Evolutionary Bioinformatics

TECHNICAL ADVANCE

POPBAM: Tools for Evolutionary Analysis of Short Read Sequence Alignments

Daniel Garrigan

Department of Biology, University of Rochester, Rochester, New York 14627 USA. Corresponding author email: dgarriga@bio.rochester.edu

Abstract

Background: While many bioinformatics tools currently exist for assembling and discovering variants from next-generation sequence data, there are very few tools available for performing evolutionary analyses from these data. Evolutionary and population genomics studies hold great promise for providing valuable insights into natural selection, the effect of mutations on phenotypes, and the origin of species. Thus, there is a need for an extensible and flexible computational tool that can function into a growing number of evolutionary bioinformatics pipelines.

Results: This paper describes the POPBAM software, which is a comprehensive set of computational tools for evolutionary analysis of whole-genome alignments consisting of multiple individuals, from multiple populations or species. POPBAM works directly from BAM-formatted assembly files, calls variant sites, and calculates a variety of commonly used evolutionary sequence statistics. POPBAM is designed primarily to perform analyses in sliding windows across chromosomes or scaffolds. POPBAM accurately measures nucleotide diversity, population divergence, linkage disequilibrium, and the frequency spectrum of mutations from two or more populations. POPBAM can also produce phylogenetic trees of all samples in a BAM file. Finally, I demonstrate that the implementation of POPBAM is both fast and memory-efficient, and also can feasibly scale to the analysis of large BAM files with many individuals and populations.

Software: The POPBAM program is written in C/C++ and is available from http://dgarriga.github.io/POPBAM. The program has few dependencies and can be built on a variety of Linux platforms. The program is open-source and users are encouraged to participate in the development of this resource.

Keywords: divergence, linkage disequilibrium, polymorphism, natural selection, next-generation sequencing

Evolutionary Bioinformatics 2013:9 343–353

doi: 10.4137/EBO.S12751

This article is available from http://www.la-press.com.

Introduction

Evolutionary genetics research routinely capitalizes on the vast amount of data made available by next-generation sequencing technologies. It is now possible to generate high-quality, high-coverage sequence of genomes from conspecific samples or samples from closely related species. These new whole-genome evolutionary analyses are yielding remarkable insights into the nature of adaptive evolution, the genetic basis of quantitative traits, and the origin of species.[1–5] Many of these new studies employ chromosome-level spatial scans to identify regions of potential interest. For example, Ellegren et al[2] scanned the genomes of two closely related species of *Ficedula* flycatcher to identify genomic regions with elevated levels of sequence divergence to aid in the identification of genomic regions involved in the evolution of reproductive isolation. Similarly, Jones et al[4] scanned the genomes of freshwater and marine sticklebacks to identify the genetic basis for adaptation to freshwater and marine environments. Finally, Garrigan et al[3] used whole-genome scans to discover regions of gene flow between species of *Drosophila* that have already evolved post-zygotic reproductive isolation.

There is a rapidly increasing number of computational tools for performing either reference-based or *de novo* genome assembly from short-read next-generation sequence data. For reference-based assembly, much of the research has concentrated on developing efficient algorithms for indexing either the reference sequence, the short-read sequences, or both (see Li and Homer[6] for a review). Similarly, the development of algorithms for *de novo* sequence assembly is also an active field of research and many methods seek to optimize the resolution of the large de Bruijn (or related) graphs into contigs and scaffolds (see Miller et al[7] for a review). However, tools for sequence assembly are not the only areas of active research. Significant progress is being made on improving algorithms for many downstream tasks, such as improving scaffold construction,[8] dealing with repetitive sequence,[9] calling single nucleotide polymorphisms (SNPs),[10] and phasing of haplotypes.[11]

Current research should be directed towards the development of computational tools for assembling and refining high-quality alignments from next-generation sequence data. However, much less development has been directed towards downstream evolutionary-based analyses, such as measuring levels of nucleotide diversity and linkage disequilibrium (LD), fitting models of population divergence, and testing for natural selection. It is evident that the analysis tools that evolutionary geneticists have relied on for so long do not feasibly scale to the level of whole-genome analysis.[12] Most evolutionary investigators working with large whole-genome alignments are forced to cobble together a custom bioinformatics pipeline and develop their own tools to parse the needed information. In this paper, I describe an effort to provide a publicly available solution for the evolutionary analysis portion of a next-generation sequencing pipeline. The POPBAM program is a comprehensive suite of evolutionary genomics tools that can be used to analyze whole-genome alignments from multiple individuals, from multiple populations. POPBAM is ideally designed to work in sliding windows across chromosomes or scaffolds. POPBAM has been in continuous development since 2010 and is currently in its third beta release. This paper describes the basic functionality, usage, and performance of the program. POPBAM is also open-source and the community is encouraged to request and help develop needed functionality.

Methods

Overview of the POPBAM program

The POPBAM program is written in a mixture of the C and C++ languages. POPBAM is provided as open-source software under the MIT license. POPBAM is currently in its third beta release. The source code is freely available through the GitHub website (http://dgarriga.github.io/POPBAM). The only dependency for compiling POPBAM is the zlib compression library (http://zlib.net). POPBAM successfully compiles with the GCC compiler (versions 4.4.6 up to 4.7.2 have been tested with the current POPBAM source) and was successfully built with the Microsoft Visual C++ compiler (version 11.0.60315). However, successful compilation requires the C++ 11 standard library.[13] Currently, POPBAM comprises more than 17,000 lines of source code. The majority of the source code for reading BAM files and constructing pileups was forked from SAMtools version 0.1.18 (https://github.com/samtools/htslib), which also includes code for the klib library for constructing generic hash tables (https://github.com/attractivechaos/klib).

Furthermore, source code from the PHYLIP version 3.69[14] program was incorporated and modified to construct neighbor-joining trees.

Inputting data into POPBAM

The sequence alignment/map (SAM) format and the compressed version (BAM) are the *de facto* standard for storing reference-based next-generation sequence assemblies. BAM files include a header containing metadata, the sequence reads, mapping and base qualities, and alignment information in an efficiently compressed format.[15] While POPBAM is capable of reading any BAM file, it is specifically designed to work with merged BAM files that contain reads from multiple samples (individuals). Furthermore, POPBAM is able to group the individual samples according to the population from which it was taken. How POPBAM deals with assigning reads to populations depends upon the information in the header of the BAM file itself. A BAM file header will typically include a sequence dictionary with information about the reference sequence used to generate the assembly and additional information about each read group. A read group typically represents the set of short read sequences generated from a single lane in an Illumina sequencer. In the BAM header, individual read groups can be annotated as coming from a particular sample, or individual (denoted with the standard tag "SM" in the "@RG" line in the BAM header). Therefore, individual samples can be represented by multiple lanes from a sequencer. As an additional layer of hierarchical grouping, POPBAM allows samples to be grouped according to population; this is achieved by the user inserting a modified "@RG" header line that includes the new "PO" tag. The "PO" tag can be any string, as long as the string is identical between samples from the same population.

As an example of how POPBAM assigns reads to samples and populations, consider a BAM file that has three read groups, which are labeled R21, R22, and R25. In this hypothetical example, the R22 and R25 read groups are from two different individuals of the fruit fly species *Drosophila melanogaster* called "MEL001" and "MEL002", respectively. The third read group, R21, is from a single individual of the related species *D. mauritiana* labeled "MAU001". Thus, the above read group information from the header of this example BAM file would appear as the following three lines:

```
@RG     ID:R21     SM:MAU001     PO:MAU
@RG     ID:R22     SM:MEL001     PO:MEL
@RG     ID:R25     SM:MEL002     PO:MEL
```

POPBAM will analyze the above example as two populations, "MAU" and "MEL", with the "MEL" population having two samples, each represented by a single read group and the "MAU" population represented by only one sample and one read group. The non-standard "PO" tag has to be added manually by the user by editing the text of the BAM header and then re-headering the BAM file with programs such as SAMtools[15] or PICARD (http://picard.sourceforge.net). Alternatively, POPBAM has an option to allow the user to input the BAM header as a separate text file. When adding the "PO" tag, please be advised that the fields of the read group entry must be tab-delimited.

SNP calling

POPBAM will call the consensus genotype for each individual included in a BAM file. POPBAM has the option of calling either a haploid or diploid genotype. The method that POPBAM uses for calling genotypes follows the method outlined in Section 3 of the Supplementary Text of Li et al.[16] Briefly, let α_{nk} be the probability that exactly k reads have errors out of a total of n reads covering a given genomic position. If the sequences of all reads are assumed to be generated independently, then we might expect that

$$\alpha_{nk} = \binom{n}{k} \prod_{i=1}^{k} \varepsilon_i \prod_{j=k+1}^{n} (1-\varepsilon_j), \qquad (1)$$

where ε_x is the probability that the base in read x is incorrectly called. Empirically, the values of ε are the error probabilities generated during sequencing. However, if errors can be correlated, then we can re-write α_{nk} as

$$\alpha_{nk} \approx c_{nk} \prod_{i=1}^{k} \varepsilon_i^{\lambda}, \qquad (2)$$

in which c_{nk} is the probability of k errors, given $k-1$ errors, and λ is a parameter that controls the error

dependency. POPBAM assumes that $\lambda = 0.85$.[16] Furthermore, POPBAM improves the accuracy of the consensus genotype call by calculating the probability in Equation 2 separately for reads coming from different strands since errors are expected to be independent between reads for the forward and reverse strands.

POPBAM has a number of options for filtering sites based on different assessments of data quality. All POPBAM commands have global options for filtering reads on quality. For example, the user can set a minimum base quality score for POPBAM to consider individuals sites. Similarly, the user can set minimum mapping qualities for individual reads to be considered, a minimum root mean square mapping score for a consensus base to be called, or a minimum SNP quality score to consider a variant as valid. All quality scores are implemented as Phred scores.[17] Quality scores are assumed to be on the Sanger scale; however, POPBAM can also convert quality scores from the Illumina 1.3+ format.[18] Lastly, users may also exclude sites that either do not have a minimum number of reads aligned for an individual sample or exceed a maximum number of mapped reads. Sites with insertions or deletions are detected by POPBAM, but not considered for further analysis.

Evolutionary analyses

Table 1 lists the seven main command options for the POPBAM program. The first command is called **snp** and it is used to output the raw genotype calls to a formatted output file, as described above. The default output for SNP calls is native POPBAM format, which is a tab-delimited text file. However, the **snp** function will also output SNP calls to a format that can be read by the SweepFinder program[19] and in the output format of the ms program,[20] which can be post-processed by programs using the libsequence library.[21]

POPBAM provides a range of options for analyzing haplotype structure and diversity. Most haplotype-based options are implemented in the **haplo** and **ld** commands and focus on statistics that are useful for detecting natural selection or population structure. Please note that POPBAM does not have the functionality to reconstruct haplotypes from unphased diploid data. There are three analysis options accessible via the **haplo** command. The first option is simply calculating the number of distinct haplotypes spanning the entire length of a window and the corresponding haplotype diversity[22] for each population. The second option calculates the site-specific extended haplotype homozygosity (*EHHS*) statistic for detecting partial selective sweeps.[23] The POPBAM implementation of the *EHHS* statistic works by first scanning each polymorphic site in a window and recording the partition induced by the site within each population (the site type). For a given variable site, if the ancestral allele is denoted as A and the derived allele as V, the site type at SNP i can be represented as the bit array $\boldsymbol{T}_i = \{[T_{i,1} = V], [T_{i,2} = V], \ldots, [T_{i,n} = V]\}$ (square brackets indicate Iverson bracket). Thus, an individual is represented by a zero at the site when the sample has a consensus base identical to the reference (or outgroup) allele and a one when that individual carries the derived allele. POPBAM then calculates *EHHS* using the vector $\boldsymbol{T}_i$ (and its complement $\boldsymbol{T}_i'$) that appears the most frequently in the window; this site is considered the "core SNP". If all site types are singletons, then there is no strong LD and the statistic has negligible biological meaning; in this case POPBAM

Table 1. A description of the seven analysis commands that are available in the current version of the POPBAM software.

Command	Description of analysis
snp	Generates consensus base calls for each sample in the alignment that passes user-specified quality filters.
haplo	Computes a variety of haplotype-based statistics.
diverge	Calculates divergence of samples or populations from the reference sequence.
tree	Outputs newick-formatted neighbor-joining trees of all samples in the file.
nucdiv	Estimates nucleotide diversity within each population and mean number of nucleotide differences between all pairs of populations.
ld	Computes several measures of linkage disequilibrium.
sfs	Calculates the site frequency spectrum statistics Tajima's *D* and the standardized Fay and Wu's *H* for each population.

will simply use the last site type that it recorded. Let h_s be the unbiased estimate of homozygosity at the core SNP,

$$h_s = \frac{j^2 + (n-j)^2 - n}{n(n-1)}, \quad (3)$$

where j is the count of the derived allele and n is the total sample size.[24] Similarly, define h_k as an unbiased estimate of extended haplotype homozygosity over the entire genomic interval,

$$h_k = \frac{\sum_{i=1}^{K} k_i^2 - n}{n(n-1)}, \quad (4)$$

in which k_i is the count of extended haplotype i and K is the total number of unique extended haplotypes. Then,

$$EHHS = \frac{h_k}{h_s} = \frac{\sum_{i=1}^{K} k_i^2 - n}{j^2 + (n-j)^2 - n}. \quad (5)$$

It is expected that $EHHS = 0$ when each individual in the sample carries a unique haplotype, while $EHHS = 1$ when the haplotype homozygosity is identical to the site homozygosity at the core SNP. The POPBAM **haplo** command also implements a measure of relative between-population haplotype sequence identity

$$G_{\min} = \frac{\min(d_{xy})}{\bar{d}_{xy}} \quad (6)$$

(see below for definitions of $\bar{d}_{xy}$ and d_{xy}), which is particularly sensitive to the effects of recent gene flow.[25]

The POPBAM **ld** command includes options to calculate Kelly's Z_{nS} statistic,[26] the $\omega_{\max}$ statistic for detecting recent selective sweeps,[27] and two site congruency statistics, Wall's B and Q.[28] The Z_{nS} statistic is simply defined as the average pairwise r^2 value,

$$Z_{nS} = \frac{2}{S(S-1)} \sum_{i=1}^{S-1} \sum_{j=i+1}^{S} r_{ij}^2, \quad (7)$$

in which r_{ij} is the correlation coefficient of allele frequencies at SNPs i and j and S is the number of segregating sites in a window.[26] POPBAM has the option of excluding singleton sites when calculating the pairwise values of r^2. Similarly, for each segregating site, LD can be partitioned on either side of the site to calculate ω:

$$\omega = \frac{\left(\binom{l}{2} + \binom{S-l}{2}\right)^{-1} \left(\sum_{i,j \in L} r_{ij}^2 + \sum_{i,j \in R} r_{ij}^2\right)}{(1/l(s-l)) \sum_{i \in L, j \in R} r_{ij}^2}. \quad (8)$$

Then, $\omega_{\max}$ is derived from the site in the window that yields the maximum value of ω.[27] Lastly, Wall's B and Q statistics focus on runs of segregating sites that partition the data in an identical pattern. If site type $\boldsymbol{T}_i = \boldsymbol{T}_{i+1}$ or $\boldsymbol{T}_i = \boldsymbol{T}'_{i+1}$ (its complement), then one is added to variable $\boldsymbol{B'}$. Wall's B statistic is then,

$$B = \frac{B'}{S-1} \quad (9)$$

and similarly, for Wall's Q,

$$Q = \frac{(B' + |M|)}{S}, \quad (10)$$

in which $|M|$ is the number of distinct partitions induced by congruent pairs of segregating sites.[28]

In addition to the haplotype-based statistics, POPBAM provides functionality for calculating nucleotide diversity within populations and the average number of nucleotide differences between populations. The POPBAM **nucdiv** command will produce unbiased estimates of nucleotide diversity (π) within each population as

$$\pi = \frac{2}{(n-1)^2} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} d_{ij}, \quad (11)$$

in which d_{ij} is the Hamming distance between sequences i and j, as well as the average number of nucleotide differences between populations,

$$\bar{d}_{xy} = \frac{1}{n_x n_y} \sum_{x=1}^{n_x} \sum_{y=1}^{n_y} d_{xy}, \quad (12)$$

in which d_{xy} is the Hamming distance between all sequences from population x with sample size n_x and population y with sample size n_y.[29] POPBAM reports both π and $\bar{d}_{xy}$ per aligned nucleotide site. From these two quantities, it is straightforward to calculate a widely-used formulation of F_{ST}[30] in a post-processing step as,

$$F_{ST}(x,y) = 1 - \frac{\pi_x + \pi_y}{2\bar{d}_{xy}}. \quad (13)$$

The **nucdiv** command also allows the user to specify the minimum number of samples required to calculate the diversity statistics, thus not requiring complete coverage across all samples.

The **diverge** command allows the user to calculate the divergence of the samples with either the reference sequence, or a specified outgroup sequence, using either a Hamming distance or a Jukes-Cantor corrected distance.[31] The **diverge** command can output either the divergence of each sample from the outgroup/reference genome or the mean divergence of each population. Since the **diverge** command is intended for calculating local substitution rates, the user may also specify whether only sites that are fixed substitutions should be used in the calculation. The output of the **diverge** command also outputs information that can be used to perform the HKA test[32] in a post-processing step.

The POPBAM **tree** command builds neighbor-joining[33] (NJ) trees of all samples present in the BAM file and outputs the newick-formatted trees for each genomic window. The NJ algorithm was selected over other tree-building methods to avoid the computational cost of using a heuristic tree-building algorithm on a potentially large number of genomic intervals. The **tree** command will only consider sites for which every individual is represented in the alignment. NJ trees can be calculated using either a Hamming distance or the Jukes-Cantor sequence distance.[31]

Finally, the **sfs** command calculates statistics that summarize the site frequency spectrum within populations. For each window in each population the count of mutations with frequency i in the sample is given by ξ_i, for which the total number of segregating sites is $S = \sum_{i=1}^{n-1} \xi_i$. The first site frequency spectrum measure output by POPBAM is Tajima's D statistic.[34] Tajima's D contrasts low- and intermediate-frequency variants and can be calculated as:

$$D = \sum_{i=1}^{n-1} \xi_i \frac{\frac{2i(n-i)}{n(n-1)} - \frac{1}{a_1}}{\sqrt{e_1 S + e_2 S(S-1)}}, \quad (14)$$

in which $a_1 = \sum_{i=1}^{n-1} 1/i$ and e_1 and e_2 are given by Tajima.[34,35] The second site frequency spectrum measure is a standardized analog of Fay and Wu's H statistic.[36,37] The H statistic contrasts high- and intermediate-frequency variants and can be calculated as:

$$H = \sum_{i=1}^{n-1} \xi_i \frac{\frac{2i(n-i)}{n(n-1)} - \frac{i}{n-1}}{\sqrt{g_1 \theta + g_2 \theta^2}}, \quad (15)$$

in which g_1 and g_2 are given in Zeng et al[37] and POPBAM estimates θ as Watterson's moment estimator of the population mutation rate.[38] Tajima's D and Fay and Wu's H were chosen because they consider both the folded and unfolded site frequency spectra, respectively. For the unfolded site frequency spectrum, POPBAM assigns ancestral and derived alleles using the reference sequence; however, the user can also manually specify which sequence should be used as an outgroup to polarize the directionality of a given mutation. The raw count of derived mutation frequencies can be obtained using the **snp** command with SweepFinder formatted output.

Output and post-processing

POPBAM is designed to calculate a variety of evolutionary-based statistics in non-overlapping genomic windows. The windows are uniformly sized relative to the reference genome coordinates and the size of the windows can be specified by the user at run time. All POPBAM output is formatted as text and is tab-delimited. The output of POPBAM is specifically tailored to be input seamlessly into the R statistical environment.[39] Each of the POPBAM commands produces differing output fields; however, the first four fields are identical across commands: the first column always prints the scaffold/chromosome name, the second column prints the position in the reference sequence where a window starts, the third column

prints the reference position of the end of the window, and the fourth column is the total number of aligned sites in the window that was used in the analysis (this includes invariant sites). The columns that follow the fourth column depend upon the POPBAM command and the output options being used. Typically, statistics will be output grouped by populations in the order in which the populations are defined in the BAM header. Exceptions to this output format are limited to the **snp** function, which generates output formatted as input for a different program or as a list of consensus genotype calls, which also give the SNP quality score and the read depth for each individual in the BAM file. If a particular calculation is not possible in a given window, POPBAM will output the string "NA", which will be read by the R statistical environment as a missing value.

I provide an example case of using POPBAM to scan the major chromosome arms of ten lines of *Drosophila melanogaster*[40] and a single line of *D. mauritiana*[3] for the signature of recent positive natural selection. The *D. melanogaster* sample comprises nine lines from sub-Saharan Africa and a single line from France. Furthermore, each *D. melanogaster* individual is represented by a single read group of paired-end 76 bp Illumina reads. The *D. mauritiana* outgroup is from a single individual with a single read group of paired-end 86 bp Illumina reads. First, the POPBAM **nucdiv** command was run in 10 kb windows on all major chromosome arms to identify regions of low nucleotide diversity. Genomic windows appearing the the lowest 1% quantile were considered candidates for recent positive natural selection. The POPBAM **ld** command in 10 kb windows with the ω_{max} option was then run on chromosome *3R* to identify patterns of linkage disequilibrium that are consistent with a model of a recent selective sweep. Lastly, the POPBAM **tree** command was used in 1 kb windows in candidate sweep regions to visualize the patterns of polymorphism.

Assessing performance

The performance of the POPBAM program was assessed using a BAM file created from the *Drosophila melanogaster* short read data described in the previous subsection. However, only the X chromosome was considered for the purposes of measuring the execution time and memory usage of the POPBAM program. The length of the reference sequence for chromosome *X* is 22,422,827 bp and the input BAM file is 6.8 Gb consisting of a total of 103,464,508 mapped reads. The POPBAM program was tested on server running the Red Hat® Enterprise Linux® 6.4 operating system with dual Intel® Xeon® L5630 processors running at 2.13 GHz with 144 Gb of RAM. POPBAM was compiled using GCC version 4.4.7-3.

Program execution time was assessed using the GNU time program (version 1.7) for the seven different POPBAM commands. Additionally, program execution time as a function of the total alignment length, number of individuals, and the window size was also measured. Furthermore, a profile of POPBAM was created using the GNU profiling tool gprof and the memory heap was profiled using the massif module of the program valgrind.

Results

Performance

Table 2 shows the execution time and peak memory usage trials for each of the major evolutionary analyses performed by POPBAM. In this case, the window size is 10 kb and there is a small amount of variation in both execution time and peak memory usage among the different POPBAM functions. The **snp** and **sfs** commands tend to have longer execution times, while the **tree** function tends to have the shortest execution times. Figure 1 shows execution times and peak memory usage for different window sizes, different total alignment lengths, and different numbers of individual samples in the BAM file. The execution time of POPBAM increases linearly with the total alignment length and the number of individuals. The execution time is greater for small windows compared to large windows because of the increased number of disk input/output operations and the number of times memory blocks are allocated and freed from the heap. Peak memory usage increases with larger window sizes because there are more SNPs to be retained in memory with larger windows. This increase in peak memory usage is greater than linear for POPBAM functions that perform pairwise calculations between sites (ie, functions in the **ld** command). Finally, the peak memory usage is not strongly influenced by the

Table 2. Time and memory trials reported separately for the main components of POPBAM functionality. The circumstances of the trials are provided in the text.

Command	Analysis	Time (seconds)	Memory (Mb)
snp	Call and output SNPs	5074.14	65.546
haplo	Calculate number of haplotypes	4998.22	65.386
	Site-specific extended haplotype homozygosity	4727.29	65.386
	Minimum between-population distance	4592.35	65.386
diverge	Calculate number of substitutions	4587.08	65.567
tree	Compute neighbor-joining trees	4497.68	65.388
nucdiv	Within and between population differences	4762.05	65.546
Ld	Calculate mean r^2	4553.11	65.385
	Calculate ω_{max}	4763.04	66.095
	Calculate Wall's congruency statistics	4902.55	65.386
sfs	Tajima's *D* and Fay and Wu's *H*	5024.26	65.386

total alignment length and it increases linearly with the number of samples in a merged BAM file.

The profiling of POPBAM indicates that the program spends nearly two-thirds of its execution time parsing the auxiliary data associated with each read and assigning individual reads to samples and populations. Over 15% of POPBAM execution time is spent generating consensus genotypes. Slightly less than 10% of the execution time is spend processing the pileup from the BAM file. Approximately 8% of the total time is spent implementing the error model. All other functions take less than 1.5% of POPBAM's total execution time. This suggests that POPBAM's run time could be shortened considerably by allowing the program to operate on an input file that consists only of previously called consensus genotypes, such as the VCF format.[41] However, most analyses performed by POPBAM would require a VCF file that also includes all invariant sites as well as variant sites.

Example output

Figure 2A plots the output of the POPBAM **nucdiv** command over all major chromosome arms of *Drosophila melanogaster* in non-overlapping 10 kb windows. The reduced levels of nucleotide diversity in regions extending away from centromeres and telomeres accords well with the known reduction in crossing-over rates in these regions.[5] However, unlike non-African *D. melanogaster*, the POPBAM analysis

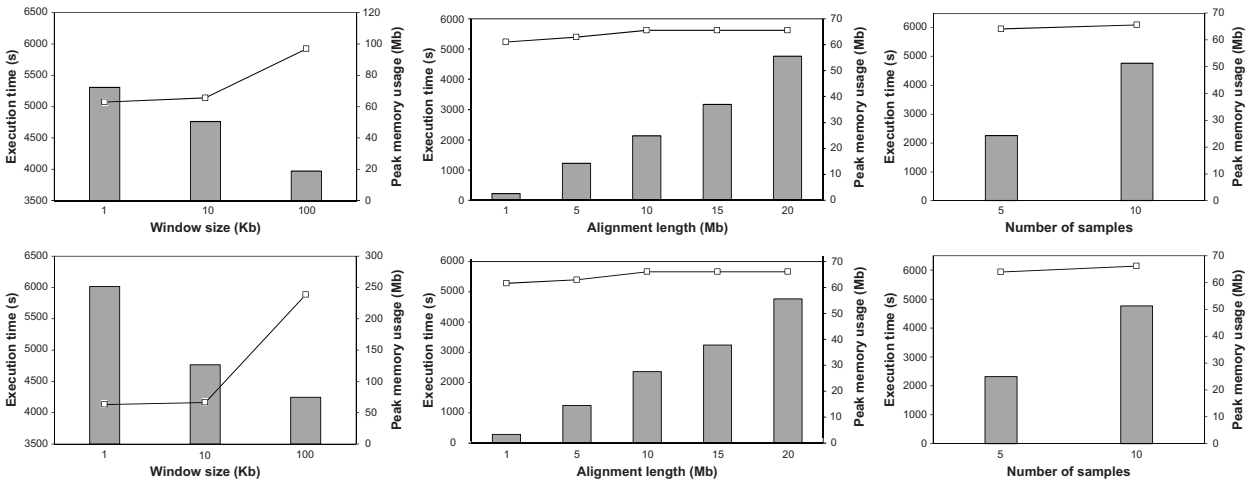

Figure 1. Time and memory trials for the **nucdiv** command (top row) and the **ld** command using the ω_{max} option (bottom row). In all graphs, the filled bars show the execution time (in seconds) and the line shows the peak memory usage in megabytes (note that megabytes are shown on the secondary vertical axis, while megabases are shown on the horizontal axis). The first column of graphs shows results for various window sizes. The second column shows results for different total alignment length. Finally, the third column of graphs shows the results for BAM files consisting of either five or ten ingroup individuals (not including one outgroup individual).

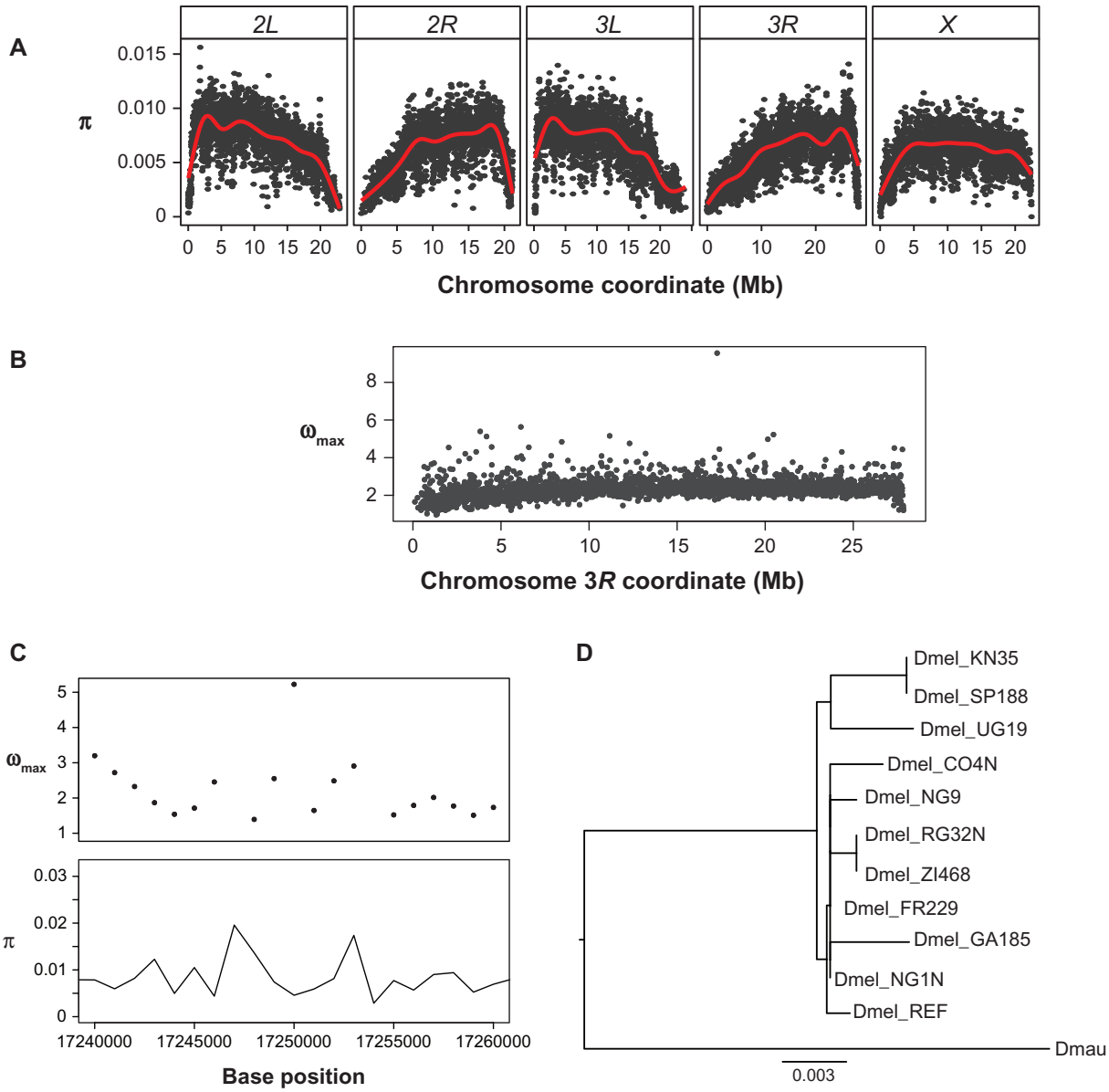

Figure 2. Panel (**A**) shows the spatial distribution of nucleotide diversity (π) in 10 kb windows across each of the major chromosome arms in ten lines of *Drosophila melanogaster*. Panel (**B**) shows the distribution of 10 kb windows on chromosome 3*R* for the linkage disequilibrium statistic, ω_{max}, which is sensitive to patterns caused by recent selective sweeps. The window beginning at position 17,250,000 has a significant outlier for ω_{max} (see text). Panel (**C**) shows 1 kb windows of ω_{max} (top) and π (bottom) in the outlier region of chromosome 3*R*. Finally, panel (**D**) shows the neighbor-joining tree for the 1 kb window spanning positions 17,250,000–17,251,000 on chromosome 3*R* with the highest ω_{max} statistic. *D. melanogaster* sequences are labeled with the prefix "Dmel" and the *D. mauritiana* sequence is labeled with the prefix "Dmau".

shows that, for the autosomes, the mean $\pi_A = 0.0064$, while for the X chromosome, the mean $\pi_X = 0.0060$. The ratio of $\pi_X/\pi_A = 0.934$, which is much less that the value of 3/4 expected under an equal breeding sex-ratio.[40] Figure 2B shows the distribution of the ω_{max} statistic in 10 kb intervals across chromosome 3*R*. POPBAM identifies an outlier window that spans positions 17,250,000–17,260,000 and has a pattern of linkage disequilibrium consistent with the recent action of positive natural selection ($\omega_{max} = 9.55$; $P < 0.05$ of randomly selecting a window with a higher, or equal, value of ω_{max}). POPBAM was then used to examine more fine-scale patterns of nucleotide diversity and linkage disequilibrium in between genome coordinates 17,240,000 and 17,260,000 to attempt to localize a putative signal of positive selection (Fig. 2C). This fine-scale scan shows a 1 kb window with elevated ω_{max} and depressed π between positions 17,250,000 and 17,251,000. POPBAM was then used to build neighbor-joining trees in 1 kb

windows across the candidate region. Figure 2D shows the shape of the putatively selected genealogy for the 17,250,000–17,251,000 interval. The above example is intended to illustrate the use of POPBAM to scan for and explore regions of genomes that have putatively experienced recent positive natural selection.

It is worth noting that, during its years of development, the numerical accuracy of the statistics calculated by POPBAM have been verified by comparing them with calculations from other programs, such as those built with the libsequence package[21] and the DnaSP program.[42]

Conclusions

The POPBAM program is a flexible and extensible tool for conducting studies of population and evolutionary genomics. Although the current beta release of POPBAM has some necessary limitations, goals for the release candidate version of POPBAM include the ability to read and write VCF files and to incorporate genome features, such as those stored in GFF3 files (http://www.sequenceontology.org/resources/gff3.html), to measure levels of nonsynonymous and synonymous polymorphism and divergence, and to analyze polymorphism and divergence by site type (eg, exon, intron, 5′ untranslated regions, etc.). It is important to note that POPBAM is not intended to perform higher-order functions, such as multi-locus or composite likelihood types of analyses. Rather, POPBAM is intended to be used as the penultimate step in any evolutionary bioinformatics pipeline, to fill the void between the genome assembly steps and the final, integrative whole-genome statistical analyses. It fulfills this role by providing an efficient means to measure many aspects of genome polymorphism and divergence. By making POPBAM open-source, I hope that additional functionality and improvements will be suggested by the community of evolutionary investigators working with large next-generation sequence data sets.

Acknowledgments

Thanks to J. Fuller, A. Geneva, S. Kingan, L. Lovato, and J. Vedanayagam for thorough alpha testing of the POPBAM program. Special thanks to Carlos Machado and the Oryza Genome Evolution project for extensive beta testing of POPBAM. Thanks to Sarah Kingan for comments on an earlier draft of this manuscript and helping run the analysis on the *Drosophila melanogaster* data. Thanks also to three anonymous reviewers for constructive comments on an earlier draft of the manuscript.

Author Contributions

Conceived and designed the experiments: DG. Analyzed the data: DG. Wrote the first draft of the manuscript: DG. Contributed to the writing of the manuscript: DG. Agree with manuscript results and conclusions: DG. Jointly developed the structure and arguments for the paper: DG. Made critical revisions and approved final version: DG. All authors reviewed and approved of the final manuscript.

Funding

This work was made possible by funds from the University of Rochester and the a grant from the National Institutes of Health (R01OD010548).

Competing Interests

Author(s) disclose no potential conflicts of interest.

Disclosures and Ethics

As a requirement of publication the authors have provided signed confirmation of their compliance with ethical and legal obligations including but not limited to compliance with ICMJE authorship and competing interests guidelines, that the article is neither under consideration for publication nor published elsewhere, of their compliance with legal and ethical guidelines concerning human and animal research participants (if applicable), and that permission has been obtained for reproduction of any copyrighted material. This article was subject to blind, independent, expert peer review. The reviewers reported no competing interests.

References

1. Cao J, Schneeberger K, Ossowski S, et al. Whole-genome sequencing of multiple *Arabidopsis thaliana* populations. *Nat Genet*. 2011;43:956–63.
2. Ellegren H, Smeds L, Burri R, et al. The genomic landscape of species divergence in *Ficedula* flycatchers. *Nature*. 2012;491:756–60.
3. Garrigan D, Kingan SB, Geneva AJ, et al. Genome sequencing reveals complex speciation in the *Drosophila simulans* clade. *Genome Res*. 2012;22:1499–511.
4. Jones FC, Grabherr MG, Chan YF, et al. The genomic basis of adaptive evolution in threespine sticklebacks. *Nature*. 2012;484:55–61.
5. Mackay TFC, Richards S, Stone EA, et al. The *Drosophila melanogaster* genetic reference panel. *Nature*. 2012;482:173–8.

6. Li H, Homer N. A survey of sequence alignment algorithms for next-generation sequencing. *Brief Bioinform*. 2010;11:473–83.
7. Miller JR, Koren S, Sutton G. Assembly algorithms for next-generation sequencing data. *Genomics*. 2010;95:315–27.
8. Kim J, Larkin DM, Cai Q, et al. Reference-assisted chromosome assembly. *Proc Natl Acad Sci U S A*. 2013;110:1785–90.
9. Treangen TJ, Salzberg SL. Repetitive DNA and next-generation sequencing: computational challenges and solutions. *Nat Rev Genet*. 2012;13:36–46.
10. Nielsen R, Paul JS, Albrechsten A, Song YS. Genotype and SNP calling from next-generation sequencing data. *Nat Rev Genet*. 2011;12:443–51.
11. Aguiar D, Istrail S. HapCompass: a fast cycle basis algorithm for accurate haplotype assembly of sequence data. *J Comput Biol*. 2012;19:577–90.
12. Sousa V, Hey J. Understanding the origin of species with genome-scale data: modelling gene flow. *Nat Rev Genet*. 2013;14:404–14.
13. International Organization for Standardization. Information technology—Programming languages—C++. ISO/IEC 14882:20112011.
14. Felsenstein J. PHYLIP (Phylogeny Inference Package) version 3.6. 2005. Distributed by the author. Department of Genome Sciences, University of Washington, Seattle.
15. Li H, Handsaker B, Wysoker A, et al. The Sequence Alignment/Map format and SAMtools. *Bioinformatics*. 2009;25:2078–9.
16. Li H, Ruan J, Durbin R. Mapping short DNA sequencing reads and calling variants using mapping quality scores. *Genome Res*. 2008;18:1851–8.
17. Ewing B, Green P. Base-calling of automated sequencer traces using Phred. II. Error probabilities. *Genome Res*. 1998;8:186–94.
18. Cock PJ, Fields CJ, Goto N, Heuer ML, Rice PM. The Sanger FASTQ file format for sequences with quality scores, and the Solexa/Illumina FASTQ variants. *Nucleic Acids Res*. 2010;38:1767–71.
19. Nielsen R, Williamson S, Kim Y, Hubisz MJ, Clark AG, Bustamante C. Genomic scans for selective sweeps using SNP data. *Genome Res*. 2005;15:1566–75.
20. Hudson RR. Generating samples under a Wright-Fisher neutral model of genetic variation. *Bioinformatics*. 2002;18:337–8.
21. Thornton K. Libsequence: a C++ class library for evolutionary genetic analysis. *Bioinformatics*. 2003;19:2325–7.
22. Nei M, Tajima F. DNA polymorphism detectable by restriction endonucleases. *Genetics*. 1981;97:145–63.
23. Tang K, Thornton KR, Stoneking M. A new approach for using genome scans to detect recent positive selection in the human genome. *PLoS Biol*. 2007;5:e171.
24. Nei M, Roychoudhury AK. Sampling variance of heterozygosity and genetic distance. *Genetics*. 1974;74:371–80.
25. Geneva AJ, Muirhead CA, Lovato LM, Kingan SB, Garrigan D. A simple statistic to detect recent gene flow from whole-genome data. *Genetics*. 2013.
26. Kelly JK. A test of neutrality based on interlocus associations. *Genetics*. 1997;146:1197–206.
27. Kim Y, Nielsen R. Linkage disequilibrium as a signature of selective sweeps. *Genetics*. 2004;167:1513–24.
28. Wall JD. Recombination and the power of statistical tests of neutrality. *Genet Res*. 1999;74:65–79.
29. Nei M, Li W-H. Mathematical model for studying genetic variation in terms of restriction endonucleases. *Proc Natl Acad Sci U S A*. 1979;76:5269–73.
30. Hudson RR, Slatkin M, Maddison WP. Estimation of levels of gene flow from DNA sequence data. *Genetics*. 1992;132:583–9.
31. Jukes TH, Cantor CR. Evolution of protein molecules. In *Mammalian Protein Metabolism, III* (Munro HN, editor) New York: Academy Press; 1969.
32. Hudson RR, Kreitman M, Aguadé M. A test of neutral molecular evolution based on nucleotide data. *Genetics*. 1987;116:153–9.
33. Saitou N, Nei M. The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Mol Biol Evol*. 1987;4:406–25.
34. Tajima F. Statistical method for testing the neutral mutation hypothesis by DNA polymorphism. *Genetics*. 1989;123:585–95.
35. Tajima F. Estimation of the amount of DNA polymorphism and statistical tests of the neutral mutation hypothesis based on DNA polymorphism. In *Progress in Population Genetics and Human Evolution* (Donnelly P, Tavaré S, editors) New York: Springer; 1997.
36. Fay JC, Wu C-I. Hitchhiking under positive Darwinian selection. *Genetics*. 2000;155:1405–13.
37. Zeng K, Fu Y-X, Shi S, Wu C-I. Statistical tests for detecting positive selection by utilizing high-frequency variants. *Genetics*. 2006;174:1431–9.
38. Watterson GA. On the number of segregating sites in genetical models without recombination. *Theor Popul Biol*. 1975;7:256–76.
39. R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing Vienna, Austria 2013.
40. Pool JE, Corbett-Detig RB, Sugino RP, et al. Population genomics of sub-Saharan *Drosophila melanogaster*: African diversity and non-African admixture. *PLoS Genet*. 2012;8:e1003080.
41. Danecek P, Auton A, Abecasis G, et al. The variant call format and VCFtools. *Bioinformatics*. 2011;27:2156–8.
42. Librado P, Rozas J. DnaSP v5: a software for comprehensive analysis of DNA polymorphism data. *Bioinformatics*. 2009;25:1451–2.