UPPSALA
UNIVERSITET

UPTEC X 16 001
Examensarbete 30 hp
Maj 2016

Exploring linkage disequilibrium to date admixture using ancient DNA

Anders Sjölander

UPPSALA UNIVERSITET

Bioinformatics Engineering Program

Uppsala University School of Engineering

UPTEC X 16 001	Date of issue 2016-05

Author

Anders Sjölander

Title (English)

Exploring linkage disequilibrium to date admixture using ancient DNA

Title (Swedish)

Abstract

Admixture studies based on contemporary DNA are well studied and results from these have shown to correlate well with other evidence. The study of using ancient DNA for determining admixture times is a less explored alternative for determining admixture. Admixture-induced Linkage Disequilibrium for Evolutionary Relationships (ALDER) is an established tool for determining admixture time from contemporary DNA. This project investigates the use of ALDER on simulated ancient DNA to determine if it can be used in actual ancient DNA studies. The results from this project suggest that ALDER can be used in actual studies, with good accuracy and reliability.

Keywords

Ancient DNA, Migration, Admixture, ALDER, Simulations

Supervisors

Torsten Günther

Uppsala University

Scientific reviewer

Martin Lascoux

Uppsala University

Project name	Sponsors
Language **English**	Security
ISSN 1401-2138	Classification
Supplementary bibliographical information	Pages **56**

Biology Education Centre Biomedical Center Husargatan 3 Uppsala

Box 592 S-75124 Uppsala Tel +46 (0)18 4710000 Fax +46 (0)18 471 4687

Exploring linkage disequilibrium to date admixture using ancient DNA

Anders Sjölander

Populärvetenskaplig sammanfattning

Populations- och migrationsstudier undersöker och kartlägger historiska folkströmningar för att kunna bestämma folkgruppers vandringar genom historien. En aspekt av dessa migrationsstudier är undersökningar av folkblandningar, där befolkningsgrupper som tidigare varit åtskilda i någon grad möts och blandas upp. Med tillgång till de separerade folkgruppernas DNA samt DNA:t hos den blandade gruppen är det möjligt att avgöra hur länge sedan folkblandningen skedde.

Dessa uppblandningsstudier utgår ofta utifrån DNA från levande individer, men det finns ytterligare information att vinna om dessa studier kunde baseras på forntida DNA som hittas i arkeologiska utgrävningar. Eftersom DNA bryts ned med tiden så är forntida DNA av mycket lägre kvalité, vilket gör det svårare att analysera och dra slutsatser.

Det finns en rad olika analysprogram för nutida DNA för att avgöra historiska befolkningsblandningar men det är osäkert huruvida dessa kan användas för att bestämma folkblandningar baserat på forntida, lågkvalitativt DNA. Med hjälp av simuleringar av forntida DNA går det att undersöka tillförlitligheten hos sådana verktyg, då analysresultaten på dessa simuleringar kan jämföras med de faktiska parametrarna som ligger bakom datan. Genom att studera träffsäkerheten hos ett sådant verktyg på simulerat forntida DNA, går det att få en indikation om det faktiskt går att använda i riktiga populationsstudier baserade på forntida DNA.

Resultaten visar att verktyget ALDER som undersökts i detta projekt mycket väl kan ge indikationer vid vilken tid folkblandning skett med hjälp av forntida DNA, med god träffsäkerhet och tillförlitlighet.

Examensarbete 30 hp

Civilingenjörsprogrammet Bioinformatik

Uppsala Universitet, maj 2016

Table of contents

Abbreviations

ALDER	Admixture-induced Linkage Disequilibrium for Evolutionary Relationships, a tool for determining admixture based on Linkage Disequllibrium
LD	Linkage Disequillibtrium
SNP	Single Nucleotide Polymorphism

1. Introduction

1.1 Background

Ever since *Homo sapiens* emerged in Africa, migration has been a constant force shaping the genetic make up of the modern humans of today. Migrating groups of humans reaching different environments have created subdivisions among humans in Africa, some remaining even to this day. Evidence suggest that these divisions occurred around 150-200 000 years ago, preceeding the migratory out of Africa-event circa 100 000 years ago. (Brown, 2006) As *Homo sapiens* emerged out of Africa, further migration followed into Europe, as well as Asia, Oceania and America. While these larger migratory events are the cause for human presence across nearly all continents, there have also been smaller migratory events causing admixture, recombining and intermixing the DNA of various groups. Intermixing of DNA between subgroups has generated a genetic landscape within humans that can be used to infer the migratory history, both recent and further back in time. (Brown, 2006)
Using contemporary DNA along with archaeological findings has been the primary method to trace migratory patterns (Forster, 2004), and new findings using these methods are still discovered (Busby et al. 2015). However, with the rise of ancient DNA retrieval and analysis methods, alternate pathways of discovering and resolving migration events have appeared.

Archaeological findings give some insight to how the population and cultural dynamics might have looked at the time. (Zeder et al. 2006) However, when two populations were too closely related, it may be too difficult telling these populations apart by archaeological findings alone. By using genetic data, it is possible to tell whether culture and language were spreading simply because they were good ideas, or if the success of these ideas allowed the associated cultures to expand. The most convincing evidence is of course when the findings from both of these disciplines overlap. (Zeder et al. 2006)

Recently, studies of ancient DNA from neolithic hunter-gatherers and farmers in Europe have suggested that these groups have laid the genetic foundation of current day Europeans. (Skoglund et al. 2012) Different ethnic groups in Europe have been shown to have varying degrees of similarity with different populations from the neolithic. These results were found by comparing the DNA of these ancient hunter-gatherer and farmer individuals to current ethnic groups around Europe and finding which groups they shared the most similarities with. (Skoglund et al. 2012)

The history of admixture between Scandinavian farmers and hunter-gatherers has been an area of investigation by the Jakobsson research group, with evidence showing that the hunter-gatherer groups were small and to some extent became incorporated into farming populations, rather than disappearing entirely. (Skoglund et al. 2014)

Similar investigations are performed by other groups as well and any new methods of determining admixture would be of great interest. By looking into less studied methods of using ancient DNA for this purpose, new applications might be found that could be very helpful into admixture studies. Recently, the idea of using the metric Linkage Disequillibrium (LD) has risen to be a popular tool of analyzing admixture when looking at high quality data. (Loh et al. 2013) Using it on data of lower quality, which is the case of ancient DNA, is less explored. Being able to use such a tool would be of interest to the Jakobsson group, as well as other researchers of the field, and this project aims to investigate the possibilities of using it.

1.2 Methods of determining migration and admixture

Using archaeological data is a very valuable method for determining population migrations and

admixture. (Groucutt et al. 2015) Both through looking at anatomical similarities and differences on bones as well as studying the cultural craftsmanship that the various populations display. With the recent rise of sequencing technology, improvement of the recovery of ancient DNA and the finding of new tools to analyze the data, computer methods allow further investigations both to confirm and in some cases give closer insight into the population dynamics at the time. (Groucutt et al. 2015)

1.3 DNA

Deoxyribonucleic acid (DNA) is the genetic code inherited to offspring from their parents. The DNA contains the instructions required for the development and functions of all known organisms. In human reproduction, DNA is inherited from parents to offspring, but is first modified through the process of recombination. During recombination, homologous chromosomes cross over the genetic information by exchanging large blocks of nucleotides. This process creates a new chromosome for the germ cell containing information from both the grandparents. Therefore, there are long strands of complete identity between one grandparent and their offspring until the point where recombination has occurred, where a sequence completely identical to the other grandparent picks up.

DNA starts deteriorating as soon as an individual dies, making DNA extraction of old samples difficult, in turn leading to several issues when performing analyses. The decay rate varies depending on the conditions, with the best preservation occuring in dry and frozen climate. The are varying types of damage occurring to DNA as it ages, including both post-mortem mutations, and complete breakage of nucleotide bindings. When retrieving DNA from samples that have gone through decay, the coverage and accuracy of all sequenced DNA will be low. Only fractions of the original DNA will be found, and even though all human individuals are diploid, only one of the nucleotides might be recovered, if at all. And due to mutations occurring as the DNA decays, even if a nucleotide is recovered, it is not necessarily correct. However, as the mutation process is not entirely random, it is possible to correct for some of these mutations. In addition, it is very difficult to know the phasing of DNA when it has deteriorated. Phasing means knowing the exact sequence as it is found in each DNA strand. When sequencing DNA these two strands may become mixed, and needs to be corrected in order to be phased. When using ancient DNA this process is very difficult if not impossible to perform, as only single nucleotides will be known for most positions, and only a fraction of all positions in total will be known. Knowing the phase of the DNA provides a lot of extra information, as it tells the specific linkage between each allele in the sequence. (Handt et al. 1994)

While recombination is an exceedingly important factor for genetic diversity, it also has consequences when studying population history. As recombination keeps occurring through generations, the DNA strand of an ancestor is being broken up into smaller and smaller chunks. This means that over time, the association of one allele in close proximity with another will remain, while the association of two alleles at a long distance will disappear. This is the basis for linkage studies.

1.4 Linkage Disequillibrium analysis of admixture

LD is a metric used when dealing with this association between alleles at a population level. (Slatkin 2008) In short, it is a measure of how well one allele or nucleotide is able to predict another. LD forms the basis of association mapping, a method of linking together phenotypic traits with genetic variants without needing to sequence an entire individual. (Slatkin 2008)

Recently, using LD has also been shown to be a useful tool when trying to estimate admixture. By

measuring how LD decays with distance in the source populations compared to the admixed population, Patterson et al. (2012) have successfully determined dates of admixture in a variety of populations. Sankararaman et al. (2012) applied a similar method to determine the date of admixture between Neanderthals and modern humans that have caused increased amounts of Neanderthal DNA in non-African humans.

Loh et al. (2013) developed the ALDER software (Admixture-induced Linkage Disequilibrium for Evolutionary Relationships) as a further improvement of these ideas, using a weighted LD statistic to determine the most likely time of admixture. The LD decay curve used in ALDER is built under a variety of weighting schemes, making it possible to estimate admixture times and ratios. There are also additional, smaller refinements made to this method improving it further not mentioned here. ALDER looks for admixture in one specified admixed population, from any number of source populations. For the purpose of this project, a case with two source populations was considered the most interesting, and potentially looking into cases with only one source population.

Attempts have not yet been made to apply ALDER or any other LD-based method to lower quality data. The lower quality data lacks much information that is normally considered the very foundation of LD studies. Determining admixture through LD assumes having the DNA sequence known for both loci for all mutation sites. These mutations, or Single Nucleotide Polymorphisms (SNPs), form the basis for nearly all DNA-based evolutionary studies. This project is aims to investigate whether LD-based methods can still be used to determine admixture when using ancient DNA, despite lacking so much information.

By simulating data with known parameters, the performance of ALDER on simulated low quality data will be measured and evaluated.

1.5 fastsimcoal2

Excoffier et al. (2013) have developed fastsimcoal2 as a software for simulation of genomic data. This software produces the output in a format suitable for analysis in ALDER. Input may specify many parameters, permitting the simulation of admixture scenarios relevant for this project. This includes scenarios with separating populations, continuous migration, continuous growth and bottlenecks. fastsimcoal2 also allows the inclusion of having population hotspots which is something that does occur in humans. It is outside the scope of this project, but it could potentially be implemented as a future extension.

By taking care both to write the input files in a proper manner as well as manipulating the output files, the simulations received from fastsimcoal2 should resemble actual ancient DNA close enough. Such manipulation would include removing SNPs, post-mortem mutations, and low coverage. Necessary input parameters are of realistic effective population sizes, recombination- and mutation rates, and low sample sizes.

1.6 Outline of the project

The goal of this project is to evaluate how well ALDER works when applied to simulated low-quality data, similar to that of ancient DNA recovered from 5 000-10 000 years ago. As no studies of this had previously been done, it was unknown how well these methods would work. By finding the lowest quality data that ALDER would still be able to analyse is reached, further investigations were performed around these parameters to have a well-researched base to draw conclusions from.

The project consisted of running simulations in fastsimcoal2 to get an understanding of the

behaviour of the LD resulting from admixed populations with fragmented and low coverage data. There were plenty of variables to adjust for in these circumstances, such as time passed since admixture, deterioration of sample qualities and sample size. The project would largely focus on exploring to what extent it will and will not be a possibility to find these admixture events under varying circumstances.

There were a large amount of simulations run through a pipeline created for this project, this pipeline and simulations took up a sizeable portion of the project.

Simulations were done on fastsimcoal2, the pipeline was written as bash scripts and Perl, with calculations done in R. Heavy computational work was performed on UPPMAX.

2. Materials and methods

2.1 Creating an analysis pipeline

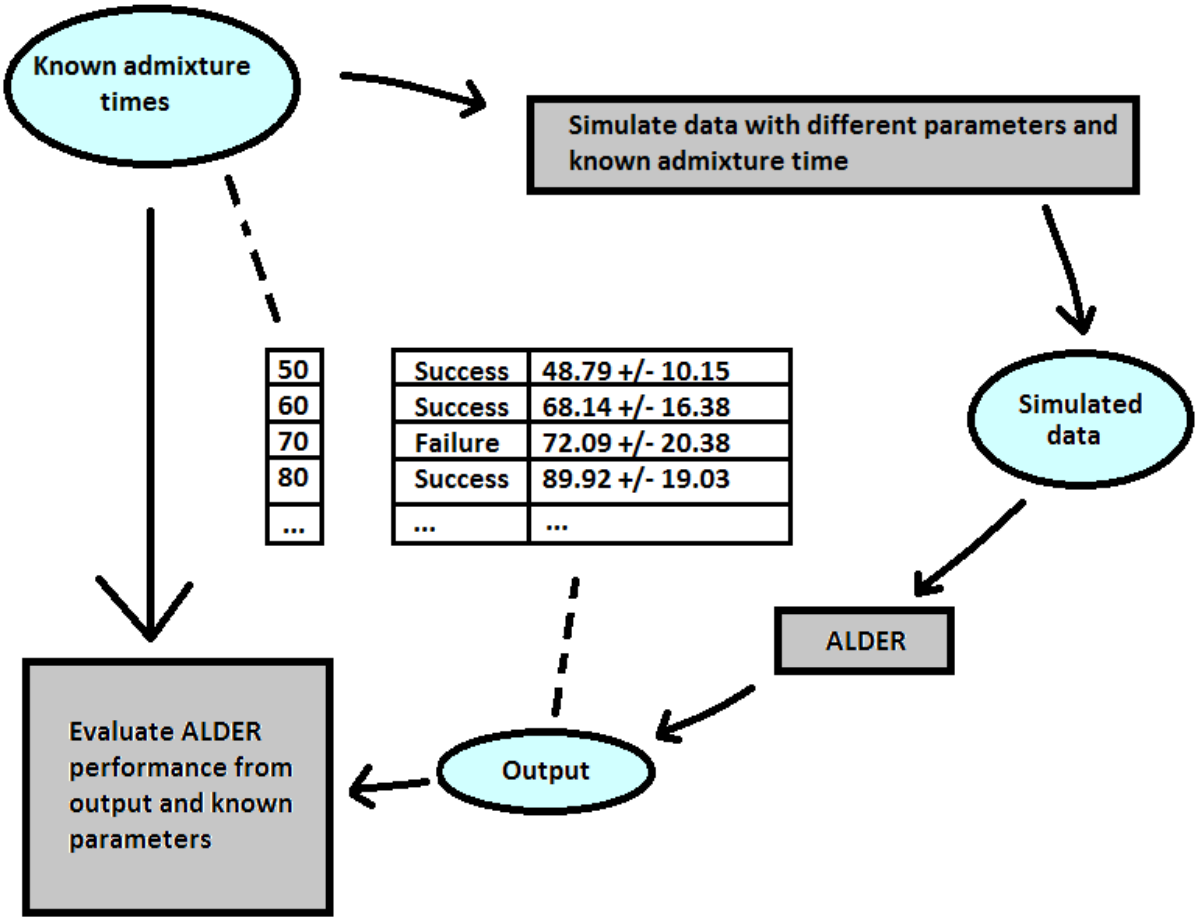


Fig. 1 An illustration of the project concept for evaluating ALDER performance.

To efficiently perform the analysis outlined for this project, a robust workflow pipeline was needed. Illustrated in fig. 1 is the outline of the automated process which easily allows to quantitatively measure the performance of ALDER, in a variety of scenarios.. By creating scripts that link all these aspects in a manner that is automated and adaptable, input could easily be created and analyzed without much manual input.

2.2 fastsimcoal2

fastsimcoal2 is a simulation software for genetic data, that was found to be the most ideal for this project. It offers quick simulations with many different features for simulating a population scenario of a specific interest, with input and output easy to deal with. It was also chosen because of its capability of sampling DNA at different points in time, but this was a functionality that ended up not being used in this project. The output files are in a format convenient to parse through and create input files for ALDER.

Population scenarios were simulated to look as that in fig. 2, and additional parameters were assigned as follows. Each individual carried 20 chromosomes, each with 156 077 602 loci, to

simulate the human genome. The recombination rate per basepair is 1.3e-8 (Dawson et al. 2002), and the mutation rate per basepair is 1.5e-8 (Lipson et al. 2015). No population growths, bottlenecks or additional continuous migration occurred unless specified.

2.3 ALDER

ALDER is capable on performing admixture analysis in cases of admixed populations, with one or two source populations. If supplied more than two source populations, ALDER performs a two-source population scenario analysis between all possible combinations of source populations. ALDER takes input files containing information about each individuals count of alleles matching a generated reference genome. An individual matching the reference genome allele homozygously will be marked with a 2, and if it is homozygous for the variant allele it is marked as 0, for zero matches. ALDER should be capable of handling missing data (marked as 9), but this did not seem to work.

ALDER takes input files from admixture scenarios with genomic data from a population of admixed individuals, as well as genomic data from one or two source populations. The analysis output files from ALDER presents the estimated admixture time as an interval, along with a p-value and z-value for the analysis.

2.4 UPPMAX

Heavy computational work was performed as project b2015168 on UPPMAX, Uppsala Universitys high performance computer cluster. 2 000 core hours/month were allocated for the project, allowing to run many demanding jobs in parallel.

2.5 Programming Environment

I created a workflow pipeline on UPPMAX, the main script written in bash with supporting scripts written in Perl. The pipeline starts by feeding manually created input files into fastsimcoal2, specifying necessary parameters for the scenario to simulate. The output is parsed into a new format for ALDER, and analyzed. The resulting analysis is saved, and further investigated in a few different R scripts, resulting in the graphs seen in fig.s 5 to 10.

3. Results

3.1 Outline of the pipeline

In order to reach the goals of this project and get a thorough investigation of the capacities of ALDER a robust pipeline was needed. The only necessary input would be the input files for fastsimcoal2, while the rest of the analysis was automated. By creating an efficient pipeline it was quick work running one scenario after another, and the only manual work required after creating the pipeline was to create the input file for fastsimcoal2. Initially, a standalone software was used to convert the output files from fastsimcoal2 into input files compatible with ALDER, but it turned out to drain a lot of memory for large files and an in-house parser had to be created for that purpose.

ALDER was initially run on various admixture scenarios to test how well it worked under these premises, and if there is anything that could not be handled. These tests were done while still getting used to and working to the entire analysis pipeline, and depending on this initial success the project could have headed different ways.

3.2 Initial testing

To begin with ALDER tests were performed on larger datasets with between 30 to 50 sampled individuals in each run. There initial tests were aiming to test the capabilities of ALDER. After a series of tests, appropriate input parameters and test scenarios were identified. After these tests, the sample size was reduced to 12 individuals as that was the minimum requirement for ALDER.

The initial tests showed that ALDER had good performance of a simple population scenario with no additional population growth, bottlenecks and an admixture rate of 0.5. ALDER estimations were reliable up to an admixture time of 220, where performance got much worse until it was entirely incapable at an admixture time of 300. Slightly more advanced scenarios with subsequent additional migration, population growth or anothern admixture rate appeared to have just as accurate performance. Adding several of these scenarios together did create problems, but the performance on the more basic scenarios was sufficient for the purposes of the project. Investigating every single advanced scenario to see what worked and what did not would take too much time, and was deemed not to be the main focus of the project. The maximum generation time for ALDER to produce reliable output was found to be around 250 generations in the past, depending slightly on the parameters. At 300 generations, there was barely any positive output, regardless of parameters.

3.2.1 Reducing available data

The minimum amount of individuals required for ALDER was to have 4 individuals in each sample group (consisting of source population 1, source population 2, and the subsequent admixed population). As it is difficult to find good ancient DNA samples in real life admixture studies, the ideal case would be if ALDER required only one individual from each of the groups. As ALDER is an analysis largely focusing on establishing LD-statistics and then inferring the admixture data, it should be possible to determine the LD based on fewer samples. Having as many as 12 good samples would be considered extremely high in the typical excavation.

At this point, the amount of available SNPs for ALDER to analyse was reduced, in order to simulate the conditions of an actual study where the DNA is largely decayed. By simulating decay in this project, it is possible to get a better estimation on how well ALDER would perform when analyzing actual ancient DNA.

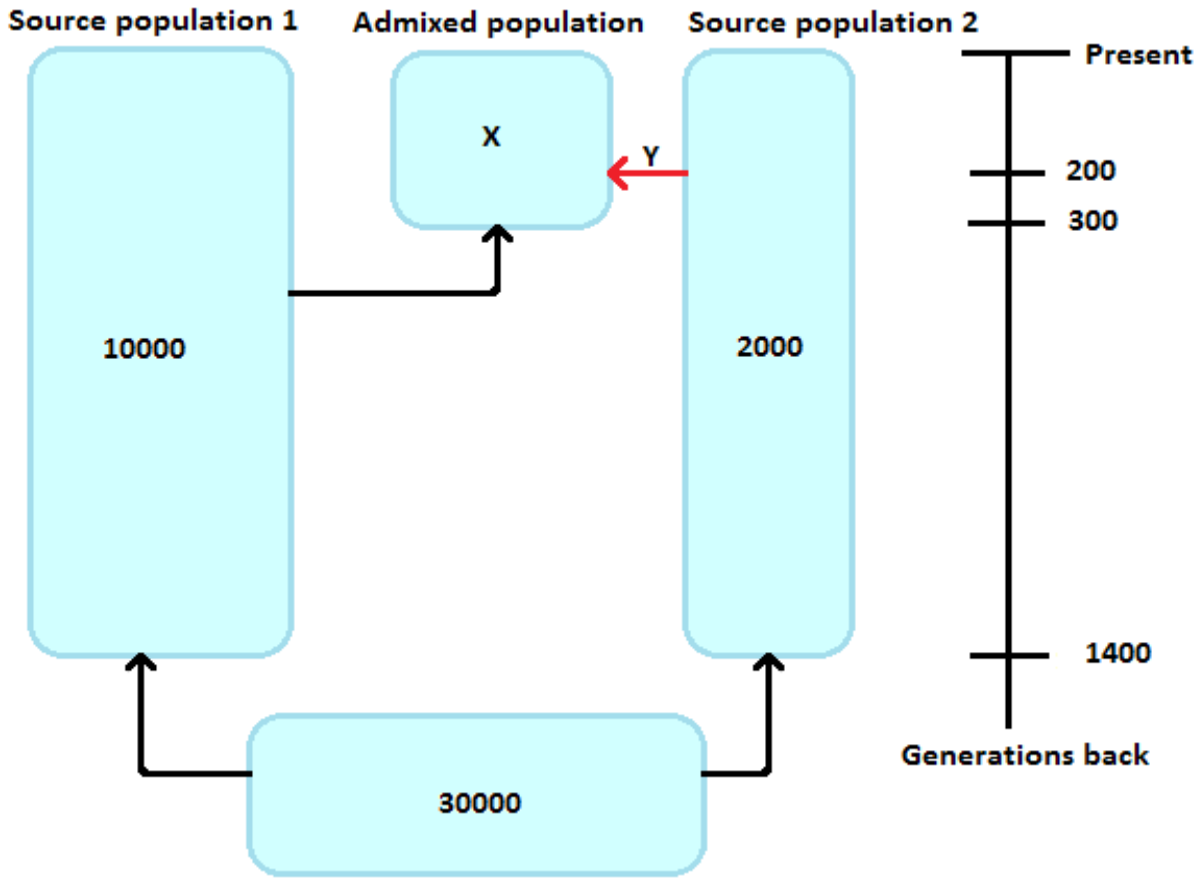


Fig. 2: Visualisation of the general population scenario to be analyzed, which is simulated in fastsimcoal2. The number in each of the boxes denotes the effective population size for that population. Black arrows denote that the source population make up the entire new population, while the red arrow denotes that a set fraction of the source population will migrate. Additional events were simulated in some of the initial tests, such as population bottlenecks, continous population growth or migration. The effective population size in the admixed population, X, as well as the migration fraction, Y, was set to different values in different scenarios. In all simulations throughout the project, each individual carries 20 chromosomes, each with 156 077 602 loci, to simulate the human genome. The recombination rate per basepair is 1.3e-8 , and the mutation rate per basepair is 1.5e-8, which is the estimated average in humans (Dawson et al. 2002) (Lipson et al. 2015). In the simulation, a set amount of samples are taken at present time from each population and produced as output from fastsimcoal2. This output is then transcribed into input files for analysis in ALDER.

Simulating this DNA decay is complicated, as it is more than a random process of lowering the amount of found nucleotides and inserting random mutations. It is a distinct process, which would be difficult to simulate as more mutations occur closer to the chromosome ends, and certain kinds of mutations being more likely than others. Simulating ancient DNA mutations this way would be outside the scope of this project, and potentially not adding much of importance. No kind of mutations would be inserted into the simulation output from fastsimcoal2. Instead, only the amount of SNPs for ALDER to analyze was reduced. This was used to produce a graph to show at what point the amount of retained SNPs became too low to produce any reliable output, as presented in fig. 3. These results showed that when the amount of retained SNPs were less than 2.56 %, ALDER estimations were no better than guessing in 50 % of cases, regardless of admixture time. Because of these findings, no investigations of estimated admixture times at retained SNP fractions would be performed in the project.

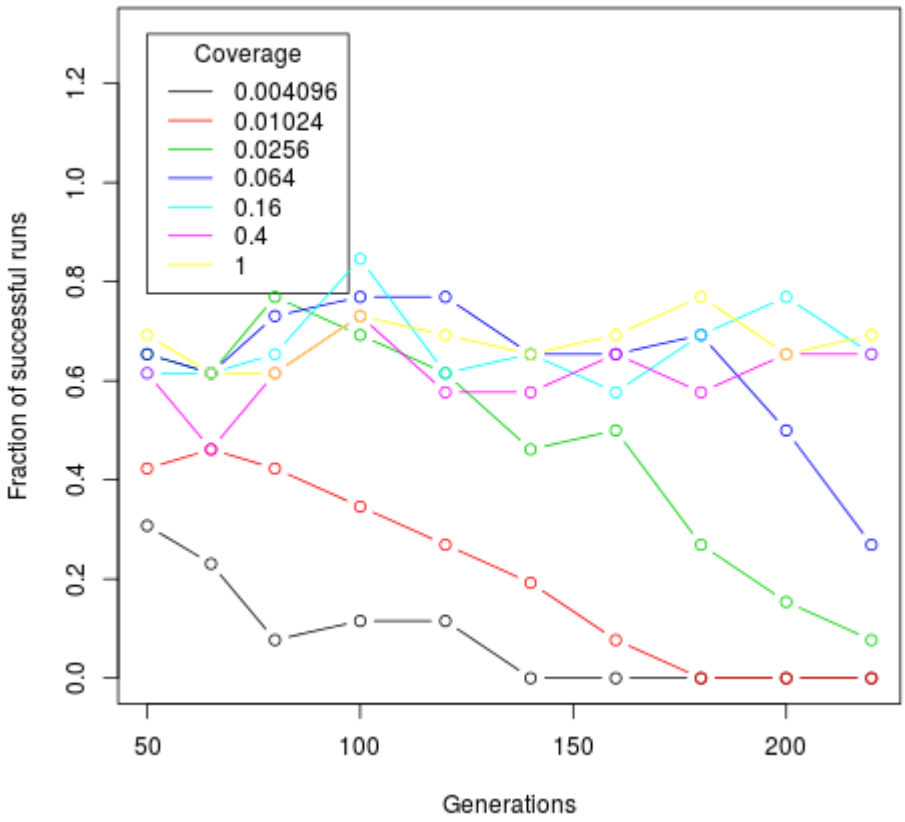


Fig. 3: ALDER analysis results of low coverage data. N=20 for each data point. Simulation was performed as presented in fig. 2, with an admixed population size of 5000 and admixture ratio of 0.4. For each population, 4 individuals were sampled. This analysis shows that when the fraction of retained SNPs were lower than 0.0256 the ALDER estimations were wrong in more than half the cases regardless of admixture time, and considered insignificant. Analyzing ALDER performance at coverage rates lower than 0.0256 would therefore not be investigated further in the project.

ALDER is supposed to be able to handle missing SNPs in single individuals, where the nucleotide in a certain position is known for some, but not all, individuals in the analysis. However, this functionality did not seem to work as intended as it resulted in ALDER giving very poor estimations even with just a few unknown nucleotides. This functionality would have been good to make use out of, as missing data often occur in actual studies. It is unlikely to identify all nucleotides for a certain locus across several individuals when the coverage is as low as it is in an ancient DNA study.

3.2.2 Modification of ALDER

Looking into the source code of ALDER, it was discovered that it is actually possible to lower the amount of samples needed. This prevented ALDER of estimating the admixture rate, only

estimating the admixture time, but the benefits of this far outweighs the negatives. In the end, only four samples in total is required, which is a huge step forward. Specifically, ALDER requires one sample from each of the source populations, and two samples from the admixed population.

However, this only works for the case of two source populations. In the case of only one source population, four samples still seems to be needed. It may be possible to change this as well, but we have not found a way to do it.

Many testing runs were performed on this set of only 4 sampled individuals, with great success. With the basic simulation parameters that were used for these tests, even when using only 4 individuals the analysis was as successful as when using 12. Success is measured as the ratio of analyses marked as "successful" by ALDER, as well as actually giving an estimated range that catches the true admixture time. However, the range gets wider with fewer samples, making it less specific and informative. It was determined that the range remained narrow enough for it to still be considered useful.

3.2.3 Simulating diploid data from ancient DNA

One of the biggest issues with ALDER for ancient DNA studies is that it requires knowledge of both nucleotides of every locus in each individual. Knowing this will very rarely be the case in actual cases, as the quality of ancient DNA is very low. Therefore, an idea was needed to simulate knowing only one nucleotide in each position, and somehow extrapolating from there.

By combining the data from two individuals from the same population group, randomly selecting one of two nucleotides from one individual, and one of two nucleotides from the other individual it is possible to combine these into a new genome. This procedure was used to create all individuals to be analysed by ALDER. This method has already been implemented in similar projects. However, in this project it did not work even for conditions that would normally produce more reliable output. As the analysis by ALDER was considered exhausted at this point, a different method was to be used to finish the last months of the project.

3.3 Alternate LD decay method

An alternate method would be needed where an indication of the admixture time would be given while still not needing two nucleotides at each position in every individual. One way of achieving this would be a survey of the simple LD-decay curves that form the basis of admixture analysis, such as the one seen in fig. 4. By sampling various data points for the admixed population in LD-decay curves, there should be signals of the kind that LD-decay-based admixture analysis stems from.

LD-decay curves were investigated in individuals from different admixture scenarios to see if there was any pattern to how they looked depending on the admixture time. This meant creating a script that would take the nucleotide sequence from each sample, calculating the LD as the covariance of allele frequencies between SNP pairs depending on their distance, and then plotting the acquired LD-decay. However, this was not successful even in test runs with knowledge of both nucleotides in every locus. While proper LD-decay curves were acquired, there was no observable difference between individuals that had admixed at 10, 100 or 200 generations ago, a sample of which can be seen in fig. 5. An observable difference should be expected, but none was found even after weeks of tampering with the script.

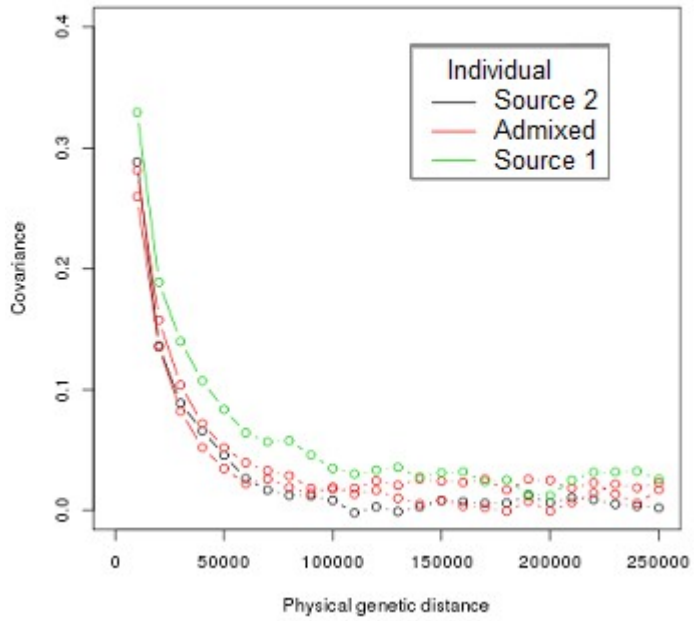


fig. 4: Linkage disequillibrium decay. Analysis of covariance between all found SNPs based on distance. The data was created from fastsimcoal2, using the simulation parameters described in fig. 1. The effective population size for the admixed population was 5 000, and the admixture rate from source 2 was set to 0.4.

3.4 Going back to ALDER

During the process of trying to figure out the alternate LD decay method, an alteration of the method we were trying with ALDER was devised. The issue of having knowledge of only one nucleotide for each locus might be resolved if this modification would work. Previously, the idea had been to use two diploid individuals to randomly select one nucleotide from each and piece together a diploid, heterozygous individual. My idea was instead to create an homozygous diploid individual from only one individual, by randomly selecting one nucleotide for each position, and making it homozygous for that nucleotide in that position. This new method was used for a basic admixture scenario, and was run through ALDER. The results were almost as successful as they had been in the trial runs of the modified ALDER. The intervals that ALDER estimated did get wider, but it retained a high success rate. These results can be seen by comparing fig. 7 and fig. 8.

It may seem surprising that this second method worked as well as it did when the first method did not work at all, as they both very nearly do the same thing. However, there actually is one difference that seems to have a big impact on ALDERs capacity to estimate the admixture time. The first method is essentially doing what method two is doing, but doing it to two individuals at once and then adding the results together. When these two individuals are added together, information is lost. At a glance this should not do much difference, but it evidently does.

With this breakthrough of the project, the investigations of finding a pattern in LD-decay curves to find admixture times was abandoned. Even if it would have ended up working, it would remain a very crude tool, and likely less accurate than a functioning ALDER analysis. For the purpose of this project, it will simply be treated as a confirmation that LD-decay curves look as they should, without looking into the detailed differences depending on various admixture scenarios.

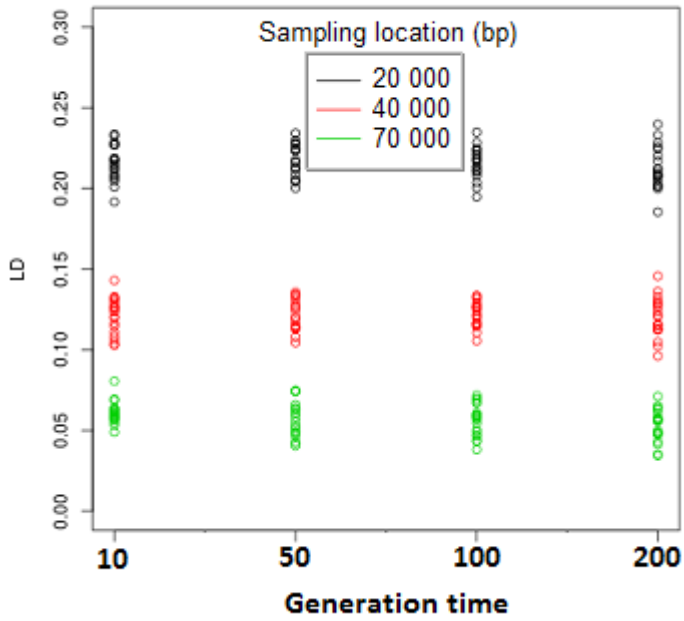


Fig. 5:Summary of the results from the alternate LD decay method. 15 LD-decay curves for each generation time were generated as presented in fig. 1. At the sampling locations specified by colors, samples of the LD was taken and mapped in this plot, to see if there was any difference in LD depending on generation time in any of the different sampling points.The results from this initial analysis showed no such signals.

3.5 Deep analysis of the final scenarios

A method that worked for ALDER analysis where the amount of samples was low, only one allele for each locus was known, and the SNPs could be reduced by a great amount had now been devised. This marked the start of the main investigations into ALDER for this project. Previous tests with ALDER had shown that there were no unexpected deviations in estimations for scenarios where there was additional migration, population growth or bottlenecks. As a significant amount of computer time running and analyzing the simulations is required, only a limited amount of scenarios could be tested. Running many simulations of a single scenario was prioritized over running fewer simulations for many scenarios, in order to get more robust data with less noise. The only requirement for testing any other scenarios is to create input files for fastsimcoal2, which is quick to do. Should interest arise in investigating scenarios that are not explored here, it is simple to prepare the input.

Table 1 is showing an overview of the six scenarios chosen for this final investigation, based on hypothetical ideas that could be the case in the actual migration and admixture of Scandinavian hunter-gatherers and farmers. Parameters and scenario are fully described in fig. 2. Scenarios I, II, and III use the same population sizes and admixture rates, but the genomic data was sampled in different ways. In scenario I, 4 individuals were sampled from each population, representing the minimum required for ALDER analysis as intended, before the ALDER source code was altered. This scenario retains the full knowledge of both alleles in all positions. Scenario II was also analyzed with full knowledge of both alleles in all positions, but the sampling was reduced to the lowest possible after altering the source code of ALDER. Scenario III is performed in the same way,

but retaining knowledge of only one allele in every position. The analysis of scenario I allows an evaluation on how much information is lost by analyzing scenarios not natively permitted by ALDER. Scenario III allows an evaluation on how much information is lost by retaining information about only one allele in every scenario. Scenarios IV, V and VI are performed using the same information as scenario III, but with altered variables to test alternative population scenarios. This results in having four different scenarios tested using the lowest possible sample size with information retained for only one allele in all positions.

Source population 1 represents the farmers, with an effective population of 10 000, also contributing the most individuals to the admixed population. Source population 2 is the hunter-gatherers, effective size of 2 000 and providing less individuals to the admixed population; 40 % in 3 of the scenarios, and 15 % in one of them. The admixed population has a varying effective population size in the different scenarios, 5 000 in two of them, and 10 000 in one scenario, 2 000 in the other. This is to see if there are any of these scenarios that ALDER fails to pick up on.

The results show that ALDER is capable of estimating admixture time in all of the scenarios with roughly equal success, with the average differences also being all the same. The interval size also stays the same.

Scenario	Description	Admixed effective population size	Admixture rate from pop. 2
I	Medium admixture population size, full knowledge of alleles, 4 sampled individuals from each population.	5000	0.4
II	Medium admixture population size, full knowledge of alleles, minimal sampling.	5000	0.4
III	Medium admixture population size, knowledge of one allele, minimal sampling.	5000	0.4
IV	High admixture population size, knowledge of one allele, minimal sampling.	10000	0.4
V	Low admixture population size, knowledge of one allele, minimal sampling.	2000	0.4
VI	Low migration rate from source population two, knowledge of one allele, minimal sampling.	5000	0.15

Table 1: Parameters of the final investigation scenarios. These populations were simulated as visualized in fig. 2. 100 simulations were performed for each specified generation time.

3.5.1 Metrics used

Six different metrics were considered interesting for analyzing and evaluating ALDER. Graphs presenting this data are seen in fig.s 6 to 11, all displaying the same metrics for different sets of data. It includes data for different amounts of retained SNPs, as a measure of fraction of original SNP remaining which can be considered to represent the overlapping sequencing coverage. By seeing how ALDERs performance varies depending on both the amount of SNPs and generation time, it is possible to identify points where the behaviour of ALDER goes wrong, and could potentially be resolved.

In graph A in each of these figures, the average estimated age by ALDER for each generation and retained fraction of SNP is plotted for analyses labeled successful but where the given interval does not overlap with the true admixture time. There is also a dashed black help line showing where the true admixture time lies. The graphs here will show if there is point in perticular where ALDER consequently either over- or underestimates the admixture time while still labeling them successful, and if it differs depending on the SNP coverage.

In graph B, the average estimated age by ALDER for each generation and retained fraction of SNP is plotted, along with a black dashed help line showing the correct admixture time. This graph will show if there is a slight bias for over- or underestimating the admixture time, even when the interval given by ALDER is correct. As the estimated admixture times are plotted for different SNP coverage rates, evidence both for a general or a coverage-specific bias may be discovered.

Graph C shows the fraction of correct successful analyses from ALDER compared to the total amount of analyses marked as successful. This will show if there is any point where the amount of false positives is too high for the output to be considered useful. Conversely, there may be a point where the fraction of true positives is very high even when the total amount of successfully analyzed runs are low, meaning that the output is very reliable in the few cases where it does work.

Graph D shows the fraction of correct successful analyses as a fraction of the total amount of analyses performed. This will show if there is any point where performance is affected when lowering the coverage, as well as if there are any perticularly difficult admixture times.

Graph E shows the interval width given from ALDER analyses for runs incorrectly labeled as successful. This could show if the interval range happens to become too narrow for some analyses, resulting in an excessive amount of failed runs.

Graph F shows the interval width for correctly labeled successful analyses. This is useful to see if the interval width goes very wide at a specific generation time and coverage, making successful analyses less useful as they are stated in too general of a manner.

There are also additional conclusions that can be drawn by combining the information from several of the graphs.

If the average estimated admixture time from graph A looks similar to the admixture time from B, but the interval width in E is significantly lower than that in F, the main issue with incorrectly labeled successes is that the interval width is too narrow in some cases.

By judging the success rate in graph D together with the interval width from F, the usefulness of the analysis can be judged. A high success rate is not as impressive if the interval width is big. On the other hand, a narrow interval width is not impressive if the success rate is low.

The data from both C and D is important when looking at any of the other graphs as they together

reveal how many ALDER runs actually were truly successful and incorrectly labeled as successful. If any point of either of these graphs is low for a certain generation and coverage, it means that the statistical basis of the other graphs at this same generation point may be low, and great care should be taken before drawing conclusions about them.

Finally, it is an important analysis to compare graphs between different scenarios. By comparing differences in successes from altering just one parameter, it should be possible to tell how these parameters actually influence the performance of ALDER.

3.5.2 Analysis of scenario I

Fig. 6 shows the results of analyses performed on data where 4 samples were taken from each population and allele information remained in full.

In Graph A, up to an admixture time of 100, the average estimated admixture time is slightly overestimated for all coverage fractions. At an admixture time of 120, all coverage fractions except for that of 0.4 is where it should be, while for the coverage of 0.4 it is underestimated. As only 2 samples were incorrectly labeled as successful at this point, it may simply be a statistical outlier. At an admixture time of 140 generations, the average for all coverage fractions is roughly correct. For an admixture time of 160, the admixture time is overestimated for all coverage fractions except that of 0.064. No correlation related to fraction of retained SNPs is appearant. At an admixture time of 180, all analyses underestimate the admixture time to roughly 165. At an admixture time of 200, the time is overestimated for some fractions, and underestimated by others, with no appearant correlation to retained SNPs. At a generation time of 220, the admixture time is overestimated for all fractions of retained SNPs, with higher amounts of retained SNPs having a larger overestimation.

From graph B, the average admixture time given by ALDER for correctly labeled analyses simulations appear to stay roughly as accurate same for all SNP coverage fractions.

From graph C, the rate of truly successful analyses as a fraction of the total amount of successfully labeled analyses has no appearant correlation with the fraction of remaining SNPs. At admixture times of 50 and 200, there is a dip in the fraction of truly successful analysis, with no analysis having a higher success rate than 60 % regardless of retained SNPs. For all other admixture times, the success rate is somewhere between 50 and 90 %.

Graph D shows how less than 60 % of runs were truly successful for an admixture time of 50, regardless of remaining SNPs. For an admixture time between 65 and 140, the success rate was around 60 to 80 %, with no appearant correlation between admixture time, remaining SNPs and success rate. At an admixture time of 160, the success rate seems to decline down to a success rate around 30 % for all fractions of remaining SNPs.

From graph E, the interval width in analyses incorrectly labeled as successful, interval width increases with higher admixture time. The average width was different for different SNP fractions, but there was no direct correlation between the fraction of retained SNPs and the interval width. In some cases the analyses with highest amount of retained SNPs had the largest interval width, and in other cases it had the lowest width.

From graph F, the interval width given by ALDER in this scenario remains roughly the same for all SNP coverage fractions, and the interval width increases linearly with increased admixture time. There are some deviations in the interval width, but with no appearant correlation between the width and fraction of retained SNPs.

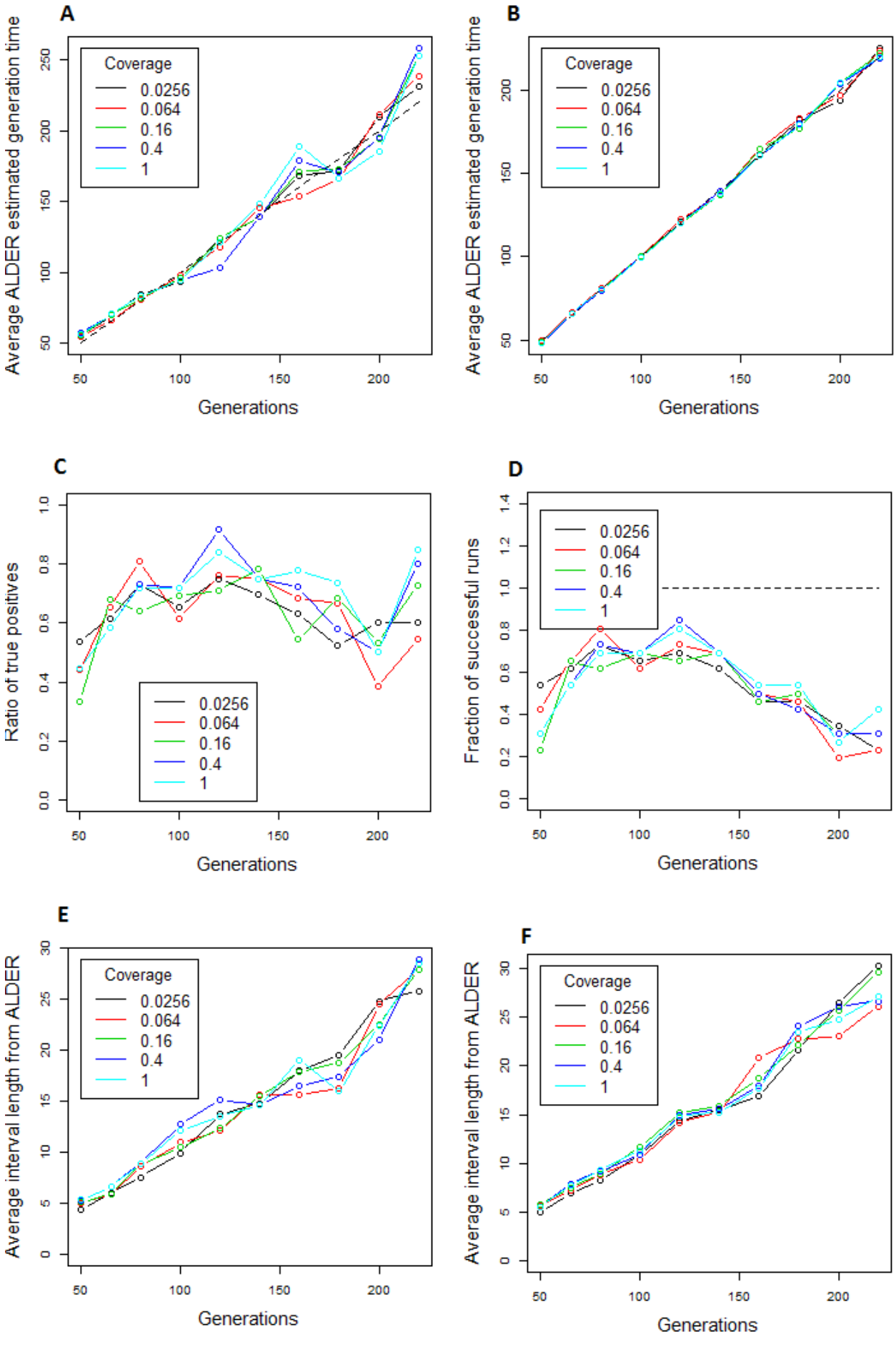

A
Average ALDER estimated generation time
Coverage
0.0256
0.064
0.16
0.4
1
Generations
B
Average ALDER estimated generation time
Coverage
0.0256
0.064
0.16
0.4
1
Generations
C
Ratio of true positives
0.0256
0.064
0.16
0.4
1
Generations
D
Fraction of successful runs
0.0256
0.064
0.16
0.4
1
Generations
E
Average interval length from ALDER
Coverage
0.0256
0.064
0.16
0.4
1
Generations
F
Average interval length from ALDER
Coverage
0.0256
0.064
0.16
0.4
1
Generations

Fig. 6: ALDER analysis results for parameters outlined in scenario I. N=25 for each generation time. The data was generated through the process described in Fig. 2, where the variables X and Y were set to 5000 and 0.4, respectively. These simulations were analyzed while retaining the full knowledge of alleles from the fastsimcoal2 output. In order to test how well ALDER performed with knowledge of less than all loci, various fractions of SNPs were retained, here denoted as Coverage. A) The median estimated generation time from ALDER in cases where ALDER considered the output successful, but was actually not. B) The median estimated generation time from ALDER in cases where ALDER considered the output successful, and was. C) The ratio of truly successful analyses among all analyses. D) The fraction of successful runs compared to the amount of runs as a whole. E) The interval width given by runs ALDER considered successful, but were actually not. F) The interval width given by runs ALDER considered successful, and were.

3.6.3 Analysis of scenario II

Fig. 7 shows the performance of ALDER analysis on a simulation scenario where information about both alleles for every locus remained intact, depending on the fraction of remaining SNPs, with one sample from each source population and two samples from the admixed population.

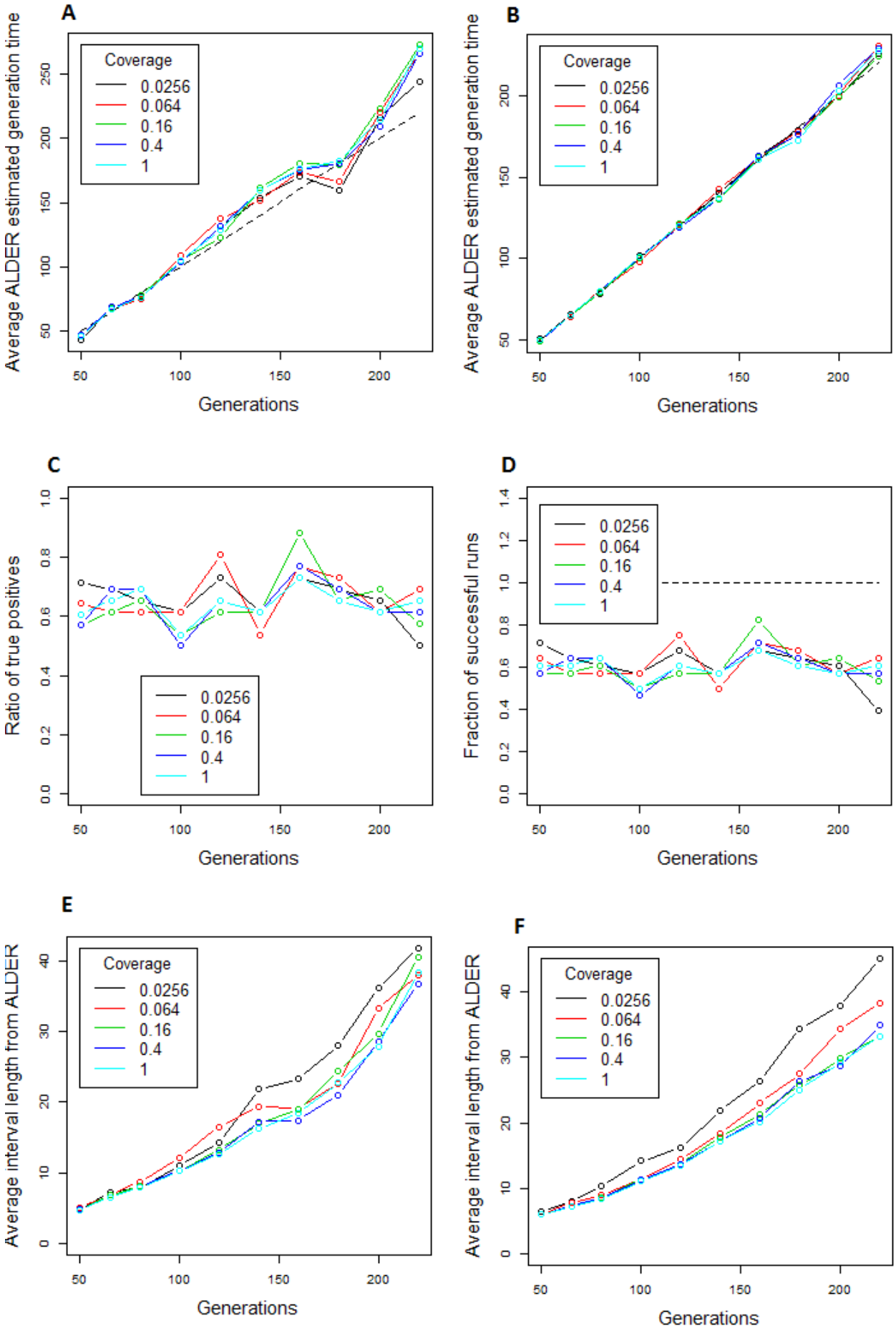
In Graph A, incorrectly labeled successful analyses have similar errors across all coverage fractions. Until 100 generations, the average estimated time is the same as the true admixture time. Between 120 and 160 generations, the average estimated time is higher than the true admixture time. At 180 generations, high coverage analyses had the same average estimated time as the true admixture time, while low coverage analyses had a lower estimated admixture time than the true admixture time. For 200 and 220 generations, regardless of coverage level, the admixture time was overestimated. From 140 generations and upwards, low coverage analyses generally had a lower estimated generation time than those from higher coverage.

From graph B, the average admixture time given by ALDER for correctly labeled analyses simulations appear to stay roughly the same for all SNP coverage fractions. At 220 generations, there is a slight bias across all coverage levels to slightly overestimate the true admixture time.

Graph C shows that the ratio of true succeses is roughly 60 %, regardless of remaining SNPs. There is a small dip in the performance at 220 generations for SNP coverage of 0.0256, where it is only successful in 40 % of cases, but there is no data at higher generation times to see if this performance remains.

Graph D is nearly identical to graph C, as ALDER labeled nearly all analyses as successful, only labeling four analyses in total as failures.

Fig. 7: ALDER analysis results for parameters outlined in scenario II. N=100 for each generation time. The data was generated through the process described in Fig. 2, where the variables X and Y were set to 5000 and 0.4, respectively. These simulations were analyzed while retaining the full knowledge of alleles from the fastsimcoal2 output. In order to test how well ALDER performed with knowledge of less than all loci, various fractions of SNPs were retained, here denoted as Coverage. A) The median estimated generation time from ALDER in cases where ALDER considered the output successful, but was actually not. B) The median estimated generation time from ALDER in cases where ALDER considered the output successful, and was. C) The ratio of truly successful analyses among all analyses. D) The fraction of successful runs compared to the amount of runs as a whole. E) The interval width given by runs ALDER considered successful, but were actually not. F) The interval width given by runs ALDER considered successful, and were.


A
Average ALDER estimated generation time
Coverage
0.0256
0.064
0.16
0.4
1
Generations
B
Average ALDER estimated generation time
Coverage
0.0256
0.064
0.16
0.4
1
Generations
C
Ratio of true positives
0.0256
0.064
0.16
0.4
1
Generations
D
Fraction of successful runs
0.0256
0.064
0.16
0.4
1
Generations
E
Average interval length from ALDER
Coverage
0.0256
0.064
0.16
0.4
1
Generations
F
Average interval length from ALDER
Coverage
0.0256
0.064
0.16
0.4
1
Generations

From graph E, the interval time of analyses incorrectly labeled as successful, lower coverage analyses had higher interval widths than those of higher coverage in some cases, but not all. The average interval width of these incorrect analyses were on the same level as those that were correct up until a generation time of 180. In generation time 200 and 220, the average interval width was larger for high coverage analyses compared to that of truly successful runs.

From graph F, the interval width given by ALDER in this scenario remains roughly the same for all SNP coverage fractions above 0.1. For 0.064 coverage, it starts getting wider than these others at 200 generations in the past. For 0.0256 coverage, it is consistantly wider than the others from 80 generations and above, and the disparage of width getting wider as the generation time increases, the interval being up to 14 generations wider at the largest point.

3.6.4 Analysis of scenario III

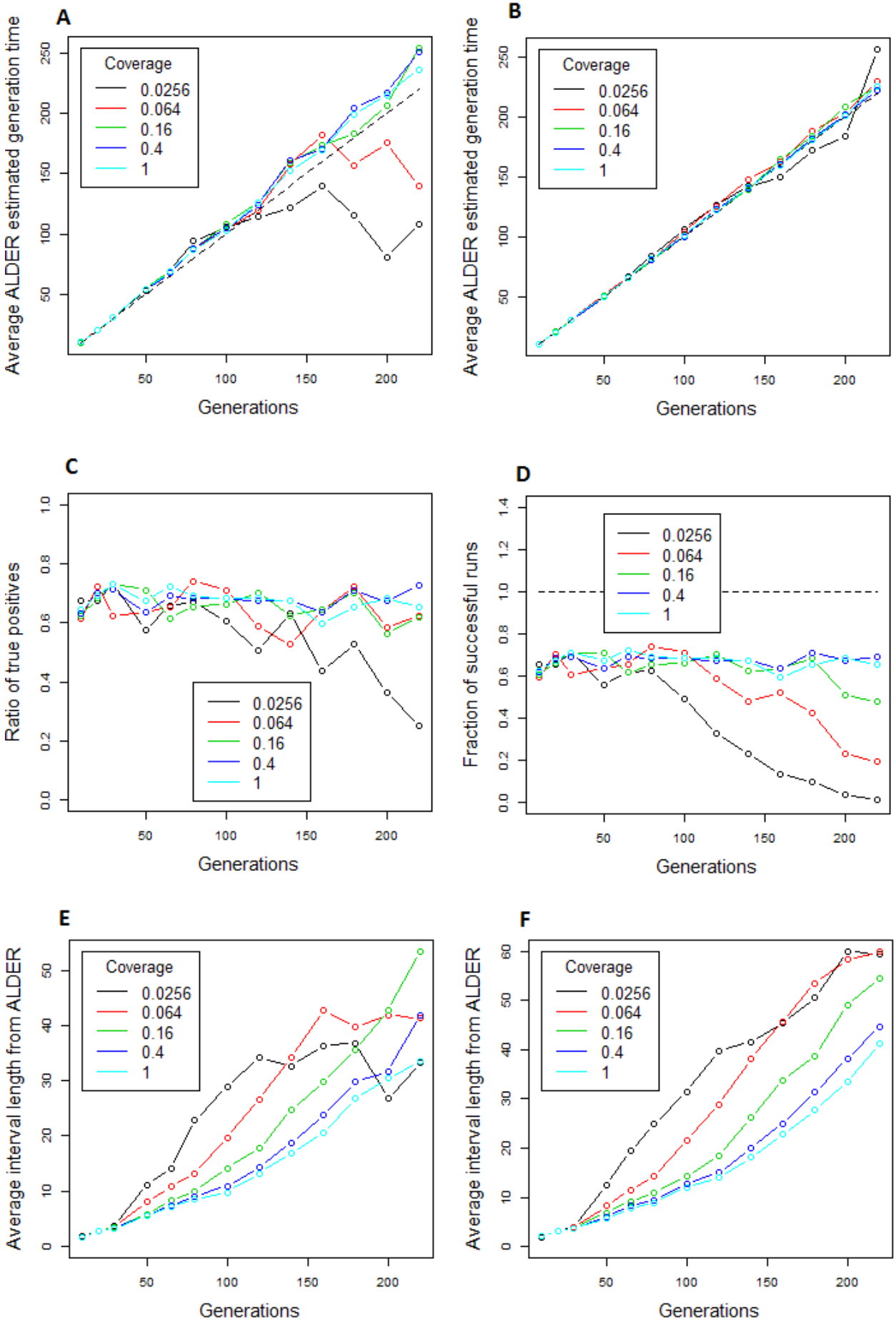
Fig. 8 shows the performance of ALDER analysis, depending on the fraction of remaining SNPs, for a simulation scenario where only information about one allele for every locus remained intact. The admixed population size was 5000, and the fraction of source population 2 in the admixed population at the migration event was 0.4.

From graph A, the average time of admixture given by ALDER in incorrectly labeled successful analyses followed the true admixture time up until an admixture time of 120. By there, incorrect analyses from 0.0256 coverage started giving analyses with an average in the regions of 110, regardless of true admixture time. Other incorrect analyses overestimated the admixture time on average from then on, with the exception of analyses performed with 0.064 coverage. It overestimated average admixture time until an admixture time of 160, where it from then on gave an admixture time in the regions of 150 generations back, regardless of true admixture time.

From graph B, the average admixture time given by ALDER for correctly labeled analyses simulations appear to stay roughly the same for all SNP coverage fractions apart from that of 0.0256 coverage. The successful 0.0256 coverage analyses first gave a lower admixture time on average compared to the correct admixture at an admixture time at 160 generations back, and then gave a higher average at the admixture time of 220.

Graph C shows that the amount of correctly labeled successful analyses among all analyses labeled as successful remains at around 60 % regardless of admixture time, except for a coverage of 0.0256 at admixture time 160 and onwards, where it starts declining down to a rate of 20 % truly successful at admixture time 220.

Fig. 8: ALDER analysis results for parameters outlined in scenario III. N=100 for each generation time. The data was generated through the process described in fig. 2, where the variables X and Y were set to 5000 and 0.4, respectively. These simulations were analyzed as having knowledge of only one allele for each locus, and assuming that the individual was homozygous for that allele. fastsimcoal2 output. In order to test how well ALDER performed with knowledge of less than all loci, various fractions of SNPs were retained, here denoted as Coverage. A) The median estimated generation time from ALDER in cases where ALDER considered the output successful, but was actually not. B) The median estimated generation time from ALDER in cases where ALDER considered the output successful, and was. C) The ratio of truly successful analyses among all analyses. D) The fraction of successful runs compared to the amount of runs as a whole. E) The interval width given by runs ALDER considered successful, but were actually not. F) The interval width given by runs ALDER considered successful, and were.


A
Average ALDER estimated generation time
Coverage
0.0256
0.064
0.16
0.4
1
Generations
B
Average ALDER estimated generation time
Coverage
0.0256
0.064
0.16
0.4
1
Generations
C
Ratio of true positives
0.0256
0.064
0.16
0.4
1
Generations
D
Fraction of successful runs
0.0256
0.064
0.16
0.4
1
Generations
E
Average interval length from ALDER
Coverage
0.0256
0.064
0.16
0.4
1
Generations
F
Average interval length from ALDER
Coverage
0.0256
0.064
0.16
0.4
1
Generations

Graph D shows that roughly 60 % of all analyses are successful and labeled as such, for coverage levels of 0.16 and upwards. When the admixture time occured at 120 generations back, the performance when having 0.064 coverage started to decline, going down to 20 % successes at a generation time for 220. When the admixture time was 100 generations, the success rate when having 0.0256 coverage declined down to 10 % by a generation time of 160.

From graph E, the interval time of analyses incorrectly labeled as successful, analyses labeled as lower coverage had higher interval widths than those of higher coverage in some cases, but not all. The average interval width of these incorrect analyses were on the same level as those that were correct, with some deviations. At generation time 120 and upwards, the 0.0256 coverage analyses gave an average generation width in the region of 30. In the case of 0.064 coverage, at an admixture time of 160 generations and upwards, the average width given was 40, regardless of admixture time.

From graph F, the interval width given by ALDER in this scenario remains roughly the same for all SNP coverage fractions up until an admixture time of 30. The interval width got higher with a lower coverage level and admixture. The interval width given by coverage levels 1.0 and 0.4 remained nearly the same regardless of generation time. At an admixture time of 160, the performance of 0.064 and 0.0256 coverage became roughly equal.

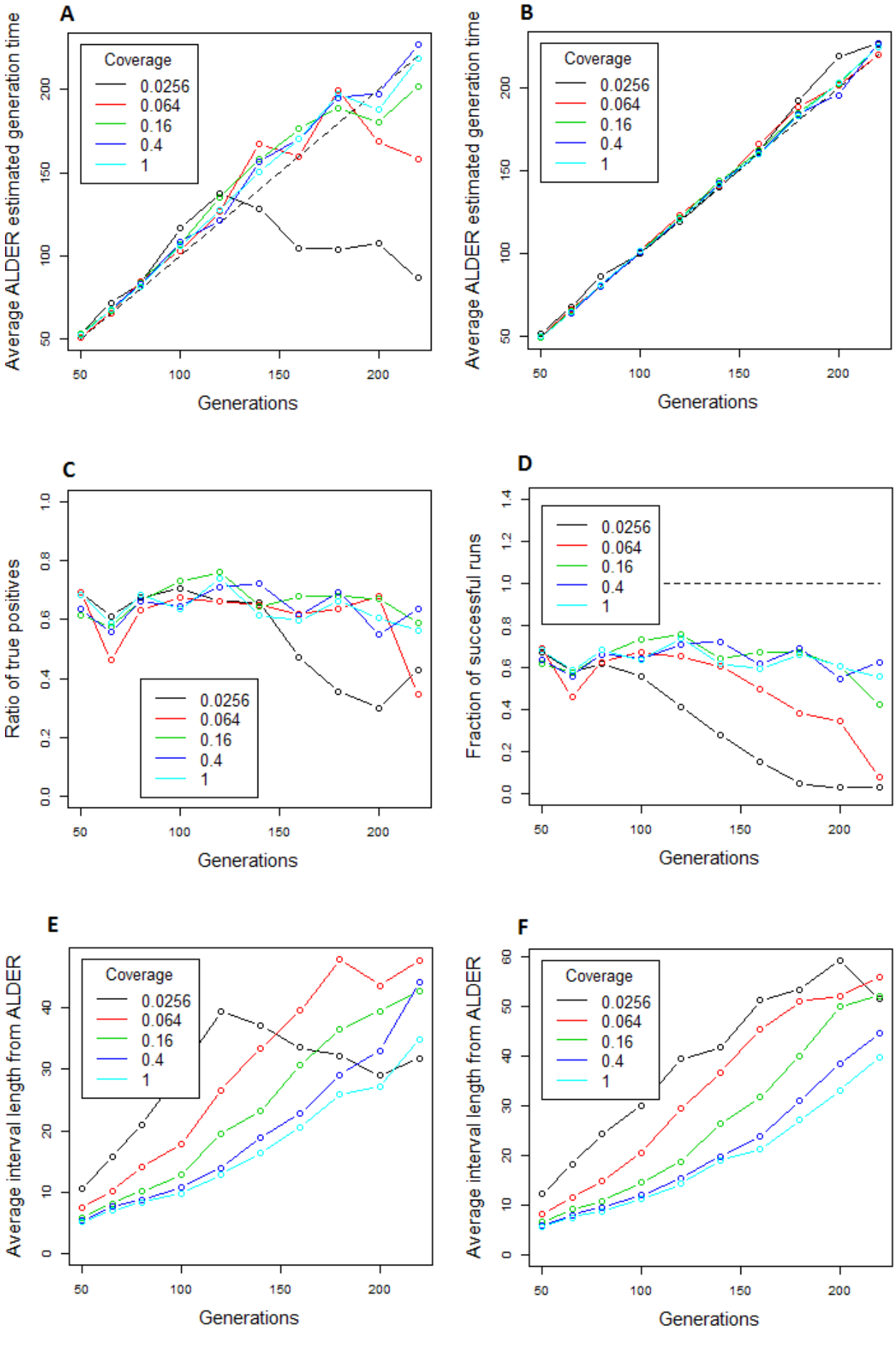
3.6.5 Analysis of scenario IV

Fig. 9 shows the performance of ALDER analysis, depending on the fraction of remaining SNPs, for a simulation scenario where only information about one allele for every locus remained intact. The admixed population size was 10000, and the fraction of source population 2 in the admixed population at the migration event was 0.4.

In Graph A, incorrectly labeled successful analyses generally overestimated the admixture time, with some exceptions. Incorrect analyses from 0.0256 coverage and an admixture time of 140 and upwards, started giving analyses with an average in the regions of 110, regardless of true admixture time. The analyses with 0.064 coverage gave far lower admixture times than the true admixture times of 200 and 220. At the admixture time of 200, all incorrect analyses underestimate the admixture time. At 220, it is underestimated from analyses with a 0.16 coverage, overestimated with 0.4 coverage, and having the correct average with 1.0 coverage.

From graph B, the average generation time given by ALDER for correctly labeled analyses simulations appear to stay roughly at the true admixture time for all SNP coverage fractions apart from that of 0.0256 coverage at an admixture time of 200 generations, where it was overestimated.

Fig. 9: ALDER analysis results for parameters outlined in scenario IV. N=100 for each generation time. The data was generated through the process described in fig. 2, where the variables X and Y were set to 10000 and 0.4, respectively. These simulations were analyzed as having knowledge of only one allele for each locus, and assuming that the individual was homozygous for that allele. fastsimcoal2 output. In order to test how well ALDER performed with knowledge of less than all loci, various fractions of SNPs were retained, here denoted as Coverage. A) The median estimated generation time from ALDER in cases where ALDER considered the output successful, but was actually not. B) The median estimated generation time from ALDER in cases where ALDER considered the output successful, and was. C) The ratio of truly successful analyses among all analyses. D) The fraction of successful runs compared to the amount of runs as a whole. E) The interval width given by runs ALDER considered successful, but were actually not. F) The interval width given by runs ALDER considered successful, and were.


A
Average ALDER estimated generation time
Coverage
0.0256
0.064
0.16
0.4
1
Generations
B
Average ALDER estimated generation time
Coverage
0.0256
0.064
0.16
0.4
1
Generations
C
Ratio of true positives
0.0256
0.064
0.16
0.4
1
Generations
D
Fraction of successful runs
0.0256
0.064
0.16
0.4
1
Generations
E
Average interval length from ALDER
Coverage
0.0256
0.064
0.16
0.4
1
Generations
F
Average interval length from ALDER
Coverage
0.0256
0.064
0.16
0.4
1
Generations

Graph C shows that the amount of correctly labeled successful analyses among all analyses labeled as correct remains at around 60 % regardless of admixture time, except for a coverage of 0.0256 at admixture time 160 and onwards, where the true success rate reduced to 40 % of all successful analyses. At an admixture time of 220, the performance when using 0.064 coverage became 35 %, while having been around 60 % at all other admixture times.

Graph D shows that roughly 60 % of all analyses are successful and labeled as such, for coverage levels of 0.16 and upwards. When the admixture occured at 160 generations back, the performance when having 0.064 coverage started to decline, going down to 10 % successes at an admixture time of 220. When the admixture time was 100 generations, the success rate when having 0.0256 coverage declined down to 10 % by a generation time of 160.

From graph E, the interval time of analyses incorrectly labeled as successful behave similar to that of graph F, with interval widths growing wider with higher admixture times and lower coverage, with one exception. At an admixture time of 140 and coverage of 0.0256, the interval width declines from 40 generations down to 30 at an admixture time of 220.

From graph F, the interval width of correct successful analyses got higher with a lower coverage level and admixture time. One exception occured, where the admixture time of 220, the average 0.0256 coverage analysis gave the same interval width as that of 0.16 coverage. However, there were few samples at this point and the statistical significance is low.

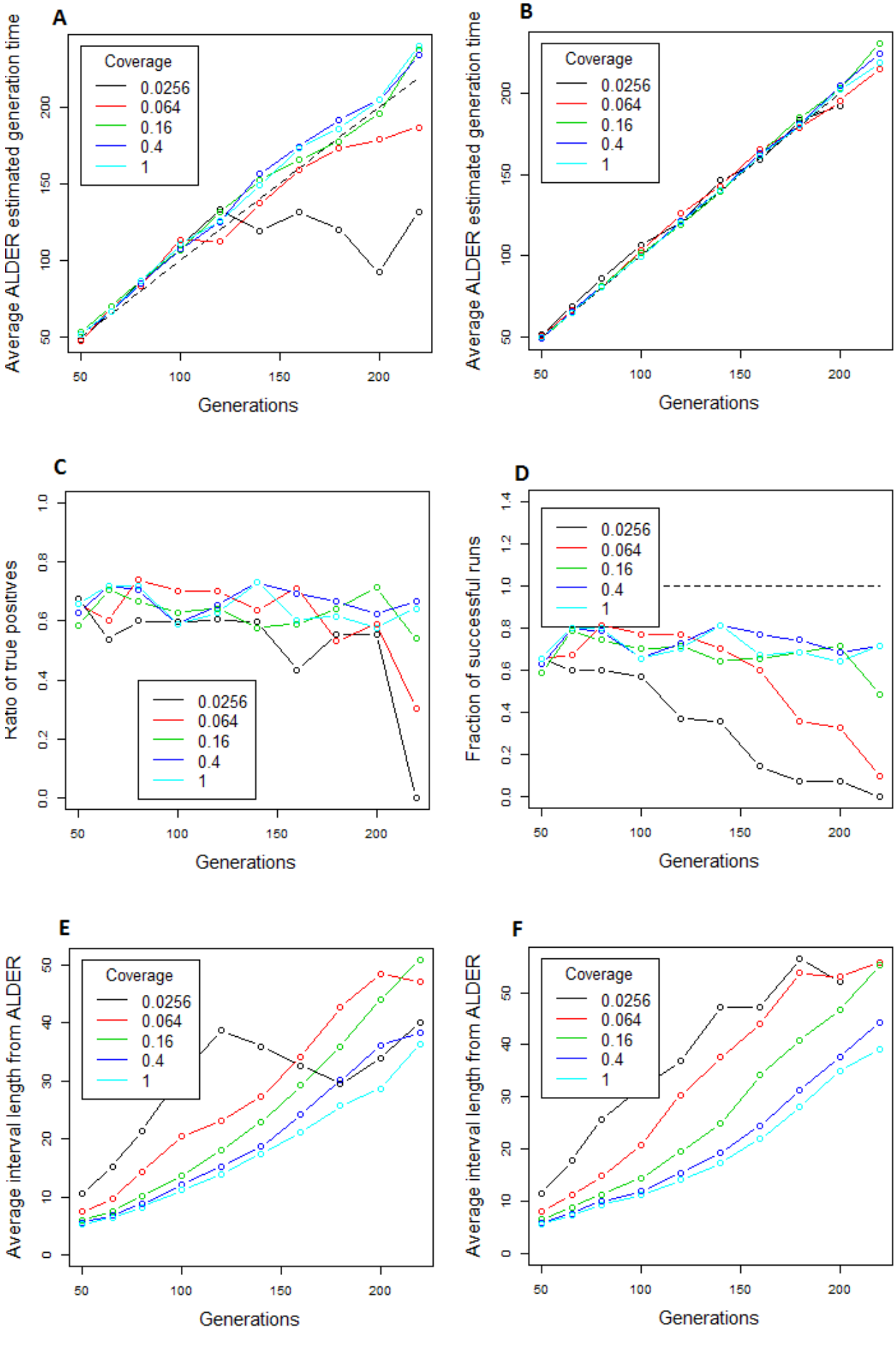
3.6.6 Analysis of scenario V

Fig. 10 shows the performance of ALDER analysis, depending on the fraction of remaining SNPs, for a simulation scenario where only information about one allele for every locus remained intact. The admixed population size was 2000, and the fraction of source population 2 in the admixed population at the migration event was 0.4.

In Graph A, incorrectly labeled successful analyses generally overestimated the admixture time, with some exceptions. Incorrect analyses from 0.0256 coverage and an admixture time of 140 and upwards, started giving analyses with an average in the regions of 110, regardless of true admixture time. The analyses with 0.064 coverage gave far lower admixture times for true admixture times of 200 and 220.

From graph B, the average admixture time given by ALDER for correctly labeled analyses simulations stays nearly on top with the actual admixture time for all SNP coverage fractions.

Fig. 10: ALDER analysis results for parameters outlined in scenario V. N=100 for each generation time. The data was generated through the process described in fig. 2, where the variables X and Y were set to 2000 and 0.4, respectively. These simulations were analyzed as having knowledge of only one allele for each locus, and assuming that the individual was homozygous for that allele. fastsimcoal2 output. In order to test how well ALDER performed with knowledge of less than all loci, various fractions of SNPs were retained, here denoted as Coverage. A) The median estimated generation time from ALDER in cases where ALDER considered the output successful, but was actually not. B) The median estimated generation time from ALDER in cases where ALDER considered the output successful, and was. C) The ratio of truly successful analyses among all analyses. D) The fraction of successful runs compared to the amount of runs as a whole. E) The interval width given by runs ALDER considered successful, but were actually not. F) The interval width given by runs ALDER considered successful, and were.


A
Average ALDER estimated generation time
Coverage
0.0256
0.064
0.16
0.4
1
Generations
B
Average ALDER estimated generation time
Coverage
0.0256
0.064
0.16
0.4
1
Generations
C
Ratio of true positives
0.0256
0.064
0.16
0.4
1
Generations
D
Fraction of successful runs
0.0256
0.064
0.16
0.4
1
Generations
E
Average interval length from ALDER
Coverage
0.0256
0.064
0.16
0.4
1
Generations
F
Average interval length from ALDER
Coverage
0.0256
0.064
0.16
0.4
1
Generations

Graph C shows that the amount of correctly labeled successful analyses among all analyses labeled as successful remains at around 60 % regardless of admixture time, with two exception at an admixture time of 220. At an admixture time of 220, the fraction of correct success analyses out of the total of success analyses when using 0.064 coverage became 35 %, while for a 0.0256 coverage zero successful analyses were had.

Graph D shows how roughly 60 % of all analyses are successful and labeled as such, for coverage levels of 0.16 and upwards. When the admixture time occured at 180 generations back, the performance when having 0.064 coverage started to decline, going down to 10 % successes at a generation time for 220. When the admixture time was 120 generations, the success rate when having 0.0256 coverage declined down to 10 % by an admixture time of 180, and no successful analyses at all were accomplished at an admixture time of 220.

From graph E, the interval time of analyses incorrectly labeled as successful behave similar to that of graph F, with interval widths growing wider with higher admixture times and lower coverage, with two exceptions. At an admixture time of 140 and coverage of 0.0256, the interval width declines from 40 generations down to 30 at an admixture time of 180, and then increases back up to 40 generations at an admixture time of 220. The interval for the analysis with 0.064 coverage narrows a little at an admixture time of 220 compared to that of 200.

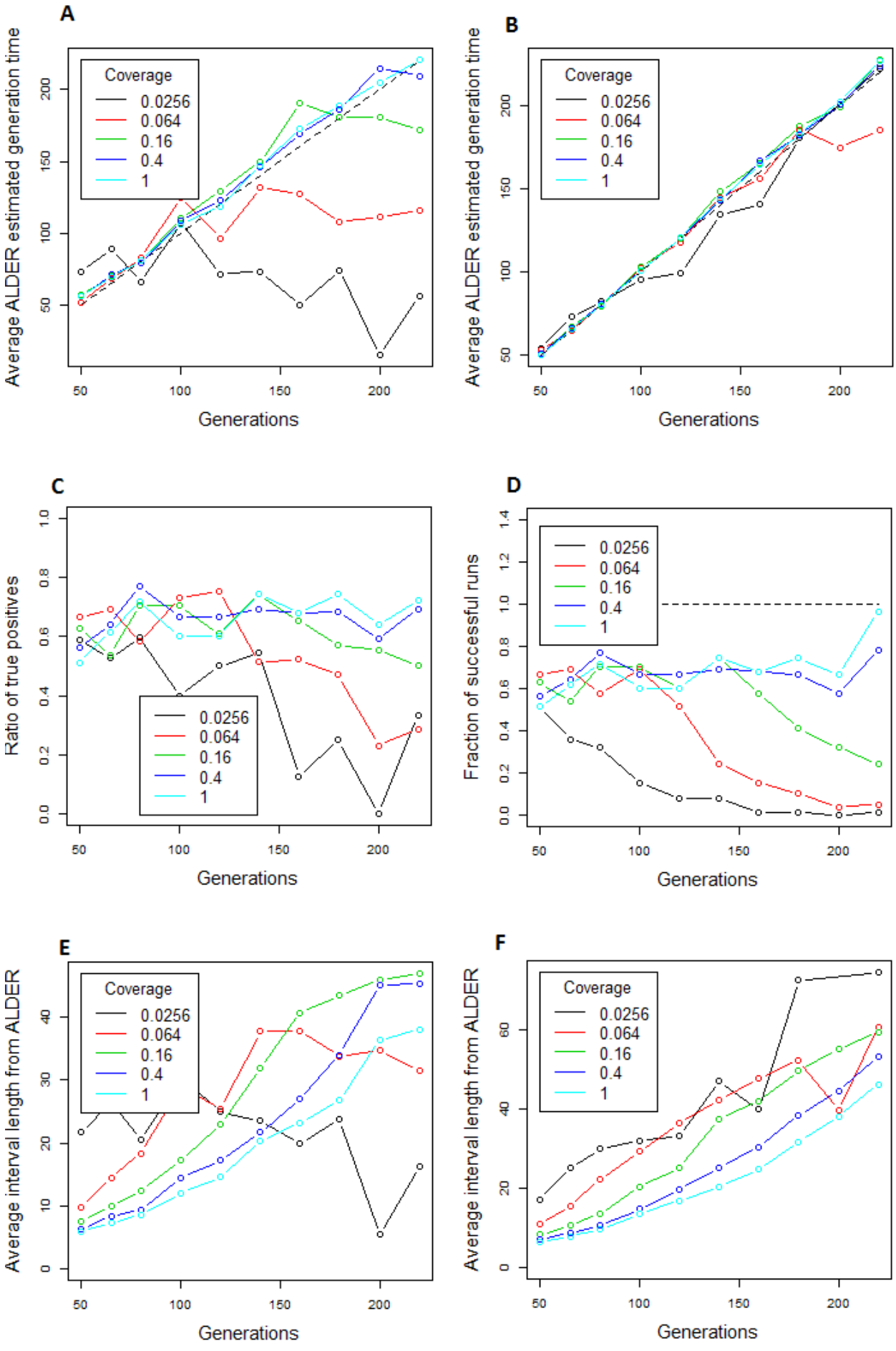
From graph F, the interval width of correct successful analyses got higher with a lower coverage level and admixture time. One exception occured for coverage of 0.0256, where the width decreased slightly at an admixture time of 200 generations. As no successful analyses had at an admixture time of 220, no width can be shown.

3.6.7 Analysis of scenario VI

Fig. 11 shows the performance of ALDER analysis, depending on the fraction of remaining SNPs, for a simulation scenario where only information about one allele for every locus remained intact. The admixed population size was 5000, and the fraction of source population 2 in the admixed population at the migration event was 0.15. There were many behaviours in this scenario that are hard to describe in words, best observed in the graphs themselves.

In Graph A, incorrectly labeled successful analyses appeared to stay roughly the same up until an admixture time of 120. By there, incorrect analyses from 0.0256 coverage started giving analyses with an average in the regions of 110, regardless of true admixture time. Other incorrect analyses overestimated the admixture time on average from then on, with the exception of analyses performed with 0.064 coverage. It overestimated average admixture time until an admixture time of 160, where it from then on gave an admixture time in the regions of 150 generations back, regardless of true admixture time.

Fig. 11: ALDER analysis results for parameters outlined in scenario VI. N=100 for each generation time. The data was generated through the process described in fig. 2, where the variables X and Y were set to 5000 and 0.15, respectively. These simulations were analyzed as having knowledge of only one allele for each locus, and assuming that the individual was homozygous for that allele. fastsimcoal2 output. In order to test how well ALDER performed with knowledge of less than all loci, various fractions of SNPs were retained, here denoted as Coverage. A) The median estimated generation time from ALDER in cases where ALDER considered the output successful, but was actually not. B) The median estimated generation time from ALDER in cases where ALDER considered the output successful, and was. C) The ratio of truly successful analyses among all analyses. D) The fraction of successful runs compared to the amount of runs as a whole. E) The interval width given by runs ALDER considered successful, but were actually not. F) The interval width given by runs ALDER considered successful, and were.


A
Average ALDER estimated generation time
Generations
Coverage
0.0256
0.064
0.16
0.4
1
B
Average ALDER estimated generation time
Generations
Coverage
0.0256
0.064
0.16
0.4
1
C
Ratio of true positives
Generations
0.0256
0.064
0.16
0.4
1
D
Fraction of successful runs
Generations
0.0256
0.064
0.16
0.4
1
E
Average interval length from ALDER
Generations
Coverage
0.0256
0.064
0.16
0.4
1
F
Average interval length from ALDER
Generations
Coverage
0.0256
0.064
0.16
0.4
1

From graph B, the average admixture time given by ALDER for correctly labeled analyses simulations appear to stay roughly the same and correct for all SNP coverage fractions apart from that of 0.0256 and 0.064 coverage. The successful 0.0256 coverage analyses first gave a lower admixture time on average compared to the correct admixture at an admixture time at 120 and 160 generations back. For 0.064 coverage, the average estimated admixture time was lower than the true value for admixture times both at 200 and 220.

Graph C shows that the amount of correctly labeled successful analyses among all analyses labeled as succesful remains at around 60 % regardless of admixture time for coverage rates of 0.16 and above. For a coverage of 0.064, it fluctuates around 60 % until it reaches an admixture time of 200, when the fraction of true successful analyses drops down to 30 %. For a coverage of 0.0256, it fluctuates around 60 % until it reaches an admixture time of 100, when the fraction of true successful analyses drops down to 40 %, increasing up to 50 % by admixture time 140, then dropping down to 20 % at admixture time 160, and having large fluctuations around that point.

Graph D shows how for a coverage level of 1.0, the performance was around 60 % regardless of admixture time, except for at 220, when the success rate was 98 %. For a coverage level of 0.4, the performance was around 60 % regardkess of admixture time. For a coverage of 0.16, the performance was around 60 % up until the admixture time was 160. From 180 to 220 the performance declined down to 30 %. For a coverage of 0.064, the performance was around 60 % up until the admixture time was 120.The performance declined down to 10 % by an admixture time of 180. For a coverage of 0.0256, the performance was at 50 % for an admixture time of 50, declining down to a performance of 10 % by an admixture time of 120.

From graph E, the interval time of analyses incorrectly labeled as successful, analyses for SNP coverages of 1, 0.4 and 0.16 behaved as that of correctly labeled successful in graph F. For the coverage of 0.064, the interval width increased to 40 by the admixture time of 140, and started decreasing slightly with increasing admixture times. For a coverage of 0.0256, no appearant correlation with the interval width and admixture time could be observed.

From graph F, the interval width of correct successful analyses got higher with a lower coverage level and admixture time, with some exceptions. At an admixture time for 200 and a coverage of 0.064, the width dropped to the same level as that of 100 % coverage for the same time. At an admixture time of 220, the width increased to being above that of 0.16 coverage. For the coverage of 0.0256, the average width dropped below that of 0.064 coverage at an admixture time of 120, and at the admixture time of 160, the width was lower than that of 0.16 coverage for the same admixture time.

4. Discussion

4.1 Current results

4.1.1 LD decay curves did not display signals of admixture

The results shown in fig. 5, with different data points taken from LD curves in different admixture scenarios, reveals no signals of admixture at all. This alternative method was intended as a less refined method to reveal admixture signals when ALDER was not able to. While this method was meant to be applied to data where only one allele was known in each position, the method did not give the desired results when both alleles were known either.

By looking at different points in the LD curve for different admixture scenarios, the expectation was to find some difference in how the LD decayed depending on the admixture time. No such differences could be observed in the tests performed within this project. However, as this is the methology that forms the basis of observing admixture through LD, traces should be possible to find. This alternate method was developed fairly quickly and not given much attention. It may have been possible that this method could have shown some indications of admixture if the method was investigated further. However, as ALDER did show to work in the circumstances that previously appeared not to work and caused the investigation of this method in the first place, this branch of the project was abandoned.

4.1.2 ALDER performance with few samples

By comparing the results from the ALDER analyses for scenario I and II, presented in fig. 6 and fig. 7 respectively, it is possible to see the changes in performance that occured when lowering the sample size below that which was originally required in ALDER. Originally ALDER required knowledge of 4 samples from each population in order to perform analysis, but changing the source code allowed to perform tests with only 4 samples required in total. Scenario I and II are both created using the same simulation parameters and retaining information about both alleles in all positions. Scenario I uses the original ALDER settings, while scenario II contains the bare minimum of samples.

In graphs E and F in fig. 6 and fig. 7, the interval width was consistantly lower for scenario I than for scenario II. This is most likely simply because scenario I contains information from 12 samples compared to the 4 samples from scenario II, and the increased information allows ALDER to give a more precise analysis. Graph B in fig. 6 and fig. 7 show that there is no consistant over- or underestimation of admixture time in any circumstance in either of the scenarios.

Graph A in fig. 6 and fig. 7 show that analyses in scenario II incorrectly labeled as successful had a tendency to overestimate the admixture time in simulations where the admixture time was anywhere between 120 to 220, with the exception of that of 180, where some underestimation occured. Scenario I only showed somewhat consistant tendencies to overestimate admixture time at 220. There may be some bias to overestimate the admixture time in scenario II at certain times where it does not occur for scenario I, and this may suggest a fault that occurs when lowering the sample size.

In graph D for fig. 6 and fig. 7, the rate of correctly labeled successful runs show that the performance of ALDER in scenario II is actually significantly better than that in scenario I at many points, and equal in some others. This is surprising, as scenario I contains much more information and ALDER should be expected to perform better than in scenario II. While ALDER did consistantly give a smaller interval width for scenario I than II, it really should provide better output

considering how much more information there is available. There should be no reasons for such a poor performance given the circumstances.

Graph C in fig. 6 and fig. 7 shows the ratio of true positives among all positives, and also shows that if anything, the performance of ALDER is better in scenario II than scenario I. The lower performance in graph D perhaps could have been justified in scenario I if more reported positives had been true positives, but that is definitely not the case either. It would be very interesting to know of the reason why this may be, but there is no clear way to determine it.

In summary, the only information loss that was observed from lowering the sample amount was that the given interval width increased slightly. Strangely, there was also a big increase in the performance of ALDER as the sample size was lowered, for unknown reasons. The simulation and analysis of scenario I was done after having altered the source code to allow input of fewer samples than permitted by default, which may have caused this strange this output. This seems unlikely when looking at the source code at a glance, but it's even more unlikely that providing more information would create worse output.

4.1.3 ALDER performance in different low quality scenarios

As a successful method of preparing pseudo-diploid individuals similar to that in ancient DNA studies was developed, further investigations could be performed on ALDERs capability of analyzing even lower quality data. By comparing the results between scenario II and III, it is possible to see the changes in performance occuring when using pseudo-diploid data compared to having full allele knowledge.

From graph F in fig. 6 and fig. 7, the interval width increases significantly with larger admixture times, and is increased even more with lower retained SNPs. For a coverage of 0.16, the suggested window from ALDER is on average 100 generations wide for an admixture time of 220. While it does technically result in successful output, the usefulness is questionable.

Graph D in fig. 8 shows that the accuracy of ALDER decays at certain points depending on the fraction of retained SNPs. While the performance seems to be consistant for a coverage of 0.16 and above, it starts dropping earlier with lower coverage. This effect was not at all observed for scenario II, and suggests that in cases where only one allele is recovered for the known positions, the coverage needs to be higher to determine admixture time the longer ago the actual admixture occurred.

Graph C in fig. 8 shows how the ratio of true positives starts lowering for a coverage of 0.0256 at an admixture time of 180. This means that even in the rare chance that an analysis is labeled as successful, it is unlikely to be correct.

Graphs A and B in fig. 8 show that there are many anomalies occuring for low coverage data at large admixture times. This is probably due to the low amount of samples for these data points, simply being statistical noise.

This initial comparison between scenario II and III shows that there is quite a bit of data loss occuring as pseudo-diploid data is constructed and analyzed in ALDER. The interval widths given get significantly higher with higher admixture times, and the accuracy when using low coverage data diminishes with higher admixture times. There could be a small bias to overestimate the admixture time as well.

4.1.4 ALDER analysis of varying low quality scenarios

Using scenario III as a basis, it is possible to see how variations of the parameters used in that scenario change the performance of ALDER. Scenario IV uses the same parameters apart from having an increased admixed population size, from 5000 in scenario III to 10000 in scenario IV. The results from ALDER is very similar between these two scenarios. The accuracy of low coverage

analysis gets lower with increasing admixture time, and the given interval widths are similarly sized at most points as well. There are some anomalies, most likely caused by noise.

Scenario V is performed in a similar way as scenario IV, but rather having a reduced admixed population size of 2000 compared to 5000 in scenario III. The similarities are once again striking between these two scenarios, where there is a reduction in performance with lower coverage for higher admixture times. The interval widths are similar, and anomalies are likely caused by noise.

Scenario VI uses the same parameters as scenario III, with the exception of having a migration fraction of 0.15 rather than 0.4 from source population 2. This caused ALDER to perform significantly worse than in the previous three low quality scenarios. The success rate with a 0.0256 coverage drops below 50 % from generation time 65 and onwards. The success rate with 0.064 coverage drops earlier than in the previous scenarios, and a coverage of 0.16 faces issues at later admixture times as well. The interval widths grow even larger, especially for large admixture times and low coverage. There are many anomalies among incorrectly labeled successful analyses, even where the statistical basis is good, where there seems to be a bias to overestimate the admixture time.

The results do suggest that ALDER is very capable even when reducing the data to the level explored in this project. While unlikely in reality, having full coverage, or even just a coverage of 0.4, would pass all the ALDER analyses very well. However, as the analysis has not worked when supplying ALDER with missing SNPs, this requires there to be an overlap of discovered SNPs of 40 % to reach the coverage as described in this project. Finally, if the admixture time is low, coverages as low as 0.0256 or possibly even lower appear to work for the purposes of ALDER.

4.2 Future prospects

4.2.1 Applying ALDER to ancient DNA studies

The results found in this project suggest that ALDER is indeed capable for ancient DNA analysis, at least if the scenarios are simple and the coverage is high enough. There are three main parameters that seem to influence if ALDER will be successful in its analysis or not: Time since admixture, fraction of retained SNPs, and complexity of the scenario. In all scenarios studied in this project, retaining as much as 40 % of all occuring SNPs gives a very reliable basis for analysis in ALDER. If the coverage is lower, there is a larger chance of the analysis being successful if the admixture occured closer in time. While the scenarios simulated in this project may be too simplified to accurately judge if ALDER will be successful in the case of ancient DNA studies, this project has determined that ALDER is certainly capable of working with data of this kind.

4.2.2 Further studies of advanced admixture scenarios

This project has displayed the capabilities of ALDER in simple, low data quality settings. The project has not explored how the performance of ALDER is in more complex scenarios, nor potential performance improvements from using more samples. Investigating ALDER performance in these scenarios will allow to further judge the accuracy and reliability of ALDER in actual ancient DNA studies.

Nearly all parts of the workflow pipeline are automated, and only a minimum of manual input is required. The largest issue when performing these tests are the computer resources needed. Using 4 cores on the UPPMAX cluster requires roughly 5 to 10 minutes of computing work per simulation. This requires a large amount of computational work to create a solid statistical basis for estimations of performance. However, as the performance of ALDER appears promising for ancient DNA studies, this is time well spent.

4.2.3 Additional evaluation metrics

There are a few additional metrics that would probably be of interest when measuring ALDER performance that were not included in this project. Such metrics would include measuring the average error for successful runs in more qualitative ways, as the current measure of success only involve a simple yes or no for whether or not the true admixture time is included in the given interval. Including metrics of the average error, potentially based on the interval width given is likely to give a deeper measure of accuracy, and a better sense of ALDER performance.

4.3 Conclusions and concluding remarks

The aim of this project was to investigate means of estimating admixture times using a very reduced set of data. The success of the investigations allowed further studies, looking into how reduced the data may be before not producing any valuable output any longer.

The results from using ALDER to estimate admixture times have continuously been very positive and encouraging. During most simulations ALDER has proven to estimate the true admixture times in 80 % of the cases, when having over 6.4 % of SNPs intact. The performance gets worse at lower generation times when using lower coverage. Analysing the output from ALDER has shown that there are no biases in over- or underestimating the admixture times. Had there been such a bias, this could potentially have been accounted for and adjusted accordingly.

ALDER has shown that it is capable of estimating admixture times even when reducing the amount of samples below what is originally possible in the software. The estimations also remained correct when cutting down the amount of SNPs for analysis, as well as when retaining only one nucleotide in each locus. ALDER does get less specific as the data quality lowers, but that is only to be expected. The output itself does give the correct intervals in most of the cases, and it will be a balancing act to consider whether or not the interval is too big to be of interest.

There are a couple of shortcomings when using ALDER in ancient DNA admixture analysis. While it did turn out to be possible to greatly reduce the amount of needed samples below what is required by default, there is still a requirement of providing two sampled individuals from the admixed population. Ideally, there would be a need of only one, although diploid data would then be necessary to calculate the LD. There may be a possibility reduce this further, but it has not been found. Furthermore, it would be very useful if it was possible to actually have missing nucleotides in the ALDER analysis data. This is something that will probably be prevalent in ancient DNA, but there may be other ways to get around this issue. Otherwise, sites with missing information in any individual will have to be excluded in this analysis, which is wasting information that could be useful. When trying to use ALDER with only one source population, there is still a requirement for 4 individuals from each population, which makes this functionality of less interest in ancient DNA projects. It might be possible to change this though, as it was with the two source populations scenario.

From this project ALDER seems very capable of analysing ancient DNA, even when reducing the amount of SNPs and removing knowledge of both nucleotides in each locus. There are some investigations that could be of interest, trying out more advanced samples than those tried in this project. Additionally, it would be of interest to make an in-depth study of how much information is gained from having more samples. The scripts created for this project are all capable of doing those studies, and all that is needed is more computer time to actually run those analyses.
Further investigations: gain of accuracy by increasing samples, accuracy in advanced scenarios. Tools are there, only computer time is needed.

Acknowledgements

I wish to thank my supervisor, Torsten Günther, for providing very helpful support, input, encouragement and ideas during the entire course of the project. I wish to thank my scientific reviewer, Martin Lascoux, for being very encouraging and providing a lot of good input on the report.

Most of all I wish to thank Linnea, for being a mountain of support, encouragement and help throughout the entire project. Ths project would not have been possible without you.

References

1. Brown, T.A, (2006) *Genomics*. 3rd ed. New York: Garland Science Publishing
2. Busby, G.B.J. et al. (2015), **The Role of Recent Admixture in Forming the Contemporary West Eurasian Genomic Landscape** *Current Biology*, vol. 25, pp. 2518-2526; doi:10.1016/j.cub.2015.08.007
3. Dawson, E. et al. (2002) **A First-Generation Linkage Disequilibrium Map of Human Chromosome 22** *Nature* vol. 418, pp. 544-548; doi:10.1038/nature00864
4. Excoffier, L., Dupanloup, I., Huerta-Sánchez, E., Sousa, V.C., Foll, M. 2013. **Robust Demographic Inference from Genomic and SNP data.** *PLOS Genetics,* vol. 9; doi:10.1371/journal.pgen.1003905
5. Forster, P. (2004) **Ice Ages and the Mitochondrial DNA Chronology of Human Dispersals: a Review** *Philosophical Transactions of the Royal Society B: Biological Sciences,* vol. 359, pp 255-264; doi:10.1098/rstb.2003.1394.
6. Groucutt, H.S. et al. (2015), **Rethinking the Dispersal of Homo Sapiens out of Africa.** *Evolutionary Anthropology* vol. 24, pp 149–164. doi:10.1002/evan.21455
7. Handt, O. et al. (1994) **Ancient DNA: Methodological Challenges**, *Experientia* vol. 50, pp. 524-529, doi:10.1007/BF01921720
8. Loh, P.R et al. (2013) **Inferring Admixture Histories of Human Populations Using Linkage Disequilibrium**. *Genetics* 2013 vol. 193, pp. 1233-1254; doi:10.1534/genetics.112.147330
9. Lipson M. et al. (2015) **Calibrating the Human Mutation Rate via Ancestral Recombination Density in Diploid Genomes**. *PLoS Genetics* vol. 11; doi:10.1371/journal.pgen.1005550
10. Patterson, N. et al. (2012) **Ancient Admixture in Human History**. *Genetics* vol. 192, pp. 1065-1093; doi:10.1534/genetics.112.145037
11. Sankararaman S, Patterson N, Li H, Pääbo S, Reich D (2012) **The Date of Interbreeding between Neandertals and Modern Humans**. *PLoS Genetics*, vol. 8; doi:10.1371/journal.pgen.1002947
12. Skoglund P, et al. (2012) **Origins and Genetic Legacy of Neolithic Farmers and Hunter-Gatherers in Europe**. *Science* vol. 336, pp. 466–469; doi:10.1126/science.1216304
13. Skoglund P, et al. (2014) **Genomic Diversity and Admixture Differs for Stone-Age Scandinavian Foragers and Farmers**. *Science* vol. 344, pp. 747–750; doi:10.1126/science.1253448
14. Slatkin, M. (2008) **Linkage Disequilibrium — Understanding the Evolutionary Past and Mapping the Medical Future**. *Nature Reviews Genetics* vol. 9, pp. 477-485; doi:10.1038/nrg2361
15. Zeder, M.A., Emshwiller, E., Smith, B.D. & Bradley, D.G. (2006), **Documenting Domestication: the Intersection of Genetics and Archaeology**, *Trends in Genetics* vol. 22, pp. 139-155; doi:10.1016/j.tig.2006.01.007

Appendix

Appendix A

Script files used in workflow pipeline

runfullscratchshuffle.sh

```
#!/bin/bash
echo -e "Enter input filename: \c "
read filename
echo -e "How long should the program run for? \c "
read runtimeone
echo -e "How many cores should the program use? \c "
read cores
echo -e "What fraction of mutations should be kept? \c "
read mutremove
echo -e "How many fractionloops? \c "
read fractionloops
cp Miniloopscratchshuffle.sh Miniloopfull_${filename}.sh
perl bashfullparser.pl ${filename}  ${runtimeone}  ${cores} ${mutremove} $
{fractionloops}
sbatch Miniloopfull_${filename}.sh
```

bashfullparser.pl

```
use strict;
use warnings;
use Tie::File;



my $filename = $ARGV[0] or die;
my $runtimeone = $ARGV[1] or die;
my $cores = $ARGV[2] or die;
my $mutremove = $ARGV[3] or die;
my $loops = $ARGV[4] or die;
my $numofchr;

tie my @array, 'Tie::File', "$filename.tpl" or die;
my $bashline = "";
for my $i (0 .. $#array){
      $bashline = $array[$i];
      if($bashline =~ "\/\/Number of independent"){
            $numofchr = (split '\s', $array[$i+1])[0];
            print "OMGSSSS $array[$i+1]\n $numofchr";
```

```
	}
}
untie @array;
tie @array, 'Tie::File', "Miniloopfull_$filename.sh" or die;
$bashline = "";
for my $i (0 .. $#array){

	$bashline = $array[$i];
	if($bashline =~ "Fastsimcoal2/fsc_linux64/./fsc252"){
		#$array[$i] = "Fastsimcoal2/fsc_linux64/./fsc252 -t ${filename}.tpl
-e ${filename}.est -E${numest}  -n${numsim} -g -q -c1 -B$numofchr";

		$array[$i] = "Fastsimcoal2/fsc_linux64/./fsc252 -t ${filename}.tpl
-f ${filename}.def -n1 -g -q -c$cores -B$numofchr";
	}
	if($bashline =~ "filename="){
		$array[$i] = "filename=$filename";
	}
	if($bashline =~ "#SBATCH -t"){
		$array[$i] = "#SBATCH -t $runtimeone";
	}
	if($bashline =~ "mutremove="){
		$array[$i] = "mutremove=$mutremove";
	}
	if($bashline =~ "loops="){
		$array[$i] = "loops=$loops";
	}
	if($bashline =~ "cores="){
		$array[$i] = "cores=$cores";
	}
	if($bashline =~ "#SBATCH -n"){
		$array[$i] = "#SBATCH -n $cores";
	}
}
untie @array;
```

Miniloopscratchshuffle.sh

```
#!/bin/bash

#SBATCH -A b2015168
#SBATCH -p core
#SBATCH -n 2
#SBATCH -t 30:00
```

```
#SBATCH -J SmallRuns


filename=mixfixtest
numsim=1
numest=1
mutremove=0.40000
loops=4
mkdir -p /proj/b2015168/Alder_${filename}_0/
rm /proj/b2015168/Alder_${filename}/alder_log_summary_${filename}.log
Fastsimcoal2/fsc_linux64/./fsc252 -t mixfixtest.tpl -f mixfixtest.def -n1 -g -q
-c2 -B20
files=($(ls -v ${filename}/*.arp))
i=0
filename=mixfixtest
cp alderparam.par alderparam_${filename}.par
cores=2
i=0
loops=4
sed -i '6s|.*|numthreads:\t1|' alderparam_${filename}.par
task(){
      local num=$(($i))
      local mutloop=$mutremove
      date +%c
      RET1=$(perl check.pl $f)
      if [ $RET1 = 1 ]
      then
            #### REFER TO THE FILE ap.pl RATHER THAN apded.pl IF YOU WISH TO USE
INFORMATION OF BOTH ALLELES RATHER THAN JUST ONE.
            perl apded.pl $f $TMPDIR/${filename}_${num}_0_0.geno $TMPDIR/$
{filename}_${num}_0_0.snp $TMPDIR/${filename}_${num}_0.ind
            date +%c
            local j=1
            while [ $j -le ${loops} ]
            do
                  perl removemuts.pl $TMPDIR/${filename}_${num}_0_0.geno
$TMPDIR/${filename}_${num}_0_0.snp $TMPDIR/${filename}_${num}_${j}_0.geno
$TMPDIR/${filename}_${num}_${j}_0.snp ${mutloop}
                  local mutloop=$(echo $mutloop*$mutremove | bc)
                  local j=$(($j+1))
            done
      fi
}
      for f in "${files[@]}"; do
            i=$(($i+1))
```

```
            task "$f" &
      echo -e "Running task $i"
      if [ $(($i % $cores)) = 0 ]
      then
            wait
      fi

done
wait

num=0
for f in "${files[@]}"; do
      j=0
      num=$(($num+1))

      while [ $j -le ${loops} ]
      do
            date +%c
            sed -i '3s|.*|indivname:\t'$TMPDIR'/'${filename}_${num}_0'.ind|'
alderparam_${filename}.par
            sed -i '1s|.*|genotypename:\t'$TMPDIR'/'${filename}_${num}_$
{j}_0'.geno|' alderparam_${filename}.par
            sed -i '2s|.*|snpname:\t'$TMPDIR'/'${filename}_${num}_${j}_0'.snp|'
alderparam_${filename}.par
            sed -i '4s/.*/admixpop:\tSample_2/'  alderparam_${filename}.par
            sed -i '5s/.*/refpops:\tSample_1;Sample_3/'   alderparam_$
{filename}.par
            alder/./alder -p alderparam_${filename}.par > /proj/b2015168/Alder_$
{filename}_0/alder_log_${filename}_${num}_${j}.log
            date +%c
            j=$(($j+1))
      done
      j=0
      let currPar=($num+$numsim-1)/$numsim
      perl appendinfo.pl ${filename}.def /proj/b2015168/Alder_$
{filename}_0/alder_log_summary_${filename}.log  ${currPar}
      while [ $j -le ${loops} ]
      do
            perl minieval.pl /proj/b2015168/Alder_${filename}_0/alder_log_$
{filename}_${num}_${j}.log  /proj/b2015168/Alder_$
{filename}_0/alder_log_summary_${filename}.log
            echo >> /proj/b2015168/Alder_${filename}_0/alder_log_summary_$
{filename}.log
            j=$(($j+1))
      done
```

```
done

rm alderparam_${filename}.par

echo -e "All analysis is completed. Alder output is available in Alder_$
{filename}/ . in Script ending at time: "

date +%c
```

ap.pl

```
#use strict;
use warnings;
use File::Path;
use Tie::File;
use POSIX;
use Time::HiRes qw/ time sleep /;

my $morgan = (1/(1.3e-8));
my @arlstring;
my @mutposarray;
my @chrpos;
my @geno;
my @samples;
my @references;
my @variants;
my @tmparray;
my $i = 0;
my $j = 0;
my $length = 0;
my $chr = 0;
my $readytoread = 0;
my $arlline;
my $arllinetwo;
my $firstgeno = 1;
my @genoarray;
my $c;
my $d;
open my $arl, $ARGV[0] or die;
#Parsing .arl-file, building .geno-file.
my $start = time;
my $scorestring;
while ($arlline = <$arl>) {
#     $i++;
#     print "$i\n";
      if($arlline =~ /# \d/) {
            @arlstring = split(' ', $arlline);
            push @chrpos, $arlstring[1];
                              # is launched if it's the line following a
chromosome declaration in the .arl-file.
                              # Setting readytoread to 0 means anotheri chr-
declaration needs to be found before entering this block again.
                              # This block is only reached after a chr-
declaration has been found, which is written in a block further down.
            $arlline = <$arl>;
            chomp($arlline);
            @arlstring = split(', ', $arlline);       # splits the line input
down into every single chromosome position
            chomp(@arlstring);                  # where mutations have occured,
and removes
            $arlstring[0] =~ s/\W//g;           # extra characters and newlines.
            #code for reading/writing .snp-file:
            for my $k (0 .. $#arlstring){
                  push @mutposarray, $arlstring[$k];
            }
      }
```

```
	elsif($arlline =~ /SampleSize=/){
		$arlline =~ s/SampleSize=//g;
		push @samples, $arlline;
	}
	elsif($arlline =~ /\d\_\d/){
		$scorestring = "";
		$arllinetwo = <$arl>;
		chomp($arlline);
		chomp($arllinetwo);
		$arlline = (split /\t/, $arlline)[2];
		$arllinetwo = (split /\t/, $arllinetwo)[2];
		if($firstgeno == 1){
			$firstgeno = 0;
			@references[length($arlline) - 1] = "X";
			@references = split('', $arlline);
			@variants = ('X')x$#references;
			while(my $c = chop $arllinetwo){
				$length = length($arllinetwo);
				if($c =~ $references[$length]){
					$scorestring = $scorestring . "2";
				}
				else{
					$scorestring = $scorestring . "1";
					$variants[$length] = $c;
				}
			}
		}
		else{
			while(my $c = chop $arllinetwo){
				my $d = chop $arlline;
				$length = length($arllinetwo);
				if($c =~ $references[$length]){
					if($d =~ $c){
						$scorestring = $scorestring . "2";
					}
					else{
						$scorestring = $scorestring . "1";
						$variants[$length] = $d;
					}
				}
				elsif($d =~ $references[$length]){
					$scorestring = $scorestring . "1";
					$variants[$length] = $c;
				}
				else{
					$scorestring = $scorestring . "0";
					$variants[$length] = $c;
				}
			}

		}
		push @genoarray, $scorestring;
	}
}
my $end   = time;
print 'Parsed arlfile in: ', ( $end - $start ) , "\n";

print "$#references\t$#variants\t$#mutposarray\n";
close $arl;

$start = time;
open(my $genooutput, '>' , $ARGV[1]);
while (length($genoarray[0]) > 0){
	$outline = "";
```

```
      for my $j (0 .. $#genoarray){
            $c = chop $genoarray[$j];
            $outline = $outline . $c;
      }
      print $genooutput $outline . "\n";
}
$end   = time;
print 'Transposed genofile in:  ', ( $end - $start ) , "\n";
close $genooutput;


#Building .ind-file.
open(my $indoutput, '>' , $ARGV[3]);
for my $i (1 .. ($#samples + 1)){
      for my $j (1 .. $samples[$i - 1]){
            print $indoutput "$i\_$j\_Sample\_$i\tU\tSample\_$i\n";
      }
}
close $indoutput;

#Building .snp-file.
open(my $snpoutput, '>' , $ARGV[2]);
my $k = 0;
$j = 0;

print "$#references\t$#variants\t$#mutposarray\n";

my $snpstring;
for my $i (0 .. $#mutposarray){
      if($j > ($chrpos[$k] - 1)){
            $k++;
            $j = 0;
      }
      $j++;
      #print $snpoutput "locus_" . ($i + 1) . "\t" . ($k + 1) . "\t" .
$mutposarray[$i]/130000000 .
"\t$mutposarray[$i]\t$references[$i]\t$variants[$i]\n";

      $snpstring = "locus_" . ($i + 1) . "\t" . ($k + 1) . "\t" .
$mutposarray[$i]/$morgan .
"\t$mutposarray[$i]\t$references[$i]\t$variants[$i]\n";
      print $snpoutput $snpstring;
}
close $snpoutput;
```

apded.pl

```
#use strict;
use warnings;
use File::Path;
use Tie::File;
use POSIX;
use Time::HiRes qw/ time sleep /;

my $morgan = (1/(1.3e-8));
my @arlstring;
my @mutposarray;
my @chrpos;
my @geno;
my @samples;
my @references;
my @variants;
```

```
my @tmparray;
my $i = 0;
my $j = 0;
my $length = 0;
my $chr = 0;
my $readytoread = 0;
my $arlline;
my $arllinetwo;
my $firstgeno = 1;
my @genoarray;
my $c;
my $d;
open my $arl, $ARGV[0] or die;
#Parsing .arl-file, building .geno-file.
my $start = time;
my $scorestring;
while ($arlline = <$arl>) {
#     $i++;
#     print "$i\n";
      if($arlline =~ /# \d/) {
            @arlstring = split(' ', $arlline);
            push @chrpos, $arlstring[1];
                              # is launched if it's the line following a
chromosome declaration in the .arl-file.
                              # Setting readytoread to 0 means anotheri chr-
declaration needs to be found before entering this block again.
                              # This block is only reached after a chr-
declaration has been found, which is written in a block further down.
            $arlline = <$arl>;
            chomp($arlline);
            @arlstring = split(', ', $arlline);       # splits the line input
down into every single chromosome position
            chomp(@arlstring);                  # where mutations have occured,
and removes
            $arlstring[0] =~ s/\W//g;           # extra characters and newlines.
            #code for reading/writing .snp-file:
            for my $k (0 .. $#arlstring){
                  push @mutposarray, $arlstring[$k];
            }
      }
      elsif($arlline =~ /SampleSize=/){
            $arlline =~ s/SampleSize=//g;
            push @samples, $arlline;
      }
      elsif($arlline =~ /\d\_\d/){
            $scorestring = "";
            $arllinetwo = <$arl>;
            my $templine = "";
            $arlline = (split /\t/, $arlline)[2];
            $arllinetwo = (split /\t/, $arllinetwo)[2];
            while (length($arlline) > 0){
                  $c = chop $arlline;
                  $d = chop $arllinetwo;
                  if(rand() > 0.5){
                        $templine = $templine . $c;
                  }
                  else{
                        $templine = $templine . $d;
                  }
            }
            $arlline = reverse($templine);
            $templine = "";

            chomp($arlline);
```

```
            if($firstgeno == 1){    ## When encountering the first row of
nucleotide reads, it's used to get a reference sequence.

                  $firstgeno = 0;
                  $scorestring = '2'x(length($arlline));
                  @references[length($arlline) - 1] = "X";
                  @references = split('', $arlline);
                  @variants = ('X')x($#references+1); ## The variant sequence is
initialized as X in every position, but should be replaced with actual
nucleotides everywhere as the script proceeds.
            }
            else{
                  while(my $c = chop $arlline){
                        if($c =~ $references[$length]){
                              $scorestring = $scorestring . "2";
                        }
                        else{
                              $scorestring = $scorestring . "0";
                              $variants[$length] = $c;
                        }
                  }

            }
            push @genoarray, $scorestring; # The scorestring used for storing
matching data is actually reversed, due to the chop command working from the end
to the beginning. However, the chop command is used again when writing to the
.geno-file, undoing the reverse.
      }
}
print length($genoarray[0]) . ' ' . $#references . ' ' . $#variants . '\n';
my $end   = time;
print 'Parsed arlfile in: ', ( $end - $start ) , "\n";
close $arl;


#Building .geno-file.
$start = time;
open(my $genooutput, ">", $ARGV[1]) or die "Couldn't open: $!";
#open(my $genooutput, '>' , $ARGV[1]);
while (length($genoarray[0]) > 0){
      $outline = "";
      for my $j (0 .. $#genoarray){
            $c = chop $genoarray[$j];
            $outline = $outline . $c;
      }
      print $genooutput $outline . "\n";
}
$end   = time;
print 'Wrote genofile in:  ', ( $end - $start ) , "\n";
close $genooutput;


#Building .ind-file.
open(my $indoutput, '>' , $ARGV[3]);
for my $i (1 .. ($#samples + 1)){
      for my $j (1 .. $samples[$i - 1]){
            print $indoutput "$i\_$j\_Sample\_$i\tU\tSample\_$i\n";
      }
}
close $indoutput;

#Building .snp-file.
open(my $snpoutput, '>' , $ARGV[2]);
my $k = 0;
```

```
$j = 0;
my $snpstring;
for my $i (0 .. $#mutposarray){
	if($j > ($chrpos[$k] - 1)){
		$k++;
		$j = 0;
	}
	$j++;

	$snpstring = "locus_" . ($i + 1) . "\t" . ($k + 1) . "\t" .
$mutposarray[$i]/$morgan .
"\t$mutposarray[$i]\t$references[$i]\t$variants[$i]\n";
	print $snpoutput $snpstring;
}
close $snpoutput;
```

removemuts.pl

```
use strict;
use warnings;

open my $ingeno, $ARGV[0] or die;
open my $insnp, $ARGV[1] or die;
open(my $outgeno, '>' , $ARGV[2]);
open(my $outsnp, '>' , $ARGV[3]);
my $keep = $ARGV[4];
while (my $genoline = <$ingeno>){
	my $snpline = <$insnp>;
	if(rand() < $keep){
		print $outgeno $genoline;
		print $outsnp $snpline;
	}
}
```

minieval.pl

```
open my $bothpop, $ARGV[0] or die;
open(my $output, '>>' , $ARGV[1]);
my $bothpops;
my @bothpopstring;
while ($bothpops = <$bothpop>) {
	if($bothpops =~ /DATA:/) {
		@bothpopstring = split('\t', $bothpops);
		if($bothpopstring[1] =~ "success"){
			if($bothpopstring[2] == 0){ #bothpopstring[2] contains the p-
value, which is 0 if there is too much long-range LD.
				print $output "Test failed miserably, no data to
show.\n";
				while ($bothpops = <$bothpop>){}
			}
			else{
				print $output "Test succeeded" .
checkWarnings($bothpopstring[1]) . "> $bothpopstring[2]\t$bothpopstring[10]\n";
			}
		}
		if($bothpopstring[1] =~ "failure"){
			print $output "Test failed" . checkWarnings($bothpopstring[1])
. "> $bothpopstring[2]\t$bothpopstring[10]\n";
		}
	}
	elsif($bothpops =~ /fit start = inf/) {
		print $output "Test failed miserably, no data to show.\n";
	}
}
```

```
sub checkWarnings{
      my $outputstring = '';
      if($_[0] =~ /warning/){
            $outputstring = ", with inconsistent decay:\n";
      }
      else{
            $outputstring = ":\n";
      }
      return $outputstring
}
```

appendinfo.pl

```
use POSIX qw(ceil);
use Tie::File;

my $file = $ARGV[0];
tie my @arr, 'Tie::File', $file;
#open my $parfile, $ARGV[0] or die;
open(my $output, '>>' , $ARGV[1]);
my $currPar = $ARGV[2];
my @stringone = split('\t', $arr[0]);
my $stringtwos = $arr[$currPar];
my @stringtwo = split('\t', $stringtwos);
for my $i (0 .. $#stringtwo){
      print $output "$stringone[$i]: $stringtwo[$i]\t";
}
print $output "\n";
```

Script files used for visualization

alder_fullreport.pl

```
use strict;
use warnings;
use File::Path;
use Tie::File;
use POSIX;


open my $summary, $ARGV[0] or die;
open (my $output, '>', $ARGV[1]) or die;
my $mutremove = $ARGV[2] or die;
my $truevalue = 0;
my $loopcount = 0;
my @arr;
my $inline;
my $i = 0;
while (my $inline = <$summary>) {
      if($inline =~ "Test"){
            my $print = 0;
            my $outline = "";
            $i++;
            $outline = $mutremove**($i-1) . " ";
```

```
            $outline = $outline . "$truevalue ";
            if($inline =~ "Test succeeded"){
                  $inline = <$summary>;
                  @arr = split('\s+', $inline);
                  if(($truevalue >= ($arr[2] - $arr[4])) && ($truevalue <=
($arr[2] + $arr[4]))){
                        $outline = $outline . "1 $arr[2] $arr[4] ";
                  }
                  else{
                        $outline = $outline . "2 $arr[2] $arr[4] ";
                  }
            }
            elsif($inline =~ "Test failed"){
                  $inline = <$summary>;
                  $outline = $outline . "3 500 500 ";
            }
            else{
                  my $inline = <$summary>;
            }
            print $output "$outline\n";
            my $inline = <$summary>;
      }
      else{
            $i = 0;
            @arr = split('\s+', $inline);
            $truevalue = $arr[5];
      }
}
```

Plotsummary.r

```
fname = "fullreport_mixcopybigtest";
mytable <- read.table(paste(fname, ".output", sep=""));
max <- length(mytable[(mytable$V1==mytable$V1[1] &
mytable$V2==mytable$V2[1]),1])
mytabletrue <- mytable[mytable[,3]=="1",]
mytablefalse <- mytable[mytable[,3]=="2",]
mytabletruefalse <- data.frame(table(mytable[c(-4,-5)]))
mytabletff <- mytabletruefalse[mytabletruefalse[,3]=="2",]
mytabletft <- mytabletruefalse[mytabletruefalse[,3]=="1",]




mytabletrue <- mytabletrue[c(-3)]
```

```
mytablefalse <- mytablefalse[c(-3)]

mytabletruediff <- mytabletrue[c(-4)]

mytabletruewidth <- mytabletrue[c(-3)]

mytablefalsediff <- mytablefalse[c(-4)]

mytablefalsewidth <- mytablefalse[c(-3)]

meanvaluestruediff <-
aggregate(mytabletruediff[,3],list(mytabletruediff[,2],mytabletruediff[,1]),mean
)

meanvaluestruewidth <-
aggregate(mytabletruewidth[,3],list(mytabletruewidth[,2],mytabletruewidth[,1]),m
ean)

meanvaluesfalsediff <-
aggregate(mytablefalsediff[,3],list(mytablefalsediff[,2],mytablefalsediff[,1]),m
ean)

meanvaluesfalsewidth <-
aggregate(mytablefalsewidth[,3],list(mytablefalsewidth[,2],mytablefalsewidth[,1]
),mean)


################ This block is used for plotting average ALDER estimated
generation time for successfully labeled analyses within the correct range

meanvalues <- meanvaluestruediff;

cov <- as.numeric(as.character(meanvalues[,2]))


#png(paste(paste("Plot_alderdifftrue_", fname, sep=""), ".png", sep=""))

#par(cex.main=c(2),cex.sub=c(2),ps=c(50))

png(file=paste(paste("Testplot_alderdifftrue_", fname, sep=""), ".png", sep=""),
units="in", width=6.0, height=6.0, res=60)

#png(file=paste(paste("Testplot_alderdifftrue_", fname, sep=""), ".png",
sep=""))


#png(file="mag_feb.png", units="in", width=7.2, height=7.2, res=300)

plot(meanvalues[1][meanvalues[2]==cov[1],], meanvalues[1]
[meanvalues[2]==cov[1],], col=1, type="l", lty=2, xlim=range(mytable$V2),
ylim=c(min(meanvalues[3],meanvalues[1]),max(meanvalues[3],meanvalues[1])),
xlab="Generations", ylab="Average ALDER estimated generation time",cex.lab=1.5)



for (i in 1:length(unique(cov))){

      lines(meanvalues[1][meanvalues[2]==unique(cov)[i],], meanvalues[3]
[meanvalues[2]==unique(cov)[i]], col=i, type="b")

}


legend(min(mytable$V2),max(meanvalues[3]), unique(cov), title="Coverage",
lty=rep(c(1), each=length(unique(cov))), lwd=rep(c(1),
each=length(unique(cov))),,col=c(1:(length(unique(cov)))),cex=1.3)

dev.off()

##########################################
```

```
################ This block is used for plotting average ALDER estimated
generation time for successfully labeled analyses outside of the correct range

meanvalues <- meanvaluesfalsediff;

cov <- as.numeric(as.character(meanvalues[,2]))


png(file=paste(paste("Testplot_alderdifffalse_", fname, sep=""), ".png",
sep=""), units="in", width=6.0, height=6.0, res=60)

#png(paste(paste("Plot_alderdifffalse_", fname, sep=""), ".png", sep=""))


plot(meanvalues[1][meanvalues[2]==cov[1],], meanvalues[1]
[meanvalues[2]==cov[1],], col=1, type="l", lty=2, pch="-",
xlim=range(mytable$V2),
ylim=c(min(meanvalues[3],meanvalues[1]),max(meanvalues[3],meanvalues[1])),
xlab="Generations", ylab="Average ALDER estimated generation time",cex.lab=1.5)


for (i in 1:length(unique(cov))){

      lines(meanvalues[1][meanvalues[2]==unique(cov)[i],], meanvalues[3]
[meanvalues[2]==unique(cov)[i]], col=i, type="b")

}


legend(min(mytable$V2),max(meanvalues[3]), unique(cov), title="Coverage",
lty=rep(c(1), each=length(unique(cov))), lwd=rep(c(1),
each=length(unique(cov))),,col=c(1:(length(unique(cov)))),cex=1.3)

dev.off()

##########################################


################ This block is used for plotting average ALDER estimated
interval width for successfully labeled analyses within the correct range

meanvalues <- meanvaluestruewidth;

cov <- as.numeric(as.character(meanvalues[,2]))


png(file=paste(paste("Testplot_alderwidthtrue_", fname, sep=""), ".png",
sep=""), units="in", width=6.0, height=6.0, res=60)

#png(paste(paste("Plot_alderwidthtrue_", fname, sep=""), ".png", sep=""))


plot(meanvalues[1][meanvalues[2]==unique(cov)[1],], meanvalues[3]
[meanvalues[2]==unique(cov)[1]], col=1, type="b", xlim=range(mytable$V2),
ylim=c(0,max(meanvalues[3])), xlab="Generations", ylab="Average interval length
from ALDER",cex.lab=1.5)



for (i in 2:length(unique(cov))){

      lines(meanvalues[1][meanvalues[2]==unique(cov)[i],], meanvalues[3]
[meanvalues[2]==unique(cov)[i]], col=i, type="b")

}


legend(min(mytable$V2),max(meanvalues[3]), unique(cov), title="Coverage",
```

```
lty=rep(c(1), each=length(unique(cov))), lwd=rep(c(1),
each=length(unique(cov))),,col=c(1:(length(unique(cov)))),cex=1.3)

dev.off()

########################################


################ This block is used for plotting average ALDER estimated
interval width for successfully labeled analyses outside of the correct range

meanvalues <- meanvaluesfalsewidth;

cov <- as.numeric(as.character(meanvalues[,2]))


png(file=paste(paste("Testplot_alderwidthfalse_", fname, sep=""), ".png",
sep=""), units="in", width=6.0, height=6.0, res=60)

#png(paste(paste("Plot_alderwidthfalse_", fname, sep=""), ".png", sep=""))


plot(meanvalues[1][meanvalues[2]==unique(cov)[1],], meanvalues[3]
[meanvalues[2]==unique(cov)[1]], col=1, type="b", xlim=range(mytable$V2),
ylim=c(0,max(meanvalues[3])), xlab="Generations", ylab="Average interval length
from ALDER",cex.lab=1.5)



for (i in 2:length(unique(cov))){

      lines(meanvalues[1][meanvalues[2]==unique(cov)[i],], meanvalues[3]
[meanvalues[2]==unique(cov)[i]], col=i, type="b")

}


legend(min(mytable$V2),max(meanvalues[3]), unique(cov), title="Coverage",
lty=rep(c(1), each=length(unique(cov))), lwd=rep(c(1),
each=length(unique(cov))),,col=c(1:(length(unique(cov)))),cex=1.3)

dev.off()

########################################

################ This block is used for plotting ALDER success rate out of
analyses labeled successful

result <- mytabletruefalse[4]

frame <- data.frame(table(mytabletft))

cov <- as.numeric(as.character(frame[,1]))

gen <- as.numeric(as.character(frame[,2]))

max <- length(mytable[(mytable$V1==mytable$V1[1] &
mytable$V2==mytable$V2[1]),1])

acc <- c(1)

dist[1] <- c(1)

for (i in 1:length(mytabletft$V3)){

      dist[1] <- mytabletft$Freq[mytabletft$V2==mytabletft$V2[i] &
mytabletft$V1==mytabletft$V1[i]]

      acc[i] <- dist[1]/max

}
```

```
acc <- rapply(as.data.frame(acc), f=function(x) ifelse(is.nan(x),0,x),
how="replace" )

###

png(file=paste(paste("Testplot_alderratiorighttotal_", fname, sep=""), ".png",
sep=""), units="in", width=6.0, height=6.0, res=60)

#png(paste(paste("Plot_alderratiorighttotal_", fname, sep=""), ".png", sep=""))

plot(gen[cov==unique(cov[1])], as.data.frame(acc)[cov==unique(cov[1]),1], col=1,
type="b", xlim=range(gen), ylim=c(0,1.4), xlab="Generations", ylab="Fraction of
successful runs",cex.lab=1.5)




lines(meanvalues[1][meanvalues[2]==cov[1],], rep(1, length(unique(gen))), col=1,
type="l", lty=2)


legend(50,1.37, unique(cov), lty=rep(c(1), each=length(unique(cov))),
lwd=rep(c(1), each=length(unique(cov))),,col=c(1:length(unique(cov))),cex=1.3)

for (i in 2:length(unique(cov))){

      lines(gen[cov==unique(cov)[i]], as.data.frame(acc)[cov==unique(cov)[i],1],
col=i, type="b")

}


dev.off()

##########################################

result <- mytabletruefalse[4]

frame <- data.frame(table(mytabletft))

cov <- as.numeric(as.character(frame[,1]))

gen <- as.numeric(as.character(frame[,2]))

acc <- c(1)

for (i in 1:length(mytabletft$V3)){

      dist[1] <- mytabletft$Freq[mytabletft$V2==mytabletft$V2[i] &
mytabletft$V1==mytabletft$V1[i]]

      dist[2] <- mytabletff$Freq[mytabletff$V2==mytabletff$V2[i] &
mytabletff$V1==mytabletff$V1[i]]

      acc[i] <- dist[1]/(dist[1]+dist[2])

}

acc <- rapply(as.data.frame(acc), f=function(x) ifelse(is.nan(x),0,x),
how="replace" )

png(file=paste(paste("Testplot_alderratiorightwrong_", fname, sep=""), ".png",
sep=""), units="in", width=6.0, height=6.0, res=60)

#png(paste(paste("Plot_alderratiorightwrong_", fname, sep=""), ".png", sep=""))

plot(gen[cov==unique(cov[1])], as.data.frame(acc)[cov==unique(cov[1]),1], col=1,
type="b", xlim=range(gen), ylim=c(0,1.0), xlab="Generations", ylab="Ratio of
true positives",cex.lab=1.5)
```

```
legend(80,0.4, unique(cov), lty=rep(c(1), each=length(unique(cov))),
lwd=rep(c(1), each=length(unique(cov))),,col=c(1:length(unique(cov))),cex=1.3)

for (i in 2:length(unique(cov))){

      lines(gen[cov==unique(cov)[i]], as.data.frame(acc)[cov==unique(cov)[i],1],
col=i, type="b")

}


dev.off()
```