The Journal of Open Source Software

IdentityByDescentDispersal.jl: Inferring dispersal rates with identity-by-descent blocks

Francisco Campuzano-Jiménez [1¶], **Arthur Zwaenepoel** [1], **Els Lea R De Keyzer** [1], and **Hannes Svardal** [1,2]

1 University of Antwerp, Belgium **2** Naturalis Biodiversity Center, Leiden, Netherlands ¶ Corresponding author

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Evan Spotte-Smith

Reviewers:

- @slwu89
- @g-rppl

Submitted: 08 July 2025
Published: unpublished

Summary

The population density and per-generation dispersal rate of a population are central parameters in the study of evolution and ecology. The dispersal rate is particularly relevant for conservation management of fragmented or invasive species (Driscoll et al., 2014). There is a growing interest in developing statistical methods that exploit the increasingly available genetic data to estimate the effective population density and effective dispersal rate (Ringbauer et al., 2017; Rousset, 1997; Chris C. R. Smith et al., 2023; Chris C. R. Smith & Kern, 2023).

The distribution of recent coalescent events between individuals in space can be used to estimate such quantities through the distribution of identity-by-descent (IBD) blocks (Ringbauer et al., 2017). An IBD block is defined as a segment of DNA that has been inherited by a pair of individuals from a common ancestor without being broken by recombination. Here we present `IdentityByDescentDispersal.jl`, a Julia package for estimating effective population densities and dispersal rates from observed spatial patterns of IBD shared blocks.

Statement of need

Ringbauer et al. (2017) proposed an inference scheme for the estimation of effective population density and effective dispersal rate from shared IBD blocks. Despite their promising results, there is to this date no general-purpose software implementation of their method.

In order to make the inference approach available to the broader audience of evolutionary biologists and conservation scientists, we present `IdentityByDescentDispersal.jl`, a Julia (Bezanson et al., 2017) package with an efficient and easy-to-use implementation of the method. The package implements the core equations proposed by Ringbauer et al. (2017) and can be used to perform composite likelihood-based inference using either maximum-likelihood estimation (MLE) or Bayesian inference.

The method of Ringbauer et al. (2017) was limited to a family of functions for the change in effective population density over time of the form $D_e(t) = Dt^{-\beta}$, for which the theory was analytically tractable. In addition, in the paper describing the original approach, the authors used gradient-free optimization to calculate maximum likelihood estimates (MLEs). Our implementation makes two major software contributions. First, we admit composite likelihood calculations for arbitrary functions $D_e(t)$ by evaluating the relevant integrals numerically through Gaussian quadrature rules (Johnson, 2013). This significantly enlarges the space of biologically relevant models that can be fitted. Second, our implementation takes advantage of the powerful Julia ecosystem and the work of Geoga et al. (2022) to provide a version of the composite likelihood that is fully compatible with automatic differentiation (AD), including AD with respect to β. By having a fully AD-compatible composite likelihood, `IdentityByDescentDispersal.jl`

The Journal of Open Source Software

can be used together with standard gradient-based optimization and sampling methods available in the Julia ecosystem, which are typically more efficient than gradient-free methods.

Lastly, our package comes with a template to simulate synthetic datasets and a pipeline for end-to-end analysis from VCF files to final estimates. We believe it will encourage a broader audience to adopt the inference scheme proposed by Ringbauer et al. (2017), motivate further developments and expand its applications.

Other related software includes `spacetrees` (Osmond & Coop, 2024), which estimates dispersal rates from inferred genome-wide genealogies, and `disperseNN2` (Chris C. R. Smith & Kern, 2023), a machine learning framework for predicting the expected per-generation displacement distance from unphased genotypes. `IdentityByDescentDispersal.jl` differs in the nature of the data it uses for inference, but also in that it allows for flexible model-based inference of the effective population density. Moreover, we expect `IdentityByDescentDispersal.jl` to be several orders of magnitude faster than `disperseNN2`. For comparison, training disperseNN2 with n=100 diploids took up to a week on a GPU according to Chris C. R. Smith & Kern (2023), compared to under a minute for IdentityByDescentDispersal.jl to find the MLE on a single CPU.

Overview

`IdentityByDescentDispersal.jl` contains two main sets of functions. The first set has the prefix `expected_ibd_blocks` and allows users to calculate the expected density of IBD blocks per pair of individuals and per unit of block length for various demographic models by solving Equation 1.

$$\mathbb{E}[N_L|r,\theta] = \int_0^\infty G4t^2 \exp(-2Lt) \cdot \Phi(t|r,\theta)\, dt \tag{1}$$

where G is the length of the genome (in Morgan), t is time (generations in the past), L is the length of the block (Morgan) and r is the geographical distance in the present (at time $t=0$) between the two individuals. $\Phi(t|r,\theta)$ is the instantaneous coalescence rate at time t of two homologous loci that are initially r units apart under the demographic model with parameters θ. A slightly more complicated expression that accounts for chromosomal edges and diploidy is the default in `IdentityByDescentDispersal.jl`.

The second set of functions has the prefix `composite_loglikelihood` and allows users to directly compute the composite likelihood of the data by assuming the observed number of IBD blocks whose lengths fall in a small bin $[L, L+\Delta L]$ and are shared by a pair of individuals r units apart follows a Poisson distribution with mean $\lambda = \mathbb{E}[N_L|r,\theta]\Delta L$.

`IdentityByDescentDispersal.jl` allows for three different parameterizations of the effective population density function: a constant density, a power-density, and a user-defined density (see Table 1).

Table 1: `IdentityByDescentDispersal.jl` functions support three different parameterizations that are indicated by their respective suffixes.

Function suffix	$D_e(t)$ formula	Parameters	Solver
`constant_density`	$D_e(t) = D$	D, σ	Analytically
`power_density`	$D_e(t) = Dt^{-\beta}$	D, β, σ	Analytically
`custom`	User-defined	User-defined and σ	Numerically

The Julia package is accompanied by two additional resources. First, we provide a simulation template in SLiM for forward-in-time population genetics simulation in a continuous space with tree-sequence recording (Haller et al., 2019; Haller & Messer, 2023). This template can be used to assess model assumptions, guide empirical analysis, and perform simulation-based calibration. Assessing the performance of the method with synthetic datasets is a crucial step, as it is known that errors in the detection of IBD blocks are common (S. R. Browning & Browning, 2012) and that inferences based on composite likelihood tend to be overconfident, underestimating posterior uncertainty and yielding too narrow confidence intervals.

Second, we have also implemented a bioinformatics pipeline that carries out a complete analysis from detecting IBD blocks to finding the MLE of the effective population density and the effective dispersal rate. It is shared as a Snakemake pipeline, a popular bioinformatics workflow management tool (Mölder et al., 2021). It takes as input a set of phased VCF files, their corresponding genetic maps and a CSV file containing pairwise geographical distances between individuals. The pipeline detects IBD blocks using HapIBD (Zhou et al., 2020), post-processes them with Refined IBD (B. L. Browning & Browning, 2013) and produces a CSV file compatible with subsequent analysis with `IdentityByDescentDispersal.jl` via the `preprocess_dataset` function.

Both the SLiM simulation template and the Snakemake pipeline can be found in the GitHub repository at https://github.com/currocam/IdentityByDescentDispersal.jl.

Example

In this section, we demonstrate how `IdentityByDescentDispersal.jl` can be used together with `Turing.jl` (Fjelde et al., 2025), a popular probabilistic programming language for Bayesian inference, to analyse a dataset we simulate in the documentation. We analyse error-free IBD blocks shared by 100 diploid individuals from a constant-density population with parameters $D_{\text{true}} \approx 200$ diploids/km² and $\sigma_{\text{true}} \approx 0.100$ km/generation.

`IdentityByDescentDispersal.jl` has extensive documentation that covers the underlying theory behind the method, how to effectively simulate synthetic datasets, various demographic models, and inference algorithms. We refer the reader to the documentation for more details, which can be found at https://currocam.github.io/IdentityByDescentDispersal.jl/.

Thanks to `Turing.jl`, we can perform Bayesian inference with a wide range of popular Monte Carlo algorithms. Figure 1 shows the estimated pseudo-posterior obtained through doing inference with the composite likelihood.

The Journal of Open Source Software

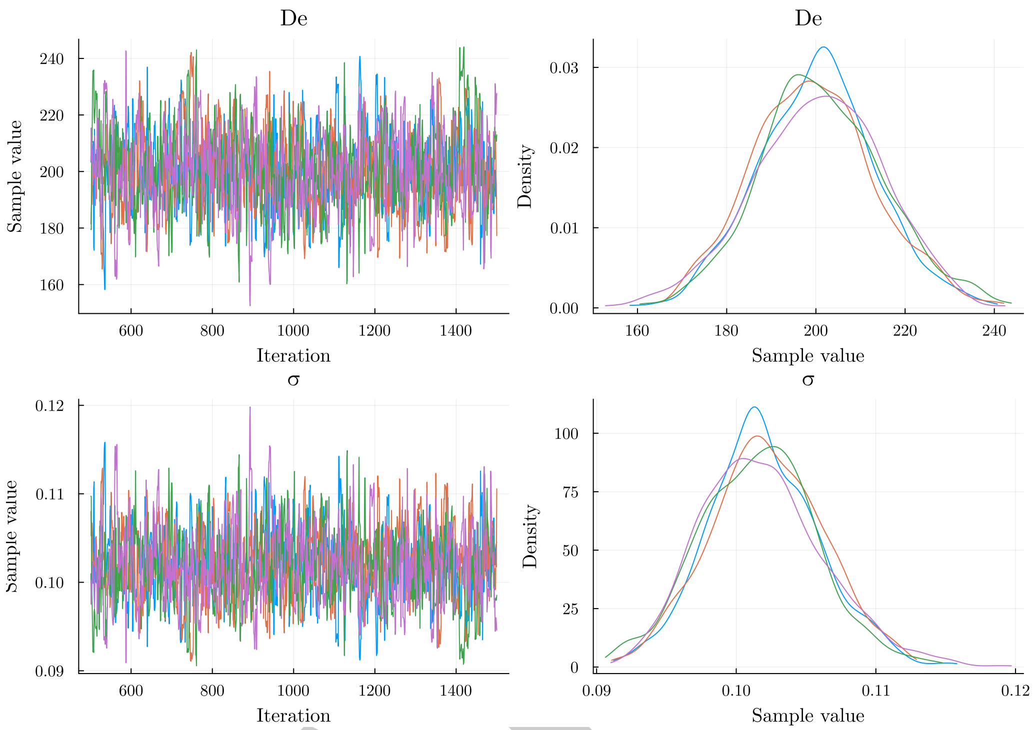

Figure 1: Estimated pseudo-posterior obtained by doing inference with the composite likelihood. The pseudo-posterior concentrates around the true values ($\mathbb{E}[D|\text{data}] \approx 200$ and $\mathbb{E}[\sigma|\text{data}] \approx 0.102$, respectively). However, we don't generally expect to be well-calibrated, and we suggest performing simulation-based calibration checking.

Figure 1 was generated by the following snippet of Julia code, which reads the processed data CSV from the provided Snakemake pipeline.

```
using CSV, DataFrames, Turing, StatsPlots, IdentityByDescentDispersal
df = CSV.read("ibd_dispersal_data.csv", DataFrame)
contig_lengths = [1.0]
@model function constant_density(df, contig_lengths)
    De ~ Truncated(Normal(1000, 100), 0, Inf)
    σ ~ InverseGamma(1, 1)
    Turing.@addlogprob! composite_loglikelihood_constant_density(
      De, σ, df, contig_lengths
    )
end
m = constant_density(df, contig_lengths)
chains = sample(m, NUTS(), MCMCThreads(), 1000, 4)
plot(chains)
```

We can also easily compute the MLEs of the same demographic model,

```
mle_estimate = maximum_likelihood(
  m; lb=[0.0, 0.0], ub=[1e8, 1e8]
)
coeftable(mle_estimate)
```

which estimates $D_{\text{MLE}} \approx 199$ diploids/km² (95% CI: 171–226) and $\sigma_{\text{MLE}} \approx 0.102$ km/generation (95% CI: 0.094–0.110). The 95% confidence interval is computed from the Fisher information matrix.

Availability

`IdentityByDescentDispersal.jl` is a registered Julia package available through the official General registry. Its source code is hosted on GitHub at https://github.com/currocam/IdentityByDescentDispersal.jl.

Acknowledgements

We acknowledge financial support from the Research Foundation - Flanders (FWO). This work was supported by FWO-11A2P26N (F.C.J, H.S), FWO-G0A9B24N (F.C.J, H.S), FWO-1272625N (A.Z., H.S) and FWO-12A8423N (E.L.R.D.K., H.S).

References

Bezanson, J., Edelman, A., Karpinski, S., & Shah, V. B. (2017). Julia: A Fresh Approach to Numerical Computing. *SIAM Review*, *59*(1), 65–98. https://doi.org/10.1137/141000671

Browning, B. L., & Browning, S. R. (2013). Improving the accuracy and efficiency of identity-by-descent detection in population data. *Genetics*, *194*(2), 459–471. https://doi.org/10.1534/genetics.113.150029

Browning, S. R., & Browning, B. L. (2012). Identity by descent between distant relatives: Detection and applications. *Annual Review of Genetics*, *46*, 617–633. https://doi.org/10.1146/annurev-genet-110711-155534

Driscoll, D. A., Banks, S. C., Barton, P. S., Ikin, K., Lentini, P., Lindenmayer, D. B., Smith, A. L., Berry, L. E., Burns, E. L., Edworthy, A., Evans, M. J., Gibson, R., Heinsohn, R., Howland, B., Kay, G., Munro, N., Scheele, B. C., Stirnemann, I., Stojanovic, D., … Westgate, M. J. (2014). The Trajectory of Dispersal Research in Conservation Biology. Systematic Review. *PLOS ONE*, *9*(4), e95053. https://doi.org/10.1371/journal.pone.0095053

Fjelde, T. E., Xu, K., Widmann, D., Tarek, M., Pfiffer, C., Trapp, M., Axen, S. D., Sun, X., Hauru, M., Yong, P., Tebbutt, W., Ghahramani, Z., & Ge, H. (2025). Turing.jl: A general-purpose probabilistic programming language. *ACM Trans. Probab. Mach. Learn.* https://doi.org/10.1145/3711897

Geoga, C. J., Marin, O., Schanen, M., & Stein, M. L. (2022). *Fitting Matérn Smoothness Parameters Using Automatic Differentiation*. arXiv. https://doi.org/10.48550/ARXIV.2201.00090

Haller, B. C., Galloway, J., Kelleher, J., Messer, P. W., & Ralph, P. L. (2019). Tree-sequence recording in SLiM opens new horizons for forward-time simulation of whole genomes. *Molecular Ecology Resources*, *19*(2), 552–566. https://doi.org/10.1111/1755-0998.12968

Haller, B. C., & Messer, P. W. (2023). SLiM 4: Multispecies Eco-Evolutionary Modeling. *The American Naturalist*, *201*(5), E127–E139. https://doi.org/10.1086/723601

Johnson, S. G. (2013). *QuadGK.jl: Gauss–Kronrod integration in Julia*. https://github.com/JuliaMath/QuadGK.jl.

Mölder, F., Jablonski, K. P., Letcher, B., Hall, M. B., Tomkins-Tinch, C. H., Sochat, V., Forster, J., Lee, S., Twardziok, S. O., Kanitz, A., Wilm, A., Holtgrewe, M., Rahmann, S., Nahnsen, S., & Köster, J. (2021). Sustainable data analysis with Snakemake. *F1000Research*, *10*, 33. https://doi.org/10.12688/f1000research.29032.2

Osmond, M., & Coop, G. (2024). Estimating dispersal rates and locating genetic ancestors with genome-wide genealogies. *eLife*, *13*, e72177. https://doi.org/10.7554/eLife.72177

Ringbauer, H., Coop, G., & Barton, N. H. (2017). Inferring Recent Demography from

The Journal of Open Source Software

Isolation by Distance of Long Shared Sequence Blocks. *Genetics*, *205*(3), 1335–1351. https://doi.org/10.1534/genetics.116.196220

Rousset, F. (1997). Genetic Differentiation and Estimation of Gene Flow from F-Statistics Under Isolation by Distance. *Genetics*, *145*(4), 1219–1228. https://doi.org/10.1093/genetics/145.4.1219

Smith, Chris C. R., & Kern, A. D. (2023). disperseNN2: A neural network for estimating dispersal distance from georeferenced polymorphism data. *BMC Bioinformatics*, *24*(1), 385. https://doi.org/10.1186/s12859-023-05522-7

Smith, Chris C. R., Tittes, S., Ralph, P. L., & Kern, A. D. (2023). Dispersal inference from population genetic variation using a convolutional neural network. *Genetics*, *224*(2), iyad068. https://doi.org/10.1093/genetics/iyad068

Zhou, Y., Browning, S. R., & Browning, B. L. (2020). A Fast and Simple Method for Detecting Identity-by-Descent Segments in Large-Scale Data. *American Journal of Human Genetics*, *106*(4), 426–437. https://doi.org/10.1016/j.ajhg.2020.02.010