}$$

for some $\mathcal{M}_{j+1} \subset \mathbb{Z}_+^K$ of finite cardinality. The next proposition (which summarizes Proposition 3.6 in Ascolani et al. (2023)) shows how to compute $\mathcal{L}\left(X_t|\mathbf{Y}_{0:T}\right)$ from (7) and (8).

Proposition 3.1. *Consider model (1) and let $\mathcal{L}\left(X_{t_{j-1}}|\mathbf{Y}_{0:j-1}\right)$ and $\mathcal{L}\left(X_{t_{j+1}}|\mathbf{Y}_{j+1:p}\right)$ be as in (7) and (8). Then*

$$\mathcal{L}\left(X_t|\mathbf{Y}_{0:T}\right) = \sum_{\mathbf{m}_{i-1}\in M_{j-1}} \sum_{\mathbf{m}_{i+1}\in M_{j+1}} u_{\mathbf{m}_{i-1}} v_{\mathbf{m}_{i+1}} f(\mathbf{m}_{i-1}, \mathbf{m}_{i+1}),$$

where

$$f(\mathbf{m}_{i-1}, \mathbf{m}_{i+1}) = \sum_{\substack{(\mathbf{k}_{i-1},\mathbf{k}_{i+1}):\\ \mathbf{k}_{i-1}\leq\mathbf{m}_{i-1},\mathbf{k}_{i+1}\leq\mathbf{m}_{i+1}}} \tilde{p}^{\mathbf{m}_{i-1},\mathbf{m}_{i+1}}_{\mathbf{k}_{i-1},\mathbf{k}_{i+1}} q(\mathbf{k}_{i-1}, \mathbf{n}_j, \mathbf{k}_{i+1}) \Pi_{\alpha+\sum_{k=1}^{K}(\mathbf{k}_{j-1,k}+\mathbf{n}_{j,k}+\mathbf{k}_{j+1,k})\delta_{y_k^*}}$$

$$\tilde{p}^{\mathbf{m}_{i-1},\mathbf{m}_{i+1}}_{\mathbf{k}_{i-1},\mathbf{k}_{i+1}} = p_{\mathbf{m}_{i-1},\mathbf{k}_{i-1}}(t_j - t_{j-1}) p_{\mathbf{m}_{i+1},\mathbf{k}_{i+1}}(t_{j+1} - t_j)$$

with $p_{\mathbf{m},\mathbf{n}}(s)$ as in (5) and $q(\mathbf{k}_{i-1}, \mathbf{n}_j, \mathbf{k}_{i+1})$ as given below.

By readjusting the weights, it is then easy to obtain the representation (5) in the main document. The explicit formula for the weights $q(\mathbf{k}_{i-1}, \mathbf{n}_j, \mathbf{k}_{i+1})$ depends on the baseline measure α, as given by Proposition 3.6 in Ascolani et al. (2023). In particular:

- if α is *atomic* on $(y_1^*, \ldots, y_K^*)$, these can be expressed as

$$q(\mathbf{k}_{i-1}, \mathbf{n}_j, \mathbf{k}_{i+1}) \propto \frac{m(\mathbf{k}_{i-1} + \mathbf{n}_j + \mathbf{k}_{i+1})}{m(\mathbf{k}_{i-1}) m(\mathbf{n}_j) m(\mathbf{k}_{i+1})}$$

with

$$m(\mathbf{n}) = \frac{B(\boldsymbol{\alpha}+\mathbf{n})}{B(\boldsymbol{\alpha})}, \quad B(\boldsymbol{\alpha}) := \frac{\prod_{k=1}^{K}\Gamma(\alpha_k)}{\Gamma(|\boldsymbol{\alpha}|)}, \quad \boldsymbol{\alpha} = (\alpha(y_1^*),\ldots,\alpha(y_K^*)).$$

- if α is *nonatomic*, define the sets

$$\mathcal{D}_{j-1}^{\mathbf{m}_{i-1},\mathbf{m}_{i+1}} := \{k \in \{1,\ldots,K\} : m_{j-1,k} > 0 \text{ and either } n_{j,k} > 0 \text{ or } m_{j+1,k} > 0\},$$

$$\mathcal{D}_{j+1}^{\mathbf{m}_{i-1},\mathbf{m}_{i+1}} := \{k \in \{1,\ldots,K\} : m_{j+1,k} > 0 \text{ and either } n_{j,k} > 0 \text{ or } m_{j-1,k} > 0\}$$

and

$$\mathcal{S}^{\mathbf{n}_{i-1},\mathbf{n}_{i+1}} := \mathcal{D}_{j-1}^{\mathbf{n}_{i-1},\mathbf{n}_{i+1}} \cup \mathcal{D}_{j+1}^{\mathbf{n}_{i-1},\mathbf{n}_{i+1}}$$

to express the indices of shared values among different times. Then

$$\begin{aligned}\mathcal{D}^{\mathbf{m}_{i-1},\mathbf{m}_{i+1}} = \{(\mathbf{k}_{i-1},\mathbf{k}_{i+1}) :& \mathbf{k}_{i-1} \le \mathbf{m}_{i-1} \text{ and } k_{j-1,k} > 0 \;\forall\; k \in \mathcal{D}_{j-1}^{\mathbf{m}_{i-1},\mathbf{m}_{i+1}},\\ & \mathbf{k}_{i+1} \le \mathbf{m}_{i+1} \text{ and } k_{j+1,k} > 0 \;\forall\; k \in \mathcal{D}_{j+1}^{\mathbf{m}_{i-1},\mathbf{m}_{i+1}}\}\end{aligned}$$

and the weights are such that:

  - if $\mathcal{S}^{\mathbf{m}_{i-1},\mathbf{m}_{i+1}} = \emptyset$:

$$q(\mathbf{k}_{i-1},\mathbf{n}_j,\mathbf{k}_{i+1}) \;\propto\; \frac{\theta^{(|\mathbf{k}_{i-1}|)}\theta^{(|\mathbf{k}_{i+1}|)}}{(\theta+|\mathbf{n}_j|)^{(|\mathbf{k}_{i-1}|+|\mathbf{k}_{i+1}|)}}$$

  - if $\mathcal{S}^{\mathbf{m}_{i-1},\mathbf{m}_{i+1}} \neq \emptyset$:

$$\begin{aligned}q(\mathbf{k}_{i-1},\mathbf{n}_j,\mathbf{k}_{i+1}) \;\propto& \frac{\theta^{(|\mathbf{k}_{i-1}|)}\theta^{(|\mathbf{k}_{i+1}|)}}{(\theta+|\mathbf{m}_j|)^{(|\mathbf{k}_{i-1}|+|\mathbf{k}_{i+1}|)}}\cdot\\ & \cdot\prod_{k\in\mathcal{S}}\frac{(k_{j-1,k}+n_{j,k}+k_{j+1,k}-1)!}{(k_{j-1,k}-1)!(n_{j,k}-1)!(k_{j-1,jk}-1)!}\end{aligned}$$

    if $(\mathbf{k}_{i-1},\mathbf{k}_{i+1}) \in \mathcal{D}^{\mathbf{m}_{i-1},\mathbf{m}_{i+1}}$, and 0 otherwise, under the convention that $(-1)! = 1$.

Notice that, when α is nonatomic and a type y_k^* is observed at different times, the smoothing distribution imposes that the probability mass is concentrated on components of the mixture in which the type y_k^* is available (i.e., $m_k > 0$). This implies that the smoothing operation automatically implies a pruning of the graph, eliminating nodes which do not carry shared information available in the data. This implication is illustrated in Figure 3. The Figure displays the weights $q(\mathbf{k}_{i-1},\mathbf{n}_j,\mathbf{k}_{i+1})$ (as bubbles) associated to each pair $(\mathbf{k}_{i-1},\mathbf{k}_{i+1})$ (sorted along the axes) for different combinations of $\mathbf{m}_{i-1}$, $\mathbf{m}_j$ and $\mathbf{m}_{i+1}$. In particular the figure shows how the probability masses

for certain nodes are exactly zero after the update (crosses), typically near one or more sides of the lattice or, as discussed above, when the information is not shared among different collection times. The probability mass also appears to vary strongly among the nodes, which suggests many nodes may carry very small if not negligible probability.

4 Approximating filtering and smoothing distributions

Given the representation discussed at the end of Section 2, each distribution of interest is characterized by a graph on $\mathbb{Z}_+^K$ and the corresponding weights. By Proposition 2.2 each propagation step increases the number of elements in the graph and it can be shown that the latter grows polynomially with the number of collected datapoints (see, e.g., Section 3.2 of Ascolani et al. (2023)). This phenomenon makes exact computation of the weights impractical even with datasets of moderate size. However, as discussed in the main document, it is possible to exploit the sparsity properties of the involved distributions to devise an efficient Monte Carlo approximation. In the Section "Methods" of the main document it is shown how to approximate the weights of the smoothing distribution starting from the (forward and backward) filtering distributions in (7) and (8). Here we complete that discussion by showing how to approximate the filtering distributions.

Notice that updating the finite mixture with current observations (see Proposition 2.3) leaves the number of nodes in the graph unchanged. See third and fourth panels in Figure 2. Thus, it suffices to deal with the propagation step (see Proposition 2.2). Assume that

$$\mathcal{L}\left(X_{t_{j-1}}|\mathbf{Y}_{0:j-1}\right) \approx \sum_{\mathbf{m}\in\mathcal{M}} w_{\mathbf{m}}\Pi_{\alpha+\sum_{k=1}^K m_k\delta_{y_k^*}},$$

for some $\mathcal{M} \subset \mathbb{Z}_+^K$, where here $\mathcal{M}$ and $\{w_{\mathbf{m}}\}_{\mathbf{m}\in\mathcal{M}}$ are the graph and weights which approximate the distribution of the signal before propagating from t_j to t. Then, by Proposition 2.2, it is required to approximate $L(\mathcal{M})$ as in (4) and the corresponding weights. Following the same argument of the main document, the algorithm repeats the following procedure:

- draw $\mathbf{m} \in \mathcal{M}$ with probability $w_{\mathbf{m}}$;
- simulate a death process on $\mathbb{Z}_+^K$ starting from $\mathbf{m}$ with transition probabilities as in (5);

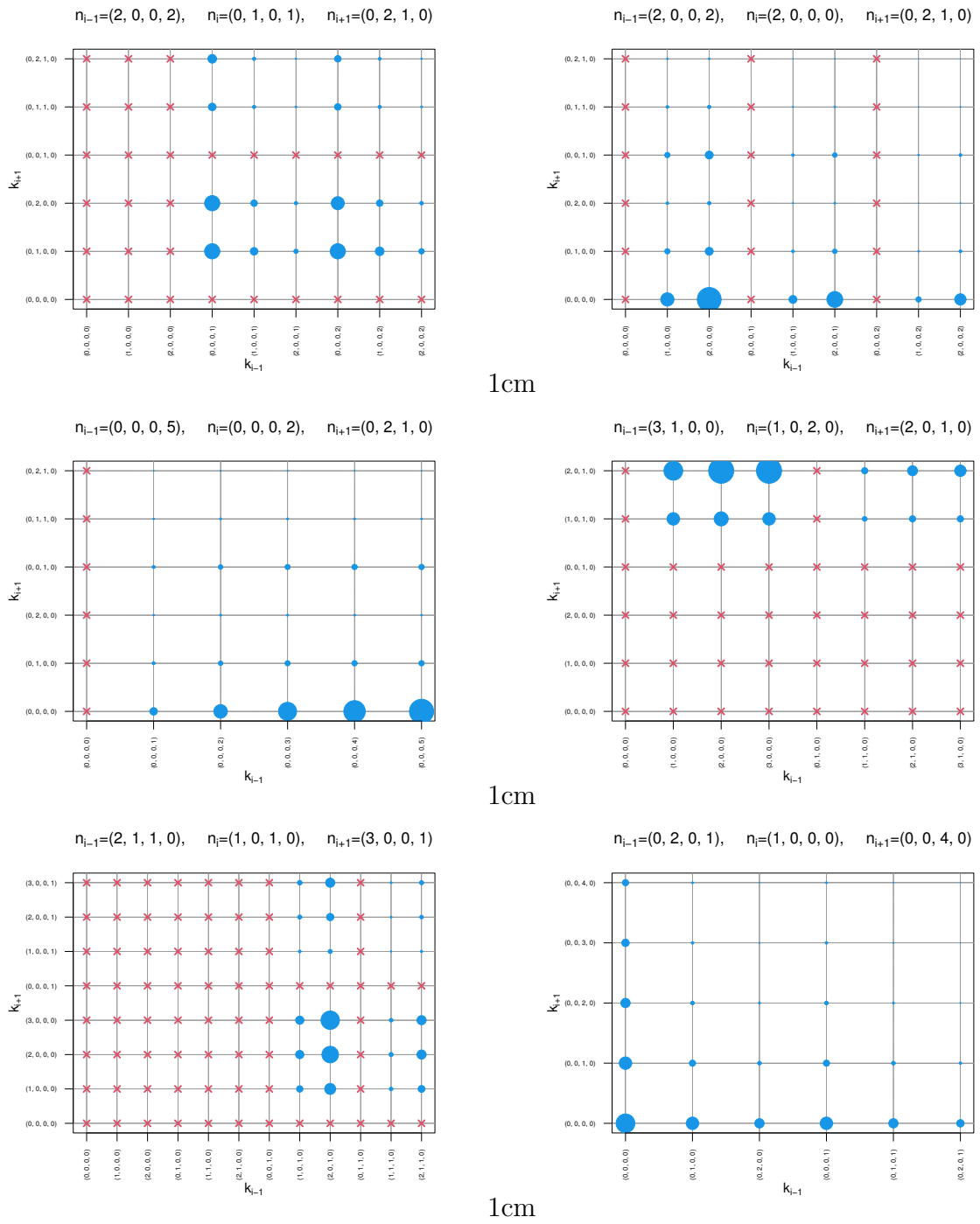

Figure 3: Weights $q(\mathbf{k}_{i-1}, \mathbf{n}_j, \mathbf{k}_{i+1})$ of the smoothing distribution in the case of a nonatomic α. The blue bubbles represent the probability mass associated to each node, while red crosses denote null mass. Each plot corresponds to a different choice of $\mathbf{m}_{j-1}$ and $\mathbf{m}_{j+1}$.

- return the arrival node $\mathbf{k}$.

The death process simulation can proceed as follows. When in $\mathbf{m}$, and until the sum of the non-homogeneous Exponential waiting times exceeds the desired interval:

- draw $S \sim \text{Exp}(\lambda_{|\mathbf{m}|})$;
- jump to $\mathbf{n}$ drawn from (6), with $|\mathbf{n}| = |\mathbf{m}| - 1$.

The above simulation should be performed for a desired number of particles, for every given $\mathbf{n}$, the new weight $\tilde{w}_{\mathbf{n}}$ is approximated by the proportion of iterations where the node $\mathbf{n}$ is the arrival point. Moreover, nodes with an empirical weights smaller than some threshold ε (typically between 10^{-9} and the machine epsilon of the computer in use) are eliminated with a pruning procedure.

Thus, the final output of the algorithm is given by nodes and weights $\tilde{\mathcal{M}}$ and $\{\tilde{w}_{\mathbf{n}}\}_{\mathbf{n}\in\tilde{\mathcal{M}}}$ such that

$$\mathcal{L}\left(X_{t_j}|\mathbf{Y}_{0:j-1}\right) \approx \sum_{\mathbf{n}\in\tilde{\mathcal{M}}} \tilde{w}_{\mathbf{n}} \Pi_{\alpha+\sum_{k=1}^{K} n_k \delta_{y_k^*}}.$$

This procedure typically leads to manageable graphs, i.e., where $\tilde{\mathcal{M}}$ has a moderate cardinality, with a large saving of computational resources. Cf. Figure 1 in the main document.

5 Details of the code

The process is implemented as an R "S3" class, i.e., as a list containing specific elements. The fixed ones, to be specified during the initialization, are necessary to describe the parameter α of the model; this is done expressing it as $\alpha = \theta P_0$ where θ is a positive real number and P_0 a probability distribution on the desired state space $\mathbb{Y}$. When the signal is updated and propagated, the relevant information is expressed via y^* (a vector), M (a matrix, where each $\mathbf{m}$ is a row) and w (a vector).

Since the majority of the operations discussed above may require a large computational cost, most of them have been implemented in C++ via the Rcpp package (Eddelbuettel and François, 2011). This helped in the use of dynamic programming to alleviate the execution times and the use of memory: for example, as $C_{|\mathbf{m}|,|\mathbf{n}|}(t)$ only depends on the cardinality of the vectors, the values of its calculation can be stored and retrieved in an appropriate array. Since the function that computes

$C_{|\mathbf{m}|,|\mathbf{n}|}(t)$ is used several times while iterating on the elements of the matrix M, it is important that the array where the results are stored is passed by reference every time that the function is called.

To calculate the smoothing distribution, a binning strategy is pursued: each ancestor pair $(\mathbf{n}_{i-1}, \mathbf{n}_{i+1})$, transformed into a single string, is used as a key for a map, which represents a bin. In turn, such bin consists of a map: in this case, the key is the descendant pair $(\mathbf{k}_{i-1}, \mathbf{k}_{i+1})$ and the associated values are two: the first is the weight $q(\mathbf{k}_{i-1}, \mathbf{n}_j, \mathbf{k}_{i+1})$, computed analytically, while the second counts how many times the pair $(\mathbf{k}_{i-1}, \mathbf{k}_{i+1})$ has been drawn from the ancestors $(\mathbf{n}_{i-1}, \mathbf{n}_{i+1})$. When all samples have been obtained, within each bin, the weights $q(\mathbf{k}_{i-1}, \mathbf{n}_j, \mathbf{k}_{i+1})$ referred to each pair of descendants are multiplied by the draw counts, and the resulting weights are then normalized to sum up to one. Finally, the weights are merged: the weight of the node $\mathbf{k}_{i-1} + \mathbf{n}_j + \mathbf{k}_{i+1}$ is computed as the sum of the weights of the pair $(\mathbf{k}_{i-1}, \mathbf{k}_{i+1})$ in each bin, multiplied by the weight of the bin $u_{\mathbf{n}_{i-1}} \cdot v_{\mathbf{n}_{i+1}}$.

This structure implies that each bin is implemented as a map whose items are themselves maps, and consequently the variable that counts how many times each node is drawn and the weights $q(\mathbf{k}_{i-1}, \mathbf{n}_j, \mathbf{k}_{i+1})$ can be accessed with constant cost. This also means that once the samples have been drawn, reordering them in the matrix M and generating the vector of weights w has a linear cost with respect to the amount of nodes generated by the approximating algorithm. This is also true regarding propagation.

Simpler operations, such as updating and sampling, have been implemented in R by taking advantage of the vectorization properties it provides.

The code is available at https://github.com/StefanoDamato/FVDDPpkg.

6 Details of Figure 1 in Main Document

The data on which the Figure 1 in Main Document is based were generated according to the following hierarchical model:

$$
\begin{aligned}
&\mathbf{Y}_i \overset{\text{iid}}{\sim} \frac{1}{2}\text{Pois}(\mu_i^{-1}) + \frac{1}{2}\text{Pois}(5 + \nu_i^{-1}),\\
&\mu_i = \mu_{i-1} + \varepsilon_i, \quad \varepsilon_i \overset{\text{iid}}{\sim} \text{Exp}(1),\\
&\nu_i = \nu_{i-1} + \eta_i, \quad \eta_i \overset{\text{iid}}{\sim} \text{Exp}(1),
\end{aligned}
\tag{9}
$$

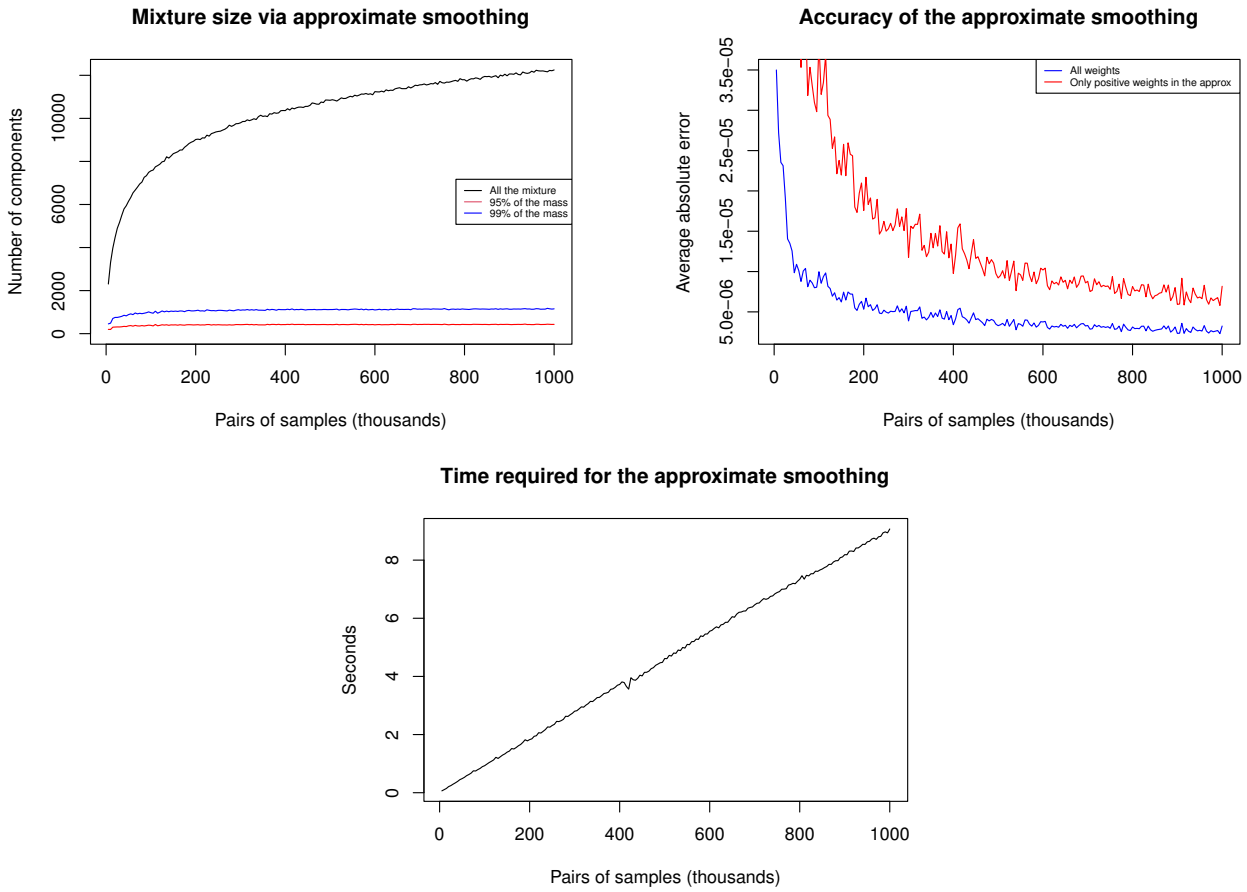

Figure 4: In the first panel, the number of components in the approximate process for different choices of the amount of Monte Carlo samples from the past and the future. In the second panel, the accuracy of the approximate weights for the same processes, where the red curve considers only nodes which were sampled at least once in the Monte Carlo approximation. In the third panel, the time required for the approximate algorithm with respect to the amount of samples. Here the data are generated according to model (9).

where $\mu_0^{-1} = \nu_0^{-1} = 5$ and μ_i, ν_i are independent. Data were collected at times $t_0 = 0$, $t_1 = 0.5$ and $t_2 = 1$, generating vectors of 10 observations for each time, denoted $\mathbf{Y}_0, \mathbf{Y}_1, \mathbf{Y}_2$. This is the same setting used in Section 4 of Ascolani et al. (2021), where the FV specification is shown to outperform other models for time dependent data.

Thus, the FV model is applied to the generated dataset, by choosing $\alpha \sim \text{NegBin}(2, 0.5)$ and the smoothing distribution is obtained by means of the appropriate algorithms outlined above, with propagation times $t_2 - t_1 = t_1 - t_0 = 0.5$. In particular the approximate algorithm is run drawing 10^6 Monte Carlo samples (this operation required less than 10 seconds on our machine).

Figure 4 shows that as the number of Monte Carlo samples used to approximate the mixture grows, the amount of mixture components it contains tends to settle at a much lower value than that of the process calculated with the exact algorithm (that is 30000). The same is true for the amount of components containing 95% and 99% of the total probablity mass. The second panel of the Figure shows how as the number of samples increases, the mixture generated by the approximate algorithm (in red) gets more accurate, as the difference of the weights from the exact one (in blue) decreases. In particular, the red curves considers only the nodes which have been sampled at least once in the Monte Carlo approximation.

The memory usage of the approximate algorithm, being proportional to the number of nodes in the resulting mixture, stabilizes at values that are significantly lower than those required by the exact algorithm. Figure 4 (first panel) depicts the effect of pruning, which significantly reduces the number of components to retain.