Timesweeper: Accurately Identifying Selective Sweeps Using Population Genomic Time Series

Logan S. Whitehouse[1] and Daniel R. Schrider[1*]

[1] Department of Genetics, University of North Carolina, Chapel Hill, North Carolina, USA

[*]Corresponding author

E-mail: drs@unc.edu

ABSTRACT

Despite decades of research, identifying selective sweeps, the genomic footprints of positive selection, remains a core problem in population genetics. Of the myriad methods that have been made towards this goal, few are designed to leverage the potential of genomic time-series data. This is because in most population genetic studies only a single, brief period of time can be sampled for a study. Recent advancements in sequencing technology, including improvements in extracting and sequencing ancient DNA, have made repeated samplings of a population possible, allowing for more direct analysis of recent evolutionary dynamics. With these advances in mind, here we present `Timesweeper`, a fast and accurate convolutional neural network-based tool for identifying selective sweeps in data consisting of multiple genomic samplings of a population over time. `Timesweeper` utilizes this serialized sampling of a population by first simulating data under a demographic model appropriate for the data of interest, training a 1D Convolutional Neural Network on said model, and inferring which polymorphisms in this serialized dataset were the direct target of a completed or ongoing selective sweep. We show that `Timesweeper` is accurate under multiple simulated demographic and sampling scenarios, and identifies selected variants with impressive resolution. In sum, we show that more accurate inferences about natural selection are possible when genomic time-series data are available; such data will continue to proliferate in coming years due to both the sequencing of ancient samples and repeated samplings of extant populations with faster generation times. Methodological advances such as `Timesweeper` thus have the potential to help resolve the controversy over the role of positive selection in the genome. We provide `Timesweeper` both as a Python package and Snakemake workflow for use by the community.

AUTHOR SUMMARY

Despite decades of research, detecting regions of the genome responsible for adaptation remains a major challenge in evolutionary biology. Such efforts have the potential to give us a clearer picture of the adaptive process, from the selective pressures that populations have had to adapt to, to the genes and mutations that have facilitated responses to these pressures. With improvements in the cost-effectiveness of DNA sequencing, it is becoming possible to procure sequence data obtained from populations at multiple timepoints, allowing for a more direct examination of the speed of evolutionary change across the genome. Such time-series data could make it easier to detect selective sweeps, in which an adaptive mutation rapidly increases in frequency, or "sweeps" through a population, thereby revealing the genetic loci responding to natural selection. We present a novel machine learning method that is capable of detect sweeps from population genomic time-series data with excellent accuracy. This method, called `Timesweeper`, is able to narrow down the genomic region containing the sweep with high precision. Timesweeper is open source, fast, flexible, easy to use, and can be applied to unphased or even un-genotyped sequencing data so long as allele frequency estimates are available.

INTRODUCTION

Over a century after the modern evolutionary synthesis, there is still a great deal of controversy over the role of natural selection in molecular evolution. In particular, the impact of positive selection, in which beneficial alleles are favored by selection and thus increase in frequency in the population, is hotly debated [1–3]. This controversy extends to the role of selective sweeps [4], in which positively selected mutations rapidly sweep through a population thereby drastically

reducing and altering patterns of genetic diversity in the vicinity of the selected locus [5,6], with some studies finding that signatures of selective sweeps alter genome-wide patterns of polymorphism [7–11] , and others purporting that these signatures may be largely or even entirely false positives [12].

A number of methods have been devised to make inferences about positive selection in the recent past, including recently completed or even ongoing selective sweeps, on the basis of snapshots of present-day variation obtained from a set of genomes sampled from the population(s) of interest. These methods typically examine one of several aspects of genetic diversity that are somewhat distinct but are each indicative of an allele that has spread rapidly enough such that there has been little time for recombination and mutation to introduce diversity into the class of haplotypes carrying the sweeping allele. These signatures of selection include: characteristic skews in the frequencies of neutral alleles linked to the selected site [13–15], reduced haplotypic diversity around the selected site [9,16,17], the presence of abnormally long-range haplotypes [18–20], increased linkage disequilibrium [21], especially on either flank of the selected site [22], and elevated divergence between closely related populations in the case of local adaptation [23].

One pernicious obstacle to efforts to detect selection is demographic change. For example, strong population bottlenecks, which involve a recovery from a period of drastically reduced population size, can result in rapid changes in allele frequencies that can produce false-positive signals of selective sweeps, especially if the demographic history is unknown [24–26]. Some recent methods, which combine each of the signatures of selection described above via machine learning in order to identify the multidimensional spatial patterns of diversity along the chromosome that sweeps produce [26–32], have proved to be far more robust to demographic events [27]. However, even these methods may experience an appreciable loss of power under

non-equilibrium demographic histories, especially if they are not adequately modeled during the training process [26,27]. Thus, in spite of decades of theoretical and empirical research, detecting selective sweeps remains a major challenge.

The difficulty in discriminating between positive selection and purely neutral processes could potentially be alleviated by using population genomic time-series data, in which genomes are sampled across a range of timepoints rather than from a single contemporaneous point. Such data allow us to more directly track the trajectories of any potentially sweeping alleles/haplotypes, and to determine if they appear to be spreading faster than other alleles segregating in the population. Population genomic time-series data are becoming more and more prevalent, in part because it is often feasible, and increasingly affordable, to collect and sequence longitudinal samples of populations from species with rapid generation times [33–36]. For example, studies conducted in collections repeated each season in *Drosophila* over a period of several years have presented evidence that seasonal oscillations in the frequencies of some alleles may be adaptive [35], and that temporal variances in allele frequencies are explained in part by positive selection [37]. Even in organisms with longer generation times, the production of time-series data may be possible in some cases. For example, a large number of ancient and archaic human genomes have been sequenced, and this has allowed researchers to directly ask whether alleles proposed to be under positive selection have indeed rapidly increased in frequency, and to more precisely infer when these frequency changes occurred [8,38–44]. Even in organisms with longer generation times, the production of time-series data may be possible in some cases. For example, a large number of ancient and archaic human genomes have been sequenced, and this has allowed researchers to directly ask whether alleles proposed to be under positive selection have indeed

rapidly increased in frequency, and to more precisely infer when these frequency changes occurred [8,38–44].

As population genomic time-series continue to proliferate, novel methods that can fully take advantage of the potential of such data are required. One area where such methods have been in development for a number of years is experimental evolve-and-resequence (E&R) studies (reviewed in [45], in which lab populations are subjected to controlled selective pressures and often sampled during the course of the experiment (e.g. [46–49]). Many of these methods take advantage of the fact that E&R studies often produce experimental replicates, allowing for more confident identification of selected loci in cases where positive selection has acted on these loci in more than one replicate [50]. However, in natural populations, controlled replicates are unavailable, and we therefore require methods that can accurately detect targets of selection in a single population genomic time-series.

Here, we describe a deep learning method for detecting positive selection from population genomic time series. This method, called `Timesweeper`, constructs an image representation of temporal changes in allele or haplotype frequencies around a focal site and then, with the aid of a convolutional neural network (CNN), uses these data to classify the site as either experiencing a recent selective sweep or not. We show that this method has remarkable power to detect selection, discriminating between selected and unselected regions with high sensitivity and specificity. `Timesweeper` is also able to localize sweeps with impressive resolution, often identifying selected polymorphisms and rejecting closely linked polymorphisms with high confidence. This method is trained on simulated data under a user-specified demographic model and sampling scheme, but we show that it can be robust to at least some scenarios of demographic model misspecification during training. Moreover, because the default mode of `Timesweeper`

examines only changes in sample allele frequencies over time, it is applicable to data for which phased haplotypes or even high-confidence genotypes are not available, including pooled population genomic sequence data or lower coverage genomes such as ancient human DNA. We demonstrate this practical utility on an E&R dataset of 10 experimental replicate population of *D. simulans* [51], finding that we are able to identify selective sweep signatures and replicate candidate sweep loci at a high rate. `Timesweeper` is also computationally efficient and does not require GPUs in order to finish the training step within several minutes. We argue that methodologies such as `Timesweeper`, which seek to better leverage population genomic time-series to detect natural selection, have the potential to not only uncover targets of positive selection with unprecedented accuracy, but to help resolve the lingering controversy over the role of positive selection in shaping patterns of diversity within and between species.

RESULTS

Representations of population genomic time-series data

We devised two representations of genomic time-series data with the goal of capturing information that would be informative about the presence of selective sweeps occurring during the sampling period. The first of these is an $l \times m$ matrix, where l is the number of polymorphisms and m is the number of sampling timepoints. The motivation for this format is that the frequency trajectories of all alleles, including that of a focal allele at the center of the window, can be tracked through time, along with spatial information along the chromosome—not only should the focal allele increase in frequency if it is positively selected, but some nearby alleles may also hitchhike to higher frequencies, with more distant polymorphisms experiencing less of a hitchhiking effect. We refer

to the mode of `Timesweeper` that uses this information to detect sweeps as the Allele Frequency Tracker (AFT).

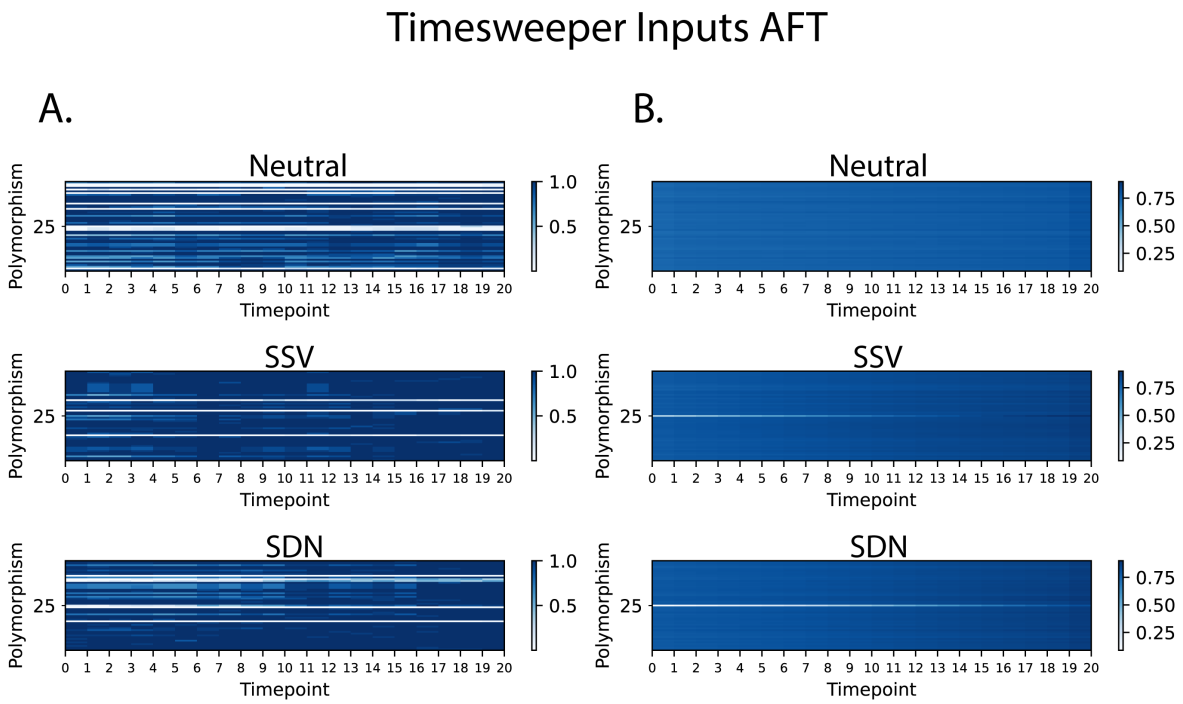

Figure 1. The allele frequency tracking (AFT) method's input representation of simulated data with 20 sampled timepoints each with a sample size of 10 diploid individuals. Each plot shows allele frequency changes over time, where the central polymorphism (position 25, as 51 polymorphisms are shown in these examples) is either evolving neutrally (Neutral), a sweeping single-origin *de novo* mutation (SDN), or a sweeping previously neutral standing variant (SSV). At each site, the frequency of the allele that has become more common from the first timepoint to the last is shown over time (*x*-axis), and polymorphisms (*y*-axis) are arranged in the same order they appear along the chromosome, such that the central polymorphism—the one to be classified—is shown in the middle. Individual example inputs into the AFT neural network for each class are shown in panel A, and the average of all simulated inputs shown in panel B.

The second data representation is a $k \times m$ matrix, where k is the number of distinct haplotypes observed and m is again the number of timepoints. The haplotypes in this matrix are sorted so that the haplotype with the highest net change in frequency across the sampling period is shown at the bottom, and the remaining haplotypes are sorted by similarity to this focal haplotype which is *a priori* the one most likely to be experiencing any positive selection. This

second representation thus tracks the frequency changes of all haplotypes observed in the entire sample, which should be especially pronounced for any haplotypes harboring a sweeping allele. We refer to the mode of `Timesweeper` that uses data in this format as the Haplotype Frequency Tracker (HFT). Both of these formats allow for easy input into convolutional neural networks, and the two modes of `Timesweeper` both use 1D convolutions over the time axis to track allele or haplotype frequency changes through time, respectively (Methods).

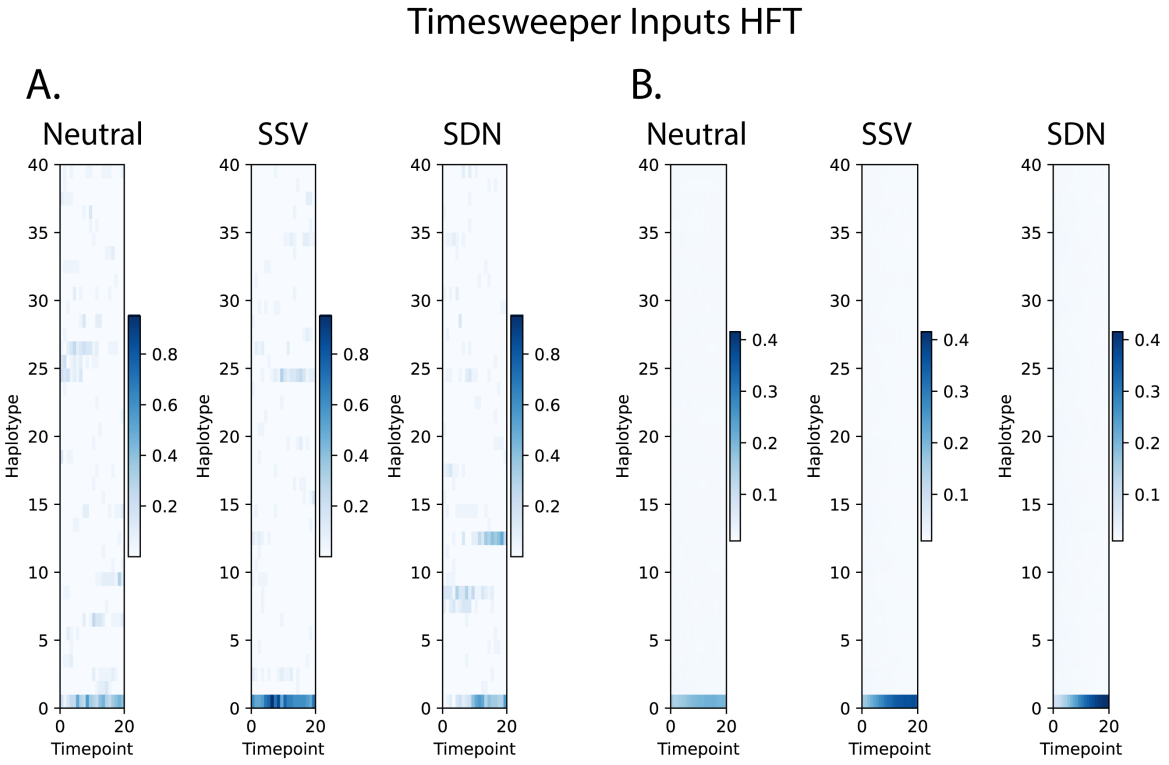

Figure 2. The haplotype frequency tracking (HFT) method's input representation of simulated data with 20 sampled timepoints each with a sample size of 10 diploid individuals. Haplotype frequencies are arranged such that the bottom-most haplotype is the one with the largest net increase in frequency over the course of the sampling period, the haplotype most similar to this is the next from the bottom, and so on. The same scenarios are shown as in Figure 1, with individual examples shown in panel A, and the average of all simulated inputs shown in panel B.

160 Figure 1 shows the expected values of each entry in these matrices, as computed from
161 forward simulations of selective sweeps under a constant population size of 10,000 and a selection
162 coefficient, *s*, of 0.05 (Methods). This visualization shows that the rapid spread of selected alleles
163 (the central row in the sweep examples shown in Figure 1B) and haplotypes (the bottom row in
164 Figure 2B) is readily observable in this representation in the average case, and the selected and
165 unselected cases are readily discernable. However, we note that individual selective sweeps may
166 depart substantially from this expectation (random individual examples from each class are shown
167 in Figure 1A and). In the next section we investigate the extent to which individual examples are
168 correctly classified as undergoing selective sweeps evolving neutrally on the basis of these
169 representations.

170

171 **Neural networks can detect and localize sweeps from time-series data with high accuracy**

172 Having constructed the data representations described above, we sought to train neural networks
173 to distinguish between three distinct classes: 1) Neutrally evolving regions, 2) Sweeps caused by
174 selection on a single-origin *de novo* variant (the SDN model), and 3) selection on standing variation
175 (the SSV model). Other approaches using these data are possible—for example, one could adapt
176 existing composite-likelihood ratio approaches [24,52] to use the information in the AFT
177 representation to detect sweeps from time-series data. However, a simulation-based CNN would
178 allow us to model the joint distribution of all of the allele or haplotype frequency changes observed
179 in the vicinity of a sweep. As we show below, this approach yielded excellent inferential power.

180 Here and in the following section, we show that allele frequency trajectories are highly
181 informative in the context of 1D CNNs, demonstrating high accuracy across all time-series

schemes tested. We began by benchmarking performance on a scenario of 20 timepoints with 10 individuals at each timepoint on a test data set held out during training, and found that the AFT method achieves an area under the ROC curve (AUC) of 0.95 when s=0.05 and the sweep began 50 generations prior to sampling (Figure 3B); we examine more combinations of these parameters in the section below. Closer inspection of class predictions on test data reveals the allele frequency model performs well when discriminating between hard and

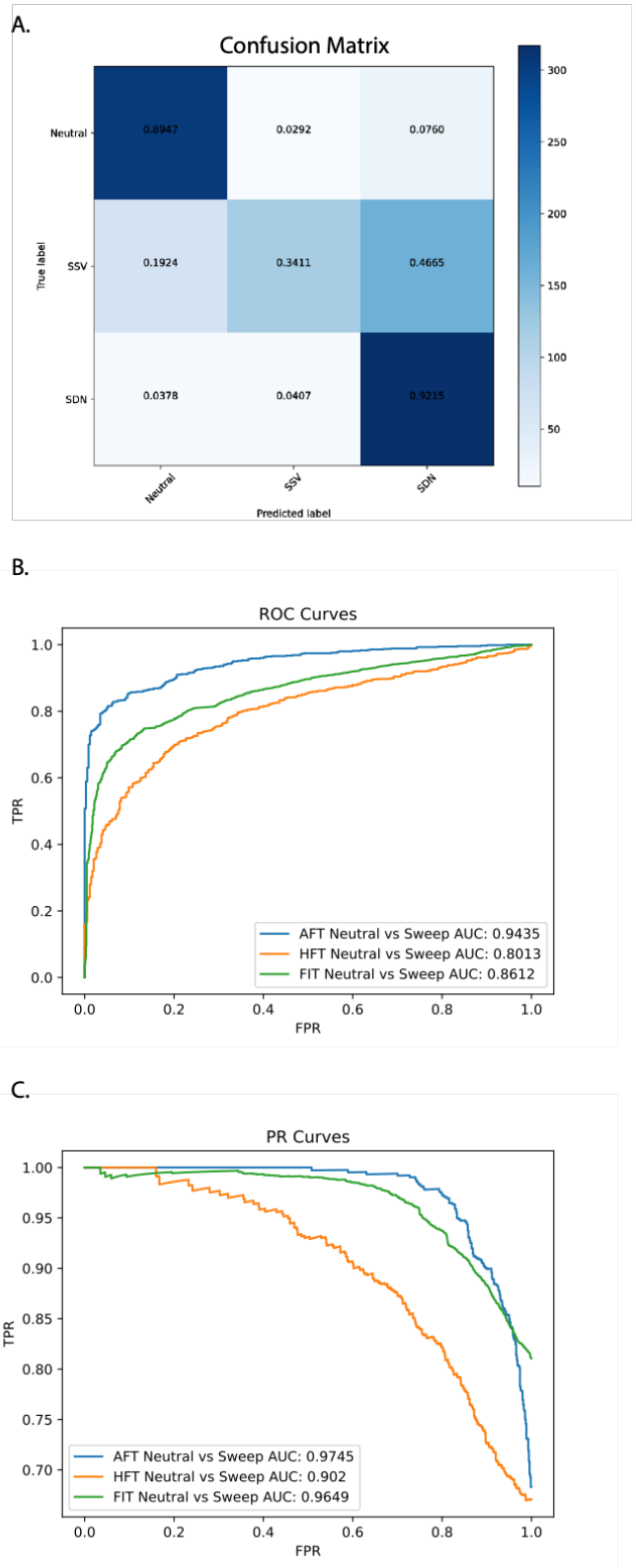

Figure 3. Benchmarking `Timesweeper` using single-locus test data. (I.e. only one classification—that made for the central polymorphism—is done for each simulated example.) **A.** Confusion matrix showing the AFT method's accuracy on simulated single-locus test data with a population size of 10,000 diploid individuals, a selection coefficient of 0.05, 20 sampled timepoints with 10 diploids sampled in each, and a sweep onset time of 50 generations prior to the sampling period. **B-C.** ROC and PR curves for AFT, HFT, and FIT on single-locus test data. For AFT and HFT, SSV and SDN classification scores were summed at each call to yield a single binarized sweep probability to produce these curves, and for FIT scores we used 1 minus the *p*-value.

195 soft sweeps (Figure 3A). The HFT method is somewhat less accurate, achieving an AUC of 0.78,
196 and is noticeably less capable of differentiating soft sweeps from neutral and hard scenarios. FIT's
197 performance falls in the middle of these two methods, with an AUC of 0.86, although we note that
198 FIT does not attempt to distinguish between hard and soft sweeps as it is a hypothesis test and can
199 thus only reject neutrality or not. Qualitatively similar results are shown in the precision-recall
200 curves in Figure 3C.

201 The results described above were obtained by examining simulated test data where either
202 the central polymorphism was selected, or not, and making a single classification for this focal site
203 (hereafter referred to as single-locus testing). However, in practice one would scan across an entire
204 chromosome and make classifications at each polymorphism before moving on to the next. Thus,
205 we next benchmarked the same `Timesweeper` classifiers described above on a set simulated
206 chromosomes 500 kilobases long—100 each of the neutral, SSV, and SDN scenarios. We found
207 that `Timesweeper` was not only able to accurately detect true sweep locations but has a low
208 false positive rate across the flanking neutral regions along the chromosome (Figure 4). The HFT
209 method, on the other hand, exhibits comparatively low resolution in localizing sweeps (S1 Fig),
210 likely because it examines a representation of haplotype frequencies that has no spatial information
211 about any potentially sweeping alleles. We find that FIT has a low false positive rate outside of
212 the sweep site, but has relatively low sensitivity to detect sweeps at a *p*-value cutoff of 0.05 (S1
213 Fig), although we note that its good performance relative to HFT shown in the ROC and precision-
214 recall curve in Figure 4 suggests that this method does effectively rank selected sites above
215 unselected sites.

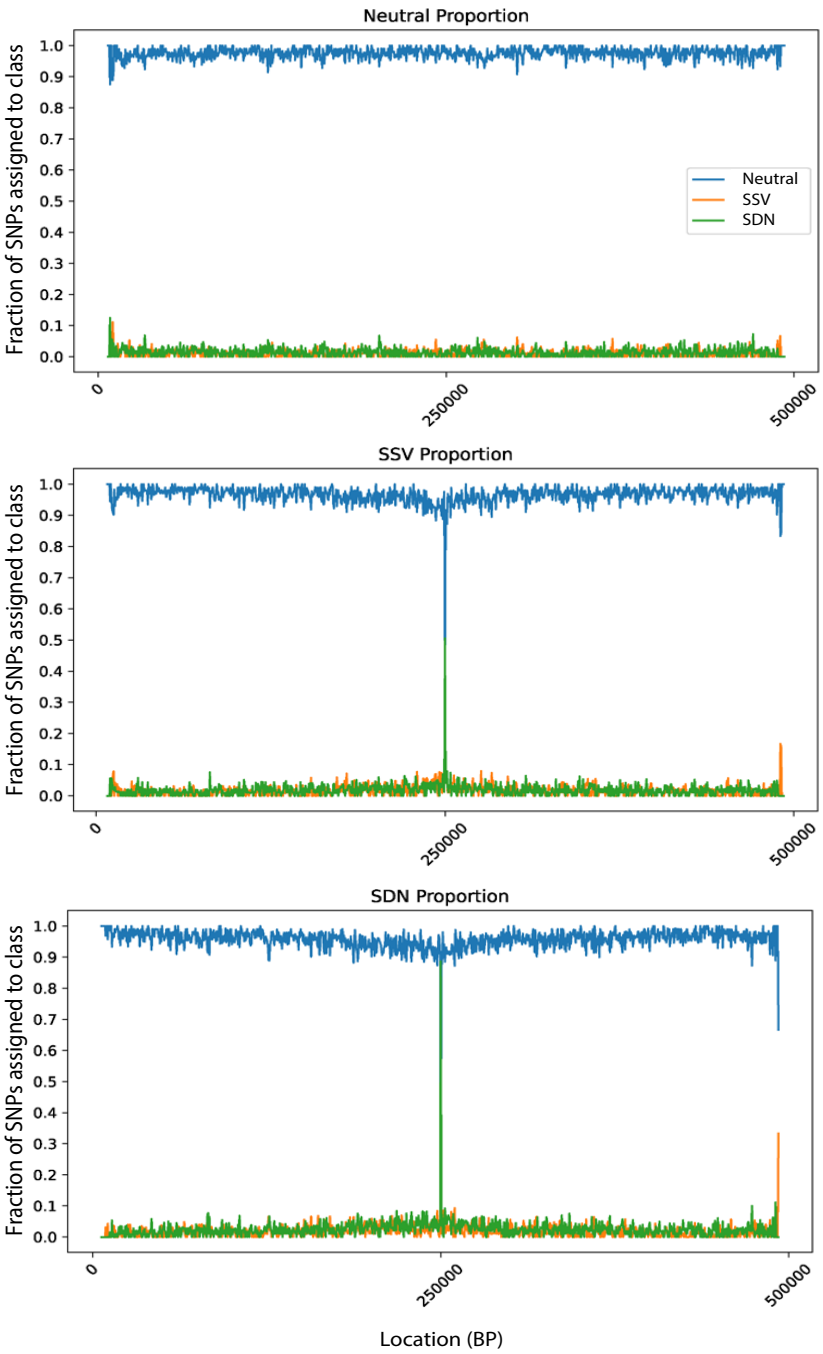

Figure 4. Resolution of `Timesweeper`'s sweep inferences on simulated 500 kb chromosomes. The top panel shows, for neutral simulations, the fraction of polymorphisms at each location on the chromosome classified as neutral (blue), an SSV sweep (orange), and an SDN sweep (green). The bottom two panels show the same information for simulated SSV and SDN sweeps occurring in the center of the chromosome, respectively. Within each panel, polymorphisms are binned into 1 kb windows and the average classification within those windows is shown, with the exception of the central polymorphism which is placed in a bin of its own.

217 `Timesweeper` performs well with a variety of selection coefficients using the sampling
218 schema described above (Figure S8). Testing on a range of *s* we found that ROC AUC is minimized
219 at 0.71 with *s*=0.005 and maximized at 0.96 with *s*=0.05. We note that extremely strong values of
220 *s* tend to reduce the testing ROC AUC slightly (e.g. *s*=0.5 results in ROC AUC of 0.89), this is
221 likely due to the speed of the sweep moving faster than the sampling window resulting in reduced
222 amounts of information along the time series.

223 Training each network on 7,500 replicates (2,500 of each class) without GPUs required
224 roughly one minute on 4 CPU cores and less than 1Gb of RAM. Classifying a simulated test set of
225 1008 polymorphisms with 10 diploid individuals sampled at each of 20 timepoints required less
226 than 60 seconds for each trained neural network, resulting in a total of 3 minutes runtime when
227 both the AFT and HFT methods as well as FIT are utilized.

228

229 **Testing the effect of sampling scheme and selection strength on accuracy to detect selection**

230 To test effectiveness of `Timesweeper` under different sampling scenarios we experimented with
231 varying three parameters: the sample size per timepoint, the number of timepoints sampled, the
232 timing of sampling relative to the onset of selection, and the selection coefficient. For each
233 experiment described below, the parameters of the training set matched those of the test set. With
234 the exception of these parameter changes, all simulation, training, and testing was performed in
235 the same manner as above. Accuracy reported in this section was estimated from 7,500 test
236 replicates (2,500 each for the SDN, SSV, and neutral classes) for each parameter combination
237 tested.

238 Sample Size: In order to test the effect of sample size on classification accuracy we trained a
239 `Timesweeper` classifier for each of four scenarios: 2, 5, 10, and 20 diploid individuals sampled
240 at each of 20 timepoints. Sampling started 50 generations post-selection in the SDN and SSV
241 scenarios, and other than sample sizes all other components of the simulation were identical
242 (example and average inputs shown in S2 Fig). We found that increasing number of samples
243 marginally increases performance of the AFT model: with two individuals per timepoint the AUC
244 for the AFT method was 0.95, and with 20 individuals per timepoint this had increased to 0.97
245 (Figure 5 and S3 Fig. This indicates that `Timesweeper` is flexible enough to handle low sample
246 sizes and still perform adequately.

247 Number of Timepoints: Next, we tested the effect of adjusting the number of timepoints on
248 `Timesweeper`'s accuracy. In this scenario the total number of sampled individuals was 200 for
249 every sampling scheme, so accuracy would be solely affected by the timing and number of samples
250 as opposed to the total amount of genetic data being examined. Scenarios tested included 100, 40,
251 20, 10, 5, 2, and 1 timepoints (example and average inputs shown in S4 Fig). In all scenarios, aside
252 from the single-timepoint scenario, the sampling times were evenly distributed across a 200-
253 generation span starting 50 generations after the onset of selection. For the single-timepoint
254 scenario only the final generation of this 200-generation span was sampled (i.e. a sample of 200
255 individuals was taken 250 generations after the onset of selection). Among time-series schemes,
256 there is an initial accuracy increase from 1 (0.50 AUC) to 2 (0.85 AUC) to 5 timepoints (0.92
257 AUC), however there is only a marginal increase in performance for 10 to 100 timepoints (0.93 to
258 0.96 AUC) (Figure 5 and S5 Fig). There is a similar increase in performance for the 3-class
259 classification, the majority of misclassifications being between SDN and SSV scenarios, with few
260 false positives or negatives (sweeps labeled as neutral or vice versa; see confusion matrices in S5

261 Fig). Overall 3-class classification performance improves with more than 10 timepoints (67%
262 accuracy) to 72% with 20 timepoints, and only marginal differences among 20, 40, and 100
263 timepoints. These results indicate `Timesweeper` is accurate under a variety of sampling
264 schemes, and even small numbers of sampling timepoints may yield substantially better
265 performance than single-timepoint data.

266 Post-selection Timing: We also simulated a set of scenarios that test the effect of varying when
267 sampling occurs relative to the start the sweep. In these experiments all scenarios were simulated
268 with 20 timepoints each with 10 sampled individuals. The sampling window for each remained a
269 span of 200 generations with the 20 timepoints evenly spaced within. The starting time for the
270 sampling window was set to either 0, 25, 50, 100, or 200 generations after the onset of selection
271 (example and average inputs shown in S6 Fig). Timesweeper performed best (0.96 AUC) on data
272 sampled beginning 0 to 100 generations post-selection (AUC 0.96-0.98), with the lowest AUC
273 observed being 0.86 AUC for sampling beginning 200 generations post-selection. Reduced
274 accuracy under sampling schemes occurring very late relative to the onset of selection is expected,
275 as some sweeps may reach fixation and their signatures may start to degrade prior to the completion
276 of sampling. However, we note that the changes in accuracy are almost entirely due to errors in
277 classing sweeps as SDN vs. SSV, while sweeps and neutrally evolving regions are distinguished
278 from one another relatively accurately (Figure 6 and S7 Fig). These tests show that
279 `Timesweeper` can be trained to accurately detects sweeps that occurred at a variety of times
280 relative to the sampling period, and is best able to differentiate between SDN and SSV sweeps
281 when the sampling scheme captures as much of the sweep trajectory as possible.

Selection coefficient: We simulated training and test sets with selection coefficients ranging from 0.005 to 0.5. For each of these simulations, we sampled at 20 timepoints, beginning 50 generations after the onset of the sweep, and sampled 10 diploid individuals at each timepoint (example and average inputs shown in S8 Fig). We find that, for the AFT, accuracy is fairly low when s=0.005 (AUC= 0.71 for distinguishing between sweeps and neutrality) but increases rapidly with s (AUC=0.81 and 0.96 when s=0.01 and s=0.05, respectively), before tailing off at higher selection coefficients (S9 Fig), perhaps because in these cases much of the sweep has already finished prior to most of the sampling interval. We also note that the HFT has lower accuracy for most values of s, it does perform better than AFT for the largest value of s examined (S9 Fig), perhaps because this input

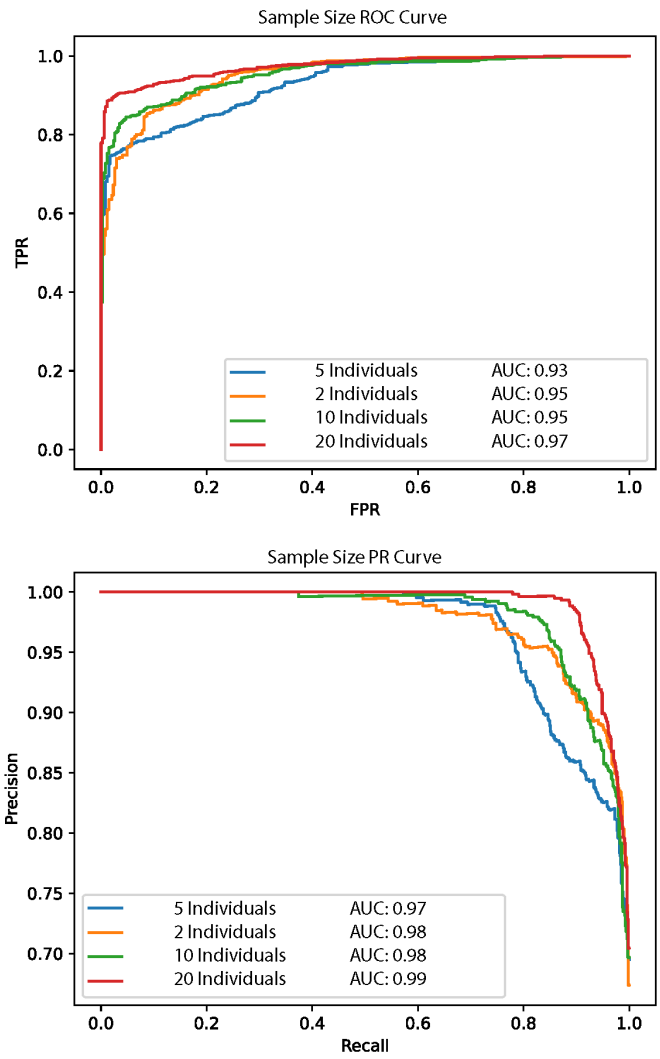

Figure 5. `Timesweeper`'s performance on data with different sample sizes. ROC Curves and PR Curves show the AFT method's performance on datasets with sample sizes ranging from 2 to 20. A total of 20 timepoints were sampled for each dataset, and these sample times are equally spaced across the 200 generation sampling period beginning 50 generations after the onset of selection. The total number of individuals sampled across the entire sampling window is equal to 20 multiplied by the sample size number. For example, "5 Individuals" corresponds to 20 samples of 5 individuals each, resulting in a total of 100 individuals.

representation is more sensitive to selective sweeps even if they have largely already completed prior to sampling.

Quantifying the effects of demographic model misspecification on `Timesweeper`'s accuracy

As with all parametric approaches to population genetic inference, the demographic model used could impact the accuracy of downstream inference [53,54]. To characterize `Timesweeper`'s accuracy in spite of highly misspecified demographic history we performed cross-model benchmarking between a constant-size demographic model (referred to as the simple model) and the Out of Africa (OoA) described in [55].

We found that model and sampling misspecification has a different effect depending on the direction of model misspecification. In the case of `Timesweeper` being trained on the OoA model and benchmarked on the simple model `Timesweeper` performs well, achieving an ROC AUC of 0.89 on the neutral vs selection classification task, whereas training on the simple model and predicting on the OoA model results in an uninformative classifier that almost always infers neutrality, perhaps due to the slower rate of drift in the expanding population sampled in our OoA model (S10 Fig and S11 Fig). These results suggest that when the true demographic model of a target dataset is not known using a model of population size change may result in higher specificity than using a very simple model of equilibrium demography. We also note that the latter strategy did result in a high false positive rate (17.4%) when using the default class probability threshold, but imposing more stringent thresholds rectifies this problem while still allowing for recovery of a large fraction of sweeps (S12 Fig).

326

327 **Detecting Sweeps in an experimentally evolved *Drosophila simulans* population**

328 In order to assess `Timesweeper`'s utility on experimental evolution data we applied it to data
329 from a publicly available *Drosophila simulans* study published by Barghi et al. [51]. This study
330 contains 10 experimental replicates, each exposed to a warm environment for 60 generations,
331 giving us the ability to assess our method's performance on real data by examining the extent to
332 which `Timesweeper`'s sweep candidates were replicated across these 10 datasets. We compare
333 our results to those obtained by Barghi et al. when using a Fisher's Exact Test (FET) for significant
334 difference between the starting versus final allele frequency for a target polymorphism within a
335 given experimental replicate.

336 In brief, we simulated a training set to match the experimental design described in Barghi
337 et al. and used these simulations to train a `Timesweeper` AFT classifier as described in Methods.
338 We note that this experiment lasted for a relatively short period of time (60 generations), such that
339 positively selected *de novo* mutations would rarely be expected to arise and reach high frequency
340 by the end of the experiment; we therefore trained `Timesweeper` to distinguish between
341 neutrality and the SSV model (example and average inputs in S13 Fig), and refer to the latter as
342 the "sweep" model for the remainder of this section. We then used the resulting classifier, which
343 we found to be highly accurate (AUC=0.96; S14 Fig) to scan allele frequency estimates from the
344 pooled-sequencing data from Barghi et al for signatures of sweeps on each of the five major
345 chromosome arms (Methods). This scan was performed independently in each of the 10
346 experimental replicates. The resulting predictions were filtered to retain the top 1% of the class
347 score distribution, these prediction scores were then compared to the top 1% of the reported -
348 log10(*p*-value) FET scores. We find that `Timesweeper` predictions have a Spearman correlation

of 0.20 with FET scores, and that generally SNPs receiving the largest sweep probabilities from `Timesweeper` have high FET scores and vice-versa (S1 Table and S2 Table).

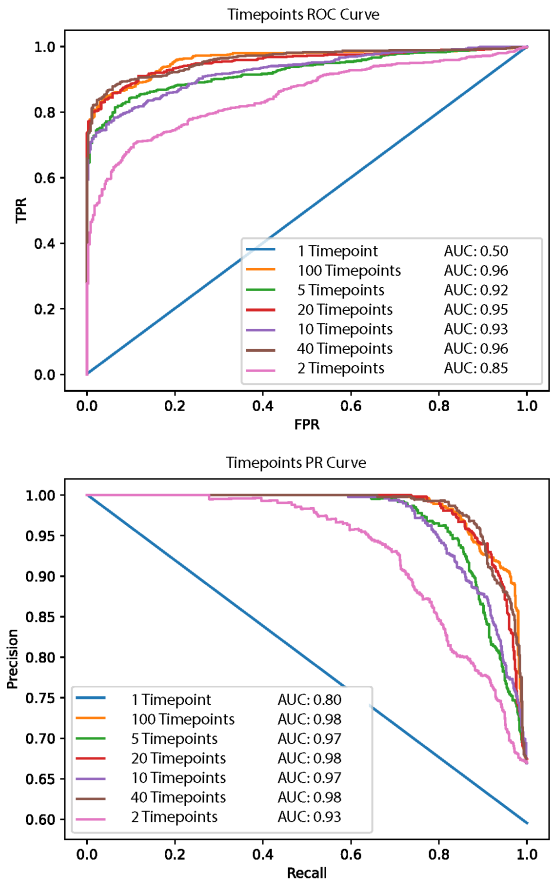

Figure 6. `Timesweeper`'s performance on data with different numbers timepoints. The top and bottom panels show ROC and PR Curves for AFT on data with varying numbers of timepoints sampled. A total of 200 individuals were sampled for each sampling scheme, and the number at each timepoint is equal to 200 divided by the number of timepoints (e.g. "40 Timepoints" corresponds to 40 timepoints, with 5 individuals sampled per-timepoint). The sampled timepoints are equally spaced across a span of 200 generations, with the first sample within this span collected 50 generations after the beginning of the sweep.

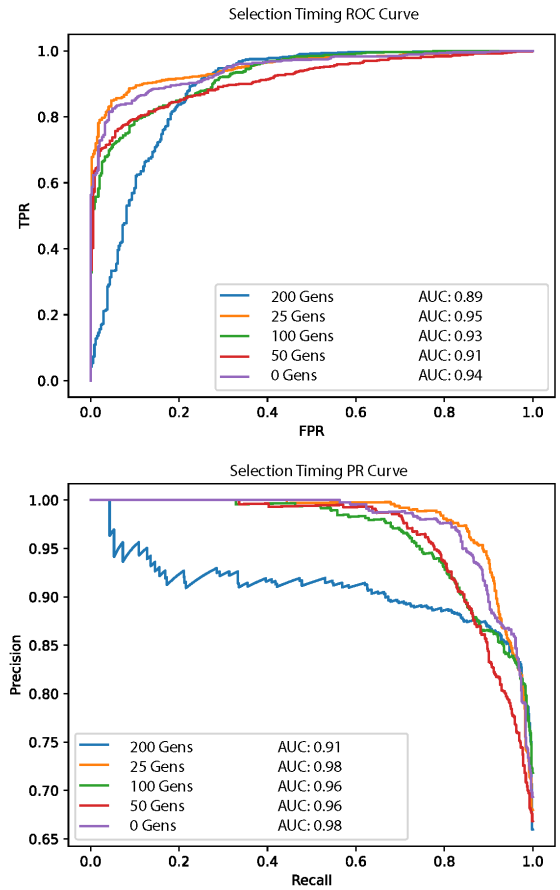

Figure 7. `Timesweeper`'s Performance on data sampled at varying times following the beginning of the sweep. ROC Curves and PR Curves for AFT performance on datasets with different post-selection sampling times. In all schemas 10 individuals were sampled at 20 timepoints sampled evenly across a 200-generation window. Values tested represent the number of generations between the introduction of selection and the first sampling point. For example, "200 Gens" refers to a dataset where the first of 20 samples was taken 200 generations after the onset of selection.

353 We next investigated the degree to which the top sweep candidate SNPs (the 100 highest-ranking
354 SNPs) from a given replicate were also classified as sweeps in other replicates (according to a less
355 stringent cutoff of whether the SNP was in
356 the top 1% of sweep scores in the replicate
357 population). This was done for both
358 `Timesweeper` and FET's top positive
359 selection candidates, using the sweep
360 class-membership probability as the score
361 for `Timesweeper`, and the -log10(*p*-
362 value) as the sweep score for FET. We
363 found that `Timesweeper` replicated a
364 larger number of hits on average than FET
365 with the exception of replicate 3 (Table 1).
366 Upon closer examination, we found that
367 this was due to the presence of a large
368 sweep region on chromosome 3 where FET predicted high scores for a large number of
369 polymorphisms in the surrounding region in multiple replicates. Importantly, `Timesweeper` was
370 able to detect and replicate the sweep signature in this region, it just did not classify as many SNPs
371 in this region as sweeps as FET did. This illustrates that these numbers of replicated classifications
372 can be somewhat inflated by linked SNPs receiving the same classification, but this is probably a
373 greater issue for the FET (which evaluates each SNP independently) than `Timesweeper`, which
374 evaluates a focal SNP while including information about flanking SNPs as well. Thus, the

Replicate	`Timesweeper`'s Replicated Hits	FET's Replicated Hits
1	44	11
2	97	86
3	44	144
4	43	43
5	37	17
6	43	18
7	30	46
8	20	8
9	52	18
10	47	21

Table 1. The numbers of top sweep candidates replicated by Timesweeper and Fisher's Exact Test (FET) on data from experimental *D. simulans* populations. SNPs in the top 100 scoring sweep candidates for a given replicate were considered to be replicated if they were also found in the top-scoring 1% predictions by that same method in another replicate. Note that some candidate SNPs were replicated in more than one of the 9 additional experimental replicates, and thus a total number of replicated hits >100 is possible.

consistent ability of `Timesweeper` to identify regions that show signals of selective sweeps across replicates underscores its ability to detect positive selection in real data sets.

DISCUSSION

Detecting recent positive selection is an important problem in population genetics. Such efforts can help reveal the extend and mode of recent adaptation, as well as clues about the genetic and phenotypic targets of selection [9,20,24]. Moreover, signatures of selective sweeps have been shown to co-localize with disease-associated mutations [11,56], but see [57]. One possible explanation is that deleterious alleles on the positively selected haplotype may hitchhike to higher frequencies than they might otherwise reach [58], although other potential explanations exist [59,60].

For these reasons, there has been a great deal of effort to develop methods to detect the signatures of positive selection from a single sample of recently collected genomes (e.g. [13,14,17–22,22,61]). Recent approaches have incorporated combinations of these tests through the use of machine learning (e.g. [27–29,32,62,63]), and it may even be feasible to completely bypass the step of computing summary statistics by training deep neural networks to operate directly on population genomic sequence alignments as input [64], as has been done effectively for other problems in population genetics [65–67].

While the use of modern data to make inferences about the past is the cornerstone of evolutionary genomics, one could potentially achieve greater statistical power by including genomic samples across a variety of timepoints, allowing for a more direct interrogation of evolutionary history. Genomic time-series data has been used to great effect to find positively

397 selected loci in experimentally controlled studies (reviewed in [45]). There is growing interest in
398 performing similar analyses in natural populations, as the acquisition of time-series genomic data
399 has become feasible both in species with short generation times such as *Drosophila* [35,36,68,69],
400 and in those for which ancient DNA is available, such as humans [70].

401 With the potential benefits of time-series data and machine learning methods for population
402 genetics in mind, we developed `Timesweeper`, a deep learning method for detecting selective
403 sweeps from population genomic time series. We experimented with two approaches: one using a
404 matrix representation of haplotype frequencies estimated in each sampled timepoint, and one
405 tracking allele frequencies through time in a window of polymorphisms centered around a focal
406 polymorphism. We found that while the haplotype frequency tracker (HFT) had decent power to
407 detect selective sweeps, the allele frequency tracker (AFT) had superior accuracy, outperformed
408 the frequency increment test [48], and was able to localize selected variants with much higher
409 resolution. We also experimented with training the AFT method on simulated data with a constant
410 population size history and then testing the classifier on data with a more complicated demographic
411 history modeled after the out-of-Africa migration in humans [55], and vice-versa. The AFT method
412 performed poorly in the extreme scenario of unmodeled population size change, but performed
413 well when trained on the out-of-Africa model and applied to a constant-population size model.
414 More work testing on additional demographic histories would be required to fully understand how
415 our method and other time-series methods would perform in various scenarios of demographic
416 model misspecification. Another important, albeit intuitive result of our analyses is that AFT
417 generally performs better on smaller samples obtained across a larger number of timepoints than
418 on datasets consisting of the same total number of individuals sampled from a smaller number of
419 timepoints. This implies that researchers interested in detecting adaptation from longitudinal

420 collections may wish to prioritize sampling individuals across a wider range of times, even if the
421 number of individuals collected at each timepoint is fairly small—five individuals per timepoint
422 may be sufficient if many timepoints are sampled.

423 The relative failure of our haplotype-based method may contain lessons for future
424 improvements. The goal of this approach was to contain all of the information present in a Muller
425 plot, which shows the frequency trajectories of each haplotype present in a (typically clonal)
426 population sampled at regular intervals. Although this information is very useful for revealing
427 whether a haplotype may have been favored by selection, it does not contain any information about
428 the degree of dissimilarity between distinct haplotypes, nor does it contain and information about
429 the locations of any variants that may be present on multiple haplotypes. The latter shortcoming
430 implies that this approach should have little power to distinguish between the targets of selection
431 and closely linked loci, and this is supported by our findings shown in Figure 4. Future work to
432 develop haplotype-frequency tracking representations that contain information both about the
433 location of polymorphisms differing among haplotypes, and the degree of dissimilarity between
434 haplotypes sharing a given allele, could yield improvements in detecting and localizing selected
435 polymorphisms. Such advancements could also aid in distinguishing between SDN and SSV
436 models of sweeps—a problem that both our HFT and AFT methods struggled with, and for which
437 information about the degree of dissimilarity among haplotypes bearing the adaptive allele is
438 paramount [9].

439 Our AFT method, on the other hand, was highly successful at localizing sweeps. This is
440 perhaps because this approach makes an inference for a single focal polymorphism while taking
441 into account spatial information about the presence or absence of nearby hitchhiking variants,
442 before sliding on to the next focal polymorphism. Although this method does not explicitly use

443 haplotype information, it may implicitly be able to track LD among polymorphisms by seeing
444 correlated shifts in allele frequencies (see Fig. 1). Thus, it is possible that not much information is
445 lost by relying on allele frequency trajectories rather than haplotype frequencies.

446 An added benefit of the AFT approach is that it can be used with unphased data, or even
447 pooled sequencing data. This allowed us to test the method on data from an evolve-and-resequence
448 study that repeatedly sampled experimental populations of *D. simulans* over a period of 60
449 generations of selection to thermal tolerance [51]. We showed that `Timesweeper` was able to
450 detect regions with alleles that experienced large shifts in frequency over the course of this
451 selection experiment. Moreover, `Timesweeper` often detected the same putatively sweeping
452 alleles in multiple experimental replicates, suggesting that its results are accurate. We stress that
453 these encouraging results were obtained in spite of relatively minimal adjustment of our method
454 to these data: the founder lines of the *D. simulans* experiment exhibit a very high density of
455 polymorphism, and thus our approach to examine 51-SNP windows caused us to examine only a
456 relatively small genetic map distance when making each classification—incorporating a larger
457 flanking window in this setting would likely yield even better performance. We believe that the
458 successful application of `Timesweeper` to these data in spite of this limitation bodes well for the
459 broad applicability and robustness of our approach.

460 In light of the method's strong performance on simulated and real genomic data, we have
461 released the `Timesweeper` package to the public. We strived to make this package easy to use
462 and computationally efficient, as we believe that fast, user-friendly, and powerful computational
463 tools are essential if we wish to realize the potential of population genomic time-series data. We
464 hope that further research in this area of population genetic methods development, combined with
465 the application of these methods to time-series data from natural populations, will help us better

466 characterize the genomic targets and evolutionary importance of positive selection. For example,
467 the continued development and analysis of these data sets can help us resolve the controversy over
468 the impact of positive selection in nature [1–3]. Such work may also play an important role in the
469 genomic surveillance of both organisms relevant to human health (vectors and pathogens) as well
470 as populations threatened by climate change or other anthropogenic disruptions.

471

472 **METHODS**

473 **Overview of the `Timesweeper` workflow**

474 `Timesweeper` is a Python package consisting of multiple interconnected modules that can be
475 used in sequence to simulate training and test data consisting of genomic regions with and without
476 selective sweeps given a user-specified demographic model and sampling scheme with multiple
477 sampling times, process the resulting simulation output and convert it into useable data formats,
478 use these data to train neural networks to detect selective sweeps, and finally to perform inference
479 on real data using the trained classifier. Each of these steps is described in detail below.
480 `Timesweeper` is developed as separate modules as opposed to a single workflow for modularity
481 and customizability in mind—the code is intended to be used out of the box but can also adapted
482 to a user's needs by modifying or replacing any of the constituent modules if desired.
483 `Timesweeper` can detect sweeps using one of two pieces of information: allele frequency
484 trajectories surrounding a focal polymorphism, and haplotype frequency trajectories within a focal
485 window. All modules in `Timesweeper` can be run using a single YAML configuration file across
486 the workflow or by supplying command line arguments for each step. Most modules are optimized
487 for multiprocessing and allow for specification of the number of threads using the `--threads`

flag at runtime. `Timesweeper`'s simulation modules allow for the partitioning of replicate ranges, allowing for straightforward parallelization across a high-performance computing system during what is the most computational-heavy stage of the workflow.

Simulating using custom SLiM scripts

`Timesweeper` has two modules for simulating training data using SLiM [71]: a module for simulating from custom-made SLiM scripts and another for using pre-made `stdpopsim` [72] demographic models. The former is relatively straightforward: a user supplies a working directory, the path to their SLiM script, the path to the SLiM executable, and the number of replicate simulations they'd like to produce. Other optional parameters supplied at runtime are passed to SLiM through the `simulate_custom` module of `Timesweeper`. This module is intended to be modified to suit the needs of the experiment, and is written such that it is easy to incorporate utility functions, stochastic variable draws, and more into each replicate.

Users must write their SLiM script such that it accepts two constant parameters at runtime: the sweep type, and the output file name. The sweep constant must accept the arguments "neut", "hard", or "soft", though a user may write their SLiM script to use these arguments however they wish. In our workflow, they specify the class of training/test data to be generated, with "neut" denoting simulations lacking a selective sweep, "hard" denoting simulations with a selective sweep on a single-origin *de novo* mutation (hereafter referred to as the SDN model), and "soft" denoting simulations with a selective sweep from standing variation (the SSV model). An example SLiM script is included in the software package.

510 **Simulating using `stdpopsim`**

511 The second option for simulating training data is by injecting time-series sampling code into a
512 SLiM script generated by the `stdpopsim` package [72]. By using `stdpopsim` with the `slim`
513 and `--slim-script` flags, that software will write out a SLiM script for running whichever
514 demographic model the user selects from the `stdpopsim` catalog. This script can then be used
515 as input to `Timesweeper`'s `simulate_stdpopsim` module, along with information about
516 sampling times (specified in years as per `stdpopsim`'s convention), sampling sizes, the identity
517 of the population that is sampled and that experiences any positive selection (hereafter referred to
518 as the "target population"), and the range of selection coefficients to use in a log-uniform
519 distribution. This module then adds the desired events to the `stdpopsim` SLiM script before
520 running the simulation. The output of this module is identical in format to that generated when
521 using the `simulate_custom` module described above, and is processed identically
522 downstream. We note that although `stdpopsim` supports selective sweeps with some degree of
523 customizable parameters, this feature does not currently support all of the scenarios we desired for
524 our workflow (e.g. selection on a randomly chosen polymorphism that was previously fitness-
525 neutral, while specifying only the time at which the derived allele later became beneficial). As
526 more features are added to the `stdpopsim` software, we may update our workflow to better take
527 advantage of this resource.

528

529 **VCF file processing post-simulation**

530 For both simulation modules described above, at each sampling timepoint, SLiM draws a random
531 sample of the specified number of individuals from the target population without replacement, and

532 concatenates a VCF entry to an output file, hereafter referred to as a "multiVCF", for that replicate.
533 After a simulation replicate is complete, the multiVCF is split into separate files consisting of one
534 timepoint each, labeled in numerical order according to when they were sampled in the series.
535 Note that because our workflow uses forward simulations, the sweeping mutation may often be
536 lost, requiring the simulation to jump back to an earlier point. This may result in repeated
537 samplings drawn for one or more sampled timepoints, but our workflow ensures that only the last
538 sample for each timepoint (corresponding to the simulation run where the sweeping mutation was
539 not lost) is retained. VCF files are then sorted and indexed using BCFtools [61,73] sort and index.
540 Individual VCFs are then merged using the command `bcftools merge --force-samples`
541 to create a VCF containing all polymorphisms among the set of all genomes sampled across all
542 timepoints. Information from these VCFs can then be loaded into a format suitable for input to our
543 neural network as described in the next two sections.

544

545 **Time-series allele frequency matrices**

546 `Timesweeper`'s allele frequency-tracking (AFT) method seeks to classify each individual
547 polymorphism on the basis of allele frequencies at each of l biallelic polymorphisms in a
548 surrounding window that consists of the central polymorphism and $(l - 1)/2$ polymorphisms on
549 either side, measured at m distinct timepoints. We therefore chose to represent allele frequencies
550 around a focal polymorphism by an $l \times m$ matrix, which serves as our input for both training and
551 prediction. This was constructed by first obtaining sample allele frequency trajectories from
552 merged VCF files. For a single merged VCF, whether obtained from either real or simulated data,
553 genotypes are loaded using scikit-allel [74], and the two alleles at each polymorphism are labeled
554 as "net-increasing" or "net-decreasing" according to their change in frequency between the first

555 and final timepoint. For a single merged VCF, whether obtained from either real or simulated data,
556 genotypes are loaded using scikit-allel [74], and the two alleles at each polymorphism are labeled
557 as "net-increasing" or "net-decreasing" according to their change in frequency between the first
558 and final timepoint. The desired $m \times l$ matrix showing the frequency of the net-increasing allele
559 for each polymorphism at each timepoint is then created. For this paper, we set l to 51 (i.e. 25
560 flanking polymorphisms are examined around each focal polymorphism to be classified). When
561 dealing with simulated data, the number of polymorphisms obtained was always greater than 51,
562 and thus the matrix was cropped to obtain 51 polymorphisms as described below in "Classifier
563 Training and Validation." For real data, a sliding window of 51 polymorphisms can be used to
564 obtain a classification for each individual polymorphism (see "Detecting Sweeps in Genomics
565 Windows" below). Given that some users may wish to examine a larger window, we give users
566 the option of changing the value of l when running `Timesweeper`.

567

568 **Time-series haplotype frequency matrices**

569 Timesweeper's haplotype frequency-tracking method constructs matrices containing information
570 about haplotype frequency trajectories. For a given window of l polymorphisms, the frequencies
571 of all haplotypes are recorded at each timepoint using information from the merged VCF (again
572 processed using scikit-allel), and recorded in a $k \times m$ matrix, where m is again the number of
573 timepoints and k is the number of distinct haplotypes observed across all samples. This matrix is
574 then sorted such that the haplotype with the highest net increase in frequency between the first and
575 last timepoints is located at the "bottom" of the frequency matrix, i.e. the 0th index. The remaining
576 haplotypes are then sorted by their similarity to the haplotype at the 0th index (i.e. the haplotype at

577 index 1 is the most similar to the haplotype with the highest net increase in frequency, the
578 haplotype at index 2 is the second-most similar, and so on).

579

580 **Binary classification using the Frequency Increment Test**

581 The Frequency Increment Test (FIT) was implemented in Python following Feder et. al. [48]. In
582 brief: SLiM output was parsed identically to the method described above for the time-series allele
583 frequency spectrum, and the rescaled allele frequency increments for each polymorphism in the
584 simulated region are calculated as $t_{\mathrm{FI}}(\mathrm{Data}) = \frac{\bar{Y}}{\sqrt{S^2/L}}$, where

585 $$\bar{Y} = \frac{1}{L}\sum_{i=1}^{L} Y_i \quad \text{and} \quad S^2 = \frac{1}{L-1}\sum_{i=1}^{L}(Y_i - \bar{Y})^2 \text{ and } Y_i = \frac{\nu_i - \nu_{i-1}}{\sqrt{2\nu_{i-1}(1-\nu_{i-1})(t_i - t_{i-1})}}, \quad i = 1,2,\ldots,L.$$

586 and ν_i and t_i refer to the allele frequency and time at timepoint i of L, respectively. The frequency
587 increments Y_i are used in a one-sample student's *t*-test for each polymorphism. To classify a region
588 as either experiencing a sweep or evolving neutrally (no effort is made to distinguish between
589 sweep types using this test), we simply examined the minimum *p*-value among all tested
590 polymorphisms in the region: if this *p*-value was less than 0.05, the region was classified as a
591 sweep, and otherwise the region was classified as neutrally evolving.

592

593 **Neural network architectures**

594 We implemented two neural network models in Keras [75] with Tensorflow backends: a shallow
595 1D convolutional neural network (1DCNN) for time-series data and a shallow fully-connected
596 network (FCN) for single-timepoint data. We implemented two neural network models in Keras

597 [75] with Tensorflow backends: a shallow 1D convolutional neural network (1DCNN) for time-
598 series data and a shallow fully-connected network (FCN) for single-timepoint data. Architectures
599 for both data types (allele frequencies and haplotype frequencies) are identical aside from the
600 adjustment in input layer dimensions.

601 The time-series network begins with two 1D convolutional layers with 64 filters and a kernel size
602 of 3 each (i.e. the first convolutional layer runs rectangular filters across an area of all
603 polymorphisms/haplotypes by 3 timepoints at a time, sliding across timepoints with a stride length
604 of 1. These are followed by a 1D maximum pooling layer with a pool size of 3, followed by a
605 dropout layer with a dropout rate of 0.15. A flattening layer then proceeds two blocks of 264-unit
606 dense layers followed by dropout layers with a rate of 0.2. This is followed by a 128-unit dense
607 layer, a dropout layer with dropout rate of 0.3, and a final dense layer with 3 output units. All
608 layers described use ReLu activation function other than the final output layer, which uses a
609 softmax activation function.

610 The one-sample network is a fully connected network consisting of two blocks of 512-unit
611 dense layers followed by dropout layers with rates of 0.2, followed by a 128-unit dense layer with
612 dropout of 0.1, and a final dense layer with 3 output units. This network also uses ReLu activations
613 for all layers aside from the final output layer, which uses a softmax activation function.

614

615 **Classifier training and validation**

616 To create training labeled data for models we first process simulated VCFs as discussed above.
617 During this process the mutation type specified in the SLiM simulation is obtained from each VCF
618 entry, allowing us to retrieve the selected mutation (the only non-neutral mutation present in the

simulation). For the AFT method, to construct training sets for the SDN and SSV scenarios, the selected mutation is retrieved and set to be the center of the 51-polymorphism window (i.e. only the 25 closest polymorphisms on either side of the selected mutation are retained). For the neutral scenario the central-most polymorphism is set to be the center of the 51-polymorphism window. For the HFT method, the data are formatted as described above without regard to the selected mutation (i.e. the haplotype found at the highest frequency is placed at the bottom of the matrix, regardless of whether or not it contains a selected mutation). Windows of the designated size are saved in a compressed pickle file and loaded into memory at training time.

Data is split into training/validation/testing partitions consisting of 70%/15%/15% of the entire dataset, and each partition is stratified such that classes are evenly split among them. Models are trained for a maximum of 40 epochs with the potential for early stopping based on validation accuracy. If early stopping occurs model weights are restored to the epoch with the highest validation accuracy.

Detecting sweeps in genomic windows

`Timesweeper`'s final module, cleverly named `find_sweeps`, reports AFT classifier scores and optionally HFT classifier scores and FIT *p*-values for each sliding window of polymorphisms across all polymorphisms in a given VCF, whether obtained from simulated data (for testing) or real data (for inference). Trained neural networks are first loaded into memory and the input VCF is loaded in chunks. For each focal polymorphism in a chunk, data is processed into the required formats described above for each method prior to classification/inference. The `find_sweeps` module also allows for testing on simulated data using the `--benchmark` flag, which causes the

641 module to check each prediction against the ground truth from the simulation, allowing for easy
642 validation of all methods on labeled test data.

643

644 **Simulations for testing `Timesweeper` on time-series data from a constant-sized population**

645 As in the standard `Timesweeper` workflows described above, all simulations used to evaluate
646 `Timesweeper`'s accuracy were performed via SLiM [71]. For the constant-sized population
647 scenarios a population of 500 individuals was created with a mutation rate of 1×10^{-7} and
648 recombination rate of 1×10^{-7}. Simulations underwent burn-in for $20N$ generations. In the SDN
649 scenario a *de novo* mutation is introduced at the physical midpoint of the chromosome with a given
650 selection coefficient. In the SSV scenario the closest standing neutral mutation to the center of the
651 chromosome is selected and the selection coefficient of that mutation is set to the desired value. A
652 number of sampling schemes and sweep parameterizations were tested under this constant-size
653 scenario, as described in the Results. For each of these experiments 7,500 replicates were
654 simulated, 2,500 of each scenario: sweeps under the SDN model, SSV sweeps, and neutrality. Data
655 were then partitioned into training, testing, and validation sets as described above.

656

657 **Testing the effects of demographic model misspecification on `Timesweeper`'s accuracy**

658 We simulated two demographic models: a constant-sized population (described above), herein
659 referred to as the "simple model", as well as the Out of Africa model described in [55] as
660 implemented in stdpopsim, herein "OoA model". For the simple model we sampled 20 evenly-
661 spaced timepoints with 10 diploid individuals at each. Sampling started 50 generations post-
662 selection and proceeded for a total of 200 generations. We similarly sampled 20 timepoints for the

663 OoA model using an identical sampling schema. We simulated 2,500 replicates of neutrality, SDN,
664 and SSV scenarios under both models and trained `Timesweeper` as described above. We then
665 used each trained CNN to detect sweeps on 100 independent replicates from the demographic
666 model not trained on. Performance metrics were then calculated on the summed predictions across
667 all replicates.

668

669 **Applying `Timesweeper` to a *Drosophila simulans* evolve-and-resequence dataset**

670 To demonstrate `Timesweeper`'s flexibility and applicability to real data, we applied the AFT
671 method to a dataset published by Barghi et al. [51] comprised of time-series *Drosophila simulans*
672 pool-sequencing data from an evolve-and-resequence study. In brief, Barghi et al. exposed 10
673 replicates of an ancestral population to extreme temperatures in 12-hour cycles for 60 generations.
674 Each experimental replicate population was sampled every 10 generations, yielding a total of 7
675 sampled timepoints including the initial ancestral population. Barghi et al. [51] then performed
676 pooled sequencing, read mapping, and genotyping on each replicate population for each of these
677 timepoints. We downloaded the read counts for each allele at each SNP at each timepoint from
678 https://doi.org/10.5061/dryad.rr137kn. The data for each timepoint were converted into allele
679 frequency estimates by simply taking, for each allele, the number of reads supporting that allele
680 divided by the total number of reads mapped to that site. Net alleles frequency changes were then
681 calculated by taking the allele with the largest net frequency increase over the course of the
682 experiment, and the allele with the largest frequency of all the remaining alleles (i.e. only two
683 alleles were considered at each site), and renormalizing so that these two allele frequency estimates
684 summed to one for each timepoint. This process was repeated for each experimental replicate.

685 The training data for this analysis were simulated using a modified `simulate_custom`
686 module of Timesweeper to allow for population-size rescaling by a factor of 100 during the burn-
687 in, with recombination and mutation rates multiplied by this factor to compensate, followed by the
688 remainder of the simulation (i.e. the 60 generations of selection) being carried out without
689 rescaling. The purpose of rescaling during the burn-in only was that the population size was small
690 during the experiment (N=1000), but most likely dramatically larger in the ancestral population.
691 The unscaled recombination rate was pulled from a uniform distribution between 0 and 2e-8 and
692 the unscaled mutation rate was set to 5×10^{-9}, with only neutral mutations occurring. The simulation
693 began with a burn-in period of 20 times the scaled population size (N=2,500, corresponding to an
694 unscaled size of 250,000), or 50,000 generations. We note that this ancestral population size is a
695 rough estimate that we obtained by calculating nucleotide diversity within each replicate's founder
696 sample (≈0.005), and is likely smaller than the *D. simulans* population used to found the
697 experimental lines as it does not account for the impact of repetitive/low quality regions or natural
698 selection on diversity, probably resulting in simulations with a somewhat lower density of
699 polymorphism than found in the real data. Following the burn-in, the population then contracted
700 to 1,000 individuals, where it remained for the rest of the simulation, and a sample was taken to
701 represent the starting "ancestral" state of the replicate, following the design used by Barghi et al.
702 [51]. In simulations with selection, the mutation closest to the physical center of the chromosome
703 was then changed from neutral to beneficial with a selection coefficient drawn from a log-uniform
704 distribution with a lower bound of s=0.02 and an upper bound of s=0.2—note that only SSV
705 sweeps were generated under this model, as the SDN model is unlikely to be relevant in such a
706 small population under selection for such a short period of time. The simulations then proceeded
707 for another 60 generations post-selection with sampling occurring every 10 generations, resulting

in a total of 7 sampled timepoints as in in Barghi et al. [51]. For each of these 7 timepoints in each replicate population, we sampled 100 diploid individuals. This was performed for a total of 2500 replicate simulations for both the neutral and sweep SSV conditions that were then processed and used to train a `Timesweeper` network using the AFT method as described above, which was then used to detect sweeps in the *D. simulans* data using the `find_sweeps` module.

Data and code availability

`Timesweeper`'s source code and documentation can be found at https://github.com/SchriderLab/timeSeriesSweeps, it is available to be built from source as well as installed as conda package. All scripts and workflows can be found at https://github.com/SchriderLab/timesweeper-experiments to allow for a complete reconstruction of all data/figures used in this manuscript.

723 **REFERENCES**

724 1. Hahn MW. Toward a Selection Theory of Molecular Evolution. Evolution. 2008;62: 255–
725 265. doi:10.1111/j.1558-5646.2007.00308.x

726 2. Kern AD, Hahn MW. The Neutral Theory in Light of Natural Selection. Mol Biol Evol.
727 2018;35: 1366–1371. doi:10.1093/molbev/msy092

728 3. Jensen JD, Payseur BA, Stephan W, Aquadro CF, Lynch M, Charlesworth D, et al. The
729 importance of the Neutral Theory in 1968 and 50 years on: A response to Kern and Hahn
730 2018. Evolution. 2019;73: 111–114. doi:10.1111/evo.13650

731 4. Stephan W. Genetic hitchhiking versus background selection: the controversy and its
732 implications. Philos Trans R Soc B Biol Sci. 2010;365: 1245–1253.
733 doi:10.1098/rstb.2009.0278

734 5. Kaplan NL, Hudson RR, Langley CH. The "hitchhiking effect" revisited. Genetics.
735 1989;123: 887–899. doi:10.1093/genetics/123.4.887

736 6. Smith JM, Haigh J. The hitch-hiking effect of a favourable gene. Genet Res. 2009/04/14 ed.
737 1974;23: 23–35. doi:10.1017/S0016672300014634

738 7. Booker TR, Jackson BC, Craig RJ, Charlesworth B, Keightley PD. Selective sweeps
739 influence diversity over large regions of the mouse genome. bioRxiv; 2021. p.
740 2021.06.10.447924. doi:10.1101/2021.06.10.447924

741 8. Enard D, Messer PW, Petrov DA. Genome-wide signals of positive selection in human
742 evolution. Genome Res. 2014;24: 885–895. doi:10.1101/gr.164822.113

743 9. Garud NR, Messer PW, Buzbas EO, Petrov DA. Recent Selective Sweeps in North
744 American Drosophila melanogaster Show Signatures of Soft Sweeps. PLOS Genet.
745 2015;11: e1005004. doi:10.1371/journal.pgen.1005004

746 10. Garud NR, Petrov DA. Elevated Linkage Disequilibrium and Signatures of Soft Sweeps
747 Are Common in Drosophila melanogaster. Genetics. 2016;203: 863–880.
748 doi:10.1534/genetics.115.184002

749 11. Schrider DR, Kern AD. Soft Sweeps Are the Dominant Mode of Adaptation in the Human
750 Genome. Mol Biol Evol. 2017;34: 1863–1877. doi:10.1093/molbev/msx154

751 12. Harris RB, Sackman A, Jensen JD. On the unfounded enthusiasm for soft selective sweeps
752 II: Examining recent evidence from humans, flies, and viruses. PLOS Genet. 2018;14:
753 e1007859. doi:10.1371/journal.pgen.1007859

754 13. Fay JC, Wu CI. Hitchhiking under positive Darwinian selection. Genetics. 2000;155: 1405–
755 1413.

14. Kim Y, Stephan W. Detecting a local signature of genetic hitchhiking along a recombining chromosome. Genetics. 2002;160: 765–777. doi:10.1093/genetics/160.2.765

15. Li H. A new test for detecting recent positive selection that is free from the confounding impacts of demography. Mol Biol Evol. 2011;28: 365–375. doi:10.1093/molbev/msq211

16. Hudson RR, Bailey K, Skarecky D, Kwiatowski J, Ayala FJ. Evidence for positive selection in the superoxide dismutase (Sod) region of Drosophila melanogaster. Genetics. 1994;136: 1329–1340. doi:10.1093/genetics/136.4.1329

17. Harris AM, DeGiorgio M. A Likelihood Approach for Uncovering Selective Sweep Signatures from Haplotype Data. Mol Biol Evol. 2020;37: 3023–3046. doi:10.1093/molbev/msaa115

18. Sabeti PC, Reich DE, Higgins JM, Levine HZP, Richter DJ, Schaffner SF, et al. Detecting recent positive selection in the human genome from haplotype structure. Nature. 2002;419: 832–837. doi:10.1038/nature01140

19. Ferrer-Admetlla A, Liang M, Korneliussen T, Nielsen R. On detecting incomplete soft or hard selective sweeps using haplotype structure. Mol Biol Evol. 2014;31: 1275–1291. doi:10.1093/molbev/msu077

20. Voight BF, Kudaravalli S, Wen X, Pritchard JK. A Map of Recent Positive Selection in the Human Genome. PLoS Biol. 2006;4: e72. doi:10.1371/journal.pbio.0040072

21. Kelly JK. A test of neutrality based on interlocus associations. Genetics. 1997;146: 1197–1206. doi:10.1093/genetics/146.3.1197

22. Kim Y, Nielsen R. Linkage disequilibrium as a signature of selective sweeps. Genetics. 2004;167: 1513–1524. doi:10.1534/genetics.103.025387

23. Chen H, Patterson N, Reich D. Population differentiation as a test for selective sweeps. Genome Res. 2010;20: 393–402. doi:10.1101/gr.100545.109

24. Nielsen R, Williamson S, Kim Y, Hubisz MJ, Clark AG, Bustamante C. Genomic scans for selective sweeps using SNP data. Genome Res. 2005;15: 1566–1575. doi:10.1101/gr.4252305

25. Jensen JD, Kim Y, DuMont VB, Aquadro CF, Bustamante CD. Distinguishing between selective sweeps and demography using DNA polymorphism data. Genetics. 2005;170: 1401–1410. doi:10.1534/genetics.104.038224

26. Mughal MR, DeGiorgio M. Localizing and Classifying Adaptive Targets with Trend Filtered Regression. Mol Biol Evol. 2019;36: 252–270. doi:10.1093/molbev/msy205

27. Schrider DR, Kern AD. S/HIC: Robust Identification of Soft and Hard Sweeps Using Machine Learning. PLOS Genet. 2016;12: e1005928. doi:10.1371/journal.pgen.1005928

790 28. Sugden LA, Atkinson EG, Fischer AP, Rong S, Henn BM, Ramachandran S. Localization
791 of adaptive variants in human genomes using averaged one-dependence estimation. Nat
792 Commun. 2018;9: 703. doi:10.1038/s41467-018-03100-7

793 29. Pybus M, Luisi P, Dall'Olio GM, Uzkudun M, Laayouni H, Bertranpetit J, et al.
794 Hierarchical boosting: a machine-learning framework to detect and classify hard selective
795 sweeps in human populations. Bioinformatics. 2015;31: 3946–3952.
796 doi:10.1093/bioinformatics/btv493

797 30. Kern AD, Schrider DR. diploS/HIC: An Updated Approach to Classifying Selective
798 Sweeps. G3 GenesGenomesGenetics. 2018;8: 1959–1970. doi:10.1534/g3.118.200262

799 31. Xue AT, Schrider DR, Kern AD, Ag1000g Consortium. Discovery of Ongoing Selective
800 Sweeps within Anopheles Mosquito Populations Using Deep Learning. Mol Biol Evol.
801 2021;38: 1168–1183. doi:10.1093/molbev/msaa259

802 32. Lin K, Li H, Schlötterer C, Futschik A. Distinguishing Positive Selection From Neutral
803 Evolution: Boosting the Performance of Summary Statistics. Genetics. 2011;187: 229–244.
804 doi:10.1534/genetics.110.122614

805 33. Pennings PS, Kryazhimskiy S, Wakeley J. Loss and Recovery of Genetic Diversity in
806 Adapting Populations of HIV. PLOS Genet. 2014;10: e1004000.
807 doi:10.1371/journal.pgen.1004000

808 34. Feder AF, Pennings PS, Petrov DA. The clarifying role of time series data in the population
809 genetics of HIV. PLOS Genet. 2021;17: e1009050. doi:10.1371/journal.pgen.1009050

810 35. Bergland AO, Behrman EL, O'Brien KR, Schmidt PS, Petrov DA. Genomic Evidence of
811 Rapid and Stable Adaptive Oscillations over Seasonal Time Scales in Drosophila. PLOS
812 Genet. 2014;10: e1004775. doi:10.1371/journal.pgen.1004775

813 36. Machado HE, Bergland AO, Taylor R, Tilk S, Behrman E, Dyer K, et al. Broad geographic
814 sampling reveals the shared basis and environmental correlates of seasonal adaptation in
815 Drosophila. Nordborg M, Wittkopp PJ, Nordborg M, editors. eLife. 2021;10: e67577.
816 doi:10.7554/eLife.67577

817 37. Bertram J. Allele frequency divergence reveals ubiquitous influence of positive selection in
818 Drosophila. PLOS Genet. 2021;17: e1009833. doi:10.1371/journal.pgen.1009833

819 38. Bollback JP, York TL, Nielsen R. Estimation of 2Nes From Temporal Allele Frequency
820 Data. Genetics. 2008;179: 497–502. doi:10.1534/genetics.107.085019

821 39. Hummel S, Schmidt D, Kremeyer B, Herrmann B, Oppermann M. Detection of the CCR5-
822 Δ32 HIV resistance gene in Bronze Age skeletons. Genes Immun. 2005;6: 371–374.
823 doi:10.1038/sj.gene.6364172

824 40. Malaspinas A-S. Methods to characterize selective sweeps using time serial samples: an
825 ancient DNA perspective. Mol Ecol. 2016;25: 24–41. doi:10.1111/mec.13492

41. Wilde S, Timpson A, Kirsanow K, Kaiser E, Kayser M, Unterländer M, et al. Direct evidence for positive selection of skin, hair, and eye pigmentation in Europeans during the last 5,000 y. Proc Natl Acad Sci. 2014;111: 4832–4837. doi:10.1073/pnas.1316513111

42. Sverrisdóttir OÓ, Timpson A, Toombs J, Lecoeur C, Froguel P, Carretero JM, et al. Direct Estimates of Natural Selection in Iberia Indicate Calcium Absorption Was Not the Only Driver of Lactase Persistence in Europe. Mol Biol Evol. 2014;31: 975–983. doi:10.1093/molbev/msu049

43. Olalde I, Mallick S, Patterson N, Rohland N, Villalba-Mouco V, Silva M, et al. The genomic history of the Iberian Peninsula over the past 8000 years. Science. 2019;363: 1230–1234. doi:10.1126/science.aav4040

44. Jeong Choongwon, Ozga Andrew T., Witonsky David B., Malmström Helena, Edlund Hanna, Hofman Courtney A., et al. Long-term genetic stability and a high-altitude East Asian origin for the peoples of the high valleys of the Himalayan arc. Proc Natl Acad Sci. 2016;113: 7485–7490. doi:10.1073/pnas.1520844113

45. Schlötterer C, Kofler R, Versace E, Tobler R, Franssen SU. Combining experimental evolution with next-generation sequencing: a powerful tool to study adaptation from standing genetic variation. Heredity. 2015;114: 431–440. doi:10.1038/hdy.2014.86

46. Terhorst J, Schlötterer C, Song YS. Multi-locus Analysis of Genomic Time Series Data from Experimental Evolution. PLOS Genet. 2015;11: e1005069. doi:10.1371/journal.pgen.1005069

47. Otte KA, Schlötterer C. Detecting selected haplotype blocks in evolve and resequence experiments. Mol Ecol Resour. 2021;21: 93–109. doi:10.1111/1755-0998.13244

48. Feder AF, Kryazhimskiy S, Plotkin JB. Identifying signatures of selection in genetic time series. Genetics. 2014;196: 509–522. doi:10.1534/genetics.113.158220

49. Illingworth CJR, Parts L, Schiffels S, Liti G, Mustonen V. Quantifying Selection Acting on a Complex Trait Using Allele Frequency Time Series Data. Mol Biol Evol. 2012;29: 1187–1197. doi:10.1093/molbev/msr289

50. Vlachos C, Burny C, Pelizzola M, Borges R, Futschik A, Kofler R, et al. Benchmarking software tools for detecting and quantifying selection in evolve and resequencing studies. Genome Biol. 2019;20: 169. doi:10.1186/s13059-019-1770-8

51. Barghi N, Tobler R, Nolte V, Jakšić AM, Mallard F, Otte KA, et al. Genetic redundancy fuels polygenic adaptation in Drosophila. PLoS Biol. 2019;17. doi:10.1371/journal.pbio.3000128

52. Vy HMT, Kim Y. A Composite-Likelihood Method for Detecting Incomplete Selective Sweep from Population Genomic Data. Genetics. 2015;200: 633–649. doi:10.1534/genetics.115.175380

53. Johri P, Eyre-Walker A, Gutenkunst RN, Lohmueller KE, Jensen JD. On the prospect of achieving accurate joint estimation of selection with population history. Genome Biol Evol. 2022; evac088. doi:10.1093/gbe/evac088

54. Johri P, Aquadro CF, Beaumont M, Charlesworth B, Excoffier L, Eyre-Walker A, et al. Recommendations for improving statistical inference in population genomics. PLOS Biol. 2022;20: e3001669. doi:10.1371/journal.pbio.3001669

55. Gutenkunst RN, Hernandez RD, Williamson SH, Bustamante CD. Inferring the Joint Demographic History of Multiple Populations from Multidimensional SNP Frequency Data. PLOS Genet. 2009;5: e1000695. doi:10.1371/journal.pgen.1000695

56. Blekhman R, Man O, Herrmann L, Boyko AR, Indap A, Kosiol C, et al. Natural selection on genes that underlie human disease susceptibility. Curr Biol CB. 2008;18: 883–889. doi:10.1016/j.cub.2008.04.074

57. Di C, Murga Moreno J, Salazar-Tortosa DF, Lauterbur ME, Enard D. Decreased recent adaptation at human mendelian disease genes as a possible consequence of interference between advantageous and deleterious variants. Wittkopp PJ, Barreiro L, Lohmueller KE, editors. eLife. 2021;10: e69026. doi:10.7554/eLife.69026

58. Chun S, Fay JC. Evidence for Hitchhiking of Deleterious Mutations within the Human Genome. PLOS Genet. 2011;7: e1002240. doi:10.1371/journal.pgen.1002240

59. Otto SP. Two steps forward, one step back: the pleiotropic effects of favoured alleles. Proc R Soc B Biol Sci. 2004;271: 705–714.

60. Corbett S, Courtiol A, Lummaa V, Moorad J, Stearns S. The transition to modernity and chronic disease: mismatch and natural selection. Nat Rev Genet. 2018;19: 419–430. doi:10.1038/s41576-018-0012-3

61. Li H. A statistical framework for SNP calling, mutation discovery, association mapping and population genetical parameter estimation from sequencing data. Bioinformatics. 2011;27: 2987–2993. doi:10.1093/bioinformatics/btr509

62. Ronen R, Udpa N, Halperin E, Bafna V. Learning Natural Selection from the Site Frequency Spectrum. Genetics. 2013;195: 181–193. doi:10.1534/genetics.113.152587

63. Gower G, Picazo PI, Fumagalli M, Racimo F. Detecting adaptive introgression in human evolution using convolutional neural networks. eLife. 2021;10: e64669. doi:10.7554/eLife.64669

64. Flagel L, Brandvain Y, Schrider DR. The Unreasonable Effectiveness of Convolutional Neural Networks in Population Genetic Inference. Mol Biol Evol. 2019;36: 220–238. doi:10.1093/molbev/msy224

65. Adrion JR, Galloway JG, Kern AD. Predicting the Landscape of Recombination Using Deep Learning. Mol Biol Evol. 2020;37: 1790–1808. doi:10.1093/molbev/msaa038

66. Sanchez T, Cury J, Charpiat G, Jay F. Deep learning for population size history inference: Design, comparison and combination with approximate Bayesian computation. Mol Ecol Resour. 2021;21: 2645–2660. doi:10.1111/1755-0998.13224

67. Chan J, Perrone V, Spence J, Jenkins P, Mathieson S, Song Y. A Likelihood-Free Inference Framework for Population Genetic Data using Exchangeable Neural Networks. Advances in Neural Information Processing Systems. Curran Associates, Inc.; 2018. Available: https://proceedings.neurips.cc/paper/2018/hash/2e9f978b222a956ba6bdf427efbd9ab3-Abstract.html

68. Lange JD, Bastide H, Lack JB, Pool JE. A Population Genomic Assessment of Three Decades of Evolution in a Natural Drosophila Population. Mol Biol Evol. 2022;39: msab368. doi:10.1093/molbev/msab368

69. Kapun M, Nunez JCB, Bogaerts-Márquez M, Murga-Moreno J, Paris M, Outten J, et al. Drosophila Evolution over Space and Time (DEST): A New Population Genomics Resource. Mol Biol Evol. 2021;38: 5782–5805. doi:10.1093/molbev/msab259

70. Allentoft ME, Sikora M, Refoyo-Martínez A, Irving-Pease EK, Fischer A, Barrie W, et al. Population Genomics of Stone Age Eurasia. bioRxiv; 2022. p. 2022.05.04.490594. doi:10.1101/2022.05.04.490594

71. Haller BC, Messer PW. SLiM 3: Forward Genetic Simulations Beyond the Wright–Fisher Model. Hernandez R, editor. Mol Biol Evol. 2019;36: 632–637. doi:10.1093/molbev/msy228

72. Adrion JR, Cole CB, Dukler N, Galloway JG, Gladstein AL, Gower G, et al. A community-maintained standard library of population genetic models. Coop G, Wittkopp PJ, Novembre J, Sethuraman A, Mathieson S, editors. eLife. 2020;9: e54967. doi:10.7554/eLife.54967

73. Danecek P, Bonfield JK, Liddle J, Marshall J, Ohan V, Pollard MO, et al. Twelve years of SAMtools and BCFtools. GigaScience. 2021;10: giab008. doi:10.1093/gigascience/giab008

74. Miles A, bot pyup io, R M, Ralph P, Harding N, Pisupati R, et al. cggh/scikit-allel: v1.3.3. Zenodo; 2021. doi:10.5281/zenodo.4759368

75. Chollet F, others. Keras. 2015. Available: https://keras.io

SUPPORTING INFORMATION LEGENDS

S1 Fig: Resolution of the AFT, HFT, and FIT methods assessed on simulated 500 kb chromosomes. (A) The fraction of polymorphisms classified as a sweep at varying locations along the 500 kb chromosome. When a sweep is present (SSV and SDN columns), it occurs in the center

931 of the chromosome. The top row shows the classification results of the AFT method, and is
932 identical to the three panels in Figure 4. The middle row shows the results of the HFT method. The
933 bottom row shows results for FIT, which can only reject or not reject the null hypothesis of
934 neutrality, so only two lines are shown: blue (sweep), and orange (no sweep). (B) The average
935 "sweep score" produced by each method at varying locations along the chromosome. For AFT and
936 HFT, the sweep score is the some of the posterior class membership probabilities for the SSV and
937 SDN classes, while for FIT it is 1 minus the *p*-value.

938

939 **S2 Fig: Example and average `Timesweeper` inputs for datasets of varying sample size.** (A)
940 and (B) show individual example inputs and average inputs of the Neutral, SSV, and SDN classes
941 for the AFT method where each input was constructed from a data set of 20 sampled timepoints
942 with two diploid individuals sampled per timepoint. (C) and (D) show the same for the HFT
943 method. The remaining panels show the same information for increasing sample sizes: 5
944 individuals per timepoint in panels (E) – (H), 10 individuals per timepoint in (I) – (L), and 20
945 individuals per timepoint in (M) – (P). For all simulations that included sweeps, sampling began
946 50 generations after the beginning of the sweep, and the selection coefficient was set to 0.05.

947

948 **S3 Fig: `Timesweeper`'s classification performance on datasets of varying sample size.**
949 `Timesweeper` was trained and tested on each of the scenarios whose inputs are shown in S2 Fig.
950 (A) – (D) Confusion matrix, ROC curve, training trajectory, and precision-recall (PR) curve
951 showing the performance of the AFT method on data with two individuals per timepoint. (E) – (H)
952 show the same for data with 5 individuals per timepoint, (I) – (L) show the same for data with 10

individuals per timepoint, and (M) – (O) show the same per data with 20 individuals per timepoint. Panels (Q) – (FF) show the same as (A) – (O) but for the HFT method. For the simulations that included sweeps, sampling began 50 generations after the beginning of the sweep, and the selection coefficient was set to 0.05.

S4 Fig: Example and average `Timesweeper` inputs for datasets of varying number of sampled timepoints. For all simulations, a total of 200 diploid individuals were sampled, whether all from one timepoint or spread across several timepoints. (A) and (B) show individual example inputs and average inputs of the Neutral, SSV, and SDN classes for the AFT method where each input was constructed from a data set of 1 sampled timepoint. (C) and (D) show the same for the HFT method. The remaining panels show the same information for increasing numbers of timepoints: 2 timepoints in panels (E) – (H), 5 timepoints in panels (I) – (L), 10 timepoints in panels (M) – (P), 20 timepoints in panels (Q) – (T), 40 timepoints in panels (U) – (X), and 100 timepoints in panels (Y) – (BB). For the simulations that included sweeps, sampling began 50 generations after the beginning of the sweep, and the selection coefficient was set to 0.05.

S5 Fig: `Timesweeper`'s classification performance on datasets of varying number of sampled timepoints. Same as S3 Fig, but with `Timesweeper` trained and tested on each of the scenarios whose inputs are shown in S4 Fig.

S6 Fig: Example and average `Timesweeper` inputs for datasets of varying number of generations between the beginning of the sweep and the beginning of the sampling period. Initial sampling times range from coincident with the sweep (A) – (D) to beginning 200

975 generations after the start of the sweep (Q) – (T). 10 diploid individuals were sampled at each of
976 20 timepoints, and for all simulations that included sweeps, and selection coefficient was set to
977 0.05.

978

979 **S7 Fig: `Timesweeper`'s classification performance on datasets of varying number of**
980 **number of generations between the beginning of the sweep and the beginning of the sampling**
981 **period.** Same as S3 Fig, but with `Timesweeper` trained and tested on each of the scenarios
982 whose inputs are shown in S6 Fig.

983

984 **S8 Fig: Example and average `Timesweeper` inputs for datasets of varying selection**
985 **strength of selection.** Selection coefficients range from 0.005 (in panels A through D) to 0.5
986 (panels Q through T). 10 diploid individuals were sampled at each of 20 timepoints, and for all
987 simulations that included sweeps, sampling began 50 generations after the start of the sweep.

988

989 **S9 Fig: `Timesweeper`'s classification performance on datasets of varying selection strength**
990 **of selection.** Same as S3 Fig, but with `Timesweeper` trained and tested on each of the scenarios
991 whose inputs are shown in S8 Fig.

992

993 **S10 Fig: `Timesweeper`'s AFT method's performance when trained on the out-of-Africa**
994 **model and tested on data simulated under equilibrium demography.** (A) Confusion matrix.
995 (B) ROC Curve for discriminating between sweeps and neutrality. (C) PR curve for discriminating

996 between sweeps and neutrality. (D) The proportions of polymorphisms at various locations along
997 a 500 kb that were classified as a sweep, with sweeps occurring in the center of the window in the
998 SSV and SDN models, following Figure 4.

999

1000 **S11 Fig: `Timesweeper`'s AFT method's performance when trained data simulated under**
1001 **equilibrium demography and applied to data simulated under the out-of-Africa model.** Same
1002 as S10 Fig, but with the direction of model misspecification reversed.

1003

1004 **S12 Fig: Confusion matrices showing the AFT method's performance after imposing**
1005 **increasingly stringent class membership probability thresholds for detecting sweeps.**

1006

1007 **S13 Fig: Example and average `Timesweeper` inputs from simulated training data for the**
1008 ***D. simulans* evolve-and-resequence dataset.** (A) – (C) show individual example inputs and (D)
1009 shows average inputs of the Neutral, SSV, and SDN classes for the AFT method where data were
1010 simulated for the *D. simulans* evolve-and-resequence dataset as described in the Methods.

1011

1012 **S14 Fig: `Timesweeper`'s classification performance on simulated test data for the *D.***
1013 ***simulans* evolve-and-resequence dataset.** (A) – (D) Confusion matrix, ROC curve, training
1014 trajectory, and precision-recall (PR) curve showing the performance of the AFT method on
1015 simulated *D. simulans* data as described in the Methods.

1016

1017 **Table S1: Summaries of `Timesweeper`'s classification results on *D*. simulans evolve-and-**

1018 **resequence dataset binned by Fisher's Exact Test *p*-value.**

1019

1020 **Table S2: Summaries of Fisher's Exact Test results on *D. simulans* evolve-and-resequence**

1021 **dataset binned by `Timesweeper`'s sweep probability score.**

1022