Pre-Processing Whole-Genome Datasets To Improve The Execution Time Of Selective Sweep Detection Tools

S. DE BOER, University of Twente, The Netherlands

The analysis of DNA sequence data of a sampled population allows us to determine what mutations provided an organism or virus with an advantage and caused positive selection. By determining accurately, what part of a DNA sequence is responsible for mutating the organism, we can better understand the world around us, allowing us to find solutions for modern world challenges like finding a more efficient COVID-19 vaccine. The detection of selective sweeps, an indicator for recent positive selection, is done with time consuming software tools. Other research has touched upon filtering techniques to filter out measurement errors in the input data. This research explores whether filtering can be used for input data without errors to speed up execution time. In this paper, 'SNP-processor', a software tool, that tests filtering techniques for the pre-processing of whole-genome datasets is introduced. Three techniques are introduced that speed up execution time up to 1.28 times at the cost of the TPR and accuracy.

Additional Key Words and Phrases: Positive Selection, Selective Sweep, Detection, Pre-Processing, Filtering

1 INTRODUCTION

The world around us is constantly changing. Plants, humans, animals and other organisms evolve and adapt to their environment. The set of instructions for an organism, the DNA, mutates to allow this change. This mechanism can be seen when looking at a population's genetics. In population genetics, detecting a pattern, called a selective sweep, in a DNA sequence allows researchers to discover why an organism thrives in an environment and performs better than similar organisms. Understanding what change in the DNA of a virus caused a beneficial mutation allows for faster and more effective drug treatments [13], but can also be used to analyze: the evolution of plants and fruit [28], the genetics of animals [7], and human evolution [24]. In the case of COVID-19, detecting selective sweeps was used to explain adaptability mechanisms, and to locate regions that had undergone strong positive selection [25].

When an organism has a higher chance of survival due to an advantage, it has a higher chance of becoming the dominant variant over time. The mutation that causes this advantage will fixate itself in the genetics of a population. In some cases, the positive selection of a genome leaves a particular data pattern over time, due to the hitch-hiking effect [23]. The hitch-hiking effect occurs parallel to positive selection. When an advantageous allele, a variant of a given gene, fixates in a population's DNA other alleles will hitchhike. Only after multiple generations or mutations these hitchhikers will mutate out of the DNA. The pattern that is created by this hitchhiking and the disappearance of hitchhikers over time is what we call a selective sweep.

TScIT 37, July 8, 2022, Enschede, The Netherlands

This specific pattern can be used the other way around for detection. By detecting a selective sweep we can detect where an allele mutated and provided the organism with an advantage. Simply, because this allele caused this pattern. For the detection of selective sweeps, multiple statistical tests and software packages are available [4, 8, 27, 6, 16]. DNA data is used as input, and the software outputs a list of locations in the DNA with a likelihood score. The likelihood score represents how likely there is a selective sweep at that position according to the tool. Different tools can be used to detect sweeps based on different characteristics.

With a neutrality test, genome data can be tested [19]. This tests whether the genome data is neutral and contains only normal demographic change, not a selective sweep. Detecting a sweep is done by performing a neutrality test on the data and rejecting the null hypothesis. This genome data has a significant size because every single allele in the DNA string is present in the dataset. A real-world example is the 1000 genome project [3]. With a sample size of 2,184, the first chromosome already contains 2,896,960 single nucleotide polymorphisms (SNPs). An SNP represents a difference in a single nucleotide between generations, another word for mutation as used before. In terms of memory storage, this comes down to 87 GBs of data, according to the experiment used to validate SweeD [20]. On one hand, detecting selective sweeps in large datasets significantly increases processing time, and can even make the software package fail execution. On the other hand, however, distinguishing selective sweeps becomes easier when the sample size increases [12].

1.1 Aim

Although research has been done on statistical tests and tools to detect selective sweeps effectively and accurately, filtering the data before processing is an unexplored field. Existing detection tools can only process a limited number of sequences or are extremely time-consuming. In this paper, we will treat the existing selective sweep software packages as black boxes. This paper will focus on the phase before processing, and seek possible optimizations by re-scaling error free input data, such that fewer data needs to be analyzed while having a limited effect on the accuracy. Where existing research filters the data with the aim of finding more accurate results [22, 18], our research explores filtering aiming to speed up the processing time.

The following research question is raised: How can whole-genome datasets be pre-processed to improve the execution time of selective sweep detection tools?

We can answer this question with the help of the following subquestions:

(1) What pre-processing techniques can be applied to the sequence data?
(2) What is the result in terms of file-size, execution time, and accuracy of these pre-processing techniques?

1.2 Contribution

First, this paper will introduce a software tool, 'SNP-processor', that implements and tests filtering techniques that pre-process whole-genome datasets before selective sweep detection. Secondly, we will test this tool with three different filtering techniques, analyze their effectiveness and draw conclusions accordingly. And finally, we will make recommendations for future work.

2 RELATED WORK

In the field of population genetics, selective sweep detection is seen as an effective way to detect changes in a population, and to pinpoint what changes in its DNA caused this. Summary statistics are used as a neutrality test for whole-genome data to limit the computational complexity and processing time [19]. Software packages like SweepFinder, SweepFinder2, SweeD, OmegaPlus and RAiSD [17, 5, 20, 1, 2] calculate test statistics for the input data, and report on the likelihood of a sweep in the data.

These software tools detect a selective sweep based on one or a selection of the three characteristics of a selective sweep: 1. Around a selective sweep a reduced variation of SNPs can be detected [23], 2. The Site Frequency Spectrum (SFS) shifts towards high- and low- frequencies [4], and 3. A specific Linkage Disequilibrium (LD) pattern between SNPs on the different sides of the favored allele can be detected [15, 21].

When a selective sweep occurs an allele in the DNA sequence causes an advantage for the organism in comparison to other alleles. This means that over time we will find this allele more often in a sample of the population, alleles around the favored allele tend to hitchhike [23]. In other words at the location and around the location of the favored allele a reduction in SNPs can be detected. The second characteristic describes the distribution of allele frequencies in the DNA sample. This distribution shifts towards higher- and lower- frequencies when a selective sweep has occurred. Lastly, as mentioned before when an allele is favored on the left and right side other alleles will hitchhike. The specific pattern that can be observed is that alleles on the same side are strongly linked to each other, meaning if one disappears due to mutation the other allele has likely also disappeared. These alleles are, however, not linked to the other side of the favored allele. This can be explained by the fact that the hitchhiking alleles bring no advantage to the organism and will disappear eventually. Removing one does not affect the other. The observation of this, can also be used as a characteristic to detect a sweep.

The different software packages detect a sweep based on different characteristics. SweepFinder [17], SweepFinder2 [5] and SweeD [20] for example relie on the SFS and implement a Composite Likelihood Ratio as introduced by [16]. OmegaPlus on the other hand detects selective sweeps with the help of a statistic also introduced by [16] that uses the Linkage Disequilibrium instead of the Site Frequency Spectrum. Tools like SweeD and OmegaPlus that test for different characteristics of a sweep cannot substitute one another, because they work differently. They can be used to complement each other and detect more selective sweeps in a whole-genome data-set [20]. Tools that take all three characteristics into account, like RAiSD [2] also exist. When comparing this tool to SweeD and OmegaPlus it outperforms them, and is more effective in detecting selective sweeps, in terms of processing time.

Detecting sweeps is difficult because of a number of reasons. Two of which are normal demographic effects and bottlenecks. Normal demographic effects refer to mutations in a DNA string that provide no advantage. Organisms mutate, but most mutations have no effect on the chance of survival. When detecting sweeps this can be seen as white noise, the detection tools will detect normal demographic effects and label them as sweeps, because of the similarity in characteristics of the two, increasing the False Positive Rate of the tools. Background selection also lowers the local diversity of alleles, thus additional correlation tests have to be performed to determine which of the two it is [19]. The second difficulty is the presence of bottlenecks, a series of mutations that need to occur first before evolution can continue. A bottleneck leaves a very similar pattern as a selective sweep, and occurs in three phases; We first have a large effective population size after which we have a smaller population size and finally a large population size again. the bottleneck is the phase where we have a small population size. When this occurs this produces a very similar pattern to a selective sweep. The paper that introduces RAiSD [2] also mentions migration models, recombination hot-spots and soft selective sweeps as challenges for the sweep detection software. This shows that detecting sweep is more challenging than recognizing the three characteristics mentioned above.

3 METHODOLOGY AND APPROACH

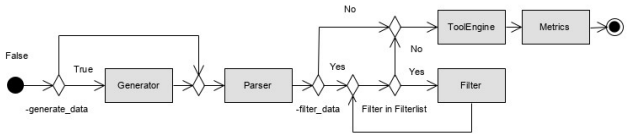

Fig. 1. Structure of objects within SNP-processor

Conducting the research to answer the research question consists out of multiple steps. First, sample data needs to be generated that can be used to test the detection tools. This data needs to be parsed by our software after generation so we can filter it later. After that, new filter code needs to be written and the filter needs to be applied to the simulated datasets. The next step is to export the filtered data and feed that data to detection tools. For this project we will use two different detection tools to analyze the results: 1. OmegaPlus [1] detecting sweep based on the Linkage Disequilibrium [16], 2. SweeD [20] detecting sweeps based on the Site Frequency Spectrum and Composite Likelihood Ratio. By including multiple tools we do not only evaluate tools that rely on one of the three characteristics of a selective sweep. The next step is to parse the output reports to analyze the difference in execution time, the distance between the detected and theoretical position of the sweep and the True Positive Rate of the detection tool. The final step is analyzing the

outcomes and making recommendations for the future. All these steps were integrated in a new software tool which will be labelled as 'SNP-processor' from now on. The tool is structured in such a way that all functionalities can be used separately, by using the right parameters. The data generation, filtering, detection and metrics part are all split. Through this design different filter approaches could be tested, while generating the sample file only once. A thorough explanation on how to use the tool and the tool itself can be found here: (https://gitlab.utwente.nl/s2396041/snp-preprocessing).

3.1 Tool Creation and Execution

SNP-processor was written using the Windows operating system and executed with Windows Subsystem for Linux. The final execution was done on a native Linux machine with the following specifications:

- Operating System: Ubuntu 22.04 LTS
- Processor: Intel i7-4710HQ 2.50 GHZ
- Internal Memory: 8 GB DDR3
- Graphics Card: NVIDIA® GeForce® GTX 850M

3.2 Data Generation and Parsing

For the generation of sample data we used Hudson's ms and mssel [11, 10] software tools, ms is used to generate neutral sets of SNPs and mssel is used to generate a dataset with selection at the center of the simulated SNP data. These software tools generate a binary-like file with a relative position and a 0 or 1 representation of an allele. Where a 1 represents a mutation. The ms format is accepted by OmegaPlus and SweeD. The convenience of the ms software is that the theoretical location of the sweep is located exactly at the center of the DNA sequence. This allows us to easily calculate performance metrics later on. SNP-processor reads the data from an external file and loads it into variables and lists, such that we can edit the data with filters later. The same ms and mssel commands from the evaluation set-up by Alachiotis et al.[2].

3.3 Data Filtering

Although there are endless ways to filter data, for the sake of showing the functionality of our software tool we will only focus on a few in this paper. A total number of three filter groups were tested with in total seven different filter variations in an iterative process. An overview of the different filters can be found in Table 1. The way we filter the ms input file is by removing rows from the input data file. This way we will be decreasing the number of SNP sequences as input.

3.3.1 Hamming Distance Filter. For the first filter we aim at merging DNA sequences, to decrease the number of sequences used as input. In a real world scenario measuring errors can be decreased when using this approach. In case sequences are actually similar, but because of errors appear different merging them into one would reverse this error. Before we can merge sequences we need to cluster them in groups and for that we need an evaluation metric. The evaluation metric decides when sequences are similar. For this we use the Hamming Distance [9], which originally was introduced to detect errors. The Hamming distance where S_p and S_q represent an SNP sequence of length n is calculated as follows:

$$d_H(S_p, S_q) = \sum_{i=1}^{n} I(S_{ip} \neq S_{iq}) \tag{1}$$

In a paper by Wang et al. [26] they use a metric based on the Hamming distance to cluster SNP sequences and perform a population study. We use a metric that calculates the average Hamming Distance between clusters to determine their similarity:

$$d(C_1, C_2) = \frac{1}{n(C_1) \times n(C_2)} \sum_{\substack{S_p \in C_1 \\ S_q \in C_2}} d_H(S_p, S_q) \tag{2}$$

Where,

$$C_k = (S_1, S_2, ..., S_n) \tag{3}$$

,C_k is a cluster that contains n SNP sequences. Each sequence in a population is added to an empty cluster at the start. After the Hamming Distance of all possible unique combinations is calculated, all combinations with a distance below a certain threshold are clustered. SNP-processor stops clustering sequences when every possible combination falls below this threshold. For our research we set the Hamming Distance threshold to be 5% of the length of one SNP sequence. This means that if 95% of two sequences or clusters match they will be clustered. After all possible combinations fall below the threshold all remaining clusters will be merged into a single sequence. We tested three different ways of merging clusters to see in what way they affect the results. 1. Frequency merge: if more than half of all sequences contain a mutation in a column the resulting sequence will too, 2. AND merge: if all sequences contain a mutation in a column the resulting sequence will too, and 3. OR merge: if one of the sequences contains a mutation in a column the resulting sequence will too.

3.3.2 Allele Mutation Frequency. With this filter we aim to remove sequences from a population. For every population in our dataset we will look at every sequence in the population and calculate the number of alleles that have mutated. In ms / mssel format this means there is a $'1'$ at that position, this results in the following formula:

$$f_{mutation}(S_n) = \sum_{n \in S_n} I(n) \tag{4}$$

, where S_n is a sequence of size n. After calculating the Allele Mutation Frequency for each sequence the sequences will be sorted from highest to lowest frequency. We, once again, tested multiple variations of this filter: 1. The sequences with the lowest frequency are removed from the population, and 2. The sequences with the highest frequency are removed from the population. For this research we removed 5% from the bottom or top of the sorted list of sequences. Meaning the resulting dataset contains the most or least mutated sequences of the original dataset and is 95% of the original size.

3.4 A Combination of Filters

The final experiment we carried out was chaining different filters. The two best performing variations of the Hamming Distance Filter and Allele Mutation Frequency Filter were used. There are two possible variations: 1. Allele Frequency -> Hamming Distance, and

Table 1. Overview of filters researched

Filter Group	Variation	Label
(1) Hamming Distance	Frequency Merge	HF
	AND Merge	HA
	OR Merge	HO
(2) Allele Mutation Frequency	Low Frequency	ML
	High Frequency	MH
(3) Filter Chaining	(1) Hamming, (2) Mutation	C-HM
	(2) Mutation, (1) Hamming	C-MH

2. Hamming Distance -> Allele Frequency. The order matters, since the Hamming Distance filter creates new sequences by merging others.

3.5 Exporting and Processing

The filtered data is loaded from memory and saved as an external file with the exact same format as Hudson's ms. To test if the file exported by Hudson's ms has the exact same format as the file exported by SNP-processor the Linux 'diff' command was used. This is a built-in method that compares two files and reports what the exact differences are between the two. Running this with a not parsed, unfiltered ms file and a parsed, but unfiltered ms file resulted in no differences. After the file is exported our code runs one of the three detection tools. In order to run the detection tools the right parameters have to be provided by SNP-processor. SweeD requires the following parameters: -name (not relevant), -input (the exported file), -length (the length of the analyzed DNA sequence), and -grid (number of positions the likelihood will be calculated). For the length of our input file we used 100,000 for all code executions. The grid size was set to 1000. OmegaPlus uses the same parameters, but also requires: -minwin, and -maxwin (minimum and maximum window for calculating the linkage disequilibrium between SNP values). In this paper we set the -minwin to 1,000 and the -maxwin to 5,000. The output files generated by the tools are moved into the Output folder after the tool completes running.

3.6 Output Analysis

The filtered and unfiltered datasets are processed by the detection tools and as mentioned above a report is generated. SNP-processor parses this report and calculates performance metrics according to this report. For this we use a similar approach as was used by Alachiotis et al. [2]. Hudson's ms is used to generate neutral sets of SNPs that do not contain selection and Hudson's mssel is used to generate datasets that do contain a sweep [11, 10]. In this paper we use an edited version of ms and mssel, such that it prints out the relative position of SNPs more accurately as described in the SweeD 3.0 manual [20]. We calculate two metrics for our filtered and unfiltered datasets: 1. The distance from the observed position to the actual position, and 2. The True Positive Rate of the detected sweeps. For the first metric, distance, we read the report of the detection tool. This report lists for each population (we used 100 populations) all observed positions and their likelihood score. SNP-processor takes the position for each population with the best likelihood score. We end up with one score for each population (100 likelihood scores). These scores are saved in a new text file, 'summary_report.txt', with their position. When we calculate the distance metric we use the following formulas:

$$d(P_k) = |theoreticalPosition - position(x)| \tag{5}$$

$$x = max(likelihood(P_{11}), ..., likelihood(P_{ij})) \tag{6}$$

, where $P_k = (S_1, S_2..., S_n)$ represents a population consisting of n sequences S, and where P_{ij} is the jth location of population i for which the likelihood score was calculated. The final score of a dataset $D_k = (P_1, P_2, ..., P_n)$ is calculated as follows:

$$d(D_k) = \frac{1}{n(D_k)} \sum_{P_z \in D_k} d(P_z) \tag{7}$$

For this paper the length was set to 100,000 thus the $theoreticalPosition$ is at position 50,000. The second performance metric, True Positive Rate, is calculated with the help of ms and mssel commands from the paper by Alachiotis et al. [2] and the following formula:

$$TPR(dataset) = \frac{1}{n} \sum_{i=1}^{n} I(likelihood(i) > FPR_{threshold}) \tag{8}$$

, where $dataset$ is a list of positions with the maximum likelihood score for each population. First a BASE dataset is generated with Hudon's ms. This is our neutral set. We generate 100 populations containing 20 sequences, and feed this into the detection tools. From the output report we generate, as mentioned before, the summary file 'summary_report.txt'. Which contains the highest likelihood score for each population. For this paper, we assumed a FPR (False Positive Rate) of 5% for the detection tools. This means, that we assume that 5% of the sweeps detected by one of the tools is wrongly classified. From our summary report we remove the top 5% with the highest likelihood scores. The now highest likelihood score is our $FPR_{threshold}$ which we will use later on. After that, a dataset containing a sweep is generated with Hudson's mssel. Mssel uses a trajectory file, which was also used from the RAiSD paper. Now that we have a data file containing a sweep we will run the detection tools for the unfiltered and filtered datasets. The summary report of these sets is parsed, leaving us once again with the highest likelihood score and position of all 100 populations. To calculate the True Positive Rate we compare each likelihood score with the threshold from the BASE file. If the score is higher it is a True Positive detection. We count all the scores that are higher than the threshold leaving us with the True Positive Rate of the sweep detection. The execution time, number of populations, number of sequences removed, the

distance (eq. 7) and the TPR (eq. 8) are all saved in a metric report file.

3.7 Test Data Selection

For this paper, ms and mssel commands from the RAiSD paper [2] were used to generate 3 datasets. The datasets picked are 1, 20 and 60. These three datasets all contain hard sweeps and have a bottleneck as confounding factor. Set 60 was included because it has a low True Positive Rate and set 1 and 20 were included as sets that have an average TPR.

4 RESULTS

The results of the different filtering approaches are displayed in table 2. The tabel shows for each dataset (1, 20 and 60): 1. The detection tool used (SweeD or OmegaPlus), 2. the filter used (the 'label' column corresponds to table 1), 3. the processing time of this run in seconds, 4. the distance to the theoretical sweep position (calculated according to eq 7), 5. the True Positive Rate (calculated according to eq 8), 6. the number of filtered sequences from the entire dataset, and 7. a relative column for Time, Distance, TPR and filtered. We validated the software tool 'SNP-processor' by using it to execute all steps required for testing the pre-processing of detection tool input data. The results are graphed in figure 2, to compare the results of filter variations with the unfiltered dataset.

4.1 Hamming Distance Filter

The results in table 2 show that the variations of the Hamming Distance Filter do not affect the results of OmegaPlus a lot. The maximum change in TPR was -0.07% for dataset 60. The execution time for OmegaPlus did only improve 1.01 times as a maximum. For SweeD this filter performed better by speeding up execution time by 1.02 - 1.17 times, while increasing the distance only slightly (3.44% at most). The TPR rate did decrease between 0 and 8.64%. The variation with AND-merge shown in table 1, performed best out of the three variations and was thus used for the chained filter.

4.2 Allele Mutation Frequency

This filter did improve execution time for OmegaPlus for all three tested datasets. Removing sequences with a high frequency of mutations improved the distance for two out of three datasets, and improved the TPR for one. Removing sequences with a low frequency of mutations appears to have the exact opposite effect, increasing the distance between 1 - 10%. This filter interestingly enough affects the execution time of SweeD negatively, but increases its accuracy in distance in almost all cases. Since, our goal is to improve the execution time we will use the filter that filters out high frequencies for our chained filter.

4.3 Chaining Filters

Chaining the two filters discussed prior had the most impact on execution time out of the three categories of filters. At the same time, it influenced the TPR and distance the most. Especially for SweeD and dataset 20, this can be seen in fig 2l. It appears the filter removed too many sequences resulting in a significant loss of accuracy.

4.4 Overall Comparison

Overall it is clear that the filters applied to the datasets at the moment are not that effective. The execution time is sped up between 0.99 - 1.25 times. Meanwhile, the distance and TPR change quite a lot. Introducing a trade off, where the researcher needs to decide what is more important: accuracy, or execution time. It becomes clear that filtering based on mutation frequency affects OmegaPlus [1], a LD based detection tool, more than SweeD [20], a SFS based detection tool in terms of execution time. And that filtering based on hamming distance affects SweeD more than OmegaPlus.

5 DISCUSSION

In this paper, many choices had to be made for the methodology. The main choices being the data generation commands, the FPR, the population size and filter thresholds. The data generation commands were used from the RAiSD [2] paper. This paper aimed to prove that RAiSD was better than comparable tools. The datasets used for this might not have been effective for our research, because each population only contains 20 sequences. After filtering a population we are left with less than 20 sequences per population. The detection of sweeps is easier for larger populations [12]. Thus, picking a different data generation command or test set with moderate size populations (>50), might have far better results, as removing the same percentage of sequences would still leave us with a moderate size sample. We expect that filtering datasets of moderate size samples improves execution time more in relation to the decrease in TPR and distance. Execution time does not grow linearly for larger datasets, the relative saved time likely grows with it. To save time, as the experiments had to be performed in 10 weeks time, we generated only 100 instead of 1000 populations. Affecting the accuracy of our results. The FPR is assumed at 5% for this paper, which directly affects the observed TPR. Assuming a higher FPR would increase the TPR and limit the effect of filtering on the TPR. The relative and absolute filter thresholds used for this paper (see Methodology), are not supported by any research. Tuning these thresholds likely improves the effectiveness of the filters and their effect on detection tool accuracy. Dataset 1, 20 and 60 are all hard sweeps with a bottleneck as confounding factor. Filtering soft sweeps or datasets with a different confounding factor would likely affect the outcome of the filtering technique. This paper used altered versions of Hudson's ms, mssel and SweeD [11, 10, 20]. Ms and mssel were changed such that the relative position of the SNPs was printed out more accurately. SweeD contained a bug where it would not work for filtered binary like files. The source code had to be altered to prevent wrong memory allocation and fail running. Lastly, the time it took to filter the data was not taken into consideration. For the hamming distance filters this took a maximum of 8.0 seconds, but for larger datasets and populations this could increase significantly due to the use of combinations in the clustering process.

6 CONCLUSIONS

In this paper, we successfully introduced a software tool, SNP-processor, that tests filtering techniques for the pre-processing of whole-genome datasets. This tool integrates sample data generation,

Table 2. Overview of filter results. Underlined are the best execution time, distance and TPR values for each combination of dataset. The label column provides info on the filter and filter variation used, see Table 1. Time is reported in speedups. Distance and TPR as a percentage change from the unfiltered dataset. The number of rows filtered is out of the entire dataset, containing 2,000 rows

Set	Tool	Label	Time	Distance	TPR	Filtered	x Time	% Distance	% TPR	% Filtered
1	OmegaPlus	-	00:48.2	*1235.9085*	*1.00*	0	1.00	*0.00%*	*0.00%*	0.00%
		ML	00:46.3	1380.1005	*1.00*	100	1.04	11.67%	*0.00%*	5.00%
		MH	00:46.1	1259.8077	*1.00*	100	1.04	1.93%	*0.00%*	5.00%
		HA	00:48.2	1243.9171	*1.00*	19	1.00	0.65%	*0.00%*	0.95%
		HF	00:48.0	1243.9171	*1.00*	19	1.00	0.65%	*0.00%*	0.95%
		HO	00:47.7	1243.9171	*1.00*	19	1.01	0.65%	*0.00%*	0.95%
		C-HM	00:46.1	1291.4548	*1.00*	*119*	1.05	4.49%	*0.00%*	*5.95%*
		C-MH	*00:46.0*	1291.4548	*1.00*	*119*	*1.05*	4.49%	*0.00%*	*5.95%*
	SweeD	-	07:17.0	3194.110604	*0.97*	0	1.00	0.00%	*0.00%*	0.00%
		ML	07:16.2	3623.019722	*0.97*	100	1.00	13.43%	*0.00%*	5.00%
		MH	07:20.4	*3114.35618*	0.96	100	0.99	*-2.50%*	-1.03%	5.00%
		HA	*07:03.4*	3304.18837	0.95	19	*1.03*	3.45%	-2.06%	0.95%
		HF	07:05.7	3234.128113	0.95	19	1.03	1.25%	-2.06%	0.95%
		HO	07:03.9	3234.128113	0.95	19	1.03	1.25%	-2.06%	0.95%
		C-HM	*07:03.4*	3167.496	0.95	*119*	*1.03*	-0.83%	-2.06%	*5.95%*
		C-MH	07:05.5	3167.496	0.95	*119*	1.03	-0.83%	-2.06%	*5.95%*
20	OmegaPlus	-	00:28.2	1759.9884	0.99	0	1.00	0.00%	0.00%	0.00%
		ML	00:27.4	1940.1788	0.99	100	1.03	10.24%	0.00%	5.00%
		MH	*00:27.4*	*1569.9925*	*1.00*	100	*1.03*	*-10.80%*	*1.01%*	5.00%
		HA	00:28.4	1819.0816	0.99	13	0.99	3.36%	0.00%	0.65%
		HF	00:28.2	1819.0816	0.99	13	1.00	3.36%	0.00%	0.65%
		HO	00:28.2	1819.0816	0.99	13	1.00	3.36%	0.00%	0.65%
		C-HM	00:27.3	1570.0327	*1.00*	*113*	1.03	-10.79%	*1.01%*	*5.65%*
		C-MH	*00:27.3*	*1569.9925*	*1.00*	112	*1.03*	*-10.80%*	*1.01%*	5.60%
	SweeD	-	06:35.4	2849.788046	0.81	0	1.00	0.00%	0.00%	0.00%
		ML	06:35.6	3125.61169	*0.85*	100	1.00	9.68%	*4.94%*	5.00%
		MH	06:36.9	*2841.340105*	0.83	100	1.00	*-0.30%*	2.47%	5.00%
		HA	06:29.3	2947.713283	0.74	13	1.02	3.44%	-8.64%	0.65%
		HF	06:28.9	2947.713283	0.74	13	1.02	3.44%	-8.64%	0.65%
		HO	06:28.8	2947.713283	0.74	13	*1.02*	3.44%	-8.64%	0.65%
		C-HM	05:08.9	3193.884	0.28	*113*	1.28	12.07%	-65.43%	*5.65%*
		C-MH	*05:08.8*	3192.843	0.29	112	*1.28*	12.04%	-64.20%	5.60%
60	OmegaPlus	-	00:07.7	15898.0197	*0.07*	0	1.00	0.00%	*0.00%*	0.00%
		ML	00:07.6	16112.343	0.00	100	1.01	1.35%	-100.00%	5.00%
		MH	*00:07.6*	15786.1492	0.00	100	*1.02*	-0.70%	-100.00%	5.00%
		HA	00:07.7	*14271.8764*	0.00	159	1.00	*-10.23%*	-100.00%	7.95%
		HF	00:07.7	15981.049	0.00	159	1.01	0.52%	-100.00%	7.95%
		HO	00:07.7	16689.927	0.00	159	1.00	4.98%	-100.00%	7.95%
		C-HM	00:07.5	14280.5412	0	*258*	1.04	-10.17%	-100.00%	*12.90%*
		C-MH	00:07.5	14398.7967	0	255	1.04	-9.43%	-100.00%	12.75%
	SweeD	-	03:26.4	27121.06642	0.00	0	1.00	0.00%	0.00	0.00%
		ML	03:33.7	28156.57746	0.01	100	0.97	3.82%	0.01	5.00%
		MH	03:25.3	27430.2857	*0.02*	100	1.01	1.14%	*0.02*	5.00%
		HA	02:59.9	26885.7069	0.00	159	1.15	-0.87%	0.00	7.95%
		HF	02:55.7	26875.2479	0.00	159	1.17	*-0.91%*	0.00	7.95%
		HO	02:56.3	26893.26233	0.00	159	1.17	-0.84%	0.00	7.95%
		C-HM	*02:45.0*	*26408.96844*	0.01	*258*	*1.25*	*-2.63%*	+0.01	*12.90%*
		C-MH	02:45.1	26895.50368	0.01	255	1.25	-0.83%	+0.01	12.75%

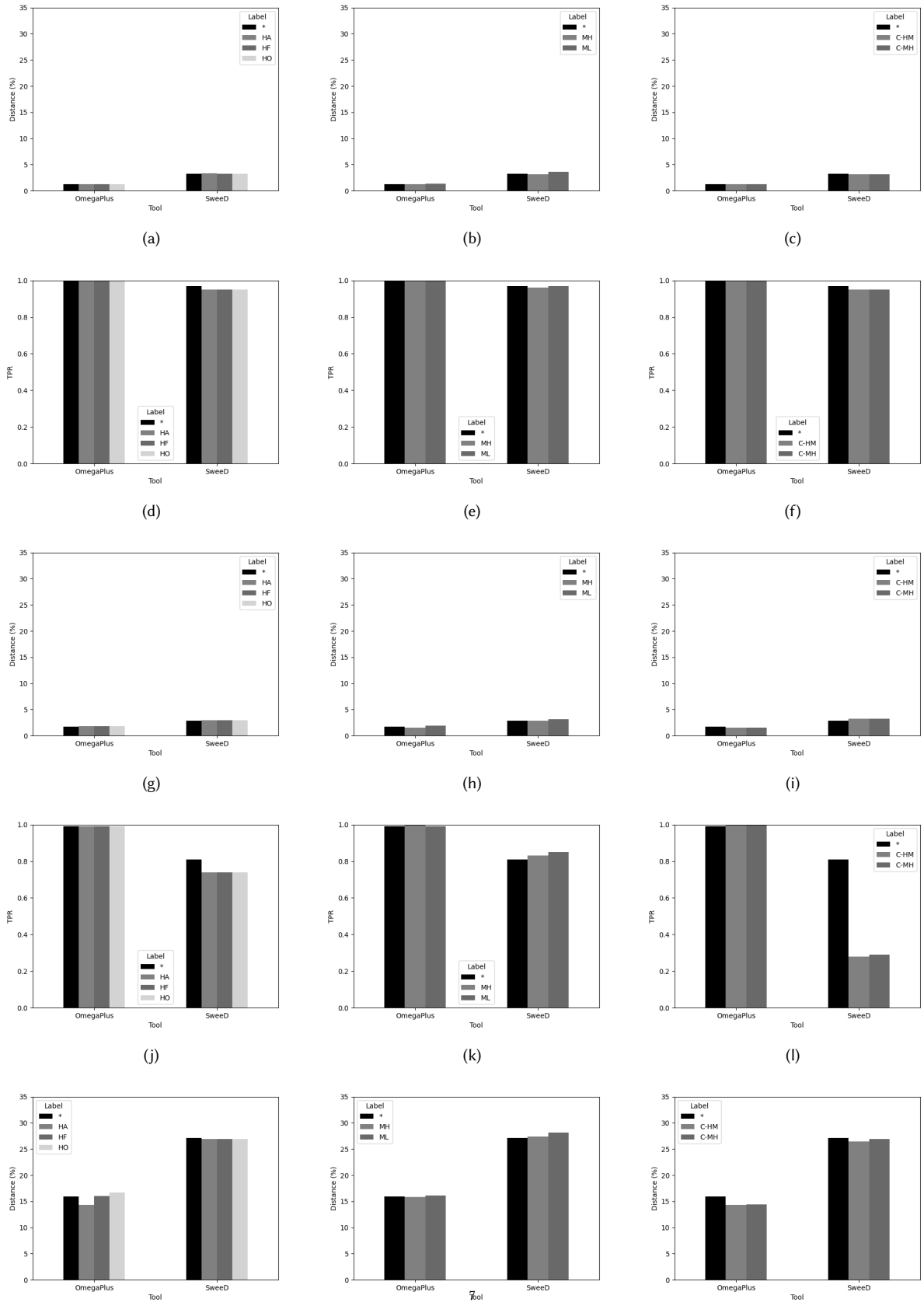

Fig. 2. Comparison of Distance and TPR, for datasets 1 (2a,2b,2c,2d,2e,2f), 20 (2g,2h,2i,2j,2k,2l) and 60 (2m,2n,2o). The TPR of dataset 60 is not shown, because of the small differences in TPR, providing no visual difference

sample data parsing, data filtering, exporting of filtered data, detection tool execution and result evaluation all in one package. We introduced three possible filtering techniques and explained their underlying theory. And we presented the results of filtering whole-genome datasets with these filters and analyzed their effectiveness. We found that for a small population size of 20 these filters:

- Speed up the execution time up to 1.28 times.
- Increase the distance between observed and actual position up to 13.43%.
- Decrease the True Positive Rate up to 52 percentage points.

Filtering whole-genome datasets based on hamming distance, allele frequency, or both does not reliably decrease execution time while retaining TPR and distance at the same level. The benefits of a faster execution time need to be weighed carefully against the downsides of decreased distance accuracy and TPR. Lastly, we found for the filtering techniques in this paper that:

- A filtering technique based on Hamming Distance has more effect on a detection tool relying on the SFS characteristic of a sweep.
- A filtering technique based on Allele Mutation Frequency has more effect on a detection tool relying on the LD characteristic of a sweep.

7 FUTURE WORK

This paper made a first attempt to filter whole-genome datasets by introducing 'SNP-processor'. As this was exploratory research into filtering error free SNP data many things have not been explored yet. Possible future work can be categorized into three different categories: 1. The extension of SNP-processor, 2. Using different sample data, 3. Finding new filtering techniques. Extending SNP-processor can be done by adding new detection tools. DiploSHIC [14] would be a good addition, as it is a machine learning approach to selective sweep detection instead of an SFS or LD approach, like SweeD [20] and OmegaPlus [1]. Including diploSHIC would ensure SNP-processor tests techniques for more use cases. The effectiveness of the proposed filtering techniques should be tested for a sample dataset that contains more populations (e.g. 1000) to improve accuracy and populations that consist of many sequences (50+) to limit the effect of filtering on the results of the detection tools [12]. Datasets with different confounding factors than a bottleneck could be explored, together with datasets that contain soft sweeps. Filtering possibly affects the results differently in those cases. Lastly, in future work different filtering techniques should be explored with the help of SNP-processor.

REFERENCES

[1] N. Alachiotis, A. Stamatakis, and P. Pavlidis. "OmegaPlus: A scalable tool for rapid detection of selective sweeps in whole-genome datasets". In: *Bioinformatics* 28.17 (2012). ISSN: 13674803. DOI: 10.1093/bioinformatics/bts419.

[2] Nikolaos Alachiotis and Pavlos Pavlidis. "RAiSD detects positive selection based on multiple signatures of a selective sweep and SNP vectors". In: *Communications Biology* 1.1 (2018). ISSN: 23993642. DOI: 10.1038/s42003-018-0085-8.

[3] Adam Auton et al. *A global reference for human genetic variation*. 2015. DOI: 10.1038/nature15393.

[4] J. M. Braverman et al. "The hitchhiking effect on the site frequency spectrum of DNA polymorphisms". In: *Genetics* 140.2 (1995). ISSN: 00166731. DOI: 10.1093/genetics/140.2.783.

[5] Michael Degiorgio et al. "SweepFinder2: Increased sensitivity, robustness and flexibility". In: *Bioinformatics* 32.12 (2016). ISSN: 14602059. DOI: 10.1093/bioinformatics/btw051.

[6] F. Depaulis and M. Veuille. *Neutrality tests based on the distribution of haplotypes under an infinite-site model [3]*. 1998. DOI: 10.1093/oxfordjournals.molbev.a025905.

[7] Clio Der Sarkissian et al. "Evolutionary genomics and conservation of the endangered Przewalski's horse". In: *Current Biology* 25.19 (2015). ISSN: 18790445. DOI: 10.1016/j.cub.2015.08.032.

[8] Justin C. Fay and Chung I. Wu. "Hitchhiking under positive Darwinian selection". In: *Genetics* 155.3 (2000). ISSN: 00166731. DOI: 10.1093/genetics/155.3.1405.

[9] R. W. Hamming. "Error Detecting and Error Correcting Codes". In: *Bell System Technical Journal* 29.2 (1950). ISSN: 15387305. DOI: 10.1002/j.1538-7305.1950.tb00463.x.

[10] R. R. Hudson. "Generating samples under a Wright-Fisher neutral model of genetic variation". In: *Bioinformatics* 18.2 (Feb. 2002), pp. 337–338. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/18.2.337.

[11] Richard R. Hudson. "Estimating the recombination parameter of a finite population model without selection". In: *Genetical Research* 50.3 (Dec. 1987), pp. 245–250. ISSN: 0016-6723. DOI: 10.1017/S0016672300023776.

[12] Jeffrey D. Jensen et al. "On the utility of linkage disequilibrium as a statistic for identifying targets of positive selection in nonequilibrium populations". In: *Genetics* 176.4 (2007). ISSN: 00166731. DOI: 10.1534/genetics.106.069450.

[13] Lin Kang et al. "A selective sweep in the Spike gene has driven SARS-CoV-2 human adaptation". In: *Cell* 184.17 (2021). ISSN: 10974172. DOI: 10.1016/j.cell.2021.07.007.

[14] Andrew D. Kern and Daniel R. Schrider. "DiploS/HIC: An updated approach to classifying selective sweeps". In: *G3: Genes, Genomes, Genetics* 8.6 (2018). ISSN: 21601836. DOI: 10.1534/g3.118.200262.

[15] Yuseob Kim and Rasmus Neilsen. "Linkage disequilibrium as a signature of selective sweeps". In: *Genetics* 167.3 (2004). ISSN: 00166731. DOI: 10.1534/genetics.103.025387.

[16] Yuseob Kim and Wolfgang Stephan. "Detecting a local signature of genetic hitchhiking along a recombining chromosome". In: *Genetics* 160.2 (2002). ISSN: 00166731. DOI: 10.1093/genetics/160.2.765.

[17] Rasmus Nielsen et al. "Genomic scans for selective sweeps using SNP data". In: *Genome Research* 15.11 (2005). ISSN: 10889051. DOI: 10.1101/gr.4252305.

[18] Shannon J. O'Leary et al. "These aren't the loci you're looking for: Principles of effective SNP filtering for molecular ecologists". In: *Molecular Ecology* 27.16 (2018). ISSN: 1365294X. DOI: 10.1111/mec.14792.

[19] Pavlos Pavlidis and Nikolaos Alachiotis. "A survey of methods and tools to detect recent and strong positive selection". In: *Journal of Biological Research (Greece)* 24.1 (2017). ISSN: 22415793. DOI: 10.1186/s40709-017-0064-0.

[20] Pavlos Pavlidis et al. "SweeD: Likelihood-based detection of selective sweeps in thousands of genomes". In: *Molecular Biology and Evolution* 30.9 (2013). ISSN: 07374038. DOI: 10.1093/molbev/mst112.

[21] P. Pfaffelhuber, A. Lehnert, and W. Stephan. "Linkage disequilibrium under genetic hitchhiking in finite populations". In: *Genetics* 179.1 (2008). ISSN: 00166731. DOI: 10.1534/genetics.107.081497.

[22] Nab R. Roshyara et al. "Impact of pre-imputation SNP-filtering on genotype imputation results". In: *BMC Genetics* 15.1 (2014). ISSN: 14712156. DOI: 10.1186/s12863-014-0088-5.

[23] John Maynard Smith and John Haigh. "The hitch-hiking effect of a favourable gene". In: *Genetical Research* 23.1 (1974). ISSN: 14695073. DOI: 10.1017/S0016672300014634.

[24] Lauren Alpert Sugden et al. "Localization of adaptive variants in human genomes using averaged one-dependence estimation". In: *Nature Communications* 9.1 (2018). ISSN: 20411723. DOI: 10.1038/s41467-018-03100-7.

[25] Maria Vasilarou et al. "Population genomics insights into the first wave of covid-19". In: *Life* 11.2 (2021). ISSN: 20751729. DOI: 10.3390/life11020129.

[26] Charlotte Wang et al. "Using hamming distance as information for SNP-sets clustering and testing in disease association studies". In: *PLoS ONE* 10.8 (2015). ISSN: 19326203. DOI: 10.1371/journal.pone.0135918.

[27] Kai Zeng et al. "Statistical tests for detecting positive selection by utilizing high-frequency variants". In: *Genetics* 174.3 (2006). ISSN: 00166731. DOI: 10.1534/genetics.106.061432.

[28] Yongfeng Zhou et al. "Evolutionary genomics of grape (Vitis vinifera ssp. vinifera) domestication". In: *Proceedings of the National Academy of Sciences of the United States of America* 114.44 (2017). ISSN: 10916490. DOI: 10.1073/pnas.1709257114.