

**University of São Paulo  
“Luiz de Queiroz” College of Agriculture**

**Building highly saturated genetic maps with OneMap 3.0: new  
approaches using workflows**

**Cristiane Hayumi Taniguti**

Thesis presented to obtain the degree of Doctor in Science.  
Area: Genetics and Plant Breeding

**Piracicaba  
2021**

**Cristiane Hayumi Taniguti**  
**Bachelor in Biotechnology**

**Building highly saturated genetic maps with OneMap 3.0: new  
approaches using workflows**

versão revisada de acordo com a resolução CoPGr 6018 de 2011

Advisor:

Prof. Dr. **ANTONIO AUGUSTO FRANCO GARCIA**

Thesis presented to obtain the degree of Doctor in Science.

Area: Genetics and Plant Breeding

**Piracicaba**  
**2021**



**Dados Internacionais de Catalogação na Publicação**  
**DIVISÃO DE BIBLIOTECA - DIBD/ESALQ/USP**

Taniguti , Cristiane Hayumi

Building highly saturated genetic maps with OneMap 3.0: new approaches using workflows / Cristiane Hayumi Taniguti . -- versão revisada de acordo com a resolução CoPGr 6018 de 2011. -- Piracicaba, 2021 .

154 p.

Tese (Doutorado) -- USP / Escola Superior de Agricultura "Luiz de Queiroz".

1. Mapa de ligação 2. Haplótipo 3. Reprodutibilidade 4. Erro de genotipagem I. Título.

To my parents Nanci and Marco Taniguti,  
with all my love.

In honor of all women scientists and educators who inspire me.  
In particular, my grandmother, the math teacher Edna Donabela Taniguti.

## ACKNOWLEDGEMENTS

I am thankful to CNPq (“Conselho Nacional de Desenvolvimento Científico e Tecnológico”) for the financial support, which made possible my professional qualification, the research, and the results here presented. I also thank the University of São Paulo, especially the “Luiz de Queiroz” College of Agriculture (ESALQ) for providing the structure needed for all these years of work and for joining so many amazing scientists, professors, and students, who I had the opportunity to meet. I am also grateful to other Brazilian institutions such as the Federal University of São Carlos (UFSCar), Federal University of Lavras (UFLA), and EMPRAPA (Empresa Brasileira de Pesquisa Agropecuária) for providing me environments of important scientific discussions.

I especially thank my advisor, Augusto Garcia, who gave support to all my ideas, precisely oriented me, and became a dear friend. He is for me an example of a scientist and optimistic person. I admire his capacity as a professor and the brilliant scientific team he has been qualifying, with significant impact in Brazilian and worldwide context. I do not have enough words to describe the impact of Augusto’s influence on my personal and professional development. I also thank him for introducing me to his wife, Luciana, and to his talented son, João.

I am grateful to all the co-workers in the Statistical Genetics Lab (ESALQ/USP) for all the scientific discussions and friendship. In this pandemic time, I miss our coffee breaks. Since my master’s, I have known some brilliant and kind people in this lab. I missed a lot the ones who defended the thesis first and left: João Feijó, Guilherme Pereira, Felipe Ferrão, Marianella Quezada, Letícia Lara, Danilo Cursi, Rodrigo Amadeu, Matheus Krause, Jhonathan Pedroso, Rafael Nalin, Pedro Jaloto, Mayara Oliveira, Mariana Niederheitmann, and Jéssica. The ones who just stayed for a while: Elaine Batista and Raíssa. I also thank the ones who are currently members of the lab: Gabriel Gesteira, Wellingson Araújo, Getúlio Ferreira, Camila Godoy, Vitória Bizão, Rafaela, Kaio Olímpio, and Laura. I especially thank Guilherme Pereira for helping me with ideas about the genotyping errors, multiallelic markers, and the design of the workflows for both empirical and simulated data; Marianella Quezada for providing datasets to tests; Jhonathan Pedroso and Getúlio Ferreira for coming with the idea of using splines to simulate the genetic map; Gabriel Gesteira for several works together and for helping me test the workflows in other systems; and Kaio Olímpio for the help with statistical analysis. Also, Rodrigo Amadeu and Letícia Lara for trusting me as their maid of honor. I spent really good times with all these people.

I am grateful to those who first developed OneMap: Gabriel Margarido, Marcelo Mollinari, and Augusto Garcia. And also to all those who contributed somehow: Guilherme Pereira, Rodrigo Amadeu, Getúlio Ferreira, Karl Broman, IdoBar, and all users that sent me an email with questions or suggestions. I especially thank Marcelo Mollinari for also developing MAPpoly, from which several of OneMap updates were based.

I also thank researchers from other groups in ESALQ, with whom I interacted a lot: Bioinformatics Laboratory Applied to Bioenergy, Conservation Genetics and Genomics group, Laboratory of Genetics of Microorganisms, Plant Ecological Genetics Laboratory, Allogamous Plant Breeding Laboratory, and our neighbors, the Cytogenomics and Epigenetics Laboratory. I especially thank the members of Bioinformatics Laboratory Applied to Bioenergy: Gabriel Margarido for the great classes, the discussions, and the help with OneMap updates; Fernando Correr for the friendship, for being a great partner in several activities, and for also helping me with workflow tests in other systems and with ideas about genotyping errors; Lorena Batista for the productive discussions and the trust in my potential; Victor Mello, Amanda Avelar, Ana Letycia, and Guilherme Hosaka. Evelyn Couto and Tamylin for the friendship, and Igor Araújo (Conservation Genetics and Genomics group) for the friendship and the help with evolution and population genetics. Allison and Carol (Plant Ecological Genetics Laboratory) for the company in lunch and sports activities. Maria Letícia (Microorganisms, Plant Ecological Genetics Lab-

oratory) for the talks about scientific communications. Júlia and Fernando (Allogamous Plant Breeding Laboratory) for all activities together. And Thiago Oliveira from Statistical Department (currently in Roslin Institute), for the friendship, very nice conversations, and for helping me with this work statistical analysis.

I acknowledge all the staff and professors of the Genetics Department, specially Prof. Maria Lúcia, Prof. Gabriel Margarido, Prof. Cláudia Vitorello, Prof. Maria Carolina, Prof. Giancarlo, Berdan, Fernandinho, Valdir, Léia, Macedônio, and Rafaelle.

Also, I am grateful for the opportunity to participate in the extension groups GVENCK and GENT, and interact with all members.

I would like to thank Fausto Silva and his team for providing me the joy of being called a “genetic scientist” on national open television. Empowering me even more in this profession. And the friends from Life Circo Piracicaba, PLPs Piracicaba and Cheerleading São Carlos, for the good times together.

I would like to thank my brother, Lucas Taniguti, who came with the idea of building the workflows and helped me to learn WDL syntax for the work here presented. He also had a great influence on my professional carrier choice, once since a kid, he is enthusiastic about technologies and motivated me to know new things and play video-games. I also thank my close friends Aline Nakamura, Bianca Coré, Luiza Kame, Guira Luz, Emeline Boni, Thaís Milanez, Laura Damada, Meenakshi Kannan, Fernanda Rossin, Maria Fernanda Trientini, Caio Boscariol, Rafael Pinto, Ariane Henrique, Daniel Sbravatti, Silvia de Oliveira, Nathalia Taniguti, Nathalia Alves, and Victor Rezende for the support in difficult times and for celebrating together the good times. I am grateful to Veri Firmino for being a great partner in this difficult last year and give me love and support to face all challenges that it brought.

I am mainly grateful to my parents, for making everything possible.

“I am among those who think that science has great beauty.  
A scientist in his laboratory is not only a technician,  
he is also a child place before natural phenomenon,  
which impress him like a fairy tale.”  
-*Marie Curie*

“Nothing happens in contradiction to nature,  
only in contradiction to what we know of it.”  
-*Dana Scully*

“Education is education.  
We should learn everything and then choose which path to follow.”  
-*Malala Yousafzai*

## SUMMARY

Resumo . . . . .	9
Abstract . . . . .	10
1 Introduction . . . . .	11
1.1 Literature review . . . . .	15
1.1.1 Linkage maps for outcrossing species . . . . .	15
1.1.2 High-throughput genotyping . . . . .	16
1.1.3 Genotyping errors . . . . .	18
1.1.4 Haplotype-based multiallelic markers . . . . .	20
References . . . . .	22
2 Reads2Map: Practical and reproducible workflows to build linkage maps from sequencing data . .	29
Abstract . . . . .	29
2.1 Background . . . . .	29
2.2 Implementation . . . . .	31
2.2.1 Pre-processing reads . . . . .	31
2.2.2 Sequencing reads simulation . . . . .	31
2.2.3 SNP calling . . . . .	34
2.2.4 VCF filters . . . . .	36
2.2.5 Genotype calling methods . . . . .	36
2.2.6 Linkage maps . . . . .	37
2.2.7 Read2Map Workflows App . . . . .	37
2.2.8 Selecting the pipeline . . . . .	38
2.3 Results and Discussion . . . . .	40
2.3.1 Genetic map . . . . .	40
2.3.2 Pipeline validation with linkage map . . . . .	41
2.3.3 Pipeline validation with simulations . . . . .	42
2.3.4 Genome assembly assisted by linkage maps . . . . .	43
2.3.5 Workflow flexibility . . . . .	43
2.4 Conclusions . . . . .	43
2.5 Availability and requirements . . . . .	44
References . . . . .	44
3 The effect of considering genotype probabilities and haplotype-based multiallelic markers in build- ing linkage maps with high-throughput genotyping . . . . .	49
Abstract . . . . .	49
3.1 Introduction . . . . .	49
3.2 Material and Methods . . . . .	51
3.2.1 Genotype probabilities in OneMap Hidden Markov Model . . . . .	51
3.2.2 Empirical data . . . . .	52
3.2.3 Simulation data . . . . .	53
3.2.4 Statistical analysis . . . . .	55
3.3 Results and Discussion . . . . .	57
3.3.1 SNP calling . . . . .	57
3.3.2 Genotype calling . . . . .	61
3.3.3 Linkage map building . . . . .	68
3.3.4 Haplotype-based markers and ordering algorithms . . . . .	72
3.4 Conclusion . . . . .	74

References . . . . .	75
4 Final Considerations . . . . .	81
Appendices . . . . .	83
Attachments . . . . .	115

## RESUMO

### Construção de mapas genéticos altamente saturados com OneMap 3.0: novas abordagens usando workflows

OneMap é um pacote do R desenvolvido por membros do Laboratório de Genética Estatística da ESALQ/USP (Brasil) lançado em 2008. Ele ganhou atenção da comunidade científica por ser um dos primeiros programas capazes de construir mapas genéticos integrados para populações  $F_1$  segregantes. Ele é hoje muito usado mundialmente. Entretanto, ele requer aprimoramentos para lidar com novos e abundantes marcadores provindos de técnicas de genotipagem baseada em sequenciamento. Neste trabalho, foi feito um aprimoramento significativo no OneMap para a versão 3.0, o qual inclui: maior velocidade na estimativa das distâncias genéticas; novos métodos de agrupamento e ordenamento dos marcadores; novas ferramentas gráficas para diagnóstico da qualidade dos mapas; novos recursos para realização de simulações; recursos para conversão de arquivos VCF com marcadores bialélicos e multialélicos para os arquivos de entrada do OneMap; possibilidade de incluir probabilidade de erro ou de genótipos para estimar as distâncias genéticas. Uma vez que o OneMap foi atualizado, também foram explorados passos anteriores à construção do mapa, os quais têm impacto na qualidade do mapa resultante. Para isso, foram desenvolvidos os *workflows* **Reads2Map** que realizam análises desde leituras de sequenciamento de dados empíricos ou simulados até mapas genéticos. Por ser escrito em *Workflow Description Language* (WDL), os workflows **Reads2Map** disponibilizam aos usuários códigos localizáveis, acessíveis, interoperáveis e reutilizáveis para a construção de mapas genéticos. Os workflows desenvolvidos são capazes de comparar o desempenho dos programas na construção de mapas genéticos: **freebayes**, **GATK** como identificadores de SNPs e genotipadores; **updog**, **polyRAD** e **SuperMASSA** como genotipadores; **OneMap 3.0** e **GUSMap** para construção de mapas. Além disso, foi desenvolvido o aplicativo **shiny Reads2MapApp** para avaliação gráfica dos resultados dos workflows. No caso particular do conjunto de dados de *Populus tremula*, o **freebayes** foi selecionado como identificador de SNPs e genótipos, e uma probabilidade de erro global de 5%, resultando em um mapa com 6936 marcadores e 3299.96 cM. Em seguida, também utilizando os workflows, foi testado o impacto de duas das maiores melhorias do OneMap 3.0: o uso de probabilidades genotípicas para estimativa das distâncias genéticas; e o uso de marcadores multialélicos baseados em haplótipos provindos de identificadores de SNPs. Usando sequências de leituras simuladas foi possível medir a eficiência de cada identificador de SNP e genótipo e suas influências na construção do mapa. O impacto das probabilidades dos genótipos foi variável entre os programas de acordo com o cenário simulado. Os resultados mostraram que o OneMap 3.0 é capaz de construir mapas genéticos de alta qualidade se i) os genotipadores não cometerem muitos erros e a probabilidade de erro for de 5% para todos os genótipos ou ii) se o genotipador cometer mais erros de genotipagem e atribuir probabilidades menores para os genótipos errados. Além disso, o uso dos marcadores multialélicos baseados em haplótipos revelou um aumento na qualidade de ordenamento e estimativa de distância genética. Uma vez que os processos anteriores à construção dos mapas têm grande impacto na sua qualidade, o uso combinado do OneMap 3.0, **Reads2Map** e **Reads2MapApp**, disponibiliza para os usuários ferramentas para construção de mapas genéticos desde leituras de sequenciamento, e também gráficos diagnóstico para auxílio na escolha da melhor combinação de programas e parâmetros.

**Palavras-chave:** Mapa de ligação, Haplótipo, Reprodutibilidade, Erro de genotipagem



## ABSTRACT

### Building highly saturated genetic maps with OneMap 3.0: new approaches using workflows

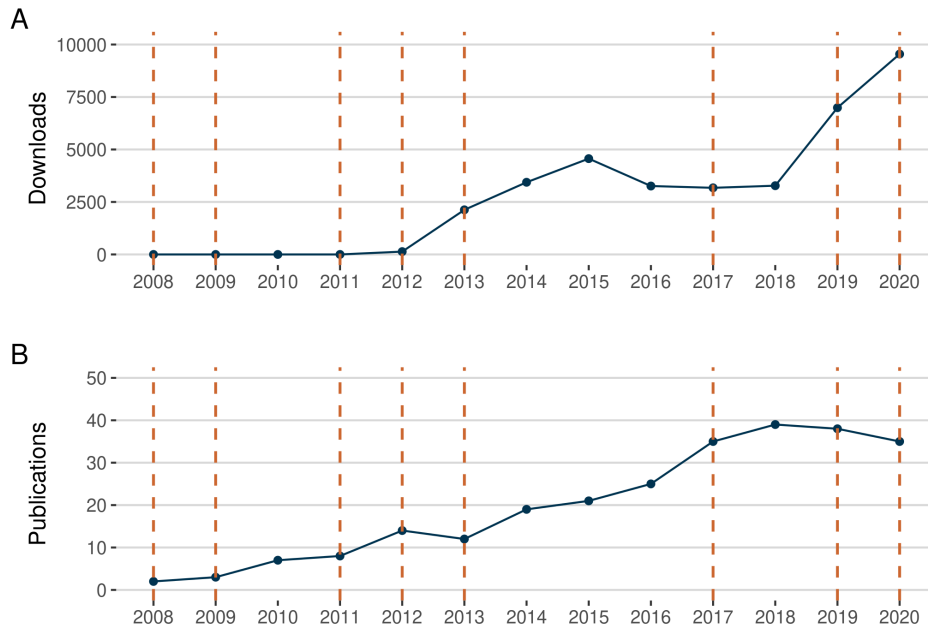
OneMap is an R package developed by members of Statistical Genetics Laboratory at ESALQ/USP (Brazil) released in 2008. It gained the attention of the scientific community for being one of the first software for building integrated genetics maps for outcrossing species. It is now highly used worldwide. However, it requires updates to deal with the new and abundant markers generated by high-throughput genotyping techniques. In this work, we made a major update of OneMap to version 3.0, which includes: higher speed of the genetic distance estimation; new methods for group and ordering markers; new quality diagnostic graphics tools; new features for making simulations; features to the conversion of VCF file with biallelic and multiallelic to OneMap input file; possibility of include error or genotype probability to estimate the genetic distances. Once OneMap was updated, we explored the steps upstream of the map building process, which has an impact on the resulted map quality. For that, we developed the **Reads2Map** workflows that perform the analysis, starting with empirical or simulated sequencing reads until the final linkage maps. Because the presented workflows are written with **Workflow Description Language** (WDL), they provide to users a findable, accessible, interoperable, and reusable code to build maps. The workflows compare the performance of software in the linkage map building: **freebayes**, **GATK** as SNP and genotype callers; **updog**, **polyRAD**, **SuperMASSA** as genotype caller; **OneMap 3.0** and **GUSMap** as linkage map builders. We also developed the shiny **Reads2MapApp** app to evaluate graphically the workflow's results. In the particular case of an example dataset from *Populus tremula*, we select the **freebayes** as SNP and genotype caller, and a global error probability of 5%, resulting in a map with 6936 markers and 3299.961 cM. After also using the workflows, we tested the impact of two of the major OneMap 3.0 updates in the linkage maps: the usage of genotype probabilities to estimate the genetic distances and the haplotype-based multiallelic markers from assembly-based SNP caller. Using simulated sequence reads data we could measure each SNP and genotype caller efficiency and its influences in the resulted map. The impact of the genotype probabilities was variable between software according to each simulated scenario. The results showed that OneMap 3.0 can build high-quality genetic maps if i) the genotype callers do not estimate wrongly many genotypes and a global error rate of 5% is applied for all genotypes or ii) if the genotype caller estimate more genotypes wrongly it also gives lower genotype probabilities for the wrong genotypes. Furthermore, the usage of haplotype-based markers reveals to increase the order and genetic distance quality. Once the procedures upstream the genetic map building have a strong influence in its quality, the combined usage of OneMap 3.0, **Reads2Map** and **Reads2MapApp** provide to users tools to build linkage maps since the sequencing reads, and also diagnostic graphics and measures to help them to choose the best combination of software and parameters.

**Keywords:** Linkage map, Haplotype, Reproducibility, Genotyping error

## 1 INTRODUCTION

The **OneMap** package (MARGARIDO *ET AL.*, 2007) was release in CRAN repository in 2008 with the novelty of building integrated genetic maps for outcrossing species. **OneMap** can build maps with all types of markers (Table A.2) of outcrossing or inbred (RIL,  $F_2$  intercross, and  $F_2$  backcross) mapping populations. At the time, **OneMap** was one of the software which opened the opportunity to better understand the genetic architecture of outcrossing species such as yellow passion fruit, loblolly pine, sugarcane, rubber tree, oil palm, eucalyptus, and salmon (OLIVEIRA *ET AL.*, 2008a; XIONG *ET AL.*, 2016; GARCIA *ET AL.*, 2006a; SOUZA *ET AL.*, 2013; JEENNOR and VOLKAERT, 2014; BARTHOLOMÉ *ET AL.*, 2015c; GONEN *ET AL.*, 2014).

**OneMap** is written in R, a free software environment for statistical computing and graphics. Users need to have a least a little knowledge of the R language to be able to use the package. According to TIOBE programming community index (THE SOFTWARE QUALITY COMPANY TIOBE, 2021), R had low (rate of 0.06%) popularity in 2007. In 2010, its popularity started to increase because more people, especially in the statistical science field, started to migrate from commercial statistical languages to R (Figure A.1). The accessibility of **OneMap** increased together with the R popularity (Figure 1.1).



**Figure 1.1.** OneMap popularity since its release in 2008. A: Total number of **OneMap** downloads from CRAN by year until 2020. Data obtained using `cranlogs` package (CSÁRDI, 2019). B: Total number of publications citing **OneMap** by year from 2008 to 2020 according to DIMENSIONS (2021) using the words 'onemap', 'linkage map' and 'genetic' as research criterias. The orange dashed lines represents **OneMap** version updates in CRAN

Members of Statistical Genetics Laboratory in ESALQ/USP (Brazil) make constant improvements and maintenance in **OneMap** algorithms and documentations, which also contribute to its success. In 2013, we created an **GitHub** (CHACON and STRAUB, 2014) repository to store the **OneMap** development version (<https://github.com/augusto-garcia/onemap>). The platform gives several advantages to code development, including the track of all changes, teamwork optimization, and allows **OneMap** to receives contributions from anyone around the world through pull requests or messages.

The improvements were made following novelties in statistical genetics research and according to users' feedbacks. Most of the users' demands came together with high-throughput genotyping platforms availability. Most of these demands focused on the conversion of VCF file format (DANECEK *ET AL.*,

2011b), the time-consuming analysis using Hidden Markov Model (HMM), the wrong group and ordering of markers, and the inflated map sizes. In this work, we made updates to attend to each one of these demands. All the updates compose a major modification in the OneMap version. Therefore, here we will refer to the OneMap before this work updates as OneMap 2.0 and after this work updates as OneMap 3.0.

The HMM (BAUM *ET AL.*, 1970; LANDER and GREEN, 1987) combined with the expectation-maximization (EM) (DEMPSTER *ET AL.*, 1977) algorithm implemented in OneMap is a very robust method to estimate the phases and genetic distance. It calculates iteratively the likelihoods for each possible phase for each marker adding them sequentially and also considering the previous information. At the end of the process, the HMM returns the most likely haplotype for each individual in the population. Thus, the recombination fraction estimated between markers are based on entire sequence information (multipoint approach). This is the best model possible to estimate the haplotypes, but demands an exhaustive method. As the total number of markers in the sequence increases, it also increases the computational resources and time needed (WU *ET AL.*, 2002d; GARCIA *ET AL.*, 2006b; MARGARIDO *ET AL.*, 2007).

OneMap version 2.0 already received updates focusing in speed up the HMM and EM algorithm. The code was rewritten with a lower-level programming language, the C++, mainly by the developers M. Mollinari and G. Margarido. The R package Rcpp (EDELBUETTEL and FRANÇOIS, 2011; EDELBUETTEL, 2013; EDELBUETTEL and BALAMUTA, 2017) allowed an easy integration of C++ with the remaining R code. Nevertheless, users still demanded more speed.

Another solution was proposed in SCHIFFTHALER *ET AL.* (2017), which includes calculating the linkage map in overlapping batches. The idea is that we do not need to perform the search for the maximum likelihood for each marker considering all previous markers in the sequence when there are many markers available. The previous information is indeed necessary to obtain accurate calculations of phase likelihoods, but the return saturates after few markers are evaluated. Keeping the search will only overload the model while offering no additional accuracy. In this case, we can limit this search using batches. The batches still keep part of the information from the previous batches using the previously estimated phases in overlapping markers. The SCHIFFTHALER *ET AL.* (2017) simulations revealed that, once the batch and overlap size are optimized, the method keeps the same accuracy of haplotype estimation as the original one. We also made our simulation to confirm that (Attachment B).

The SCHIFFTHALER *ET AL.* (2017) method also proposed the parallelization of the analysis in different computer cores. The batches can not be considered as independent processes to compute in separated cores, because of their overlapping markers. However, the parallelization can be made through the four possible phases for outcrossing marker combinations. Thus, the analysis can be divided into a maximum of four different computer cores. We also examined the possibility of parallelization through the batches to increase the possible number of cores to be used. We tested the accuracy of the estimated haplotypes by comparing the estimations in overlap markers among the batches in simulations. Despite it compromises accuracy for some marker combinations, it can be useful to obtain fast estimations (see Attachment B).

The SCHIFFTHALER *ET AL.* (2017) modification was implemented in BatchMap package, a separated fork of OneMap, released in CRAN in March 2017. However, BatchMap did not receive maintenance and was removed from CRAN in December 2019. There is still the GitHub and Docker Hub versions available. As mentioned before, GitHub platform offers several advantages to code development, however does not impose restrictions on unfunctional codes. Docker hub is a repository for container images. The containers images are lightweight, standalone, executable package of software that includes everything needed to run an application: code, runtime, system tools, system libraries and settings (MERKEL, 2014a). Thus, with Docker containers users can reproduce the exact computer environment where developers made the code functional. However, the Docker hub and the GitHub versions of BatchMap became outdated once OneMap received new updates. Here, we made the needed updates and merged

the **BatchMap** modifications with the most recent version of **OneMap**.

The ordering step of linkage map building in **OneMap** also needed improvements to adjust to high-throughput markers. All ordering algorithms implemented try to solve the "traveling salesman problem". With high-throughput markers, the traveling salesman would have "more cities to visit" and the problem became more complex. In 2016, PREEDY and HACKETT (2016) proposed the application of multidimensional scaling for ordering markers in linkage maps as a rapid and efficient method for a large number of markers. We also implemented the method in **OneMap** and made simulations to compare it with the other ordering algorithms implemented (see Attachment C). It reveals to be an excellent method for a global ordering of markers, despite keeping local misplacement.

Besides the increase in the number of markers available, the high-throughput sequencing genotyping also presented more errors. The users' demands about the wrong grouping and the inflated map sizes are consequences of these errors.

The grouping of markers in separated linkage groups is made in **OneMap** through the two-points recombination fraction estimations. The logarithm of odds (LOD) score statistics can be applied to test for linkage. LOD score is calculated by the log of the linkage probability of the observed data divided by the probability of the loci be unlinked.

Equation 1.1 shows a simple example of the probability density function used to estimate the described probabilities. This particular function is used for a backcross population.

$$l(r) = (n_1 + n_4)\log\left(\frac{1-r}{2}\right) + (n_2 + n_3)\log\left(\frac{r}{2}\right) \quad (1.1)$$

In the equation 1.1,  $n_1$ ,  $n_2$ ,  $n_3$  and  $n_4$  refers to the four possible combination of two genotypes for backcross population structure (AB|AB, AB|Ab, AB|aB and AB|ab) and  $r$  is the two-point recombination fraction value. Under  $H_0$  hypothesis (unlinked loci) the recombination fraction parameter  $r$  is 0.5.

**OneMap** suggests a LOD threshold defined considering all two-points tests that will be performed for all markers in the data set. It uses a global alpha controlling type I error with Bonferroni's correction. From this global alpha, the corresponding quantile from the chi-square distribution is taken and then converted to LOD score.

Genotyping errors can distort the two-points estimations for some markers, which makes traditional **OneMap** grouping algorithm returns wrong linkage groups. The result usually presents a total number of groups different of expected. It is tempting to increase the LOD threshold until the number of groups reaches the expected for the species. This is a common mistake made by users. However, notice that, because it refers to a log function, every slight increase or decrease in LOD value turns the analysis very permissive or rigorous, causing error types I and II, respectively.

A fast, but still not the best solution for group issues was to make use of previous information from the genome, drafts, or past linkage maps. We made an algorithm that separates sequences according to the previous information and, after, tries to group the remaining markers using the group algorithm. The **OneMap** group algorithm tests the linkage of each marker with markers already in the sequences to decide if they are together.

With the presence of genotyping errors, every step of map building turns more difficult, because the genotype frequencies used to estimate two-points recombination fractions are not trustful. The two-points recombination fraction estimation does not allow to consider an error probability for each genotype, but the HMM does. Once the groups are defined, the HMM can consider an error probability for each genotype in its emission function to estimate the genetic distances. In other words, the HMM can consider the genotypes, not as discrete values such 0 ("aa"), 1("ab"), and 2 ("bb"), but continuous probabilities referring to the chance of each possible genotype to be the true genotype. However, the success of the

approach depends on accurate genotype probabilities estimations in steps upstream **OneMap** (TANIGUTI, 2017; BILTON *ET AL.*, 2018; MOLLINARI and GARCIA, 2019; MOLLINARI *ET AL.*, 2020).

Another problem surrounding the high-throughput markers is the low-informativeness of the SNP markers. Their biallelic nature makes available only markers of types B3.7, D1.10 and D2.15 (see A.2). Depending on the population, marker types D1.10 and D2.15 are more frequent than B3.7. In this scenario, it is harder to find information to integrate parents' meiosis information. As consequence, the HMM and EM algorithms need to iterate in more phases possibilities until it reaches the best possible solution, which demands more computer efforts and includes more uncertainties in the analysis.

Some SNP calling software like **GATK** (POPLIN *ET AL.*, 2017) and **freebayes** (GARRISON and MARTH, 2012) uses an assembly-based haplotyping method to search for the polymorphisms in the data sequences. As consequence, they provide phased markers in specific regions of the genomes where they could define a local haplotype. This information can be useful to increase markers informativity in the map building process. Also, it can reduce the possible phases to be estimated and consequently reducing the HMM computational efforts. Therefore, to solve the issues about genotyping errors and low-informativeness, we need to explore the bioinformatics steps upstream of the **OneMap** analysis.

With our own experience and with users' feedback we already saw that the genetic map itself can be an interesting tool to validate the upstream process because errors observed in the maps point to a dissociation of the data from the genetic concepts. **OneMap** provides tools to measure the quality of the built map. The heatmap color graphics of the recombination fraction can highlight outlier markers breaking the expected recombination pattern. We implemented new graphical tools to draw the estimated parents and progeny haplotypes and count the number of recombination breakpoints estimated. This new tool demanded a major modification in  $F_2$  intercross algorithms for phase estimation (Attachment C). The new tool highlights the excessive recombination breakpoints estimated when something wrong happened in an upstream process, as contaminant individuals or genotyping errors. However, we need more tools for diagnostics in the entire pipeline (from read sequence to built linkage map) which are the steps affecting the map quality, or which are the ones that can bring solutions.

Even with diagnostic tools available for this upstream process, the dataset context, the software and parameters to test are too many and we can not make a single solution or recommendation. Mostly about the parameters, each software can have dozens of them and changing a single one can produce different results. Therefore, we developed in Chapter 2 the **Reads2Map** workflows. They perform the linkage map building from empirical or simulated sequencing reads datasets. The workflows are written in **Workflow Description Language** (WDL) (VOSS *ET AL.*, 2017), which provides an organized, user-friendly and reproducible structure to all the analysis. With them, users can perform the simultaneous analysis using **freebayes**, **GATK** as SNP and genotype callers; **updog**, **polyRAD**, **SuperMASSA** as genotype callers; **OneMap 3.0** and **GUSMap** as linkage map builders. The results can be visualized in the shiny app **Reads2MapApp**, which contains graphical tools for diagnostics of the entire procedure to help users the select the best combination of software and parameters for their particular case.

In Chapter 3, we used the workflows developed in Chapter 2 to measure the impact of two of the major modifications of **OneMap 3.0** in empirical and simulated data. One is the usage of genotype or error probabilities in the HMM to estimate the genetic distances. We compared the genotype probabilities profiles of the genotype callers and related them to the quality of the linkage map resulted. The second is the usage of haplotype-based multiallelic markers from the assembly-based SNP callers. We observed the capacity of each SNP caller to call these markers and compared the maps including them or not.

## 1.1 Literature review

### 1.1.1 Linkage maps for outcrossing species

The first genetic maps, made from the 1910s to 1960s, had only a few morphological markers for inbred-based populations of diploid species ( $F_2$ , backcross and RILs). In the 1970s, with the spread of electrophoresis techniques, plants genetic maps started to include isozymes markers (NIELSEN and SCANDALIOS, 1974). The DNA markers were developed in the 1980s and the PCR-based markers in the 1990s (NADEEM *ET AL.*, 2017). At this time, maps started to have from dozens to hundreds of markers, significantly improving their resolution. After, boosted by the Human Genome Project, started in 2001 an exponential decrease in price and increase in quality of sequencing technologies. Nowadays, with high-throughput sequencing, it is possible to automatically obtain millions of markers (HEATHER and CHAIN, 2016).

Following these advances, statistical approaches were developed to build maps for different mapping populations and different marker types. For diploid outcrossing species, which for some reason have inbreeding depression or a long reproductive cycle, it is not possible to generate homozygous lines. The mapping population available for these species are the  $F_1$  progenies.

The  $F_1$  mapping populations have 18 different segregation patterns (WU *ET AL.*, 2002d), once there are up to four different alleles for each loci: if a locus presents three or four different alleles, it produces four genotypes in the progeny at the 1:1:1:1 proportion; if a loci have two different alleles, with both parents heterozygous, genotypes will follow the 1:2:1 proportion; if loci have two different alleles, but one parent is homozygous, the pattern will be 1:1 (Table A.2). In this scenario, it is essential to estimate the parental linkage phase for loci with segregation 1:2:1 and 1:1 to be possible to count recombinants to build the map. Without phase information, it is not possible to distinguish parental and recombinant configuration for map distance estimation. For highly informative markers, with segregation 1:1:1:1 the phase is easily estimated because each allele is unique by parent homologs.

Focusing on the estimation of linkage phases, statistical methodologies for map building were developed to include outcrossing species, starting with pseudo-testcross method (RITTER *ET AL.*, 1990; GRATTAPAGLIA and SEDEROFF, 1994), which consider only dominant markers and provide separated maps for each parent. The method considers the segregation on the progeny to infer the parents' phase. After, ARÚS *ET AL.* (1994); RITTER and SALAMINI (1996); MALIEPAARD *ET AL.* (1997) revealed recombination fraction maximum likelihood estimators for outcrossing populations allele configurations. Also, a multipoint maximum likelihood estimator was proposed for inbred lines maps (LANDER and GREEN, 1987; JANSEN and NAP, 2001). Later, the method was expanded for outcrossing species (WU *ET AL.*, 2002d). WU *ET AL.* (2002d) presents a method to simultaneously estimates phase and the recombination fraction using maximum likelihood estimators presented in MALIEPAARD *ET AL.* (1997), which made it possible to integrate the separated maps for each outcrossing parent in only one. In WU *ET AL.* (2002c), the method was improved by using a multipoint algorithm based on HMM (Hidden Markov Model). The method proposed by WU *ET AL.* (2002d,c) was implemented in OneMap software (MARGARIDO *ET AL.*, 2007).

The integrated genetic map makes more sense for association studies, once the analyzed phenotype is from progenies (not from parents). Besides that, integrated genetic maps can join every type of marker and provide a dense genetic map. Also, with integrated genetics maps we can still measure the difference in recombination frequency between the parents (WU *ET AL.*, 2002c). However, if there are only markers of D1 and D2 types, it is not possible to joint the information of both parents, once these markers carry recombination information from only one of them.

### 1.1.2 High-throughput genotyping

In the last few decades, we saw a gradual change in the molecular laboratory protocols from efforts elaborating primers to obtain genetic markers based on fragment length, such as microsatellites, to procedures to generate DNA libraries for high-throughput genotyping platforms (SEEB *ET AL.*, 2011; GROVER and SHARMA, 2014; GARRIDO-CARDENAS *ET AL.*, 2018). These platforms are able to generate millions of molecular markers at an affordable price and without the need for much molecular laboratory efforts.

The high-throughput genotyping platforms can be differentiated as array-based or sequence-based. The array's technologies consist of a solid surface with thousands of genomic sequences called probes bonded covalently. The biological sample DNA fluorescently labeled hybridizes with the probes emitting signals which are individually detected by the platform (GARRIDO-CARDENAS *ET AL.*, 2018). It requires the previous information of the genome sequence to design the probes. Thus, it is usually applied to well-known agronomic species like eucalyptus (SILVA-JUNIOR *ET AL.*, 2015) or maize (XU *ET AL.*, 2017). It usually generates high-quality markers, but with a relatively high price. It can also contain bias if the array probes are designed for a population evolutionarily distant from the studied one (LIU *ET AL.*, 2020).

The sequence-based technologies overcome the ascertainment bias because it allows a simultaneous SNP and genotype calling. Furthermore, it usually presents lower price (LIU *ET AL.*, 2020). There are also several methods of obtaining markers from sequencing. Generally, we can differentiate them by how the DNA libraries are prepared: amplicon generation, whole-genome preparation, and target enrichment (DER AUWERA *ET AL.*, 2020).

Building genetic maps also requires a large population size to identify the recombination events. In general, higher the population size, the higher is the map resolution. It is an important parameter mostly when dealing with mapping populations which suffers only one recombination event and have large disequilibrium blocks, such as outcrossing population ( $F_1$ ),  $F_2$  and backcross (Attachment D).

The higher size of populations required for genetic mapping makes the usage of the whole-genome preparation library for this purpose unreliable because it would require sequencing all genomes several times (depth) to have trusted genotypes. Despite the decrease in sequencing price, it is still unaffordable for most researchers.

In the target enrichment method, only some pre-defined regions of the genome are sequenced. It is widely applied in exome studies. It is similar to arrays-based genotyping, but instead of an array, bait sequences (similar to PCR) primers are used to direct the sequencing to particularly regions (Figure 1.2) (DER AUWERA *ET AL.*, 2020). But, also similar to arrays disadvantage, for most of the agricultural species, we do not have much information about the genome to design the target sequences.

Sequencing genotyping technologies only started to be massively applied to plant and animal mapping populations after the development of the amplicon generation libraries method. It does not require previous information about the genome and allowed sequencing more individuals at the expense of genome coverage. It comprises the reduction of genomic complexity with restriction enzymes (RADseq methods) (BAIRD *ET AL.*, 2008; ELSHIRE *ET AL.*, 2011; PETERSON *ET AL.*, 2012). Thus, the library preparation includes digesting the genomic DNA with restriction enzymes, evaluating the sequence size generated, adding barcodes and adapters to the sequence borders, and polymerase chain reaction (PCR) amplification. The presence of barcodes in each read makes it possible to sequence several samples in the same lane. The generated amplicons are fragments of DNA present in large numbers of copies that start and stop at the same positions. There are several variants of RADseq protocol, see ANDREWS *ET AL.* (2016) for a review of each RADseq variation.

The spread of RADseq methods usage contributed to significant advances in ecological (AN-

DREWS *ET AL.*, 2016) and plant and animal breeding (KIM *ET AL.*, 2015) studies. However, sequencing a large mapping population can still be not affordable for many laboratories, even though the sequence technologies are becoming less expensive and more readily available. To make it affordable, it is common to perform low-depth sequencing combined with the RADseq methods.

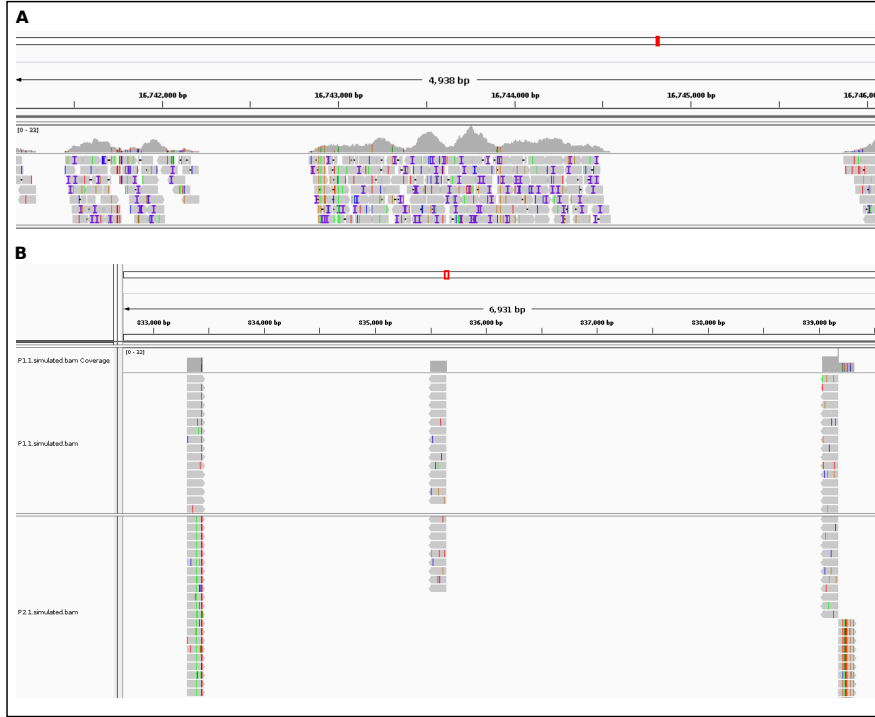
In RADseq methods, the number of loci to be sequenced, or the genome coverage, is related to the selected enzyme cut frequency in the genome. The number of samples is important to detect different recombination events and to increase map resolution. The greater the number of loci and samples per lane the lower the depth, that is, the number of reads per locus (DAVEY *ET AL.*, 2011). While the number of markers and samples affects the genome coverage and resolution, the depth of the sequencing affects the quality of the information.

The decisions around coverage, resolution, and depth of the RADseq protocol need to be made according to the genome characteristics and the study goals. For homozygous individuals, as recombinant inbred lines (RILs) populations, low-depth sequencing produces different subsets of markers with reliable genotypes for each individual. The missing data generated by the overlapping of the subsets can be easily imputed if knowing the positions of the markers in a reference genome or genetic map (XIE *ET AL.*, 2010). However, low-depth sequencing for heterozygous loci carries more uncertainty.

For heterozygous genotypes, low-depth sequencing can randomly sample only one of the alleles and erroneously call homozygous genotypes. Moreover, the low-depth emphasizes errors coming from other sources, as sequencing errors, allelic bias coming from PCR amplification, erroneous reads alignments (POMPANON *ET AL.*, 2005). As a consequence, the number of reads per allele is different than expected. Their distribution can show outliers, overdispersion, and allelic bias (GERARD *ET AL.*, 2018).

RIVERA-COLÓN *ET AL.* (2020) traced two non-mutually exclusive sources of errors in RADseq data: the unsuitable choice and application of a molecular protocol, and problems with library preparation and sequencing. Considering all possible molecular protocols available for RADseq methods (ANDREWS *ET AL.*, 2016; KIM *ET AL.*, 2015), it is important to choose the one that best fits the specific subject of the research and correctly plan an experimental design. The RADinitio (RIVERA-COLÓN *ET AL.*, 2020) software simulate read sequences according to RADseq protocols and intends to help researchers choose the best protocol for their experiments.





**Figure 1.2.** IGV (ROBINSON *ET AL.*, 2011) image showing examples sequencing reads of two types of DNA library preparation. A: the target enrichment as exome. B: the amplicon generation libraries such as RADseq. The data used in this example was simulated by SimusCoP (YU *ET AL.*, 2020) and RADinitio (RIVERA-COLÓN *ET AL.*, 2020), respectively, using the **SimulatedReads2Map** WDL workflow developed in chapter 2.

DER AUWERA *ET AL.* (2020); RIVERA-COLÓN *ET AL.* (2020) highlights the PCR-related technical bias as the leading source of genotyping errors for this type of datasets. The poor quantity and quality of DNA available for generating the library limit the number of original molecules available for the PCR enrichment and sequencing, which increases significantly its sampling bias. Particularly, the alleles dropout can also be caused by mutations in restriction enzyme cut sites. However, RIVERA-COLÓN *ET AL.* (2020) shows through simulations that the alleles lost due to the random sampling associated with library enrichment and sequencing are the main source of this type of error. The RADinitio tool is the only one available capable of simulate RADseq read sequences including the PCR-related bias and the allele dropout due to variants. Also, it allows to simulated user-defined variants and genotypes with a VCF input.

### 1.1.3 Genotyping errors

The bioinformatics software developed to perform RADseq sequencing data analysis (BRADBURY *ET AL.*, 2007; CATCHEN *ET AL.*, 2013a; GERARD *ET AL.*, 2018; CLARK *ET AL.*, 2019) have the challenge to overcome the sources of errors coming from low-depth sequencing. The bioinformatics pipelines to obtain the genetic markers and the genotypes starts with the alignment of the reads to a reference genome or between themselves (if no reference genome is available). Then, SNP and genotype calls are performed. When analyzing only a sample, SNP and genotype calling refers to the same procedure, because the presence of the polymorphism also implies a heterozygote or a non-reference homozygous genotype (MIELCZAREK and SZYDA, 2016). But, when dealing with several samples, the SNP and genotype calling are different steps. The SNP calling consists only in the identification of polymorphic loci and the genotype calling step consists in infers the genotypes of the samples at each identified polymorphic locus (NIELSEN *ET AL.*, 2011).

Some SNP and genotype calling software, like GATK (POPLIN *ET AL.*, 2017; MCKENNA *ET AL.*,

2010) and SAMtools (LI *ET AL.*, 2009a), try to overcome sources of errors even before the SNP calling. They perform first a local realignment and/or recalibration of Phred scores for the alignment and the sequencing quality (MIELCZAREK and SZYDA, 2016).

After the SNP calling, quality scores from each previous step can be applied to a priori distributions in probabilistic methods of genotype calling. Then, each genotype has a probability of being homozygote for reference allele, or alternative allele, or heterozygote. The highest probability will define the genotype and it will be normally used in software for build genetic maps.

Recently, TANIGUTI (2017); BILTON *ET AL.* (2018); MOLLINARI and GARCIA (2019); MOLLINARI *ET AL.* (2020) demonstrated the advantage of using the genotypes probabilities instead of qualitative genotypes to estimated the linkage maps. Thus, it is possible to estimate genetic distance by weighting the distance between markers with the errors associated with the genotype probabilities. As a consequence, the estimated genetic map distance is closer to the actual one, because, if not detected, the genotype errors are counted as recombinant events and inflates the map distances (BUETOW, 1991; GOLDSTEIN *ET AL.*, 1997; CHEEMA and DICKS, 2009; HACKETT and BROADFOOT, 2003; RASTAS, 2017). A similar approach using genotype probabilities instead of genotypes already has been applied to infer haplotypes and for Genome-Wide Association (GWAS) studies (KANG *ET AL.*, 2004; BROWNING and YU, 2009).

Each genotype calling software considers a specific a *priori* according to the source of errors considered at their probabilistic model and provides genotype probabilities that contains information from each procedure applied upstream.

The HMM used to estimate the genetic distances in **OneMap** software allows easy implementation of the error probabilities in its emission function. The **OneMap** first version was designed to deal with markers with few sources of errors, mostly SSRs. Thus, the error probability used in HMM received a single value of  $10^{-5}$ . This value is not compatible with error probabilities for high-throughput markers. As consequence, high-density genetic maps generated by the software accumulate errors in the genetic distance estimation. The implementation of reliable error probabilities in **OneMap** will allow the software to produce high-density genetic maps with genetic distances compatible with reality (SMITH and NAMBIAR, 2020).

Until now, the probabilities applied in the genotype calling procedure for map building in diploids considered only errors coming from a random sampling of the alleles reads (BILTON *ET AL.*, 2018). Polyploid species present more challenges in genotype calling because their heterozygotes samples can have several possible dosages. The software for calling polyploids genotypes have been more robust in modeling the sources of errors compared with those designed only for diploid species (GERARD *ET AL.*, 2018; SERANG *ET AL.*, 2012; CLARK *ET AL.*, 2019).

In MOLLINARI and GARCIA (2019), a population of 160  $F_1$  offsprings and markers identified using SolCAP Infinium 8303 potato array and genotyped with fitTetra genotype caller were used to build a linkage map for an autotetraploid potato in MAPpoly R package for polyploid map building. The MAPpoly package extended the **OneMap** HMM capabilities to polyploid species evaluations (MOLLINARI and GARCIA, 2019). For this autotetraploid potato map, they applied the genotype probabilities from fitTetra and a global error rate of 5% in the HMM to build a linkage map for the autotetraploid potato with MAPpoly R package. Despite both approaches reduced the linkage group sizes, the authors considered that by applying a global error rate of 5% provided better results. The explanation for that is that a global error rate gives the HMM more flexibility to estimate the genotypes according to the global chromosomal structure. In MOLLINARI *ET AL.* (2020), genotypes probabilities from SuperMASSA (SERANG *ET AL.*, 2012) were used to build a new map for a  $F_1$  population with 315 hexaploid sweet potato individuals. Both polyploid maps mentioned counted with high-quality array data or high-depth sequencing data obtained by an optimized genotyping-by-sequencing protocol (WADL *ET AL.*, 2018).

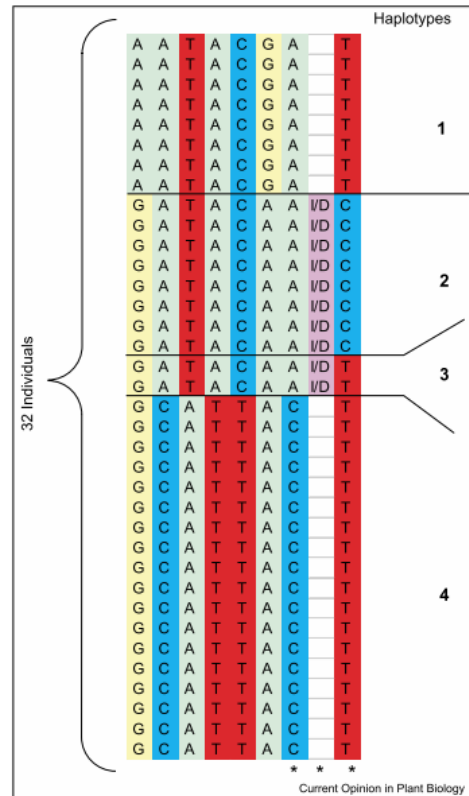
#### 1.1.4 Haplotype-based multiallelic markers

In addition to the genotype errors, the genetic mapping software are also facing a limited level of information of high-throughput markers. It is caused by the biallelic nature of SNPs (VIGNAL *ET AL.*, 2008). In theory, there are four possible bases to differentiate a single position in the genome; in practice, SNPs usually are restricted to only two. This happens because of the low frequency of single nucleotide substitution at the genomes. For *Arabidopsis thaliana* it was estimated a spontaneous mutation rate of  $7 \times 10^{-9}$  base substitutions per site per generation, with a bias favoring transitions ( $G : C \rightarrow A : T$ ) (OSSOWSKI *ET AL.*, 2010). Therefore, the probability of two independent base changes occurring at a single position is very low.

To overcome the low-informativeness, several studies of genome-wide association mapping (GWAS) and genomic prediction combines adjacent biallelic markers in the same disequilibrium block (high LD) into a single multiallelic haplotype. These haplotype-based markers showed more accuracy in association analysis compared with individual SNP (SEHGAL and DREISIGACKER, 2019; ABED and BELZILE, 2019; N'DIAYE *ET AL.*, 2017; GAWENDA *ET AL.*, 2015; LORENZ *ET AL.*, 2010; LIU *ET AL.*, 2008; JIANG *ET AL.*, 2018). N'DIAYE *ET AL.* (2017) pointed several advantages of haplotype-based markers over the single SNP in these studies as the higher capacity of epistatic interactions identification (JIANG *ET AL.*, 2018), more information to estimate identical by descent alleles and reduction of the number of tests and hence the type I error rate.

Haplotypes can be defined as sequences of consecutive loci over a chromosome which shares high similarity with  $k - 1$  other chromosomes in diploid ( $k = 2$ ) and polyploids ( $k > 2$ ) (MOTAZEDI *ET AL.*, 2018a). They do not necessarily present a lack of recombination breakpoints but have a known linkage phase.

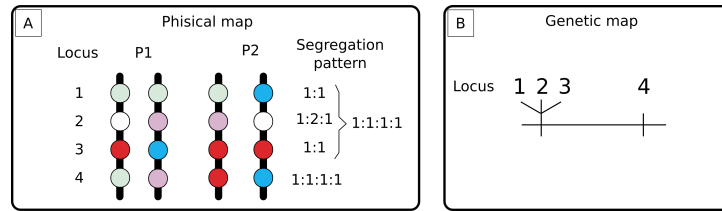
The usage of haplotypes is particularly effective when there is a high level of linkage disequilibrium (LD), which group markers in large blocks in the genome (RAFALSKI, 2002). If there are high LD and a high number of markers available covering the genome, there will be a lot of redundant markers for recombination information, because of the low resolution (see Figure AT28). Despite being considered redundant for recombination, this excess of markers can integrate haplotypes and add more allelic information. As an example, the SNPs highlighted with “\*” in Figure 1.3 can differentiate only two alleles individually, but combined it become a haplotype block able to differentiate four alleles (Figures 1.3 and 1.4).



**Figure 1.3.** Figure from RAFALSKI (2002). Example with eight SNPs and one indel identified in maize. The horizontal rows correspond to 32 individuals. If combined, the markers identify four haplotypes.

There are several methods to build the haplotypes in sequencing data available (BROWNING and BROWNING, 2011). They can be divided into three different approaches: user-defined length, sliding-window, and linkage disequilibrium. In the user-defined length approach, the genome are divided in blocks of sequences with size defined by the user and the SNPs within each sequence are combined into haplotypes. This way, the genome are divided into haplotypes blocks with same length. This is the most simple approach but requires a good quality reference genome and does not consider any biological aspect as the LD, population structure, or evolutionary history. In the sliding-window approach, the target genomic regions of interest are divided into uniform or variable size windows which compose the haplotype blocks, but similarly the user-defined length approach, the sliding-windows does not consider the LD information (SEHGAL and DREISIGACKER, 2019). Both approaches, the user-defined length and the sliding-window are sometimes also called bins (HUANG *ET AL.*, 2009a; XIE *ET AL.*, 2010).

The LD-based approaches are the most advantageous because they consider the biological aspects of the population studied (SEHGAL and DREISIGACKER, 2019). Build genetic maps is one of the methods using the linkage disequilibrium approach. It provides haplotypes referring to the parents and progeny of entire chromosomes with the phasing and measure of genetic distances between all markers. However, as mentioned before, build the entire chromosome haplotype with only biallelic markers require phases estimation comparing many possibilities when dealing with outcrossing populations. This task is even more difficult when thousands of markers are available and some of them contain genotyping errors, requiring many statistical tests and also inserting a bias in the linkage estimations.



**Figure 1.4.** A hypothetical biparental cross of an outcrossing species and the expected genotypes for four loci. Figure A shows the parents' genotypes. Locus 1, 2, and 3 are the SNPs marked with “\*” at figure 1.3. They together differentiate four haplotypes, but separated, they differentiate two alleles at each locus. Figure B shows the hypothetical genetic map, the SNPs 1, 2, and 3 are redundant for recombination information, they belong to the same disequilibrium block.

One idea to optimize the entire chromosome haplotype building via linkage maps is to combine it with another approach of haplotype building that combines SNP marker in smaller blocks of the chromosome. The smaller haplotype blocks estimated may become multiallelic markers for the genetic map, called here haplotype-based markers.

The haplotype-based markers have already being used to assist linkage map building in RILs populations (HUANG *ET AL.*, 2009a; XIE *ET AL.*, 2010; YU *ET AL.*, 2011; XU, 2013). For that, the SNP markers are ordered by physical position in a well-assembled reference genome. Combining the order and the genotypes it is possible to identify the regions that come from each parent and consequently the recombination breakpoints of each individual of the progeny. The approach used to identify the recombination breakpoints is based on a sliding window that evaluates groups of consecutive SNPs. The approach is able to identify genotype errors because isolated genotypes coming from a parent are observed inside a region with predominant genotypes from the other. However, for outcrossing species, it is not possible to identify from which parent each part of the chromosome was inherited without phase estimation, even with genomic positions available.

In the outcrossing context, we can obtain phased genotypes in haplotype-based multiallelic markers using assembly-based SNP callers. The SNP calling procedure is already needed in the linkage map building pipeline. Thus, if we use SNP callers to identify the haplotype-based markers we do not need to include extra steps in the pipeline. Software as PolyBayes (MARTH *ET AL.*, 1999), samtools (LI, 2011), freebayes (GARRISON and MARTH, 2012), GATK (POPLIN *ET AL.*, 2017; MCKENNA *ET AL.*, 2010) and Platypus (RIMMER *ET AL.*, 2014) uses assembly-based haplotyping approaches to identify variants based on the read sequences alignment to a reference genome, not only positions. These software do not focus on mapping populations studies and the approach was implemented with a different purpose than genetic map building. They are all focused on association studies in natural populations, for which genotype imputation is an important step and it is also improved with high informative markers (GARRISON and MARTH, 2012). Also, assembly-based methods based on Bruijn-like implemented in Platypus and GATK graphs increase the accuracy of indels call (RIMMER *ET AL.*, 2014; POPLIN *ET AL.*, 2017).

## References

- ABED, A. and F. BELZILE, 2019 Comparing Single-SNP, Multi-SNP, and Haplotype-Based Approaches in Association Studies for Major Traits in Barley. *The Plant Genome* **12**: 190036.
- ANDREWS, K. R., J. M. GOOD, M. R. MILLER, G. LUIKART, and P. A. HOHENLOHE, 2016 Harnessing the power of RADseq for ecological and evolutionary genomics. *Nature Reviews Genetics* **17**: 81–92.
- ARÚS, P., C. OLARTE, M. ROMERO, and F. VARGAS, 1994 Linkage analysis of ten isozyme genes in F1 segregating almond progenies. *Journal of the American Society for Horticultural Sciences* **119**: 339–344.

- BAIRD, N. A., P. D. ETTER, T. S. ATWOOD, M. C. CURREY, A. L. SHIVER, Z. A. LEWIS, E. U. SELKER, W. A. CRESKO, and E. A. JOHNSON, 2008 Rapid SNP Discovery and Genetic Mapping Using Sequenced RAD Markers. *PLoS ONE* **3**: e3376.
- BARTHOLOMÉ, J., E. MANDROU, A. MABIALA, J. JENKINS, I. NABIHOUDINE, C. KLOPP, J. SCHMUTZ, C. PLOMION, and J.-M. GION, 2015c High-resolution genetic maps of Eucalyptus improve Eucalyptus grandis genome assembly. *New Phytologist* **206**: 1283–1296.
- BAUM, E., T. PETRIE, S. G., and W. N., 1970 A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains. *The Annals of Mathematical Statistics* **41**: 164–171.
- BILTON, T. P., M. R. SCHOFIELD, M. A. BLACK, D. CHAGNÉ, P. L. WILCOX, and K. G. DODDS, 2018 Accounting for Errors in Low Coverage High-Throughput Sequencing Data When Constructing Genetic Maps Using Biparental Outcrossed Populations. *Genetics* **209**: 65–76.
- BRADBURY, P. J., Z. ZHANG, D. E. KROON, T. M. CASSTEVEN, Y. RAMDOSS, and E. S. BUCKLER, 2007 TASSEL: software for association mapping of complex traits in diverse samples. *Bioinformatics* **23**: 2633–2635.
- BROWNING, B. L. and Z. YU, 2009 Simultaneous Genotype Calling and Haplotype Phasing Improves Genotype Accuracy and Reduces False-Positive Associations for Genome-wide Association Studies. *American Journal of Human Genetics* **85**: 847–861.
- BROWNING, S. R. and B. L. BROWNING, 2011 Haplotype phasing: existing methods and new developments. *Nature Publishing Group* **12**: 703–714.
- BUETOW, K. H., 1991 Influence of aberrant observations on high-resolution linkage analysis outcomes. *American Journal of Human Genetics* **49**: 985–994.
- CATCHEN, J., P. A. HOHENLOHE, S. BASSHAM, A. AMORES, and W. A. CRESKO, 2013a Stacks: an analysis tool set for population genomics. *Molecular Ecology* **22**: 3124–40.
- CHACON, S. and B. STRAUB, 2014 *Pro Git*. Apress, USA, second edition.
- CHEEMA, J. and J. DICKS, 2009 Computational approaches and software tools for genetic linkage map estimation in plants.
- CLARK, L. V., A. E. LIPKA, and E. J. SACKS, 2019 polyRAD: Genotype Calling with Uncertainty from Sequencing Data in Polyploids and Diploids. *G3: Genes|Genomes|Genetics* **9**: g3.200913.2018.
- CSÁRDI, G., 2019 *cranlogs: Download Logs from the 'RStudio' 'CRAN' Mirror*.
- DANECEK, P., A. AUTON, G. ABECASIS, C. A. ALBERS, E. BANKS, M. A. DEPRISTO, R. E. HANDSAKER, G. LUNTER, G. T. MARTH, S. T. SHERRY, G. McVEAN, R. DURBIN, and . G. P. A. GROUP, 2011b The variant call format and VCFtools. *Bioinformatics (Oxford, England)* **27**: 2156–2158.
- DAVEY, J. W., P. A. HOHENLOHE, P. D. ETTER, J. Q. BOONE, J. M. CATCHEN, and M. L. BLAXTER, 2011 Genome-wide genetic marker discovery and genotyping using next-generation sequencing. *Nature Reviews Genetics* **12**: 499–510.
- DEMPSTER, A. P., N. M. LAID, and D. B. RUBIN, 1977 Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B* **39**: 1–38.

- DER AUWERA, G. A., G. VAN DER AUWERA, and B. D. O'CONNOR, 2020 *Genomics in the Cloud: Using Docker, GATK, and WDL in Terra*. O'Reilly Media, Incorporated.
- DIMENSIONS, 2021 OneMap publications analytical views.
- EDDELBUETTEL, D., 2013 *Seamless {R} and {C++} Integration with {Rcpp}*. Springer, New York.
- EDDELBUETTEL, D. and J. J. BALAMUTA, 2017 Extending extit{R} with extit{C++}: A Brief Introduction to extit{Rcpp}. *PeerJ Preprints* **5**: e3188v1.
- EDDELBUETTEL, D. and R. FRANÇOIS, 2011 {Rcpp}: Seamless {R} and {C++} Integration. *Journal of Statistical Software* **40**: 1–18.
- ELSHIRE, R. J., J. C. GLAUBITZ, Q. SUN, J. A. POLAND, K. KAWAMOTO, E. S. BUCKLER, and S. E. MITCHELL, 2011 A robust, simple genotyping-by-sequencing (GBS) approach for high diversity species. *PLoS ONE* **6**: e19379.
- GARCIA, A. A. F., E. A. KIDO, A. N. MEZA, H. M. B. SOUZA, L. R. PINTO, M. M. PASTINA, C. S. LEITE, J. A. G. DA SILVA, E. C. ULIAN, A. FIGUEIRA, and A. P. SOUZA, 2006a Development of an integrated genetic map of a sugarcane (*Saccharum* spp.) commercial cross, based on a maximum-likelihood approach for estimation of linkage and linkage phases. *Theoretical and Applied Genetics* **112**: 298–314.
- GARCIA, A. A. F., E. A. KIDO, A. N. MEZA, H. M. B. SOUZA, L. R. PINTO, M. M. PASTINA, C. S. LEITE, J. A. G. DA SILVA, E. C. ULIAN, A. V. FIGUEIRA, A. P. SOUZA, J. A. G. DA SILVA, E. C. ULIAN, A. V. FIGUEIRA, and A. P. SOUZA, 2006b Development of an integrated genetic map of a sugarcane (*Saccharum* spp.) commercial cross, based on a maximum-likelihood approach for estimation of linkage and linkage phases. *TAG. Theoretical and applied genetics* **112**: 298–314.
- GARRIDO-CARDENAS, J. A., C. MESA-VALLE, and F. MANZANO-AGUGLIARO, 2018 Trends in plant research using molecular markers. *Planta* **247**: 543–557.
- GARRISON, E. and G. MARTH, 2012 Haplotype-based variant detection from short-read sequencing. *ArXiv e-prints* p. 9.
- GAWENDA, I., P. THORWARTH, T. GÜNTHER, F. ORDON, and K. J. SCHMID, 2015 Genome-wide association studies in elite varieties of German winter barley using single-marker and haplotype-based methods. *Plant Breeding* **134**: 28–39.
- GERARD, D., L. F. V. FERRÃO, A. A. F. GARCIA, and M. STEPHENS, 2018 Genotyping Polyploids from Messy Sequencing Data. *Genetics* **210**: 789–807.
- GOLDSTEIN, D. R., H. ZHAO, and T. P. SPEED, 1997 The Effects of Genotyping Errors and Interference on Estimation of Genetic Distance. *Human Heredity* **47**: 86–100.
- GONEN, S., N. R. LOWE, T. CEZARD, K. GHARBI, S. C. BISHOP, and R. D. HOUSTON, 2014 Linkage maps of the Atlantic salmon (*Salmo salar*) genome derived from RAD sequencing. *BMC Genomics* **15**: 166.
- GRATTAPAGLIA, D. and R. SEDEROFF, 1994 Genetic linkage maps of *Eucalyptus grandis* and *Eucalyptus urophylla* using a pseudo-testcross: mapping strategy and RAPD markers. *Genetics* **137**: 1121–37.
- GROVER, A. and P. C. SHARMA, 2014 Development and use of molecular markers: past and present. *Critical reviews in biotechnology* **8551**: 1–13.

- HACKETT, C. A. and L. B. BROADFOOT, 2003 Effects of genotyping errors, missing values and segregation distortion in molecular marker data on the construction of linkage maps. *Heredity* **90**: 33–38.
- HEATHER, J. M. and B. CHAIN, 2016 The sequence of sequencers: The history of sequencing DNA. *Genomics* **107**: 1–8.
- HUANG, X., Q. FENG, Q. QIAN, Q. ZHAO, L. WANG, A. WANG, J. GUAN, D. FAN, Q. WENG, T. HUANG, and OTHERS, 2009a High-throughput genotyping by whole-genome resequencing. *Genome research* pp. 1068–1076.
- JANSEN, R. C. and J.-P. NAP, 2001 Genetical genomics: the added value from segregation. *Trends in genetics* **17**: 388–391.
- JEENNOR, S. and H. VOLKAERT, 2014 Mapping of quantitative trait loci (QTLs) for oil yield using SSRs and gene-based markers in African oil palm (*Elaeis guineensis* Jacq.). *Tree Genetics & Genomes* **10**: 1–14.
- JIANG, Y., R. H. SCHMIDT, and J. C. REIF, 2018 Haplotype-Based Genome-Wide Prediction Models Exploit Local Epistatic Interactions Among Markers. *G3: Genes|Genomes|Genetics* **49**: g3.300548.2017.
- KANG, H., Z. S. QIN, T. NIU, and J. S. LIU, 2004 Incorporating genotyping uncertainty in haplotype inference for single-nucleotide polymorphisms. *American journal of human genetics* **74**: 495–510.
- KIM, C., H. GUO, W. KONG, R. CHANDNANI, L.-S. SHUANG, A. H. PATERSON, ET AL KIM, C. KIM, H. GUO, W. KONG, R. CHANDNANI, L.-S. SHUANG, A. H. PATERSON, ET AL KIM, C. KIM, H. GUO, W. KONG, R. CHANDNANI, L.-S. SHUANG, and A. H. PATERSON, 2015 Application of genotyping by sequencing technology to a variety of crop breeding programs. *Plant Science* **242**: 1–9.
- LANDER, E. S. and P. GREEN, 1987 Construction of multilocus genetic linkage maps in humans. *Proc. Natl. Acad. Sci USA* **84**: 2363–2367.
- LI, H., 2011 A statistical framework for SNP calling, mutation discovery, association mapping and population genetical parameter estimation from sequencing data. *Bioinformatics* **27**: 2987–2993.
- LI, H., B. HANDSAKER, A. WYSOKER, T. FENNEL, J. RUAN, N. HOMER, G. MARTH, G. ABECASIS, and R. DURBIN, 2009a The sequence alignment/map format and SAMtools. *Bioinformatics* **25**: 2078–2079.
- LIU, C., S. SUKUMARAN, D. JARQUIN, J. CROSSA, S. DREISIGACKER, C. SANSALONI, and M. REYNOLDS, 2020 Comparison of array- and sequencing-based markers for genome-wide association mapping and genomic prediction in spring wheat. *Crop Science* **60**: 211–225.
- LIU, N., K. ZHANG, and H. B. T. A. I. G. ZHAO, 2008 Haplotype-Association Analysis. In *Genetic Dissection of Complex Traits*, volume 60, pp. 335–405, Academic Press.
- LORENZ, A. J., M. T. HAMBLIN, and J. L. JANNINK, 2010 Performance of single nucleotide polymorphisms versus haplotypes for genome-wide association analysis in barley. *PLoS ONE* **5**: 1–11.
- MALIEPAARD, C., J. JANSEN, and J. W. V. OOIJEN, 1997 Linkage analysis in a full-sib family of an outbreeding plant species : overview and consequences for applications. *Genetical Research* **70**: 237–250.
- MARGARIDO, G. R. A., A. P. SOUZA, and A. A. F. GARCIA, 2007 OneMap: software for genetic mapping in outcrossing species. *Hereditas* **144**: 78–9.



- MARTH, G. T., I. KORF, M. D. YANDELL, R. T. YEH, Z. GU, H. ZAKERI, N. O. STITZIEL, L. D. HILLIER, P. Y. KWOK, and W. R. GISH, 1999 A general approach to single-nucleotide polymorphism discovery. *Nature Genetics* **23**: 452–456.
- McKENNA, A., M. HANNA, E. BANKS, A. SIVACHENKO, K. CIBULSKIS, A. KERNYTSKY, K. GARIMELLA, D. ALTSHULER, S. GABRIEL, M. DALY, and M. A. DEPRISTO, 2010 The Genome Analysis Toolkit: A MapReduce framework for analyzing next-generation DNA sequencing data. *Genome Research* **20**: 1297–1303.
- MERKEL, D., 2014a Docker : Lightweight Linux Containers for Consistent Development and Deployment Docker : a Little Background Under the Hood. *Linux Journal* **2014**: 2–7.
- MIELCZAREK, M. and J. SZYDA, 2016 Review of alignment and SNP calling algorithms for next-generation sequencing data. *Journal of Applied Genetics* **57**: 71–79.
- MOLLINARI, M. and A. A. F. GARCIA, 2019 Linkage Analysis and Haplotype Phasing in Experimental Autopolyploid Populations with High Ploidy Level Using Hidden Markov Models. *G3: Genes|Genomes|Genetics* **9**: 3297–3314.
- MOLLINARI, M., B. A. OLUKOLU, G. D. S. PEREIRA, A. KHAN, D. GEMENET, G. C. YENCHO, and Z.-B. ZENG, 2020 Unraveling the Hexaploid Sweetpotato Inheritance Using Ultra-Dense Multilocus Mapping. *G3: Genes|Genomes|Genetics* **10**: 281–292.
- MOTAZEDI, E., D. DE RIDDER, R. FINKERS, S. BALDWIN, S. THOMSON, K. MONAGHAN, and C. MALIEPAARD, 2018a TriPoly: haplotype estimation for polyploids using sequencing data of related individuals. *Bioinformatics* pp. 1–9.
- NADEEM, M. A., M. A. NAWAZ, M. Q. SHAHID, Y. DOĞAN, G. COMERTPAY, M. YILDIZ, R. HATİPOĞLU, F. AHMAD, A. ALSALEH, N. LABHANE, H. ÖZKAN, G. CHUNG, and F. S. BALOCH, 2017 DNA molecular markers in plant breeding: current status and recent advancements in genomic selection and genome editing. *Biotechnology and Biotechnological Equipment* .
- N'DIAYE, A., J. K. HAILE, A. T. CORY, F. R. CLARKE, J. M. CLARKE, R. E. KNOX, and C. J. POZNIAK, 2017 Single marker and haplotype-based association analysis of semolina and pasta colour in elite durum wheat breeding lines using a high-density consensus map. *PLoS ONE* **12**: 1–24.
- NIELSEN, G. and J. G. SCANDALIOS, 1974 Chromosomal Location by Use of Trisomics and New Alleles of an Endopeptidase in ZEA MAYS. *Genetics* **77**: 679–86.
- NIELSEN, R., J. S. PAUL, A. ALBRECHTSEN, and Y. S. SONG, 2011 Genotype and SNP calling from next-generation sequencing data. *Nature Reviews Genetics* **12**: 443–451.
- OLIVEIRA, E. J., M. L. C. VIEIRA, A. A. F. GARCIA, C. F. MUNHOZ, G. R. MARGARIDO, L. CONSOLI, F. P. MATTA, M. C. MORAES, M. I. ZUCCHI, and M. H. P. FUNGARO, 2008a An Integrated Molecular Map of Yellow Passion Fruit Based on Simultaneous Maximum-likelihood Estimation of Linkage and Linkage Phases. *Journal of the American Society for Horticultural Science* **133**: 35–41.
- OSSOWSKI, S., K. SCHNEEBERGER, J. I. LUCAS-LLEDÓ, N. WARTHMAN, R. M. CLARK, R. G. SHAW, D. WEIGEL, and M. LYNCH, 2010 The Rate and Molecular Spectrum of Spontaneous Mutations in *Arabidopsis thaliana*. *Science* **327**: 92–94.
- PETERSON, B. K., J. N. WEBER, E. H. KAY, H. S. FISHER, and H. E. HOEKSTRA, 2012 Double Digest RADseq: An Inexpensive Method for De Novo SNP Discovery and Genotyping in Model and Non-Model Species. *PLoS ONE* **7**: e37135.

- POMPANON, F., A. BONIN, E. BELLEMAIN, and P. TABERLET, 2005 Genotyping errors: causes, consequences and solutions. *Nature Reviews Genetics* **6**: 847–846.
- POPLIN, R., V. RUANO-RUBIO, M. A. DEPRISTO, T. J. FENNELL, M. O. CARNEIRO, G. A. V. DER AUWERA, D. E. KLING, L. D. GAUTHIER, A. LEVY-MOONSHINE, D. ROAZEN, K. SHAKIR, J. THIBAUT, S. CHANDRAN, C. WHELAN, M. LEK, S. GABRIEL, M. J. DALY, B. NEALE, D. G. MACARTHUR, and E. BANKS, 2017 Scaling accurate genetic variant discovery to tens of thousands of samples. *bioRxiv* p. 201178.
- PREEDY, K. F. and C. A. HACKETT, 2016 A rapid marker ordering approach for high-density genetic linkage maps in experimental autotetraploid populations using multidimensional scaling. *Theoretical and Applied Genetics* .
- RAFALSKI, A., 2002 Applications of single nucleotide polymorphisms in crop genetics. *Current Opinion in Plant Biology* **5**: 94–100.
- RASTAS, P., 2017 Lep-MAP3: Robust linkage mapping even for low-coverage whole genome sequencing data. *Bioinformatics* **33**: 3726–3732.
- RIMMER, A., H. PHAN, I. MATHIESON, Z. IQBAL, S. R. TWIGG, A. O. WILKIE, G. MCVEAN, and G. LUNTER, 2014 Integrating mapping-, assembly- and haplotype-based approaches for calling variants in clinical sequencing applications. *Nature Genetics* **46**: 912–918.
- RITTER, E., C. GEBHARDT, and F. SALAMINI, 1990 Estimation of recombination frequencies and construction of RFLP linkage maps in plants from crosses between heterozygous parents. *Genetics* **125**: 645–654.
- RITTER, E. and F. SALAMINI, 1996 The calculation of recombination frequencies in crosses of allogamous plant species with applications to linkage mapping. *Genetical Research* **67**: 55.
- RIVERA-COLÓN, A. G., N. C. ROCHETTE, and J. M. CATCHEN, 2020 Simulation with RADinitio improves RADseq experimental design and sheds light on sources of missing data. *Molecular Ecology Resources* pp. 1–16.
- ROBINSON, J. T., H. THORVALDSDÓTTIR, W. WINCKLER, M. GUTTMAN, E. S. LANDER, G. GETZ, and J. P. MESIROV, 2011 Integrative genomics viewer. *Nature Biotechnology* **29**: 24–26.
- SCHIFFTHALER, B., C. BERNHARDSSON, P. K. INGVARSSON, and N. R. STREET, 2017 BatchMap: A parallel implementation of the OneMap R package for fast computation of F1 linkage maps in outcrossing species. *PLoS ONE* **12**: 1–12.
- SEEB, J. E., G. CARVALHO, L. HAUSER, K. NAISH, S. ROBERTS, and L. W. SEEB, 2011 Single-nucleotide polymorphism (SNP) discovery and applications of SNP genotyping in nonmodel organisms. *Molecular Ecology Resources* **11**: 1–8.
- SEHGAL, D. and S. DREISIGACKER, 2019 Haplotypes-based genetic analysis: Benefits and challenges. *Vavilovskii Zhurnal Genetiki i Seleksii* **23**: 803–808.
- SERANG, O., M. MOLLINARI, and A. A. F. GARCIA, 2012 Efficient exact maximum a posteriori computation for bayesian SNP genotyping in polyploids. *PLoS ONE* **7**: 1–13.
- SILVA-JUNIOR, O. B., D. A. FARIA, and D. GRATTAPAGLIA, 2015 A flexible multi-species genome-wide 60K SNP chip developed from pooled resequencing of 240 Eucalyptus tree genomes across 12 species pp. 1527–1540.

- SMITH, G. R. and M. NAMBIAR, 2020 New Solutions to Old Problems: Molecular Mechanisms of Meiotic Crossover Control. *Trends in Genetics* **36**: 337–346.
- SOUZA, L. M., R. GAZAFFI, C. C. MANTELLO, C. C. SILVA, D. GARCIA, V. LE GUEN, S. E. A. CARDOSO, A. A. F. GARCIA, and A. P. SOUZA, 2013 QTL Mapping of Growth-Related Traits in a Full-Sib Family of Rubber Tree (*Hevea brasiliensis*) Evaluated in a Sub-Tropical Climate. *PLoS ONE* **8**: e61238.
- TANIGUTI, C. H., 2017 *Construção do mapa genético integrado em uma progênie de irmãos-completos proveniente do cruzamento entre Eucalyptus grandis e Eucalyptus urophylla*. Ph.D. thesis, Universidade de São Paulo, Piracicaba.
- THE SOFTWARE QUALITY COMPANY TIOBE, 2021 TIOBE Index for January 2021.
- VIGNAL, A., D. MILAN, M. SANCRISTOBAL, and A. EGGEN, 2008 A review on SNPs and other types of molecular markers. *Genetics, selection, evolution : GSE* **40**: 241–264.
- VOSS, K., J. GENTRY, and G. V. D. AUWERA, 2017 Full-stack genomics pipelining with GATK4+WDL+ Cromwell [version 1; not peer reviewed]. *F1000Research* p. 4.
- WADL, P. A., B. A. OLUKOLU, S. E. BRANHAM, R. L. JARRET, G. C. YENCHO, and D. M. JACKSON, 2018 Genetic Diversity and Population Structure of the USDA Sweetpotato (*Ipomoea batatas*) Germplasm Collections Using GBSpoly. *Frontiers in Plant Science* **9**: 1166.
- WU, R., C.-X. MA, S. S. WU, and Z.-B. ZENG, 2002c Linkage mapping of sex-specific differences. *Genetical research* **79**: 85–96.
- WU, R., C.-X. C.-X. MA, I. PAINTER, and Z.-B. Z.-B. ZENG, 2002d Simultaneous maximum likelihood estimation of linkage and linkage phases in outcrossing species. *Theoretical population biology* **61**: 349–363.
- XIE, W., Q. FENG, H. YU, X. HUANG, Q. ZHAO, Y. XING, S. YU, B. HAN, and Q. ZHANG, 2010 Parent-independent genotyping for constructing an ultrahigh-density linkage map based on population sequencing. *Proceedings of the National Academy of Sciences* **107**: 10578–10583.
- XIONG, J. S., S. E. MCKEAND, F. ISIK, J. WEGRZYN, D. B. NEALE, Z.-B. ZENG, L. DA COSTA E SILVA, and R. W. WHETTEN, 2016 Quantitative trait loci influencing forking defects in an outbred pedigree of loblolly pine. *BMC genetics* **17**: 138.
- XU, C., Y. REN, Y. JIAN, Z. GUO, Y. ZHANG, C. XIE, J. FU, H. WANG, G. WANG, Y. XU, P. LI, and C. ZOU, 2017 Development of a maize 55 K SNP array with improved genome coverage for molecular breeding. *Molecular breeding : new strategies in plant improvement* **37**: 20.
- XU, S., 2013 Genetic mapping and genomic selection using recombination breakpoint data. *Genetics* **195**: 1103–1115.
- YU, H., W. XIE, J. WANG, Y. XING, C. XU, X. LI, J. XIAO, and Q. ZHANG, 2011 Gains in QTL detection using an ultra-high density SNP map based on population sequencing relative to traditional RFLP/SSR markers. *PLoS ONE* **6**.
- YU, Z., F. DU, R. BAN, and Y. ZHANG, 2020 SimuSCoP: Reliably simulate Illumina sequencing data based on position and context dependent profiles. *BMC Bioinformatics* **21**: 1–18.

## 2 READS2MAP: PRACTICAL AND REPRODUCIBLE WORKFLOWS TO BUILD LINKAGE MAPS FROM SEQUENCING DATA

### Abstract

The high-throughput genotyping methods make available millions of read sequences for genetics research. To build genetic maps from read sequences, researchers need to perform diverse bioinformatics and statistical analysis. For each step of the analysis there are several tools available, all with different methods and parameters to be selected by users. In this context, users struggle to find the best combination of software and parameters to be used and also to create reproducible pipelines for that. Workflows systems such as **Workflow Description Language** (WDL) offers a structure to organize the entire pipeline, producing a fixed structure and metadata of each step. It also guarantees reproducibility by making interface with containers environments, such as **Docker**. Here we present two workflows: **EmpiricalReads2Map** and **SimulatedReads2Map** to build linkage maps with empirical and simulated sequencing data. **SimulatedReads2Map** simulates sequencing data using **SimusCoP** for whole-genome sequencing (WGS) and exome DNA libraries; and **RADinitio** for restriction site associated DNA sequencing RADseq library. In both workflows, the analyses are performed using **GATK**, **freebayes** as SNP callers and genotype callers; **updog**, **polyRAD**, **SuperMASSA** as genotype callers; and **OneMap** and **GUSMap** as linkage map builders. We also present the shiny app **Reads2Map** to evaluate graphically the results of workflows. We exemplify their usage running the workflows and selecting the method to re-build a linkage map of *Populus tremula*. We obtained a linkage map with 6936 markers and 3299.961 cM selecting **freebayes** as SNP and genotype caller and **OneMap** as map builder.

Key words: Genetic maps; Workflow; Container; Reproducibility.

### 2.1 Background

The advances in sequencing technologies and the development of different library protocols make available millions of genetic markers able to genotype hundreds of samples in a single sequencing run. For building linkage maps, having more markers and larger sample size better is the capacity of locating where in the genome the recombination events occur (higher genetic map resolution). Also, markers from sequencing technologies overcome efforts with molecular laboratory methods and can be considered more affordable (HEATHER and CHAIN, 2016). However, it increases significantly the computer analysis efforts because of the need of using multiple analytic tools and their application to hundreds of experimental samples.

Once the samples are sequenced, many steps follow before being able to build a map. These steps can be divided as the preprocessing of reads, with demultiplexing and cleaning; the alignment to a reference genome; the SNP calling; the genotype calling; the filtering of markers; the grouping; ordering; phases and genetic distances estimations. For each of these steps, there are multiple software available and each software has specific parameters for each feature.

Users that are not familiar with bioinformatic pipelines may find difficult to run their analysis and find the best pipeline for their specific dataset. Performing separately each step can generate several independent scripts and unconnected intermediate files compromising its FAIR Data Principles (Findable, Accessible, Interoperable and Reusable - (WILKINSON ET AL., 2016)).

Metascience studies highlight a big lack of reproducibility in science, sometimes even called as “reproducibility crises” (MUNAFÒ ET AL., 2017), which have motivated research to adopt new approaches to overcome this concerning issue. For computational methods the lack of reproducibility can be measured by trying to run codes from five or ten years ago. The possibility of success is very low. In 2019, the

Nature journal (NATURE, 2019) even promoted the “Ten Years Reproducibility Challenge” launched by ReScience C journal (RESCIENCE, 2019) to researchers try to do that with their own codes.

During all the map building procedure there are some important parts with decisions that need to be made carefully according to the study of biological aspects, such as the chosen software and their specific parameters. However, other parts are only standard procedures as file conversions, software installation, and computational resources optimization, that can be a lot time-consuming and reduce the research quality if made in a wrong way (REITER *ET AL.*, 2020).

Workflow systems are a good solution to overcome technical issues, allowing users to focus on the important decision and also reach FAIR practices. It integrates the analysis in a single structure connecting each step by inputs and outputs. The steps, here called tasks, can also be combined in sub-workflows to users who want to run only specific parts of the analysis. Intermediate files are stored and organized with a standardized structure together with metadata and reports of parameters used in each task. Furthermore, conditional structures can be used to manage exceptions and options. The workflow organized structure makes it possible to generate flowcharts automatically given a complete overview of the workflow (REITER *ET AL.*, 2020).

Workflow systems also provide built-in tools to monitor and manage resource usage. The Cromwell Execution Engine can execute workflows on any computing platform (local, High Performance Computing - HPC or cloud) (VOSS *ET AL.*, 2017). Furthermore, the integration with containers like Docker (MERKEL, 2014b) and singularity (KURTZER *ET AL.*, 2017) overcome the need for software installation and offers specific software versions, making it significantly more reproducible.

Snakemake (GRÜNING *ET AL.*, 2018), Nextflow (DI TOMMASO *ET AL.*, 2017), CWL (AMSTUTZ *ET AL.*, 2016) and WDL (VOSS *ET AL.*, 2017) can be cited as four of the most widely used bioinformatics workflows system. The Workflow Description Language (WDL) presents advantages for production-level pipelines because of its capacity to deal with hundreds or thousands of samples (REITER *ET AL.*, 2020). It is successfully used by Genome Analysis Toolkit (GATK) team (VAN DER AUWERA *ET AL.*, 2013) to provide read-to-results **Best Practices** workflows that can be executed in the Terra platform (TERRA.BIO, 2020). With these optimized and tested workflows available, researchers do not need to learn the workflow system syntax to make use of its benefits.

In this work, we developed workflows to perform the analysis from read sequencing to linkage maps including some of the most used software for the SNP and genotype calling and map build: **freebayes**, GATK (POPLIN *ET AL.*, 2017; MCKENNA *ET AL.*, 2010) as SNP and genotype callers; **updog** (GERARD *ET AL.*, 2018), **polyRAD** (CLARK *ET AL.*, 2019), **SuperMASSA** (SERANG *ET AL.*, 2012) as genotype caller; **OneMap 3.0** and **GUSMap** (BILTON *ET AL.*, 2018) as linkage map builders. The provided WDL workflow performs sequencing reads simulations based on user empirical library protocol and dataset. With this cross-platform tool, users can make optimized usage of their time and available computer resources to focus on the important decisions about the best software, algorithms, and parameters for their dataset. The simulation studies can validate the analytical methods applied and indicate which requires adaptations or improvements.

The genetic map itself can also be a powerful tool to validate upstream methods. Wrong decisions in any one of the upstream steps can be identified in the outputted map, once errors make the map proprieties dissociate of biological concepts. For example, genotyping errors can generate an inflated map size showing an excessive number of recombinations during the meiosis. Since the first genetic map studies by Sturtevant in 1915, it is observed that is unlikely that crossing-overs happen too close to each other, a phenomenon described as interference (STURTEVANT, 1915). Recent studies also described meiosis molecular mechanisms confirming the low expected number of recombination events during the meiosis (SMITH and NAMBIAR, 2020). The **OneMap** package to build linkage maps for inbred (RILs,  $F_2$  intercross, and backcross) and outcrossing mapping populations (MARGARIDO *ET AL.*, 2007) have the

biological concepts of meiosis as assumptions and provides graphical tools to diagnose the dissociation between them.

Besides the proposed workflows, we also integrated them with interactive visualizations in a shiny app to visualize generated maps and intermediary results. The R package shiny (CHANG *ET AL.*, 2020) allows to build interactive web pages, where the results can be visualized in real-time according to user-specifiable parameters.

## 2.2 Implementation

Workflow Description Language (WDL) was used to generate two main workflows: i) **EmpiricalReads2Map**, that has as input the FASTQ files from empirical data, and performs SNP and genotype calling and map building (Figure 2.1); ii) **SimulatedReads2Map**: performs simulations of RADseq Illumina FASTQ files, the SNP and genotype calling, and map building (Figure 2.2).

The two main workflows are composed of sub-workflows, which are composed of tasks. Each task has a public available Docker hub (MERKEL, 2014a) image defined to perform the analysis in containers. The containers images are lightweight, standalone, executable packages of software that includes everything needed to run the specific task: code, runtime, system tools, system libraries and settings (MERKEL, 2014a). The DOCKERFILES with specifications to generate the images are also available together with the workflows. Twenty images are used in total (Table A.1).

The Broad Institute Workflow Object Model (WOM) tools were here applied to validate the workflows, generate its input files templates, and the Graphviz (ELLSON *ET AL.*, 2003) files to draw the graphic (Figures 2.1, 2.2, 2.3). Cromwell Execution Engine was used to run the analysis. An example of a command to run the workflows via prompt terminal and via cromwell server is available in the appendix listing 4.1 and Figure A.2.

### 2.2.1 Pre-processing reads

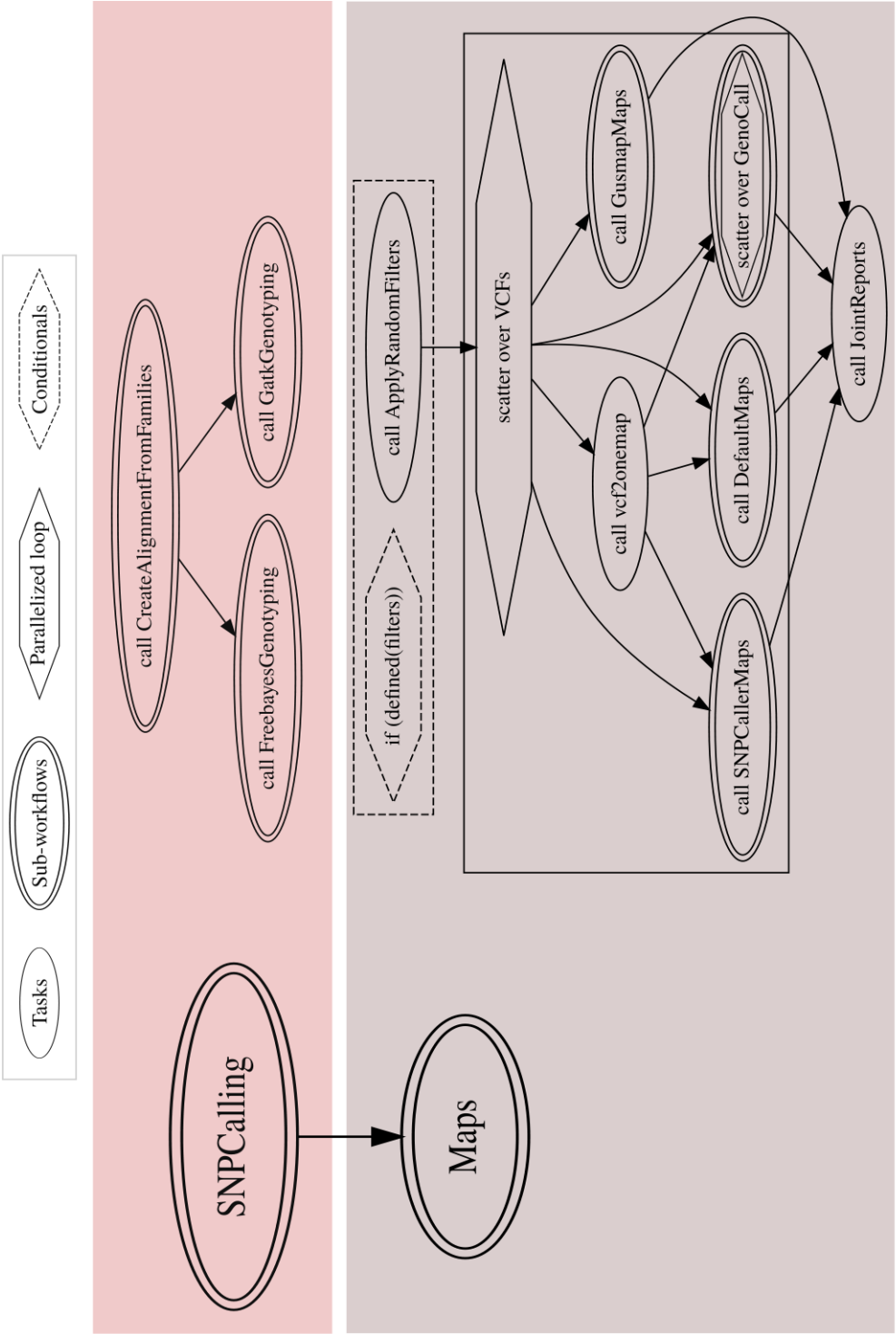
The **EmpiricalReads2Map** workflow requires demultiplexed FASTQ files already filtered for quality measurements. We made available an auxiliary workflow to perform these procedures to reads coming from restriction-site associated DNA sequencing (RADseq), technologies using one or two restriction enzymes (Figure 2.3). The workflow is composed of two tasks. The **PreprocessingReads** task uses the STACKS plugin *process\_radtags* (CATCHEN *ET AL.*, 2013a) to perform the demultiplexing, filter reads by the presence/absence of enzyme cut site, and by reads quality. The task **RemoveAdapt** uses *cutadapt* (MARTIN, 2011) to remove adapters from sequences and filter them by reads length.

The parameters to be used in this step depends on the sequencer and protocol used. We suggest defining them after an overview of available sequences using FASTQC (ANDREWS, 2010) and MULTIQC (EWELS *ET AL.*, 2016).

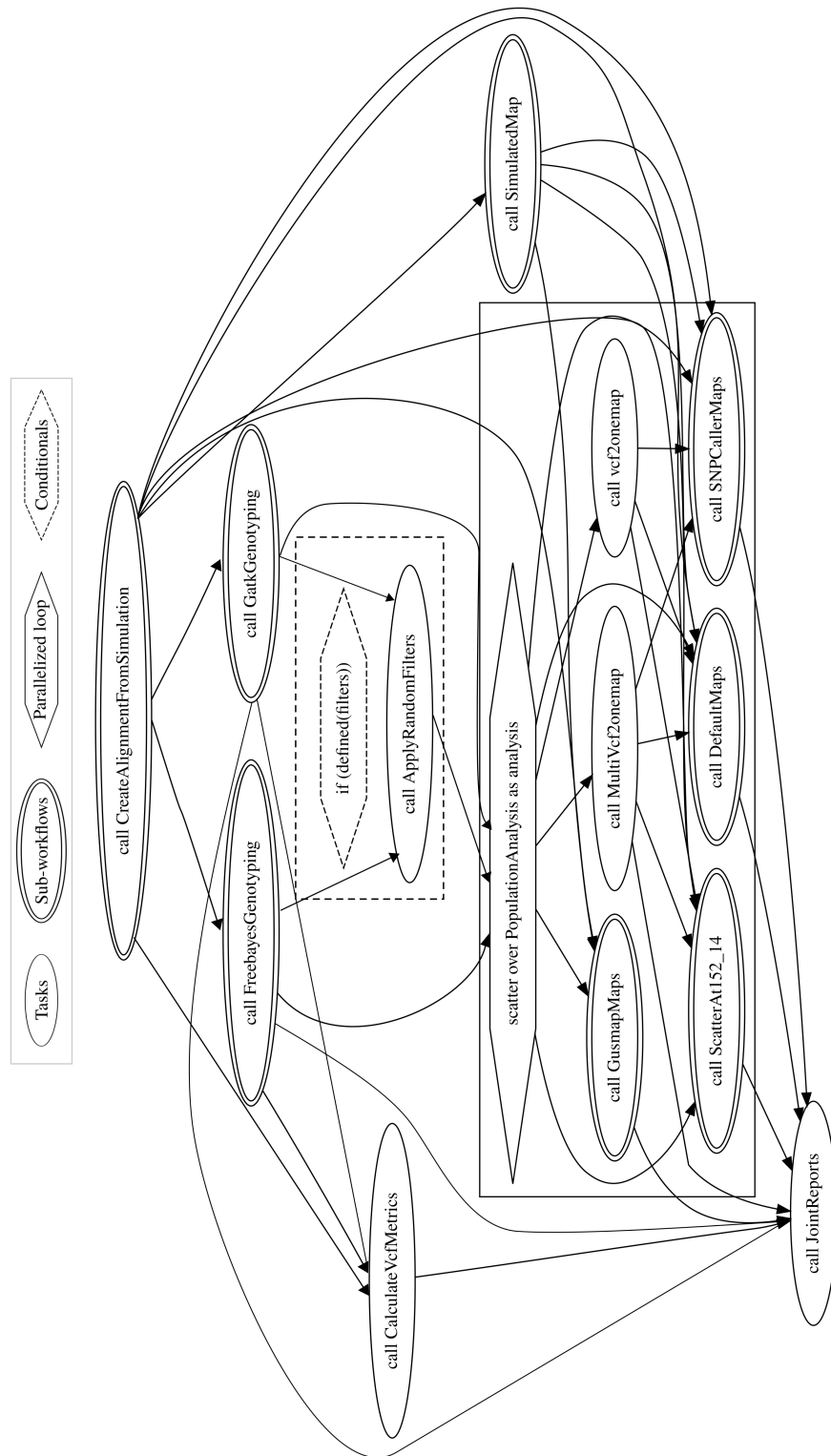
In this work, we evaluate the dataset from ZHIGUNOV *ET AL.* (2017). It is composed of RADseq sequences for 122 F1 progenies and the two parents of a bi-parental cross of *Populus tremula*. The enzymes *HindIII* and *NlaIII* were used in their library protocol and specified in the **PreprocessingReads** workflow task. The reads with uncalled bases were filtered. Reads with average quality Phred score below 10 within a sliding window with half of the total read size were discarded. The Illumina Universal Adapter was removed and reads filtered by a minimum length of 64.

### 2.2.2 Sequencing reads simulation

The simulation was based on a given reference genome chromosome sequence. If a reference linkage map and a VCF file are provided, the workflow simulates the marker genetic distances and parents'

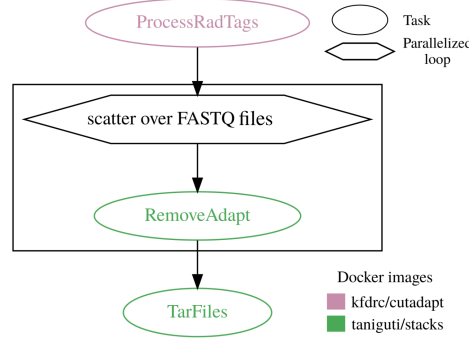


**Figure 2.1.** Workflow **EmpiricalReads2Map** to perform SNP and genotype calling, and build linkage maps.



**Figure 2.2.** Workflow **SimulatedReads2Map** to perform FASTQ files simulations of RADseq, exome or WGS reads, SNP and genotype call, and linkage map building.





**Figure 2.3.** **PreprocessingReads** workflow to filter FASTQ sequences.

genotype frequencies based on them. A cubic spline interpolation with Hyman method (HYMAN, 1983) is applied to simulate centimorgan position for each marker physical position based on this same relation in the reference linkage map provided. The genotype frequencies of parents define the frequency of marker types available for the  $F_1$  population (Table A.2).

If a reference linkage map is not provided, a uniform relation between genetic and physical distance is applied according to a given recombination rate measure in centimorgan by megabase pairs. If a VCF file is not provided, polymorphisms are simulated using **plRS** (HU *ET AL.*, 2012a) software and parents genotypes frequency must be provided by the user.

**PedigreeSim** software (VOORRIPS and MALIEPAARD, 2012) simulates the meiosis event to generate  $F_1$  progenies. The genetic map is simulated considering Haldane (HALDANE, 1919) map function. **PedigreeSim** output files are converted to VCF file using **OneMap** 3.0 function *pedsim2vcf* (Figure 2.4).

Users can also define which library protocol will be simulated. The single and double digest restriction-site associated DNA (sdRAD and ddRAD), exome and whole-genome sequencing (WGS) sequencing libraries are available. RADseq reads are simulated with **RADinitio** (RIVERA-COLÓN *ET AL.*, 2020). This software presents several parameters such as the number of PCR cycles, enzymes, reads, and sequence length. **RADinitio** outputs FASTA files, a Phred score of 40 (probability of incorrect base call is 0.0001) is added to the FASTA sequences using **SEQTK** (LI, 2020) to make its conversion to FASTQ, needed for the next steps.

For exome and WGS reads simulation we developed a package called **simuscopR** with wrapper functions for **SimuSCoP** software (YU *ET AL.*, 2020) and functions to convert VCF to its input files. The **simuscopR** package is available in <https://github.com/Cristianetaniguti/simuscopR>. For exome simulations, users must provide an alignment BAM file to define the exons regions.

To exemplify the usage, we simulate ddRAD reads for five families of 200  $F_1$  individuals from a bi-parental cross using 38% of the chromosome 10 of *Populus trichocarpa* reference genome (TUSKAN *ET AL.*, 2006) version 4.0, the linkage map, and the VCF file outputted by the **EmpiricalReads2Map** workflows. The mean depth of the simulation was set to 20 and the parent's depth was eight times higher (160). An example of workflow input JSON file and details of this specific example can also be found in appendix listing 4.2.

### 2.2.3 SNP calling

The **EmpiricalReads2Map** and **SimulatedReads2Map** share the same sub-workflows for SNP calling. The sequences in FASTQ files are aligned to a reference genome using **BWA-MEM** (LI, 2013). The sample and library information are kept in the BAM file header for further analysis. Alignment BAM files for each sample are inputs for sub-workflows with **GATK** and **freebayes** approaches. We

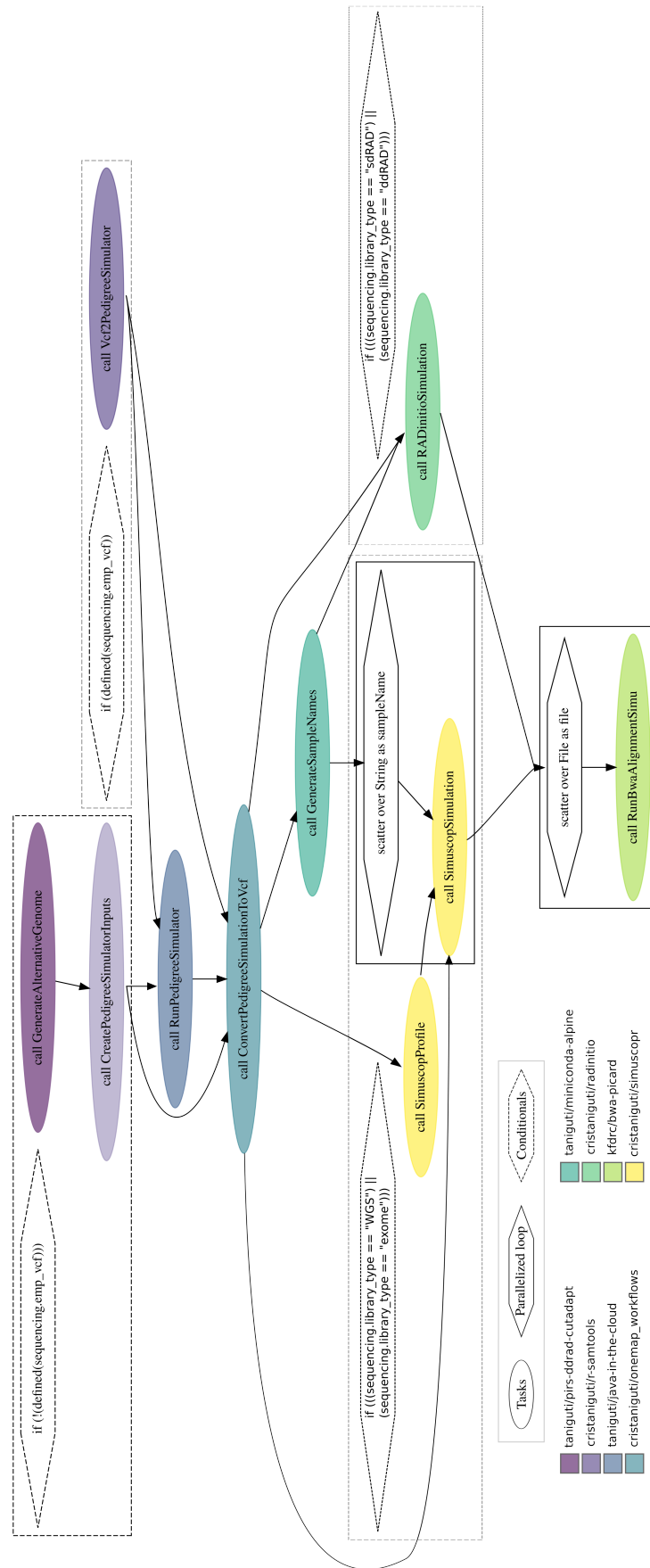


Figure 2.4. Sub-Workflow create\_alignment\_from\_read\_simulations.

followed **GATK Best Practices** to reproduce their joint genotyping via *GATK*, *GenomicsDBImport*, and *GenotypeGVCFs* and applied the suggested hard-filtering. **Freebayes** and **BWA-MEM** can be run in parallel, the maximum number of cores allowed must be defined in the input file. After obtaining the VCF files, they are normalized with **BCFtools** (LI, 2011) and multiallelic markers are separated into a new VCF file. **BCFtools** is also used to find the read depths information for each allele in BAM files and update the allele depths information in the AD (allele depth) field of VCF file. Therefore, each SNP calling method results in three VCFs: i) biallelic markers with read counts outputted by the SNP callers; ii) biallelic markers with counts from BAM files; iii) multiallelic markers. In **SimulatedReads2Map**, markers from SNP callers are compared with the simulated to estimate SNP calling efficiency.

#### 2.2.4 VCF filters

Empirical datasets can have diverse and complex variant density, read depth, and sequencing quality patterns. Before proceeding with empirical analysis, we suggest users evaluate the VCF files generated by SNP calling sub-workflow (**snpcalling\_emp**) to decide the proper filters for the identified variants. After, proceed to the map building tasks in **map\_emp** sub-workflow.

Nowadays, sequence technologies make available millions of markers to build genetic maps, and this amount of data requires high RAM to be evaluated in R environment by map building packages. As an example, if all SNPs identified for the *P. tremula* dataset would be evaluated together it would require 88 GB of RAM just to keep the **OneMap** object and the two-points recombination fraction in the R workspace. Because of this, in **Reads2Map** workflows, we filter VCF file by a selected chromosome and perform all the analysis using this filtered VCF. Users should run the complete workflow for one chromosome, identify the best combination of software and parameters, and download the complete VCF file from SNP calling workflow. With the complete VCF users can repeat the analysis for other chromosomes, using only the selected procedure and R environment (Figure A.3). Also, if users are using draft genomes and there are many markers in not assembled scaffolds, they can be included in major sequences after the workflow is concluded and the best combination of parameters and software is selected. **OneMap** has the functions *combine\_onemap*, *group\_seq*, and *try\_seq* for this purpose.

#### 2.2.5 Genotype calling methods

The genotyping software considered in this study are: **GATK** (McKENNA ET AL., 2010), **freebayes** (GARRISON and MARTH, 2012), **SuperMASSA** (SERANG ET AL., 2012), **polyRAD** (CLARK ET AL., 2019) and **updog** (GERARD ET AL., 2018). The **GATK** and **freebayes** have as input the BAM files and perform both SNP calling and genotype calling. **SuperMassa**, **polyRAD**, and **updog** only perform the genotype calling using the allele depth measures. All software provide genotype probabilities according to their procedures, which will be applied to build genetic maps by **OneMap 3.0**.

**GATK** and **Freebayes** may introduce an unexpected bias towards the reference allele when used to process low-coverage sequence data. In the case of **GATK**, it inserts the bias when reads are filtered in the local re-assembly step to avoid sequencing errors (ROS-FREIXEDES ET AL., 2018). To overcome the bias during the genotype call, the workflow applies two different measures of alleles depth, one from VCF and the other from BAM files.

We create an R package called **genotyping4onemap** to provide wrapper functions to run **updog**, **polyRAD**, and **SuperMASSA** and automatically convert the genotypes in **OneMap** objects. The package is available at <https://github.com/Cristianetaniguti/genotyping4onemap>.

### 2.2.6 Linkage maps

In the workflows, maps are built with **OneMap** 3.0 (MARGARIDO *ET AL.*, 2007) and **GUSMap** (BILTON *ET AL.*, 2018) for datasets from all combinations of SNP and genotype calling methods and considering allele counts coming from VCF and BAM files. This results in 34 maps. In **SimulatedReads2Map**, workflow maps are built for the complete dataset, including false-positive markers and also removing them. Therefore, the simulation workflow results in 68 linkage maps.

If provided, VCF with multiallelic markers is combined with the biallelic markers and receive an error probability of 0.05. VCF files can have markers that are not considered informative for the genetic map building (e.g. in  $F_1$  progeny, markers with both parents homozygous), or genotypes that are not expected by the segregation pattern. Thus, it is essential to apply other filters provided by the map builders.

In **OneMap**, our datasets filtered out markers with more than 25% of missing data, segregation pattern deviation with a p-value threshold of 0.05 after Bonferroni multiple test correction. In **GUSMap**, the datasets were filtered by minor allele frequency (MAF) of 0.05, more than 25% of missing data, segregation pattern deviation with a p-value threshold of 0.05.

In the workflows, the markers are ordered by the genome position. In **OneMap** 3.0, **BatchMap** (SCHIFFTHALER *ET AL.*, 2017) parallelization in four cores of HMM was applied to build the maps. In **OneMap** and **GUSMap**, if markers did not reach the linkage criteria of recombination fraction less than 0.5, it was automatically discarded during genetic distance estimation by HMM. In **SimulatedReads2Map**, the Haldane map function was used to convert recombination fraction to centimorgan measure. In **EmpiricalReads2Map**, the Kosambi map function was applied.

### 2.2.7 Read2Map Workflows App

Output files from the main workflows can be uploaded in **Reads2MapApp**, a shiny app with a shiny-dashboard framework. The usage of the app in our server has a limit of an input file size of 1Gb by user. For higher inputs, users need to install and run the shiny app package from the GitHub repository <https://github.com/Cristianetaniguti/Reads2MapApp>. The docker image `cristaniguti/rstudio_onemap` contains RStudio and all **OneMap** and **Reads2Map** Workflows App dependencies, which can be useful to run the app and also to perform possible manipulations in the built map using **OneMap** or **GUSMap** functions in R.

The **Reads2MapApp** provides guidance on how to use the **Reads2Map** workflows and also graphics and tables to evaluate their results. Each feature is defined in shiny dashboard icons on the left side of the app (Figure 2.5):

- Content: redirect users to manual pages depending on user interests. Manuals describing how to use **OneMap**, **Reads2Map** and **Reads2MapApp** are included;
- SNP calling efficiency (exclusive for simulations): number of simulated markers, number of SNPs, MNPs and indels markers identified and the number of false positives and false negatives markers;
- Filters: total number of informative markers, number of markers filtered by missing data, segregation pattern deviation, redundancy, and non-grouped;
- Marker type: number of markers by types. In simulations, it also shows the correct amount of markers of each type;
- Times: time spent to build each map. Only considers the processing time of the HMM;

- Depth and genotyping: shows allele counts distributions, estimated and real homozygous and heterozygous genotypes, and the genotypes probabilities provided by each software. For the real dataset, it provides a table with statistics to measure the difference between simulated and estimated genotypes;
- Genotype probabilities (exclusive for simulations): Plots the parents and progeny errors probability and the genotypes concordance using conditional probabilities;
- ROC curves (exclusive for simulations): summarizes each genotype caller model predictive power using receiver operating characteristic (ROC) curves;
- Map size each family (exclusive for simulations): shows the difference between estimated and real markers positions in each map;
- Overview of map size (exclusive for simulations): mean, variance, standard deviation, and total map distances in all simulated families;
- Map size (exclusive for empirical evaluations): plots and tables to show the linkage map interval sizes;
- Phases (exclusive for simulations): plots and tables to evaluate the difference between estimated and simulated phases for all simulated families;
- Maps: plot heatmap of recombination fraction matrix and the linkage group draw using **OneMap** and **GUSMap** functions. In this screen it is also possible download RData from the chosen pipeline map sequence (see Figure A.3);
- Plotly heatmaps (exclusive for empirical evaluations): Interactive heatmap of recombination fraction matrix;
- Progeny haplotypes (exclusive for **OneMap** maps): draw the progeny estimated haplotypes. For simulations, also plots the real haplotypes;
- Breakpoints count (exclusive for **OneMap** maps): graphics and tables to show the number of recombination breakpoints estimated in linkage maps. For simulations, graphics and tables are also generated for the real map;
- cMxMb: scatter plot with the relation between the genetic and physical position of each mapped marker;
- Workflow tasks times: the user can upload the log file of the workflow run to build an interactive graphic with the time spent to run each task.

### 2.2.8 Selecting the pipeline

To select the best method to *P. tremula* map building, we consider all available evaluations in Read2MapApp for **EmpiricalReads2Map** and **SimulatedReads2Map** results. For further discussion about all workflow results and evaluations, see Chapter 3.

We chose to create the map using **freebayes** SNP calling, read counts from VCF file, and global error of 0.05 in **OneMap** HMM. Heatmaps of recombination fraction highlight erroneous markers that break the expected color pattern in all approaches considered in the workflow. Therefore, the selected linkage group generated by **EmpiricalReads2Map** was downloaded from “Map” Read2MapApp screen (Figure A.3), and the erroneous markers were removed using **OneMap** in the R environment. We also

Reads2MapApp

- Content
- Upload data
- SimulatedReads2Map
  - SNP calling efficiency
  - Filters
  - Markers type
  - Times
  - Depth and genotyping
  - Genotype probabilities
  - ROC curves
  - Map size each family
  - Overview map size
  - Phases
  - Maps
  - Progeny haplotypes
  - Breakpoints count
  - cM x Mb
- EmpiricalReads2Map
- Workflow tasks times



This shiny app build several graphics using results from Reads2Map workflows. If you run the **SimulatedReads2Map.wdl** and/or **EmpiricalReads2Map.wdl** workflows you can upload the outputted data in **Upload SimulatedReads2Map outputs** and/or **Upload EmpiricalReads2Map outputs** sections. If you don't have your own results yet, you can explore the ones generated with the datasets described in the tables below. Select the available dataset results in **SimulatedReads2Map.wdl results** and/or **EmpiricalReads2Map.wdl results**.

SimulatedReads2Map

Upload SimulatedReads2Map results:

To not overload our server, we limited the upload size to 500MB. If your results have larger size, please run the app locally using:

```
runGitHub('Cristianetani1/Reads2MapApp')
```

If you have more than one depth value, submit all them together.

File: SimulatedReads2Map\_<depth>.tar.gz

Browse...

No file selected

See description of each dataset in the tables below.

SimulatedReads2Map.wdl results

P. tremula 37cM of chromosome 10 - without multiallelics

EmpiricalReads2Map

Upload EmpiricalReads2Map results:

To not overload our server, we limited the upload size to 500MB. If your results have larger size, please run the app locally using:

```
runGitHub('Cristianetani1/Reads2MapApp')
```

If you have more than one depth value, submit all them together.

File: EmpiricalReads2Map\_<depth>.tar.gz

Browse...

No file selected

See description of each dataset in the tables below.

EmpiricalReads2Map.wdl results

Toy sample without multiallelics

Here we describe some of the main characteristics of each dataset available in this server. It is possible to access all other arguments used in the analysis in the metadata produced by the workflows.

SimulatedReads2Map results available

Reading simulations data

Doing part 3

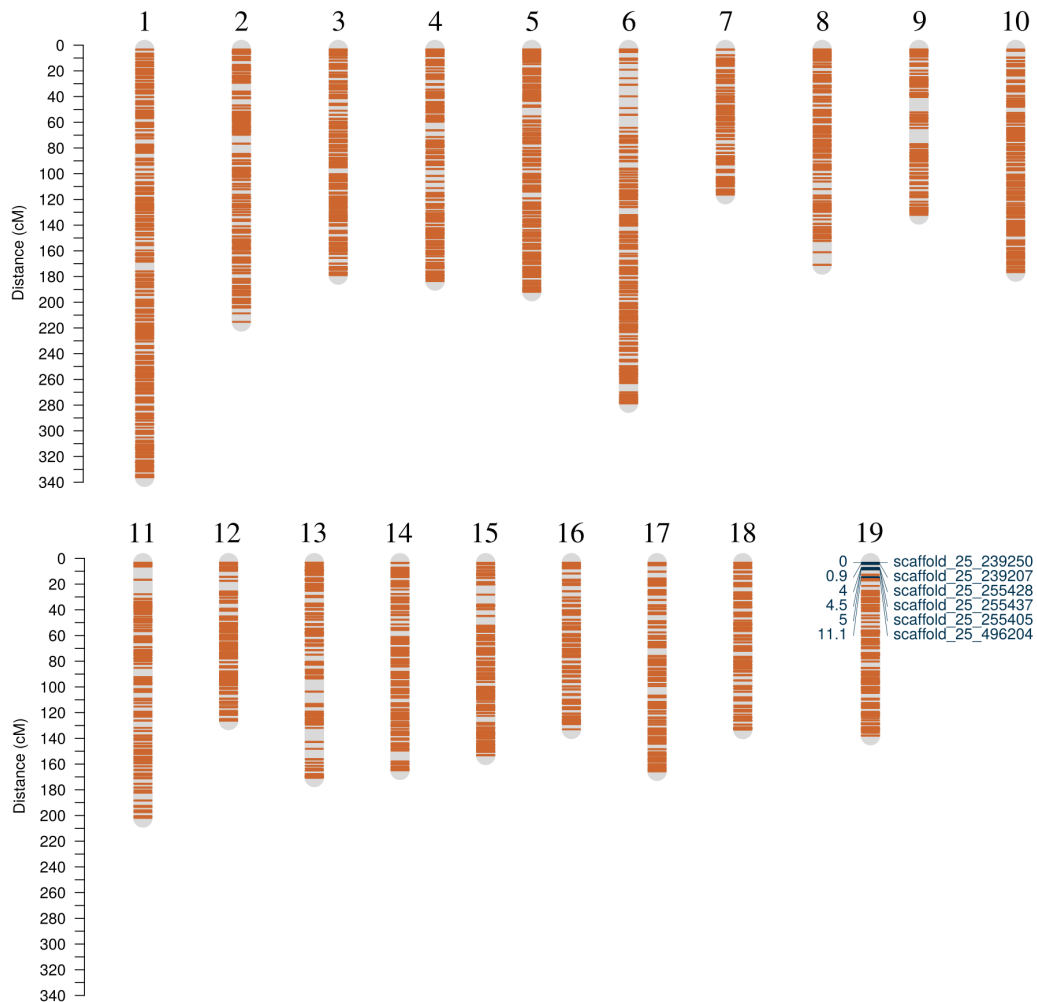
Figure 2.5. Graphical user interface of Read2Map Workflows App.

downloaded the VCF file with markers from all reference genome sequences and built the other linkage groups using the selected approach.

## 2.3 Results and Discussion

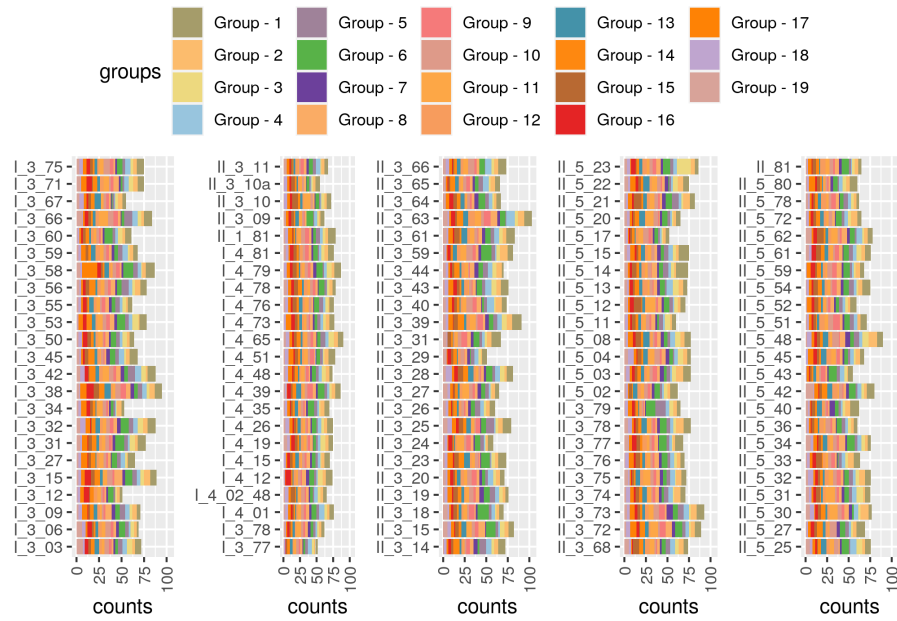
### 2.3.1 Genetic map

The **EmpiricalReads2Map** and **SimulatedReads2Map** results for the *Populus tremula* dataset can be visualized in the **Read2MapApp** example results. They showed that the best pipelines for this dataset are GATK and **freebayes** as SNP and genotype callers, using either their output genotype probabilities or a global error of 0.05. We chose to keep the pipeline with **freebayes** as SNP and genotype caller and error probability of 0.05. Expanding the pipeline to other groups and removing outlier markers, we obtained a map with 6936 markers and 3299.96 cM. **OneMap** 3.0 package also returned the progeny haplotypes, and they presented a total of 8329 recombination breakpoints in the population with a mean of 1.9 and a standard error of 0.0215 by haplotype (Figures 2.6, 2.7, Table A.3).



**Figure 2.6.** *Populus tremula* genetic map.

The study, which originated the dataset (ZHIGUNOV *ET AL.*, 2017), used the pseudo-testcross approach to build the maps. Thus, they created two maps, one for each parent. The female map having 1000 markers and 3045.99 cM, and the male map 1055 markers and 3090.56 cM. The pseudo-testcross approach was used to generate the firsts genetic maps for outcrossing populations (RITTER *ET AL.*, 1990;



**Figure 2.7.** Recombination breakpoints counts in estimated progeny haplotypes of a bi-parental cross of *Populus tremula*.

GRATTAPAGLIA *ET AL.*, 1995) when only dominant markers were available, and we it was not possible to join information from markers types D1 and D2.

The multipoint approach proposed in WU *ET AL.* (2002c) and implemented in **OneMap** allowed us to integrate this information using more informative markers (e.g., A or B) in the dataset and a multipoint approach to estimate genetic distances. Because we do not need to split the markers into two datasets, the integrated maps are denser and carry more allelic and phase information.

The built progeny haplotypes highlight the difference between the integrated and the pseudo-testcross approaches. With integrated maps, we can identify which parts of the four parents' haplotypes compose each progeny homologs (Figure A.6). The pseudo-testcross approach can identify in progeny only two of the four parent haplotypes in a separate way. For each individual in progeny, instead of having two progeny homologs composed by the combination of four possible parent haplotypes, we would have four homologs composed of two possible parent haplotypes. In **OneMap**, it would be equivalent to obtain two maps of the backcross population.

### 2.3.2 Pipeline validation with linkage map

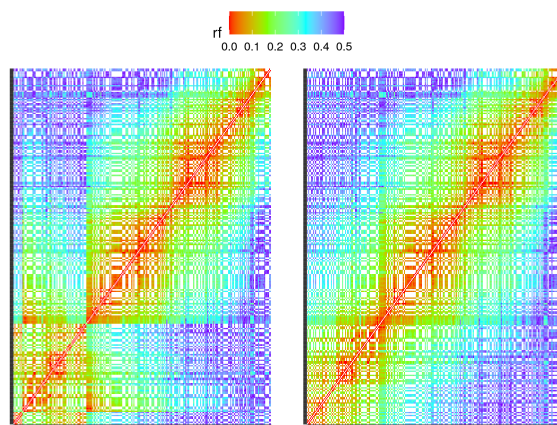
Reliable genetic maps at the end of the process are good diagnostic for all the upstream procedures, including the pollination, DNA extraction, sequencing protocol, and all bioinformatic steps in the workflow. Furthermore, we can explore abnormalities in the genetic maps to fix possible errors in previous steps.

The ZHIGUNOV *ET AL.* (2017) study identified seven contaminants (II\_3\_08, II\_1\_37, I\_4\_62, I\_4\_28, I\_4\_21, I\_2\_72, I\_3\_70) in the dataset through ancestry analysis in ADMIXTURE (ALEXANDER and LANGE, 2011). They removed these individuals from the dataset before building the map. Here, we kept these individuals to test the consequences on the map. The stability of HMM analysis in **OneMap** makes it possible to identify five of these individuals (I\_3\_08, II\_1\_37, I\_4\_62, I\_4\_21, and I\_3\_70) because they presented an excessive number of recombination breakpoints in the estimated haplotypes (Figures A.4 and 2.7). The irregularities in breakpoints counts confirm the ancestry analysis for these individuals. They are not hybrids from the studied cross. Individuals I\_4\_28 and I\_3\_72, also identified



as contaminants in ZHIGUNOV *ET AL.* (2017), did not show irregularities in breakpoints counts. However, we also removed them from the analysis to follow the previous study.

**Reads2Map** workflows require a reference genome to perform the SNP calling. By default, the workflow uses the marker genome position also to sort the markers. Differences between the reference genome markers position and their positions in the genetic map can be evaluated by a heatmap graphic of the recombination fraction matrices (Figure A.8). Wrong positioned markers use to break the expected color pattern in the heatmaps. Interactive graphics in **OneMap** help to identify and remove these markers. Without consulting this information, we can not know if the differences between genome reference and map built are consequences of biological inversions or errors in marker ordering. The heatmap of chromosome 12 presented the inversion of a sequence containing 64 markers (Figure 2.8).



**Figure 2.8.** Heatmap of recombination fraction suggest an inversion in chromosome 12 of *P. tremula* compared with *P. trichocarpa* reference genome. For well-ordered maps, we expect that the diagonal of the graphic have heat colors and the up left and down right present cold colors. It means that adjacent markers are more linked than distance markers. We can see in the left graphic that there is a break in the diagonal color pattern, indicating a disruption in the map order. In the right graphic, we inverted the sequence from the color break until the first marker in the group. After, the color pattern is more close to the expected.

In the previous study (ZHIGUNOV *ET AL.*, 2017), they found inversions in groups 1, 11, 15, 18. The difference between our map and the last built can be explained by the different approaches applied: the integrated map and the ordering based on the reference genome.

### 2.3.3 Pipeline validation with simulations

The workflows can also be used as a validation tool for software developers. As an example, running the **SimulatedReads2Map** workflow without VCF normalization (TAN *ET AL.*, 2015), we found some small inconsistencies in **freebayes** VCF markers positions (Figure A.7). Some marker's positions recorded in VCF were slightly displaced compared with the positions of the simulated markers.

**Freebayes** VCF requires normalization to be possible to make the correspondence of identified markers with the simulated ones. It is an excellent practice to always normalize the alleles representation format in VCF files once this representation is not unique when dealing with different software. Normalization involves reducing descriptions of a variant to a canonical representation. A safe representation considers parsimony of the polymorphism length and its left alignment to define the position (TAN *ET AL.*, 2015). Here we used **BCFtools** (LI, 2011) to perform the normalization of VCFs.

**Freebayes** and **GATK** identified a total of 2.399.744 (640284 indels and 1759460 SNPs) and 2.029.600 (217577 indels and 1812023 SNPs) markers, respectively, in *P. tremula* dataset. VCF normal-

ization realigned 284336 sites in *freebayes* VCF. Moreover, it converted 27460 indels into SNPs (Table A.4). The right marker positions are particularly important in our workflow because we want to compare the SNP calling efficiency comparing the position of identified markers with the simulated and also because we use these positions to recover read depth information from BAM alignment files.

### 2.3.4 Genome assembly assisted by linkage maps

As pointed out by FIERST (2015), linkage maps have been used to help reference genome assembly. Here we use the *P. thrichocarpa* reference genome (TUSKAN *ET AL.*, 2006) to define the linkage group and order the markers, but we still can help the assembly of sequences that are not included in the major scaffolds. The *P. thrichocarpa* genome was the first published genome of a tree and it is now in version 4.0, but still has 27 scaffolds not included in the 19 chromosome sequences. These scaffolds sum approximately 3 Mb of 397 Mb in the genome. *Freebayes* identified 7379 markers in these sequences. After filters, six markers were added in chromosome 19 (Figure 2.6), all from scaffold 25, suggesting a relationship between these two genome sequences.

Genome assembly and linkage map tools are complementary in the task of ordering markers or sequences. The genome assemblies joint small reads into larger sequences, solving the possible local misplacements. Still, without long sequenced reads, it faces problems in ordering markers that are far located in the genome, because some genome regions, such as repetitive regions, are difficult to assembly.

The linkage map is useful to understand where larger sequences are located. Building linkage maps requires the identification of recombination breakpoints in individuals in populations. With a limited number of individuals in progeny, the map does not have enough resolution to order markers that are located close in the genome, even if we use efficient algorithms to solve the traveling salesman problem, such as multidimensional scaling (MDS) (PREEDY and HACKETT, 2016). Thus, the capacity of the linkage map to local ordering is not precise, especially in small populations.

### 2.3.5 Workflow flexibility

The workflow system presented was created to reproduce the same analysis with different datasets. The designed workflows were limited to test two SNP callers, five genotype callers, and two genetic map builders. They are modularized and the connection between tasks or sub-workflows has few inputs and outputs, which makes it easy to locate specific functionalities (VOSS *ET AL.*, 2017). Therefore, users with few programming skills can easily make changes in parameters. Other software can also be included in our comparison, despite needing more WDL programming skills.

For development purposes, the simulations, combined with the *shiny* app, can be useful tools to identify semantic errors in produced code. It is suggested that users test their modifications first in a subset of data. Subset samples for simulations and empirical data workflows are available in the repository.

In this context, the available *Cromwell* configuration to save cache in a database is also useful and it allows access to the *Cromwell* server mode, which gives access to other tools to evaluate the process.

## 2.4 Conclusions

The HMM approach implemented in software to build linkage maps is robust and able to return the best estimation possible of genetic distances. However, it is a lot of computer resources and time-consuming. Also, many different tools can be applied in the upstream processing of genetic map building with sequencing markers, such as the SNP and genotype calling. Testing every possible scenario to select the best pipeline for the specific dataset can be very difficult.

The workflows here provided offer a tool to test and select different combinations of software and parameters to build linkage maps from sequencing reads. The entire procedure is facilitated by the available configurations, **Docker** images, and graphical interface. Thus, users can focus on statistical and genetic aspects evolving the linkage map building instead of technical issues.

We consider the **SimulatedReads2Map** workflow more useful for developers because it provides an overview of the entire procedure and facilitates the search for logical errors in codes. The **EmpiricalReas2Map** is useful for users who want to select the best combination of software and parameters to build their genetic maps. To avoid users spend too much time with this pipeline selection, we suggest running the **map\_emp** sub-workflow with a subset of data (e.g. a single chromosome). Once selected the pipeline, users can apply it for the entire dataset using **genotyping4onemap**, **OneMap** or **GUSMap** in R environment.

The shiny app **Reads2MapApp** guides the users through several quality criteria for each approach built genetic map. The main criteria to select the approach are the right color pattern in the recombination fraction heatmaps and the number of recombination breakpoints identified. The heatmaps highlight the grouping and ordering aspects of the map and the number of recombination breakpoints highlight the presence/absence of genotyping errors in the dataset.

We can validate all upstream procedures of obtention of molecular markers and their usage to genotype individuals if they can reproduce known genetic proprieties of linkage maps. The tools developed in this work provide an easy way to test protocols and software for molecular marker application studies.

## 2.5 Availability and requirements

We tested the workflows in local computer, HPC and Google cloud services. The local computer used had 8 cores and 32GB of RAM. The processor is a Intel(R) Core(TM) i7-4790 CPU @ 3.60GHz. We tried in three HPC system, two of them presented issues with singularity permissions, revealing that the workflows require proper singularity configurations by the HPC administrators. The workflows run successfully in the University of São Paulo Aguiar Cluster. It uses Slurm Workload Management (Yoo *ET AL.*, 2003) to manage the job submissions and has 128 nodes with 20 cores with 24GB RAM per core. The processor is Intel(R) Xeon(R) CPU E7- 2870 @ 2.40GHz.

Project name:	<b>Reads2Map</b>
Project home page:	<a href="https://github.com/Cristianetaniguti/Reads2Map">https://github.com/Cristianetaniguti/Reads2Map</a> <a href="https://github.com/Cristianetaniguti/Reads2MapApp">https://github.com/Cristianetaniguti/Reads2MapApp</a>
Platform:	independent
Programming language:	WDL
Other requirements:	Java and Docker
License:	GNU
Any restriction to use by non-academics:	defined by each software implemented

## References

- ALEXANDER, D. H. and K. LANGE, 2011 Enhancements to the ADMIXTURE algorithm for individual ancestry estimation. *BMC Bioinformatics* **12**: 246.
- AMSTUTZ, P., M. R. CRUSOE, N. TIJANIĆ, B. CHAPMAN, J. CHILTON, M. HEUER, A. KARTASHOV, D. LEEHR, H. MÉNAGER, M. NEDELJKOVICH, and E. AL., 2016 Common Workflow Language, v1.0.
- ANDREWS, S., 2010 FastQC: a quality control tool for high throughput sequence data. Available online at: <http://www.bioinformatics.babraham.ac.uk/projects/fastqc> .

- BILTON, T. P., M. R. SCHOFIELD, M. A. BLACK, D. CHAGNÉ, P. L. WILCOX, and K. G. DODDS, 2018 Accounting for Errors in Low Coverage High-Throughput Sequencing Data When Constructing Genetic Maps Using Biparental Outcrossed Populations. *Genetics* **209**: 65–76.
- CATCHEN, J., P. A. HOHENLOHE, S. BASSHAM, A. AMORES, and W. A. CRESKO, 2013a Stacks: an analysis tool set for population genomics. *Molecular Ecology* **22**: 3124–40.
- CHANG, W., J. CHENG, J. J. ALLAIRE, Y. XIE, and J. MCPHERSON, 2020 *shiny: Web Application Framework for R*.
- CLARK, L. V., A. E. LIPKA, and E. J. SACKS, 2019 polyRAD: Genotype Calling with Uncertainty from Sequencing Data in Polyploids and Diploids. *G3: Genes|Genomes|Genetics* **9**: g3.200913.2018.
- DI TOMMASO, P., M. CHATZOU, E. W. FLODEN, P. P. BARJA, E. PALUMBO, and C. NOTREDAME, 2017 Nextflow enables reproducible computational workflows. *Nature Biotechnology* **35**: 316–319.
- ELLSON, J., E. R. GANSNER, E. KOUTSOFIOS, S. C. NORTH, and G. WOODHULL, 2003 Graphviz and dynagraph – static and dynamic graph drawing tools. In *GRAPH DRAWING SOFTWARE*, pp. 127–148, Springer-Verlag.
- EWELS, P., M. MAGNUSSON, S. LUNDIN, and M. KÄLLER, 2016 MultiQC: Summarize analysis results for multiple tools and samples in a single report. *Bioinformatics* **32**: 3047–3048.
- FIERST, J. L., 2015 Using linkage maps to correct and scaffold de novo genome assemblies: methods, challenges, and computational tools. *Frontiers in Genetics* **6**: 1–8.
- GARRISON, E. and G. MARTH, 2012 Haplotype-based variant detection from short-read sequencing. *ArXiv e-prints* p. 9.
- GERARD, D., L. F. V. FERRÃO, A. A. F. GARCIA, and M. STEPHENS, 2018 Genotyping Polyploids from Messy Sequencing Data. *Genetics* **210**: 789–807.
- GRATTAPAGLIA, D., F. L. BERTOLUCCI, and R. R. SEDEROFF, 1995 Genetic mapping of QTLs controlling vegetative propagation in *Eucalyptus grandis* and *E. urophylla* using a pseudo-testcross strategy and RAPD markers. *Theoretical and Applied Genetics* **90**: 933–947.
- GRÜNING, B., J. CHILTON, J. KÖSTER, R. DALE, N. SORANZO, M. VAN DEN BEEK, J. GOECKS, R. BACKOFEN, A. NEKRUTENKO, and J. TAYLOR, 2018 Practical Computational Reproducibility in the Life Sciences. *Cell Systems* **6**: 631–635.
- HALDANE, J. B. S., 1919 The combination of linkage values, and the calculation of distance between linked factors. *Journal of Genetics* **8**: 299–309.
- HEATHER, J. M. and B. CHAIN, 2016 The sequence of sequencers: The history of sequencing DNA. *Genomics* **107**: 1–8.
- HU, X., J. YUAN, Y. SHI, J. LU, B. LIU, Z. LI, Y. CHEN, D. MU, H. ZHANG, N. LI, Z. YUE, F. BAI, H. LI, and W. FAN, 2012a pIRS: Profile-based Illumina pair-end reads simulator. *Bioinformatics* **28**: 1533–1535.
- HYMAN, J. M., 1983 Accurate Monotonicity Preserving Cubic Interpolation. *SIAM Journal on Scientific and Statistical Computing* **4**: 645–654.
- KURTZER, G. M., V. SOCHAT, and M. W. BAUER, 2017 Singularity: Scientific containers for mobility of compute. *PLOS ONE* **12**: e0177459.

- LI, H., 2011 A statistical framework for SNP calling, mutation discovery, association mapping and population genetical parameter estimation from sequencing data. *Bioinformatics* **27**: 2987–2993.
- LI, H., 2013 Aligning sequence reads, clone sequences and assembly contigs with BWA-MEM. *ArXiv* **1303**.
- LI, H., 2020 seqtk: Toolkit for processing sequences in FASTA/Q formats. Available online at: <https://github.com/lh3/seqtk>.
- MARGARIDO, G. R. A., A. P. SOUZA, and A. A. F. GARCIA, 2007 OneMap: software for genetic mapping in outcrossing species. *Hereditas* **144**: 78–9.
- MARTIN, M., 2011 Cutadapt removes adapter sequences from high-throughput sequencing reads. *EMBnet.journal* **17**: 10.
- McKENNA, A., M. HANNA, E. BANKS, A. SIVACHENKO, K. CIBULSKIS, A. KERNYTSKY, K. GARIMELLA, D. ALTSHULER, S. GABRIEL, M. DALY, and M. A. DEPRISTO, 2010 The Genome Analysis Toolkit: A MapReduce framework for analyzing next-generation DNA sequencing data. *Genome Research* **20**: 1297–1303.
- MERKEL, D., 2014a Docker : Lightweight Linux Containers for Consistent Development and Deployment Docker : a Little Background Under the Hood. *Linux Journal* **2014**: 2–7.
- MERKEL, D., 2014b Docker: lightweight linux containers for consistent development and deployment. *Linux journal* **2014**: 2.
- MUNAFÒ, M. R., B. A. NOSEK, D. V. BISHOP, K. S. BUTTON, C. D. CHAMBERS, N. PERCIE DU SERT, U. SIMONSOHN, E. J. WAGENMAKERS, J. J. WARE, and J. P. IOANNIDIS, 2017 A manifesto for reproducible science. *Nature Human Behaviour* **1**: 1–9.
- NATURE, 2019 Challenge to test reproducibility of old computer code.
- POPLIN, R., V. RUANO-RUBIO, M. A. DEPRISTO, T. J. FENNELL, M. O. CARNEIRO, G. A. V. DER AUWERA, D. E. KLING, L. D. GAUTHIER, A. LEVY-MOONSHINE, D. ROAZEN, K. SHAKIR, J. THIBAUT, S. CHANDRAN, C. WHELAN, M. LEK, S. GABRIEL, M. J. DALY, B. NEALE, D. G. MACARTHUR, and E. BANKS, 2017 Scaling accurate genetic variant discovery to tens of thousands of samples. *bioRxiv* p. 201178.
- PREEDY, K. F. and C. A. HACKETT, 2016 A rapid marker ordering approach for high-density genetic linkage maps in experimental autotetraploid populations using multidimensional scaling. *Theoretical and Applied Genetics*.
- REITER, T., P. BROOKS, L. IRBER, S. JOSLIN, C. REID, C. SCOTT, C. T. BROWN, and N. T. PIERCE, 2020 Streamlining Data-Intensive Biology With Workflow Systems pp. 1–19.
- RESCIENCE, 2019 Ten years of reproducibility challenge.
- RITTER, E., C. GEBHARDT, and F. SALAMINI, 1990 Estimation of recombination frequencies and construction of RFLP linkage maps in plants from crosses between heterozygous parents. *Genetics* **125**: 645–654.
- RIVERA-COLÓN, A. G., N. C. ROCHETTE, and J. M. CATCHEN, 2020 Simulation with RADinitio improves RADseq experimental design and sheds light on sources of missing data. *Molecular Ecology Resources* pp. 1–16.

- ROS-FREIXEDES, R., M. BATTAGIN, M. JOHNSON, G. GORJANC, A. J. MILEHAM, S. D. ROUNSLEY, and J. M. HICKEY, 2018 Impact of index hopping and bias towards the reference allele on accuracy of genotype calls from low-coverage sequencing. *Genetics Selection Evolution* **50**: 1–14.
- SCHIFFTHALER, B., C. BERNHARDSSON, P. K. INGVARSSON, and N. R. STREET, 2017 BatchMap: A parallel implementation of the OneMap R package for fast computation of F1 linkage maps in outcrossing species. *PLoS ONE* **12**: 1–12.
- SERANG, O., M. MOLLINARI, and A. A. F. GARCIA, 2012 Efficient exact maximum a posteriori computation for bayesian SNP genotyping in polyploids. *PLoS ONE* **7**: 1–13.
- SMITH, G. R. and M. NAMBIAR, 2020 New Solutions to Old Problems: Molecular Mechanisms of Meiotic Crossover Control. *Trends in Genetics* **36**: 337–346.
- STURTEVANT, A. H., 1915 The behavior of the chromosomes as studied through linkage. *Zeitschrift für induktive Abstammungs- und Vererbungslehre* **13**: 234–287.
- TAN, A., G. R. ABECASIS, and H. M. KANG, 2015 Unified representation of genetic variants. *Bioinformatics* **31**: 2202–2204.
- TERRA.BIO, 2020 Terra: Focus on your science. Available online at: <https://app.terra.bio/>.
- TUSKAN, G. A., S. DIFAZIO, S. JANSSON, J. BOHLMANN, I. GRIGORIEV, U. HELLSTEN, N. PUTNAM, S. RALPH, S. ROMBAUTS, A. SALAMOV, J. SCHEIN, L. STERCK, A. AERTS, R. R. BHALERAO, R. P. BHALERAO, D. BLAUDEZ, W. BOERJAN, A. BRUN, A. BRUNNER, V. BUSOV, M. CAMPBELL, J. CARLSON, M. CHALOT, J. CHAPMAN, G.-L. CHEN, D. COOPER, P. M. COUTINHO, J. COUTURIER, S. COVERT, Q. CRONK, R. CUNNINGHAM, J. DAVIS, S. DEGROEVE, A. DEJARDIN, C. DEPAMPHILIS, J. DETTER, B. DIRKS, I. DUBCHAK, S. DUPLESSIS, J. EHLTING, B. ELLIS, K. GENDLER, D. GOODSTEIN, M. GRIBSKOV, J. GRIMWOOD, A. GROOVER, L. GUNTER, B. HAMBERGER, B. HEINZE, Y. HELARIUTTA, B. HENRISSAT, D. HOLLIGAN, R. HOLT, W. HUANG, N. ISLAM-FARIDI, S. JONES, M. JONES-RHOADES, R. JORGENSEN, C. JOSHI, J. KANGASJARVI, J. KARLSSON, C. KELLEHER, R. KIRKPATRICK, M. KIRST, A. KOHLER, U. KALLURI, F. LARIMER, J. LEEBENS-MACK, J.-C. LEPLE, P. LOCASCIO, Y. LOU, S. LUCAS, F. MARTIN, B. MONTANINI, C. NAPOLI, D. R. NELSON, C. NELSON, K. NIEMINEN, O. NILSSON, V. PEREDA, G. PETER, R. PHILIPPE, G. PILATE, A. POLIAKOV, J. RAZUMOVSKAYA, P. RICHARDSON, C. RINALDI, K. RITLAND, P. ROUZE, D. RYABOY, J. SCHMUTZ, J. SCHRADER, B. SEGERMAN, H. SHIN, A. SIDDIQUI, F. STERKY, A. TERRY, C.-J. TSAI, E. UBERBACHER, P. UNNEBERG, J. VAHALA, K. WALL, S. WESSLER, G. YANG, T. YIN, C. DOUGLAS, M. MARRA, G. SANDBERG, Y. VAN DE PEER, and D. ROKHSAR, 2006 The Genome of Black Cottonwood, *Populus trichocarpa*. *Science* **313**: 1596–1604.
- VAN DER AUWERA, G. A., M. O. CARNEIRO, C. HARTL, R. POPLIN, G. DEL ANGEL, A. LEVY-MOONSHINE, T. JORDAN, K. SHAKIR, D. ROAZEN, J. THIBAUT, E. BANKS, K. V. GARIMELLA, D. ALTSHULER, S. GABRIEL, and M. A. DEPRISTO, 2013 From FastQ data to high confidence variant calls: the Genome Analysis Toolkit best practices pipeline. *Current protocols in bioinformatics* **43**: 11.10.1–11.10.33.
- VOORRIPS, R. E. and C. A. MALIEPAARD, 2012 The simulation of meiosis in diploid and tetraploid organisms using various genetic models. *BMC Bioinformatics* **13**: 248.
- VOSS, K., J. GENTRY, and G. V. D. AUWERA, 2017 Full-stack genomics pipelining with GATK4+ WDL+ Cromwell [version 1; not peer reviewed]. *F1000Research* p. 4.

- WILKINSON, M. D., M. DUMONTIER, I. J. AALBERSBERG, G. APPLETON, M. AXTON, A. BAAK, N. BLOMBERG, J.-W. BOITEN, L. B. DA SILVA SANTOS, P. E. BOURNE, J. BOUWMAN, A. J. BROOKES, T. CLARK, M. CROSAS, I. DILLO, O. DUMON, S. EDMUNDS, C. T. EVELO, R. FINKERS, A. GONZALEZ-BELTRAN, A. J. G. GRAY, P. GROTH, C. GOBLE, J. S. GRETHE, J. HERINGA, P. A. C. 'T HOEN, R. HOOFT, T. KUHN, R. KOK, J. KOK, S. J. LUSHER, M. E. MARTONE, A. MONS, A. L. PACKER, B. PERSSON, P. ROCCA-SERRA, M. ROOS, R. VAN SCHAIK, S.-A. SANSONE, E. SCHULTES, T. SENGSTAG, T. SLATER, G. STRAWN, M. A. SWERTZ, M. THOMPSON, J. VAN DER LEI, E. VAN MULLIGEN, J. VELTEROP, A. WAAGMEESTER, P. WITTENBURG, K. WOLSTENCROFT, J. ZHAO, and B. MONS, 2016 The FAIR Guiding Principles for scientific data management and stewardship. *Scientific Data* **3**: 160018.
- WU, R., C.-X. MA, S. S. WU, and Z.-B. ZENG, 2002c Linkage mapping of sex-specific differences. *Genetical research* **79**: 85–96.
- YOO, A. B., M. A. JETTE, and M. GRONDONA, 2003 SLURM: Simple Linux Utility for Resource Management. In *Job Scheduling Strategies for Parallel Processing*, edited by D. Feitelson, L. Rudolph, and U. Schwiegelshohn, pp. 44–60, Berlin, Heidelberg, Springer Berlin Heidelberg.
- YU, Z., F. DU, R. BAN, and Y. ZHANG, 2020 SimuSCoP: Reliably simulate Illumina sequencing data based on position and context dependent profiles. *BMC Bioinformatics* **21**: 1–18.
- ZHIGUNOV, A. V., P. S. ULIANICH, M. V. LEBEDEVA, P. L. CHANG, S. V. NUZHDIN, and E. K. POTOKINA, 2017 Development of F1 hybrid population and the high-density linkage map for European aspen (*Populus tremula* L.) using RADseq technology. *BMC Plant Biology* **17**.

### 3 THE EFFECT OF CONSIDERING GENOTYPE PROBABILITIES AND HAPLOTYPE-BASED MULTIALLELIC MARKERS IN BUILDING LINKAGE MAPS WITH HIGH-THROUGHPUT GENOTYPING

#### Abstract

Linkage maps are important tools for genetic studies. Nowadays, with high-throughput genotyping, there are millions of genetic markers available to build a map. The markers coming from this technique have different characteristics compared with those commonly used, such as SSRs. The software used to estimate the linkage map needed adaptations to deal with this new scenario and overcome new issues. One of these issues is the genotyping errors generated by different sources such as sequencing errors, alignment, and PCR errors. Without proper analyses, it causes inflated genetic maps. One possible solution for that is to consider the genotype probabilities coming from the genotype caller in the Hidden Markov Model (HMM) used to estimate the genetic distances. However, to the best of our knowledge, there is not yet available a clear recommendation about which genotype probability would give the best results. Here we not only implemented this feature in OneMap 3.0 package but also used the **Reads2Map** workflows to test the impact of **freebayes**, **GATK**, **updog**, **polyRAD** and **SuperMASSA** genotype probabilities in the map build process, with empirical and simulated RADseq data. Each approach presented different results depending on the choice of the parameters. The results showed that **OneMap 3.0** can build high-quality maps if i) the genotype caller did not make many mistakes and we use a global error rate of 5% or ii) if the genotype caller make mistakes but it also provides lower genotype probabilities for wrong genotypes. In our study, the scenarios that showed to be the most advantageous in terms of genotype probabilities used in the HMM were the **freebayes** as SNP caller, read counts from VCF, and a single error rate of 5% or with genotype probabilities from **SuperMASSA** and **PolyRAD**. They produced a denser map, closer to the expected recombination fraction matrix values and smaller size. The other issue found in building linkage maps with high-throughput markers is the low informativeness of the SNPs. Because of their biallelic nature, SNPs bring a lack of information for the ordering of the markers and the phase estimation in outcrossing species. The solution presented here is the usage of haplotype-based markers identified by the assembly-based haplotyping SNP callers **freebayes** and **GATK**. We also tested the differences of using or not these markers with the **Reads2Map** workflows. We observed improvements in the genetic distances and ordering in the presence of multiallelic markers, but also higher map inflation caused by these markers if they contain genotyping errors. The final approach selected was the **freebayes** as SNP and genotype caller, a global error probability of 5% and the presence of haplotype-based multiallelic markers. We re-built the *Populus tremula* chromosome 10 linkage group with 107 haplotypes-based multiallelic markers and 440 SNPs totaling 216.99 cM.

Key words: Linkage map; Genotyping error; Haplotype-based marker; RADseq.

#### 3.1 Introduction

Since the first genetic map was built by Sturtevant in 1913, methodologies for building genetic maps have been improved to deal with different and more complex genomes, like those from outcrossing and polyploids species, and with different mapping populations. All efforts that have been done are motivated by the importance of this tool for genetic researches, since genetic maps provide valuable information.

Linkage maps are commonly applied to quantitative trait loci (QTL) studies. The QTL mapping estimates the genetic architecture of traits, which include mapping the molecular polymorphisms



responsible for variation in complex traits; determining their gene frequencies and their homozygous, heterozygous, epistatic, and pleiotropic effects in multiple environments (MACKAY, 2001b).

Nowadays, with Breeding 4.0, QTL have also a great application for gene editing technologies (WALLACE *ET AL.*, 2018). QTLs allow the identification of genes responsible for important phenotypes and candidates to be edited. The gene edition allows breeders to add new alleles from germplasm in breeding programs without lose the alleles of the well-established lines. As an example, ZSÖGÖN *ET AL.* (2018) applied gene editing of six important genes of tomato to give wild lines the yield and productivity similar to cultivated ones. Previous QTL studies needed to be made to identify these six genes effects (ASHRAFI *ET AL.*, 2012; FRARY *ET AL.*, 2000).

Furthermore, it has becoming more usual to apply linkage maps to solve genome assembly issues (FIERST, 2015). For many species, especially non-model organisms with large genome size, it is still unaffordable to obtain reliable reference genome assemblies. For many species, only draft genomes are available with thousands of sequenced segments (contigs) and very limited information on how these can be assembled into chromosome sequences. Linkage maps can give content for ordering, orienting, positioning and phasing linked sequences (PENGELLY and COLLINS, 2018).

The growing accessibility of sequencing technologies is providing genome information for great understandings about its structures, and how the genome diverges between individuals. The high-throughput genotyping technology to obtain markers can provide ultra-dense genetic maps, which are promising for advances for association studies. However, it also brings challenges for genetic map building procedures. Markers now are identified on large scale automatically, without the need for handwork, which provides clear advantages to produce low-cost markers but they also have more genotyping errors. Especially when high-throughput sequencing technology is applied to reduced representation libraries (RADseq), which have an excess of duplicated sequenced sequences starting and finishing at the same point of the genome. In RADseq data, the most of duplicated sequences are products from PCR, which include errors difficult to be identified by bioinformatics tools (DER AUWERA *ET AL.*, 2020; RIVERA-COLÓN *ET AL.*, 2020).

The excess of genotyping errors in datasets have been generated inflated genetic maps, with unrealistic genetic distances (SMITH and NAMBIAR, 2020), once each genotype error is considered an extra recombination event. Studies have been made to search the sources of errors and correct them in bioinformatics and statistical methods for SNP and genotype calling procedures (HACKETT and BROADFOOT, 2003; RIVERA-COLÓN *ET AL.*, 2020; GERARD *ET AL.*, 2018; CLARK *ET AL.*, 2019). The sources of errors can also be considered in map building algorithms to provide reliable ultra-dense genetic maps (BILTON *ET AL.*, 2018; MOLLINARI and GARCIA, 2019).

Another characteristic of high-throughput genotyping is the low-informativeness that comes with the biallelic codominant nature of SNPs. In populations coming from inbred lines, this characteristic does not have a significant impact, but, in outcrossing species, this brings as consequence difficulties to integrate recombination from both parents and to ordering the markers. The low-informativeness of non-integrated genetic maps brings limitations in further QTL analysis of multiallelic traits (GAZAFFI *ET AL.*, 2020).

One possible solution to solve low-informativeness is to use assembly-based haplotyping SNP callers in previous steps of map building, such as PolyBayes (MARTH *ET AL.*, 1999), samtools (LI, 2011), freebayes (GARRISON and MARTH, 2012), GATK (POPLIN *ET AL.*, 2017; MCKENNA *ET AL.*, 2010) and Platypus (RIMMER *ET AL.*, 2014), which combine close located biallelic markers into haplotypes and output them as phased markers or multiple nucleotide markers (MNP).

The available algorithms in OneMap (MARGARIDO *ET AL.*, 2007) can build integrated genetic maps and generate the linkage group haplotypes of each individual in the mapping population. To estimate linkage phases, OneMap uses a multipoint approach with a Hidden Markov Model (HMM)

method, but it requires high computational resources, time to process, and it is not efficient if there are many genotyping errors and not at least a few informative markers. In this work, we adapted OneMap algorithms to deal with high-throughput markers. Using the **Reads2Map** workflows (chapter 2), we tested the consequences of considering genotype error probabilities and haplotype-based multiallelic markers to build linkage maps from sequencing reads. The workflows allowed the comparison using freebayes, GATK (POPLIN *ET AL.*, 2017; MCKENNA *ET AL.*, 2010) as SNP and genotype callers; updog (GERARD *ET AL.*, 2018), polyRAD (CLARK *ET AL.*, 2019), SuperMASSA (SERANG *ET AL.*, 2012) as genotype caller; OneMap 3.0 and GUSMap (BILTON *ET AL.*, 2018) as linkage map builder.

## 3.2 Material and Methods

The analysis here was made in R environment (R CORE TEAM, 2020), and through workflows written with Workflow Definition Language (WDL) and executed with Cromwell Execution Engine (VOSS *ET AL.*, 2017). These workflows are available at <https://github.com/Cristianetaniguti/Reads2Map>. The workflows results were evaluated with the shiny app Reads2MapApp (available at <https://github.com/Cristianetaniguti/Reads2MapApp>).

### 3.2.1 Genotype probabilities in OneMap Hidden Markov Model

Here we updated the OneMap package to version 3.0 to include user-defined error probabilities in the emission function of the already implemented HMM. With a combination of the HMM and EM algorithm, OneMap can perform multipoint estimation of map genetic distance for  $F_2$ , backcross, RILs, and outcrossing populations. For the multipoint estimation, OneMap algorithms have adapted code from R/QTL package (BROMAN *ET AL.*, 2003) to use HMM approach (LANDER and GREEN, 1987; BAUM *ET AL.*, 1970).

Considering the latent variable  $G_i$ ,  $i = 1, \dots, n$ , denote the true underlying genotypes for the individual at a set of  $n$  ordered loci, the observed variable  $O_i$  as the molecular phenotype (observed genotypes) for the locus  $i$ , the HMM can be represented as described in BROMAN *ET AL.* (2009):

$$P(O|G_i = g_i) = \sum_{g_1} \dots \sum_{g_{i-1}} \sum_{g_{i+1}} \dots \sum_{g_n} \pi(g_1) \prod_{j=1}^{n-1} t_j(g_j, g_{j+1}) \prod_{j=1}^n e(g_j, O_j) \quad (3.1)$$

The initial probability  $\pi(g_1)$  is the probability of having a given genotype for the first locus ( $G_1$ ) and its value depends on the cross-type. For example, for an outcrossing population, this value will be 0.25, assuming a uniform distribution of possible genotypes (AB, BA, AB, and BB). The same reasoning applies for backcross, with probabilities of 0.5, since there are only two genotypes (AA and BB).

The transition probability  $t_j(g_j, g_{j+1})$  is the probability of the genotype in a locus ( $G_{j=i+1}$ ) change to the next locus genotype ( $G_{j+1}$ ). The probability is based on the phase and recombination fraction estimated by a two-points approach using maximum likelihood estimators (MALIEPAARD *ET AL.*, 1997). For example, if the estimated recombination fraction between two specific loci is 0.1, it will be considered that there is a chance of 0.1 of that individual have a non-parental genotype for the two loci evaluated.

The emission probability  $e(g_j, O_j)$  is the probability of the observed variable given the genotype. This probability is defined by an associated genotyping error (see Appendix VIII for details). The OneMap software, until version 3.0, considered this error probability as a single value of 0.00001, for every genotype. In version 3.0, this value is kept as default, to maintain the users' code reproducibility. It can still be applied for fragment-based markers, like SSRs, which have less chance to be wrong than SNPs from

sequencing technologies. But in version 3.0 this probability can be unreliable for several situations. Thus, a variable error rate can be applied according to the quality of each genotype.

Using SNPs from sequencing technologies it is possible to infer genotypes using bioinformatics tools, which can consider diverse sources of errors and generate genotype probabilities based on that. For example, errors in the alignment of sequences or by PCR bias and errors from random sampling in low-depth sequencing. In context, **OneMap** 3.0 algorithm allows different values of error probabilities in the emission probability of HMM. Users can customize these values using *create\_probs* function. Users can provide three types of values: one global value (*global\_error*); an error probability for each inferred genotype (*genotypes\_error*); or genotype probabilities for each possible genotype in individuals (*genotypes\_probs*). Considering all available methods of SNP and genotype calling, there are many options of values to be used in the new **OneMap** feature. However, there are no recommendations yet about which are the best and how to obtain them. Users can be confused without clear selection criteria. In this work, we tested the consequences of building maps applying different genotype probabilities coming from genotype caller software and a global error rate of 0.05.

Here we used **GATK** (McKENNA *ET AL.*, 2010), **freebayes** (GARRISON and MARTH, 2012), **polyRAD** (CLARK *ET AL.*, 2019), **SuperMASSA** (SERANG *ET AL.*, 2012) and **updog** (GERARD *ET AL.*, 2018) to estimate the genotypes and genotypes probabilities. For **GATK** and **freebayes** caller, we used the Phred score genotype error (GQ FORMAT value) converted to probabilities.

**OneMap** uses the forward-backward algorithm (BAUM *ET AL.*, 1970) to compute the HMM combined with the expectation-maximization algorithm (EM). Since version 3.0, **OneMap** presents the possibility to parallelize the HMM using the approach described in SCHIFFTHALER *ET AL.* (2017). It parallelizes the procedure into a maximum of four cores. We used this new **OneMap** feature to estimate the genetic distances in this work.

### 3.2.2 Empirical data

The dataset evaluated here already have a map built in ZHIGUNOV *ET AL.* (2017). The authors made the Illumina dataset available at NCBI under the BioProject PRJNA395596. The dataset comprises sequences of a  $F_1$  full-sib mapping population from the intraspecific cross of two aspen genotypes (*P. tremula*) (ZHIGUNOV *ET AL.*, 2017). RADseq libraries were constructed using *HindIII* and *NalI* enzymes and sequenced as 150 base pair single-end reads on an Illumina HiSeq2500. Eight library replicates were built and sequenced for the parents and only one for each of the 116  $F_1$  offsprings.

The sequencing reads were filtered using the **PreprocessingReads** workflow. It uses the **Stacks** plugin *process\_radtags* (CATCHEN *ET AL.*, 2013a) to filter sequences by restriction site and sequencing quality. The reads were discarded if the average quality score of 50% of its length was below the Phred score of 10 (or 90% probability of being correct). **Cutadapt** (MARTIN, 2011) was used to remove adapters and filter by minimum read length of 64 pb. The sequences were then evaluated in the **EmpiricalReads2Map** workflow.

First, the **EmpiricalReads2Map** workflow performs the alignment with BWA mem (LI, 2013). Here, we used the *Populus trichocarpa* genome version 3.0 (TUSKAN *ET AL.*, 2006) as reference for the reads alignment. The workflow merges the same samples BAM files (LI, 2011) keeping the libraries identification on BAM header and use them to perform the SNP calling and genotype calling with **freebayes** and **GATK**. Multiallelic makers are separated from biallelic using **VCFtools** (DANECEK *ET AL.*, 2011a). Using **BCftools** (LI, 2011), the position of identified markers are used to recover the read counts for reference and alternative alleles of the biallelic markers from the BAM alignment files and create a new VCF file (DANECEK *ET AL.*, 2011a) with updated measures, as suggested in ROS-FREIXEDES *ET AL.* (2018).

For the following steps, we kept only markers in chromosome 10. The biallelic markers were filtered by Phred score of sequencing quality higher than 20, minor allele frequency (MAF) of 5%, maximum missing data allowed of 25%, and mean depth of 6 reads. The multiallelic markers were filtered by sequencing quality, MAF of 5% and maximum missing data of 25%. VCF files with biallelic markers from **freebayes** and **GATK**, and with read counts source from VCF and BAM files are input for the genotype caller software **updog**, **SuperMASSA** and **polyRAD**. The estimated genotypes and genotypes probabilities from each combination of SNP caller, genotype caller, and source of read depth are input for **OneMap** 3.0 and **GUSMap**. The workflow offers the possibility of including or not the multiallelic in the dataset to build the map; here we performed the analysis with both possibilities.

To estimate the genetic distances we kept the reference genome markers positions. Markers were also filtered by **OneMap** using maximum missing data of 25%, because the conversion of the VCF file to **OneMap** raw data includes missing data in unexpected genotypes according to the loci segregation (e.g. in a cross “AA x AB”, genotypes “BB” would be converted to missing data). We also filtered the markers with segregation distortion using alpha 0.05 and Bonferroni correction and by **GUSMap** using MAF of 5%, maximum missing data of 25%, and p-value of 5% for segregation distortion test. In **OneMap**, genetic distances are estimated by HMM multipoint (WU *ET AL.*, 2002d,c) approach using as emission probability a global error rate of  $10^{-5}$  (default in **OneMap** version < 3.0, here we will refer to this scenario as “OneMap\_version2”), a global error rate of 0.05 (5%), and the genotypes probabilities estimated by each genotype caller. All multiallelic markers received an error rate value of 0.05.

Considering all the pipeline combinations, the workflow outputted 34 possible maps for chromosome 10. Here we obtained 68 because we analyzed with and without the multiallelic approach. Table 3.1 makes an overview for notations used to refer to each evaluated scenario.

To also test the influence of the presence of the multiallelic markers in the ordering procedure, we used the built map for the chromosome 10 linkage group, and ordered its markers using **MDSMap** (PREEDY and HACKETT, 2016) (also implemented in **OneMap** 3.0, see Attachment B) and *order\_seq* ordering algorithms with and without the multiallelic markers.

### 3.2.3 Simulation data

We also run **SimulatedReads2Map** workflow to perform the simulations of ddRADseq libraries and sequencing based on the 37% of the chromosome 10 sequence of *Populus trichocarpa* version 3.0, which refers to a sequence with 8.5 megabases from a total chromosome size of about 23 megabases. This sequence comprises 38 cM of the group 10 built in chapter 2, or 21% of the linkage group total size. The **SimulatedReads2Map** workflow uses **RADinitio** software to perform the RADseq simulations. The **RADinitio** works in three separated steps: i) simulation of the variants and multiple individuals via a coalescent simulation under a user-defined demographic model using **msprime**; ii) the simulations of the RAD alleles with user-defined enzymes and considering the possibility of allele dropout; iii) the simulations of the library enrichment and sequencing, where the RAD alleles are sampled with replacement proportionally to the user-defined depth of sequencing.

The **SimulatedReads2Map** workflow skips the **RADinitio** first step to simulate natural populations. Instead, it simulates a  $F_1$  population based on user-defined parents haplotypes or a VCF file containing marker and allelic dosage information and a genetic map. Here, we used as reference the output VCF from the **GATK** obtained with **EmpiricalReads2Map** workflow evaluation of the ZHIGUNOV *ET AL.* (2017) dataset with markers filtered using **VCFTools** (DANECEK *ET AL.*, 2011a). We filtered the reference VCF with MAF 0.05, and maximum missing data of 25%. The parents’ haplotypes were simulated keeping the proportion of B3.7, D1.10, D2.15, and non-informative (“AA x BB”) marker types found in the parents reference VCF file.

The relation between physical and genetic distances of previous maps was used to train a monotone cubic (or Hermite) spline interpolation (HYMAN, 1983) model, which was applied to predict the position in centimorgan to each marker physical position in the reference VCF. We investigate the predictions based on three previous maps: from chapter 2, from ZHIGUNOV *ET AL.* (2017) and TONG *ET AL.* (2020).

The genetic map and the parents' haplotype information were used as input for **PedigreeSim** (VOORRIPS and MALIEPAARD, 2012), a software to perform the meiosis simulations and generate the progeny haplotypes. We did not consider the interference in meiosis events (HALDANE (1919) map function). **PedigreeSim** output was converted to VCF file format which is input for the step "ii" of **RADinitio** to add the polymorphisms in the reference genome sequence and simulate the ddRAD sequences. The **RADinitio** uses the inherited efficiency model (BEST *ET AL.*, 2015) to simulate a PCR amplified pool of molecules. The model includes the heterogeneity of the PCR amplification and the polymerase substitutions errors. After, **RADinitio** uses the user-defined ratio between DNA original molecules to be sequence and PCR duplicates (we used the default parameter with a proportion of 4:1) to create a distribution that will define the number of times the pool of loci is sampled, the number of duplicate molecules that are generated from a RAD locus template, and the distribution of PCR errors in the resulting reads. Besides the PCR errors inserted during the pool sampling, the software also includes Illumina sequencing normal error pattern, where 3' end of the read accumulates more errors than the 5' (GLENN, 2011).

We tested different values of PCR cycles (5, 9, and 14) and mean depth (5, 10, and 20) to simulate the FASTA files. We set the other simulation parameters to obtain 150 bases of read length, sequence size of 350, and restriction enzymes *HindIII* and *NalIII*. The mean read depth parameter for parents individuals was eight times superior to the progeny.

The **RADinitio** does not output the sequence scores of quality. By now, we converted the FASTA file format to FASTQ format including a Phred score of 40 to every base simulated using **seqtk** (LI, 2020) software. After obtaining the FASTQ files, the **SimulatedReads2Map** workflow followed the same tasks as the **EmpiricalReads2Map**, with alignment, SNP, genotype calling and linkage map build.

The **SimulatedReads2Map** workflow makes comparisons between real and estimated results within each step. The comparisons made during the entire workflow can be visualized in the shiny app **Reads2MapApp**. See details of **SimulatedReads2Map** workflow and the shiny app in chapter 2. Not all comparisons were possible to be made with **GUSMap** package because we could not have access to its intermediary information, as the estimated genotypes and their associate probability for each individual in progeny.

The combination of parameters that produced results closer to observed in empirical data was selected to perform the analysis with five repetitions (five families) and two mean depths. Each **SimulatedReads2Map** workflow run generates 68 maps with combinations of SNP caller, genotype caller, source of the reads counts, map builder packages, and the presence or absence of false-positive markers. We also run the **Simulated2Map** workflows two times in the presence and the absence of haplotype-based multiallelic markers. Therefore, the experiment has a total of  $5 \times 2 \times 2 \times 68$  (1360) maps built for the first 8.5 Mb of the chromosome 10 of *Populus trichocarpa* genome. Table 3.1 makes an overview of the notations used to refer to each evaluated scenario.

**Table 3.1.** Notation used to refer to each scenario in empirical and simulated datasets.

Workflow step	Notation	Description
Reads simulations	depth 10	Mean read depth used to simulate the dataset
	depth 20	
SNP calling	freebayes	Software used to identify the variants in the datasets read sequences
	GATK	
	BAM VCF	Source files of read counts information
Genotype calling	polyRAD	Software used to perform the estimation of genotype for a given read count information.
	SuperMASSA updog	
	freebayes/GATK	Depends of which software performed the SNP calling. For example, If the SNP calling was performed with freebayes, the genotype calling will also be.  In these cases, the only source of read counts is the VCF files, because using BAM would return the same results.
Genotype calling + map building	GUSMap	This software performs together the genotype calling and the map building.
Map building	polyRAD	Maps built with genotypes probabilities from <b>polyRAD</b>
	SuperMASSA	Maps built with genotypes probabilities from <b>SuperMASSA</b>
	updog	Maps built with genotypes probabilities from <b>updog</b>
	freebayes/GATK	Maps built with genotype probabilities from <b>freebayes</b> or <b>GATK</b> depending of which SNP caller.
	polyRAD (5%)	Map build with genotypes from <b>polyRAD</b> and global error of 0.05
	SuperMASSA (5%)	Map build with genotypes from <b>SuperMASSA</b> and global error of 0.05
	updog (5%)	Map build with genotypes from <b>updog</b> and global error of 0.05
	freebayes/GATK (5%)	Map build with genotypes from freebayes or <b>GATK</b> and global error of 0.05
	OneMap_version2	Map build with genotypes from freebayes or <b>GATK</b> and global error of 0.00001

### 3.2.4 Statistical analysis

The analysis was performed in markers from each combination of SNP caller, genotype caller, and source of read counts after they were filtered by sequencing quality, MAF, segregation distortion, redundancy, and missing data. Outlier markers breaking the pattern of the recombination fraction matrix were removed only for the ordering test with and without haplotype-based multiallelic markers in the empirical dataset.

We evaluate the estimated progenies genotypes concordance by comparing the agreement between real and estimated heterozygous, reference allele homozygous (homozygous-ref), and alternative allele homozygous (homozygous-alt). For that, we used conditional probabilities:  $P(Estimate = E | Method = M \cap Real = R)$ . It returns the probability of an estimated genotype given a method and

a real genotype. The methods are the combination of each SNP caller, genotype caller, and reads count source. We expect that a good method results in high probabilities for the same estimated and real genotypes (e.g.  $P(E = \text{heterozygous} | M \cap R = \text{heterozygous})$ ) and low probabilities when they are different (e.g.  $P(E = \text{heterozygous} | M \cap R = \text{homozygous} - \text{alt})$ ).

To summarize each genotype caller model predictive power according to their outputted genotype probabilities we used *receiver operating characteristic* (ROC) curves. It plots the sensitivity ( $\frac{\text{true positives}}{\text{true positives} + \text{false negatives}}$ ) in the vertical axis versus (1 - specificity) ( $\frac{\text{false positives}}{\text{false positives} + \text{true negatives}}$ ) on the horizontal axis for all possible thresholds  $\pi_0$  in a logistic regression (BERKSON, 1944):

$$\text{logit}[\pi(x)] = \log\left[\frac{\pi(x)}{1 - \pi(x)}\right] = \alpha + \beta x$$

or

$$\pi(x) = \frac{e^{\alpha + \beta x}}{1 + e^{\alpha + \beta x}}$$

where  $x$  is the error rate of the genotypes (1 - highest genotype probability) and the binary response variable is if they were called correctly or wrongly. The formula implies that  $\pi(x)$  changes as an S-shaped function of  $x$ . Parameter  $\beta$  determines the rate of increase or decrease of the S-shaped curve for  $\pi(x)$  (SLOANE and MORGAN, 1996).

Higher is the sensitivity value for each (1 - specificity), better this particular error rate threshold can differentiate wrong and correct genotypes. The best threshold would be the one more close to the left superior corner of the graphic. Therefore, the better the predictive power, the higher is the ROC curve. Because of this, the area under the curve provides a single value that summarizes predictive power. The greater the area, the better the predictive power of the outputted error rate from the genotype call model used (BRADLEY, 1997).

To test the capabilities of software correctly estimating the parents' genotypes, we used the same conditional probability, but, instead of measuring the similarities between individuals' genotypes, we tested the combination of both parents' genotypes. To do that we performed the conditional probabilities analysis between the marker types (e.g.  $P(E = B3.7 | M \cap R = B3.7)$ ).

Based on MOLLINARI *ET AL.* (2009), we compare the centimorgan distances of markers in the maps estimated by each method and the real map using the Euclidean distance (D):

$$D = [(m - 1)^{-1}(\hat{d} - d)'(\hat{d} - d)]^{1/2}$$

where  $m$  is the number of markers evaluated,  $\hat{d}$  is the vector of estimated distances,  $d$  is the vector of real distances, and  $'$  indicates vector transposition. A value of  $D = 1$  means that the estimated map differs with an average of 1 cM with the built map, regarding all genomic positions.

We evaluate the orders provided by the different ordering algorithms applied to empirical data with and without the haplotype-based multiallelic markers. For that, we use the absolute value of the Spearman's rank correlation coefficient  $\rho$  (SPEARMAN, 1902) calculated with:

$$\rho = 1 - 6 \sum_{i=1}^m d_i^2 / m(m^2 - 1)$$

where  $m$  is the number of markers evaluated,  $d_i$  is the difference between the rank of marker  $M_i$  on the order obtained from a given ordering procedure, and the rank of marker  $M_i$  on the reference genome position. We compared the marker's distances between each ordering algorithm with the map built with genomic position also using the Euclidean distance (D).

### 3.3 Results and Discussion

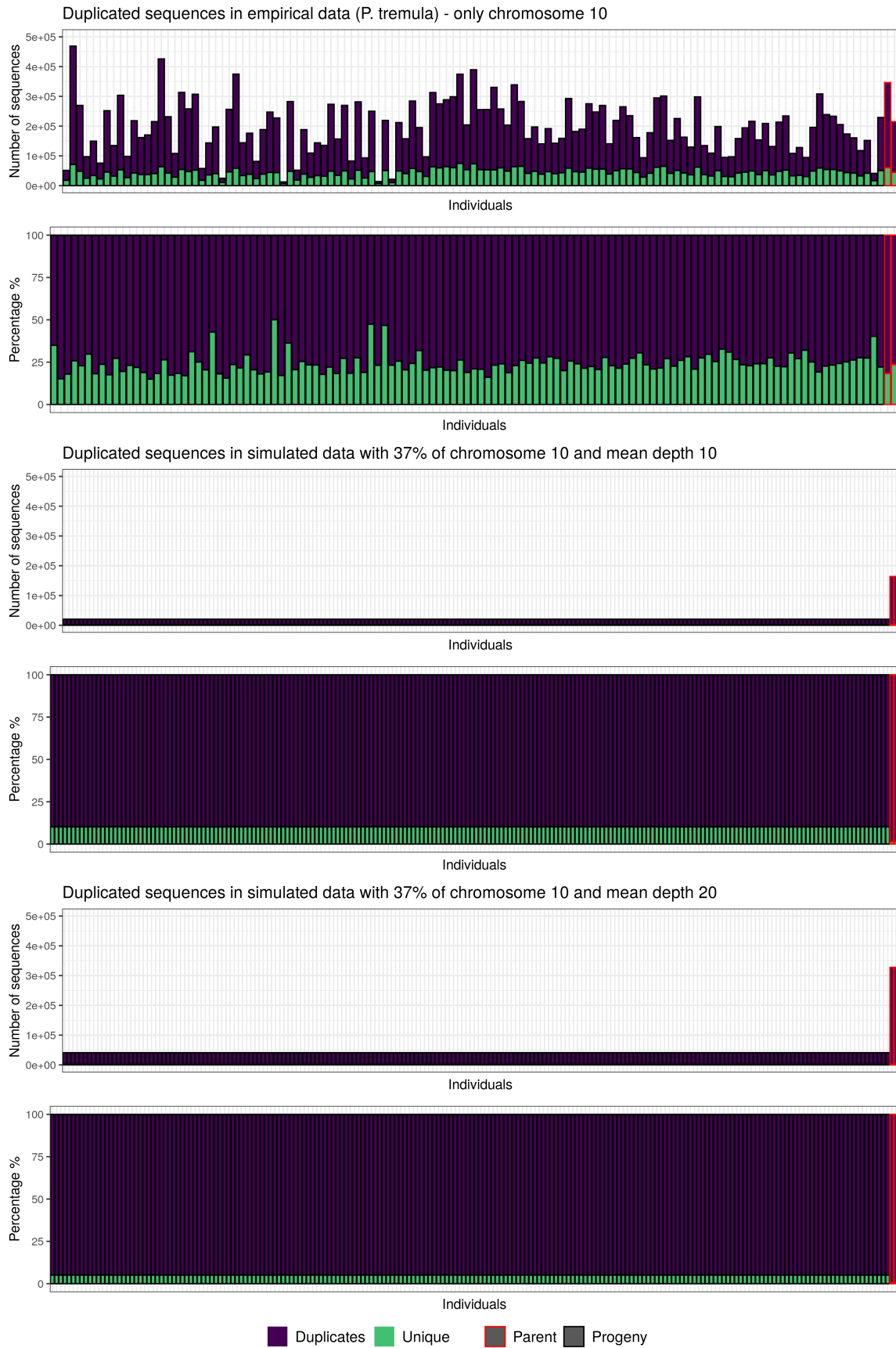
#### 3.3.1 SNP calling

After the filtering of FASTQ files by Phred quality score, the presence of the restriction enzyme cut site, and the adaptors, and only reads alignment to the chromosome 10 reference sequence, it remained in the empirical *P. tremula* dataset about 300 thousands of read sequences for the parents' samples and a highly variable number in progeny samples, with a minimum of 12 thousand and a maximum of 469 thousands of read sequences (Figure 3.1). The mean percentage of duplicated reads in the empirical dataset was 76% (SE 0.55) and in the simulated dataset with mean depth 10 was 88% (SE 0) and in the dataset with mean depth 20 was 92% (SE 0) according to `picard MarkDuplicates` tool (INSTITUTE, 2009). It shows that RADinitio simulates more duplicates by cycle than the empirical PCR performed to generate the *P. tremula* dataset, for which was performed 14 cycles (ZHIGUNOV *ET AL.*, 2017).

To our best knowledge, the RADinitio software for sequencing reads simulations is the only one available by now that was specific designed for RADseq data sets and able to simulate the duplicates. Here we did not use the RADinitio first step to simulate the individuals and variants. Instead, we based our simulations on the variants identified in the *P. tremula* empirical dataset by GATK software in the **EmpiricalReads2Map** workflow. The GATK software VCF format, with MNP markers split by row, made it possible to simulate single SNPs and the MNPs haplotypes, but indels needed to be removed because of RADinitio currents limitations. The default parameter used of RADinitio simulates a ratio of four DNA template molecules per sequenced DNA reads (4:1). The RADinitio also allows the user to define the mean depth of the sequencing. With the **Reads2Map** workflows, we tested several values for these parameters, to try to be as similar as possible with the empirical data (Figure 3.3). Changing the values of the parameters we found that with low mean depths (5) and any of the number of PCR cycles tested (5, 9, and 14) maps could not be built with markers identified by GATK, because the segregation test filters out all of them. Setting the mean to 10 and a high number of PCR cycles (9 and 14) also kept few markers in the GATK analysis. Thus, our experiments confirmed what DER AUWERA *ET AL.* (2020) already pointed about the reduced capacity of GATK method to identify markers in PCR-based libraries with a high number of duplicated sequences. However, GATK demonstrated high and stable accuracy in datasets with fewer duplicates (Table 3.3). This GATK characteristic was also demonstrated in previous comparisons (SCHILBERT *ET AL.*, 2020; YAO *ET AL.*, 2020; PEREA *ET AL.*, 2016) and in its spread usage with RADseq data. Therefore, we performed all the simulated scenarios using 5 PCR cycles with mean depths of 10 and 20.

In the empirical data, both `freebayes` and GATK SNP calling resulted in about 2 million markers considering the entire *P. tremula* genome, and about 304 and 134 thousands respectively for the chromosome 10. Filtering the VCF files by sequencing quality, MAF, missing data and depth, we only kept 6% of markers identified by `freebayes` and 7% by GATK (Table 3.2).





**Figure 3.1.** Number and percentage of duplicated reads in empirical and simulated sequences. For the empirical data, the library preparation was conducted with 14 PCR cycles. For the simulations with RADinitio, 5 PCR cycles was considered.

**Table 3.2.** The filters applied to markers identified by **freebayes** and **GATK** software in *P. tremula* RADseq data. The values in the table refer to the sum of total biallelics and multiallelics markers. Filtered and normalized VCF biallelic markers positions were used to recover allele depth information from the BAM files. \* Datasets used as inputs in **maps\_emp** workflow to build the linkage maps with 34 different approaches. \*\* Dataset reference for the simulations.

Filters	freebayes	GATK
Chr10	304468 (100%)	133782 (100%)
Chr10 - mean quality >20	229474 (75%)	133782 (100%)
Chr10 - MAF <5%	152634 (50%)	61804 (46%)
Chr10 - missing data >25%	103983 (34%)	25715 (19%)
Chr10 - mean depth by marker <6	45731 (15%)	15717 (12%)
Chr10 - mean quality >20 - MAF <5%	VCF 17621 (6%)*	VCF 9723 (7%)*
- missing data >25%	BAM 15614 (6%)*	BAM 13961 (11%)*
- mean depth by marker <6		
70 % of Chr10 - MAF <5%		
- missing data >25%	-	1003 (0.7%)**
- indels	-	-

For our simulation data, the **freebayes** and **GATK** kept a higher number of false-positive (type I errors) and false negative (type II errors) markers. Despite the simulation presented a higher number of duplicates than expected in an empirical dataset, the filters applied were able to remove most of the SNP calling errors, mainly in the markers identified by **GATK** (Table 3.3).

**Table 3.3.** Percentages of mean rate of false negative (FN) and false positive (FP) markers identified by SNP callers **freebayes** and **GATK** in the simulated data after datasets were filtered according to sequencing quality, MAF, depth, and missing data. The comparison between the VCF files was made using **GATK3 VariantsEval** tool

Filters		freebayes		GATK	
		depth 10	depth 20	depth 10	depth 20
Chr10	Total markers (#)	27707	65272	10940	5841
	FN (%)	82.6	81.4	78.7	78.8
	FP (%)	99.4	99.7	98.1	96.5
Chr10 - MAF >5%	Total markers (#)	178	174	1374	1898
	FN (%)	84.2	85.0	79.4	78.9
	FP (%)	13.2	15.5	85.3	89.1
Chr10 - missing data >25%	Total markers (#)	27774	65312	9696	4124
	FN (%)	82.6	81.7	78.8	79.2
	FP (%)	99.4	99.7	97.9	95.1
Chr10 - mean depth by marker >6	Total markers (#)	27705	65312	9713	4152
	FN (%)	83.0	81.7	78.6	78.7
	FP (%)	99.4	99.7	97.8	95.0
Chr10 - mean quality >20 - MAF >5%	Total markers (#)	178	174	200	202
	FN (%)	84.2	85.0	79.6	79.4
- missing data >25%					
- mean depth by marker >6	FP (%)	13.2	15.5	0.493	0

Each step of the pipeline, starting by collecting the samples until the built linkage map can include errors in the estimated genotypes. If markers are not filtered properly during the procedure, in the end, the map building software will demand lots of computational resources and would not be able to estimate the phases and genetic distances precisely. The approach of considering the genotype probabilities in the HMM can only identify the errors if all polymorphisms were identified correctly by

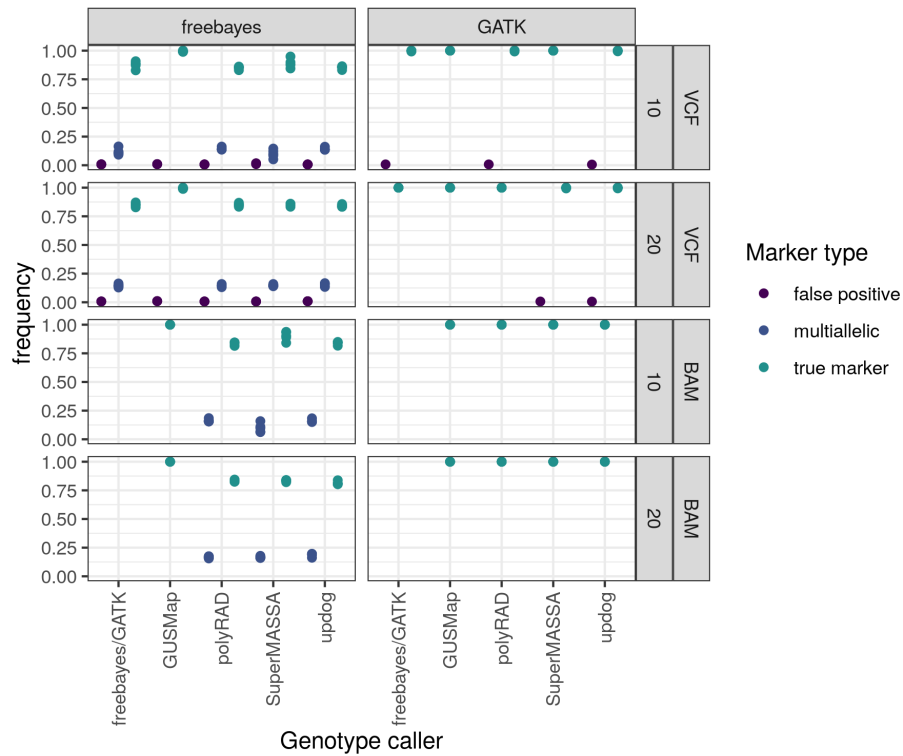
the SNP callers, if the genotype errors are not the majority of the dataset and if the markers are in the correct order.

In the SNP calling step, type II errors are consequences of wrong nucleotide bases inserted in the sequence or wrong alignment. Some of the nucleotide substitutions can be caused by DNA damage during the extraction (CHEN *ET AL.*, 2017; COSTELLO *ET AL.*, 2013), or by spontaneous deamination of methylated cytosine to uracil (CHEN *ET AL.*, 2014). Also, the polymerase errors in the PCR step (BEST *ET AL.*, 2015) and sequencer addition of wrong nucleotides to the sequence (MA *ET AL.*, 2019). The errors caused by the wrong alignment of the sequences can be reduced by filtering only reads that aligned exactly to the sequence, with a high MAPQ value (MARSHALL *ET AL.*, 2012).

For WGS and exome sequencing data is recommended to filter out duplicated sequences. These sequences come from the library preparation using the PCR or also, from erroneous detection of a single amplification cluster as multiples by the optical sensor of the sequencing instrument (DEPRISTO *ET AL.*, 2011). For those library types, we expect an overlap of just part of the reads. Thus, duplicates are redundant data that can add more errors to the analysis. In this context, the identification of the duplicates to be filtered is usually made based on the quality score of the sequence bases. However, duplicated sequences are expected for RADseq data because all sequences have a common starting point: the restriction enzyme cut site. Filtering duplicates, in this case, would filter out most of the dataset, reducing the read depth by loci and increasing the uncertainties about which presents the right sequence (KAGALE *ET AL.*, 2016). Duplicates in RADseq presents advantages to sequencing depth, however, they also bring more erroneous nucleotide substitutions from PCR.

The amount of false positives markers identified by **GATK** and **freebayes** reveal the importance of applying the proper filters to the dataset. There are different strategies for filtering variants (SCHILLING *ET AL.*, 2014). Some based on simple measures as the read depth, alignment quality, minor allele frequency, and others more robust, like those available in **GATK** toolkit, which implemented machine learning methods to model the errors in the dataset based on a validated variant database (DEPRISTO *ET AL.*, 2011; AUWERA *ET AL.*, 2013). Mapping populations have the advantage of having known segregation, once it is a controlled cross. Filters based on biological aspects used to be more efficient than others (SCHILLING *ET AL.*, 2014). Here we applied filters to empirical data to keep in the dataset only high depth markers to not overload the HMM with too many genotyping errors and also to reduce the total number of markers to be evaluated by the **maps\_emp** workflow, once the analysis is computational demanding.

For the simulations, we kept the only 37% of the original chromosome, then we could perform all the procedures without using other filters than the biological and the missing data. However, this approach kept to the next step false positive markers in some studied scenarios (Figure 3.2, see also notation Table 3.1).

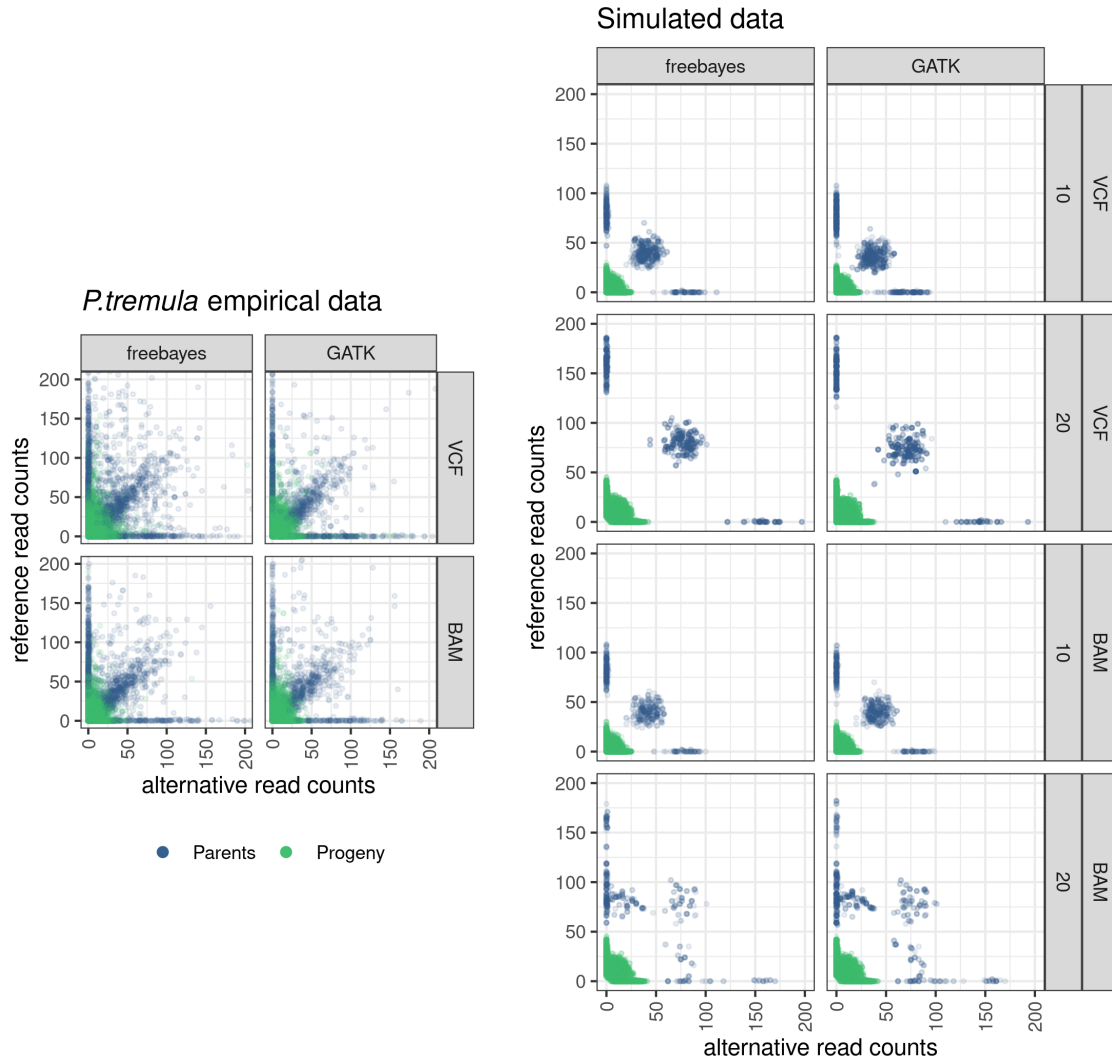


**Figure 3.2.** Frequency of correct SNPs, haplotype-based multiallelic and false positives markers identified by GATK and freebayes in the simulated data after filtered by segregation distortion, redundancy, and missing data.

The false negatives markers reduce the genome coverage of the analysis. The low coverage in RADseq is already expected because of the representation reduction of the libraries with the restriction enzymes. In the  $F_1$  population context, with a large disequilibrium size, the low coverage usually is not a problem. In our simulated data, we simulated markers close to the restriction enzyme cut site that was identified in *P. tremula* empirical data. However, the simulations consider that the efficiency of the enzyme can vary across libraries what may explain the high number of false negatives observed (Table 3.3). We used more permissive filters in the GATK VCF used as a reference for simulations to increase the possible number of SNPs to be found in the studied region of the chromosome.

### 3.3.2 Genotype calling

Once the markers are identified, the genotypes of each individual can be estimated according to the read count at each locus. Ideally, in a diploid individual, the homozygous would receive the same allele from both parents, and the heterozygous would have half of the reads containing one allele and half a different one. However, we can observe deviation of this ideal scenario using RADseq data (Figure 3.3).



**Figure 3.3.** Read counts for genotypes alternative and reference alleles from VCF and BAM files identified by freebayes and GATK in the *P. tremula* and in simulated datasets. Plots were made with a subset with  $3 \times 10^5$  genotypes.

The RADinitio simulation results in alleles read counts distribution close to observed in the progeny of the empirical data in terms of dispersion, allelic bias, and allelic dropout (RIVERA-COLÓN *ET AL.*, 2020), but it could not simulate the low-depth counts for parents and neither the outlier genotypes presented in the empirical dataset. Thus, our simulations were not able to cover these two characteristics that can be found in empirical datasets.

The allelic bias has been observed frequently in RADseq data (GERARD *ET AL.*, 2018; RIVERA-COLÓN *ET AL.*, 2020). The main source of bias of RADseq data is related to the PCR amplification step in its library preparation (DER AUWERA *ET AL.*, 2020; RIVERA-COLÓN *ET AL.*, 2020). PCR reagents randomly amplify DNA molecules with the presence of specific primers. If the reaction environment contains few or poor quality original DNA molecules, the primers will not hybridize in all unique molecular sequences at the first PCR cycle. As consequence, the target sequences identified by the primers only in later cycles will have few copies compared with those identified in the first. Or maybe some molecules would never be identified by the reaction reagents, causing the allele dropout. This is likely to happen because, after several cycles, the PCR cloned sequence are more abundant in the reaction than the original, making them easier targets for the PCR primers.

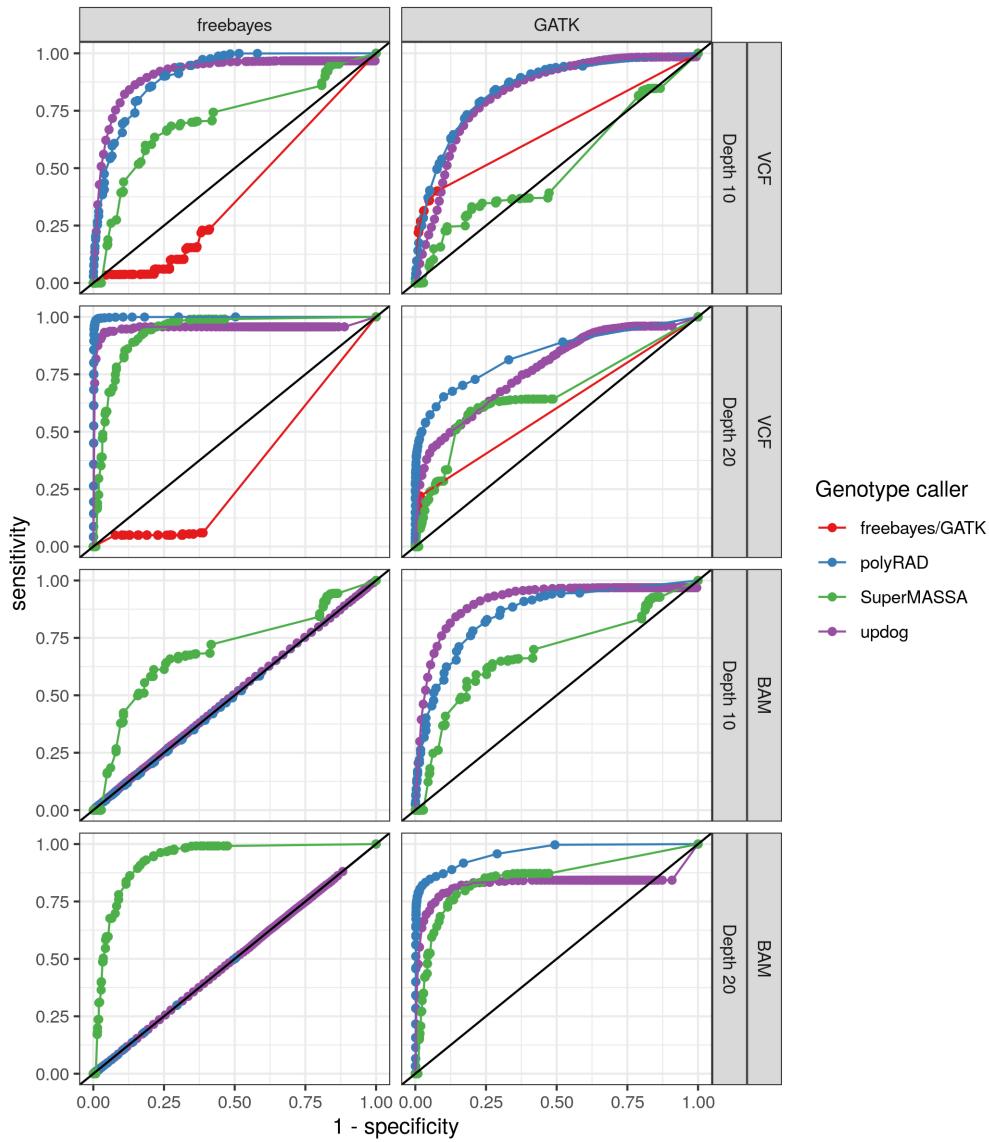
OneMap 3.0 estimate genetic distances properly if i) the genotype caller do not make mistakes estimating the genotypes and keep low error rate for these correctly inferred genotypes (e.g. in Figure A.15 column “Real:homozygous-ref/Est:homozygous-ref” where  $P(E = \text{homozygous} - \text{ref} | M \cap R = \text{homozygous} - \text{ref}) = 1$  and  $e = 0$ ) ; or ii) if doing mistakes, the software output a high error rate for the wrong genotypes (e.g  $P(E = \text{homozygous} - \text{alt} | M \cap R = \text{heterozygous}) = 0.2$  and  $e = 0.4$  in Figure A.13 column “Real: heterozygous/Est: homozygous-alt”).

With the simulations we could measure the reliability of each software outputted genotype probabilities for situation ii). Figure 3.4 shows the specificity and sensitivity profile considering different thresholds in the genotype probabilities for each scenario. Higher is the area under the curve, the higher is the genotypes probabilities reliability. Genotype probabilities thresholds closer to the left superior corner have a higher capacity to differentiate right and wrong genotypes. Scenario with freebayes SNP caller, and VCF as the source of counts presented the most reliable genotype probabilities for all genotype caller. Considering the GATK SNP caller, the probabilities are better when read count source is the BAM files. In scenario with freebayes SNP caller, polyRAD and updog genotype callers and BAM as counts source, the result was a diagonal line because almost half of the dataset is composed by wrong genotypes (Table 3.4) and the probabilities values are random distributed between right and wrong genotypes (Figures A.15, A.13 and A.14).

**Table 3.4.** Percentage of wrong genotypes in each studied scenario. The genotype call with GATK and freebayes using the read count from BAM files would output the same results as using the VCF files, then, we did not repeat the analysis and there are no results available for these scenarios.

SNPCall	CountsFrom	depth	freebayes/GATK	polyRAD	SuperMASSA	updog
freebayes	VCF	Depth 10	2.25	1.68	7.88	2.45
freebayes	VCF	Depth 20	0.16	1.15	2.08	0.17
GATK	VCF	Depth 10	7.31	5.09	14.70	10.68
GATK	VCF	Depth 20	0.98	0.67	5.12	3.08
freebayes	BAM	Depth 10	not applicable	48.99	7.77	49.00
freebayes	BAM	Depth 20	not applicable	49.69	2.07	49.78
GATK	BAM	Depth 10	not applicable	2.44	8.14	2.87
GATK	BAM	Depth 20	not applicable	1.91	2.36	0.46

In general, the results showed that all software presents more reliable genotypes for reference homozygous than alternative or heterozygous, which is a consequence of the bias for the reference allele in the dataset. GERARD *ET AL.* (2018); VAN DE GEIJN *ET AL.* (2015) proposed methods to overcome this issue. VAN DE GEIJN *ET AL.* (2015) points to the alignment step as the cause of the bias because reference alleles have more similarity with the target genome and tend to be easily aligned. The authors propose a simple bioinformatic method to avoid this problem. The method comprises align the reads two times, one with the original genome sequence and the other replacing the alternative bases identified in their specific position in the genome. Only reads aligned in both cases are kept for the analysis. However, GERARD *ET AL.* (2018) argument that the alignment could not be the only reason it happens. Thus, the authors propose on Bayesian model to adjust the bias based on allele count distribution after bioinformatics traditional approaches. The method of GERARD *ET AL.* (2018) is implemented in updog.



**Figure 3.4.** ROC curves considering the real and estimated genotypes from the five families simulated with mean depth 10 and 20 and with the firsts 8.5Mb of the chromosome 10 (38 cM). Here only biallelic markers are considered.

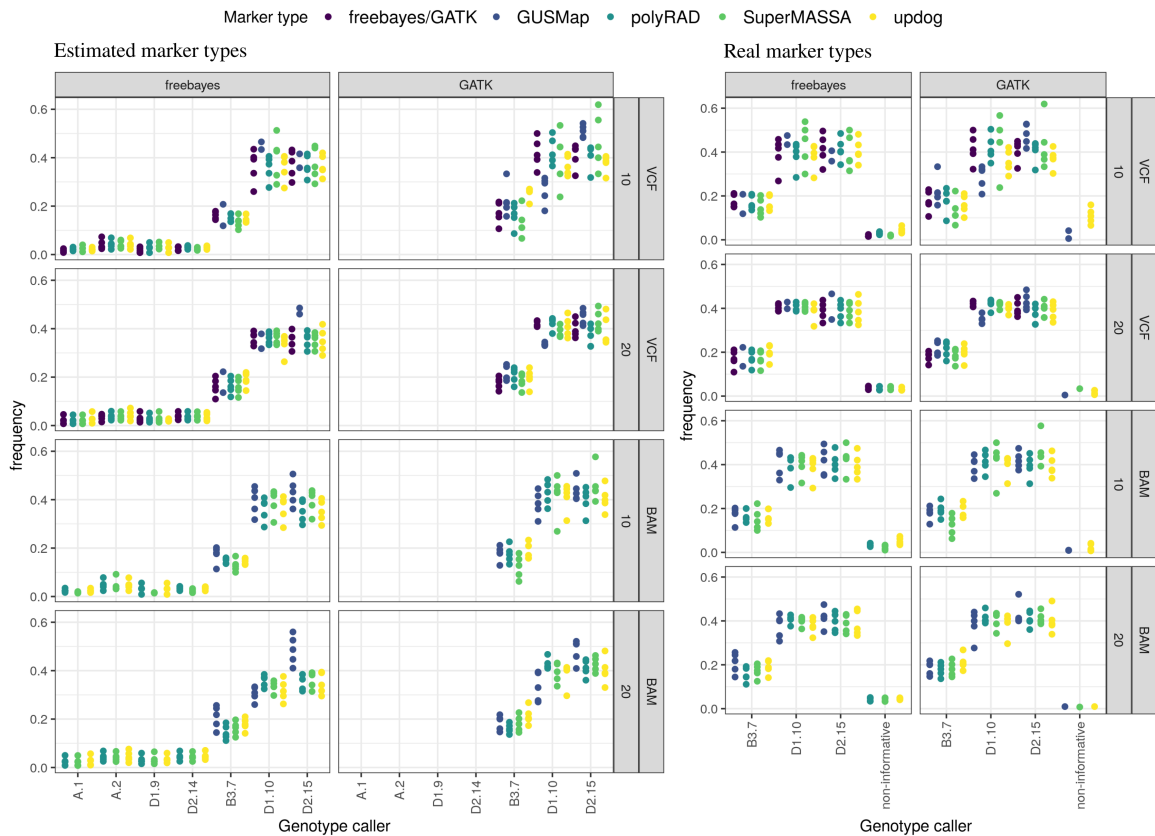
The SuperMASSA (SERANG *ET AL.*, 2012) was designed to estimate ploidy and allele dosage in polyploid species. Using a Bayesian approach it combines information of reads depths in parents and progeny and the population structure to estimate the genotypes. Here we use the  $F_1$  model and a modified version of the software to output the genotypes probabilities of each individual. Similarly with updog, it uses the population segregation as a priori to model the distribution. The polyRAD (CLARK *ET AL.*, 2019) was also designed to estimate alleles dosage in polyploid species also using the Bayesian approach and considering population structure as *priori*.

OneMap 3.0 does not consider the parents' genotype probabilities or error rates. Both parents' genotype information is combined and identified by the expected segregation for the locus cross (Table A.2). Thus, it is important to plan the sequencing experiment with high-quality parents' genotypes, because if they contain errors, they would not be identified by the map building procedure and it will cause distortions in the resulted distances and haplotypes. Here, the empirical dataset presented eight replicates for parents samples to be sequenced and we also simulated the parents with eight times more mean depth.

The informative combinations for outcrossing species with biallelic codominant markers must have at least one heterozygous genotype in one of the parents, which includes the marker types B3.7, D1.10, and D2.15. The haplotype-based multiallelic codominant markers can also present types A.1, A.2, D1.9, and D2.14.

The approach implemented in **SimulatedReads2Map** workflow simulates the parents' haplotypes using the same proportion of marker types identified in the empirical VCF file. This approach overcomes the missing data present in the empirical dataset. The VCF file used here as reference to the simulations contains 4 markers A.2, 76 B3.7, 173 D1.10, 1 D1.9, 2 D2.14, 162 D2.15, and 219 non-informative markers. We considered non-informative markers those with both parents homozygous.

To measure the genotyping concordance in parents we did not compare the real and estimated individual parents genotypes, but the combination of each locus in both parents. We separated the analysis between multiallelic and biallelic markers for each method (Figure 3.5). In our simulations, only haplotype-based markers from **freebayes** remained in the dataset after filters.



**Figure 3.5.** Frequencies of each marker type observed according to estimated and real parents genotypes. Here we did not account with false negative markers. The total number of markers here are the same in real and estimated representations.

We expect that all multiallelic markers come from combinations of biallelic marker types, then we found high conditional probabilities in this case (Appendix XII). The comparison showed the amount of B3.7, D1.10, D2.15, and non-informative markers converted to A.1, A.2, D1.9, and D2.14 haplotype-based markers. A higher proportion of B3.7 markers were converted to A.1 and A.2. The markers D1.9 and D2.14 haplotypes-based were converted from D1.10 and D2.15 SNPs combinations, respectively. Also, the haplotyping approach could combine a few non-informative into A.1, D1.9, and D2.14 markers.

Considering only biallelic markers in the conditional probabilities analysis, the identification of multiallelic marker types means erroneous estimated parents' genotypes. We observed higher number of



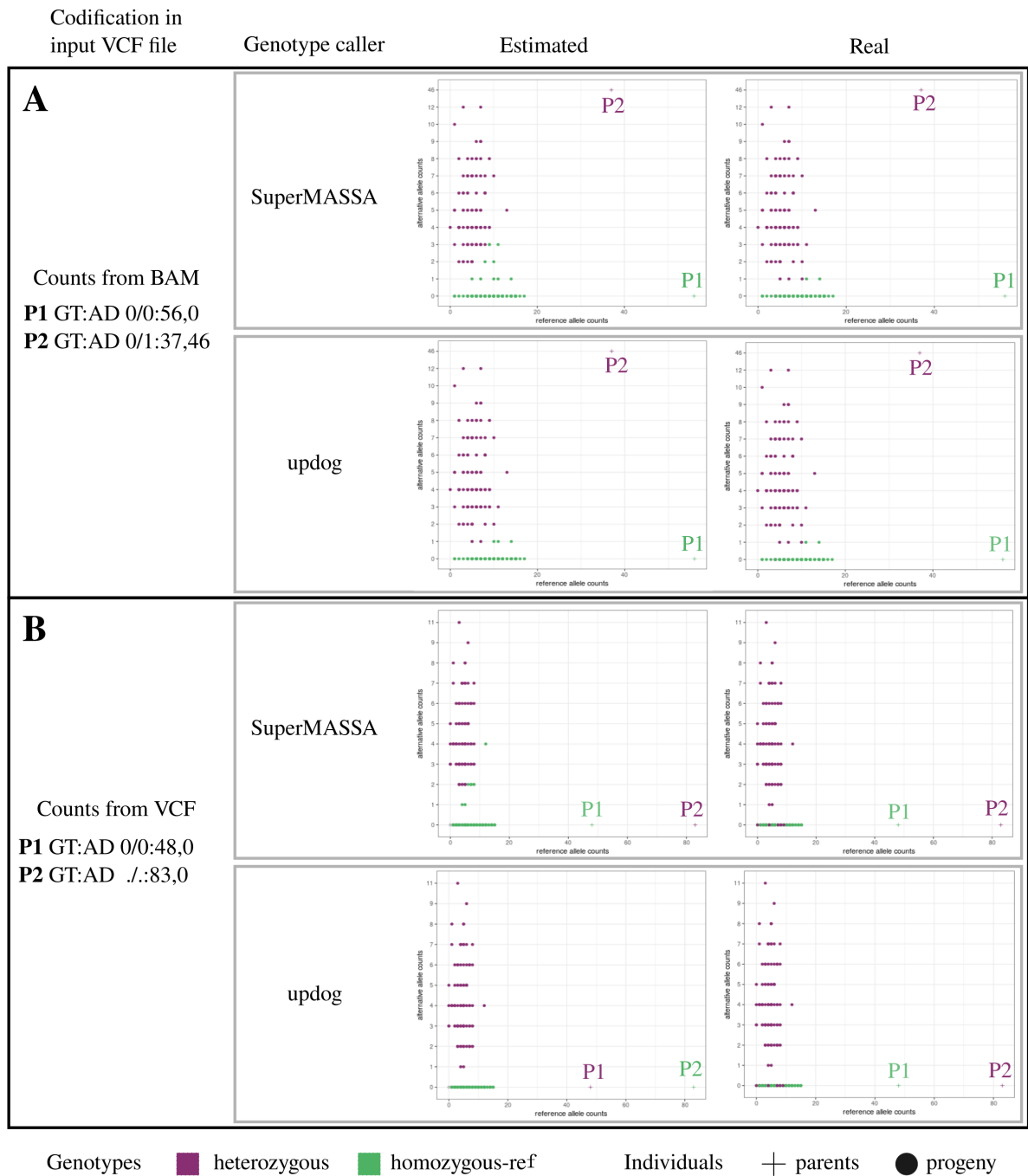
genotyping errors estimating parents' heterozygous genotypes because of the already described dispersion effect and also because they require higher sequencing depth to avoid random sampling of reads effect. As consequence, some B3.7 markers are called D1.10 or D2.15.

We also observed a relatively high proportion of estimated markers as D2.15 type given D1.10 as real and vice-versa by GUSMap, SuperMASSA, and updog in some scenarios (Figures A.22 and A.21). For the SuperMASSA and updog genotype caller, we identify the markers presenting this error and evaluated the pattern of the progeny estimated genotypes and the allele read counts. The GUSMap uses the read counts directly in the genetic map distance estimation by HMM, we could not access progeny intermediary genotype values or genotype probabilities to explore the reasons for the erroneous marker types estimated.

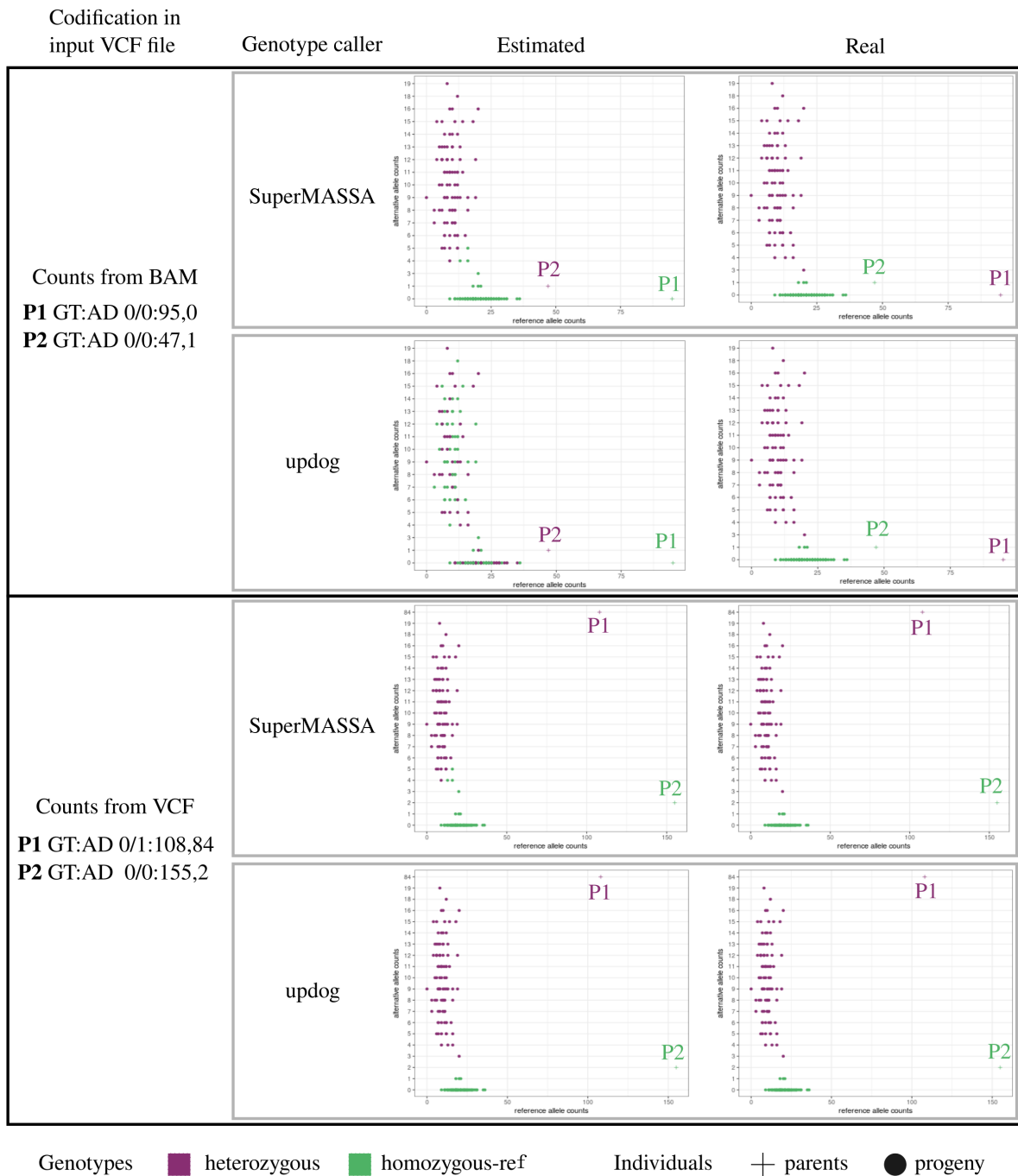
The Figure 3.6 and 3.7 represents patterns frequently observed in markers D2.15 erroneously called as D1.10 and vice-versa. In both images we can see allele dropout in some individuals' genotypes, some of them should be called heterozygous (real genotype) but only presented the reference allele. We identify different reasons for this scenario: i) if GATK software interprets a genotype as missing data according to its filtering criteria, it always outputs the total allele read counts in the reference count field of the VCF file (Figure 3.6 B). When extracting the counts directly from the VCF file, the genotype callers GUSMap, updog, SuperMASSA, and polyRAD can not identify alternative alleles for these genotypes. Using the read counts from the BAM files solves this issue (Figure 3.6 A); ii) the local-alignment of freebayes can differ significantly from the BWA alignment, using the BWA BAM files as the source of reads counts information may result in different read count proportions (Figure 3.7); iii) the allele dropout can also be a consequence of polymorphisms in enzyme cut site or the non-amplification of one allele in the PCR step. Despite we could not identify a particular marker with this pattern in our simulated dataset, it can be found in an empirical dataset (RIVERA-COLÓN *ET AL.*, 2020).

Updog and SuperMASSA genotype callers models consider the segregation pattern of population to infer the genotypes. When the progeny presents expected segregation closer to 1:1, the models estimate one of the parents as heterozygous, but they can make mistakes about which of the parents is the one. It can also be the cause of the wrong estimations of non-informative markers.

OneMap does not consider the parents' genotype probabilities in the HMM genetic map estimation, then, to avoid this problem, these erroneous markers must be removed before the linkage map building. This can be done easily by filtering the parents' genotypes according to their genotype probabilities if these values reflect the wrong estimation.



**Figure 3.6.** Allele read counts in marker Chr10\_1311451 identified by GATK considering all individuals in a simulated population with mean depth 10. Colors represent the heterozygous and homozygous genotypes. The marker was simulated as D2.15 marker type, but updog estimate it as D1.10 when read counts from VCF file was used. This marker was considered non-informative by GATK and polyRAD. This example case represents a frequently observed pattern of erroneous genotype estimations by SuperMASSA and updog.

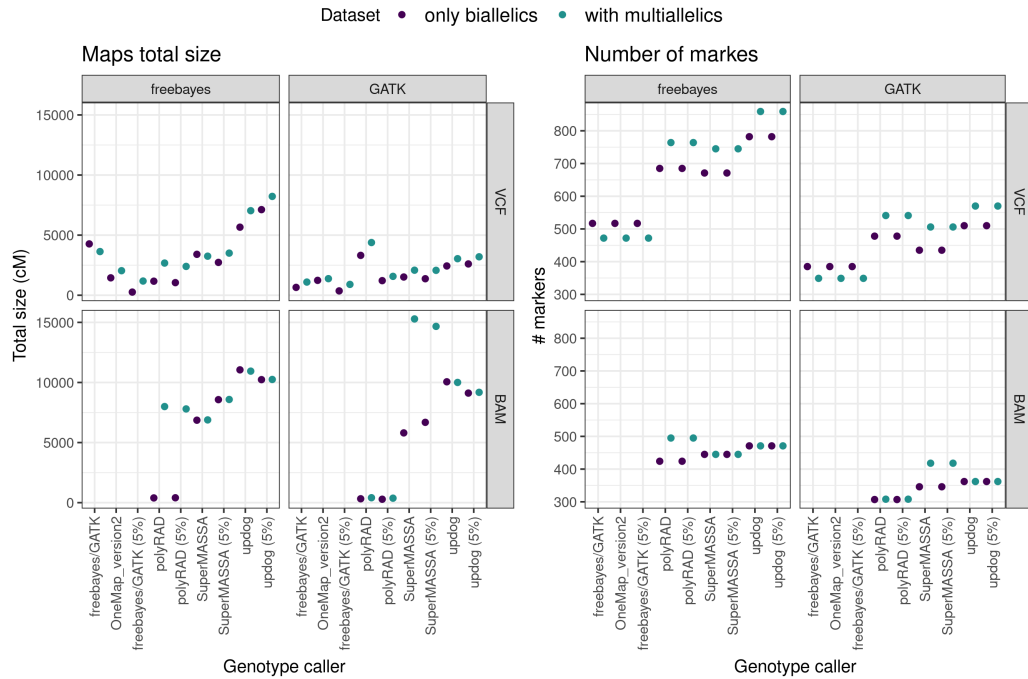


**Figure 3.7.** Allele read counts in marker Chr10\_2324067 identified by *freebayes* considering all individuals in a simulated population with mean depth 20. Colors represents the heterozygous and homozygous genotypes. The marker was simulated as D1.10 marker type, but *updog* and *SuperMASSA* estimate it as D2.15 when read counts from BAM file were used. This marker was considered non-informative according to *freebayes* genotype calling. The *polyRAD* software estimated the parents' genotypes correctly using this same marker. This example case represents a frequently observed pattern of erroneous genotype estimations by *SuperMASSA* and *updog*.

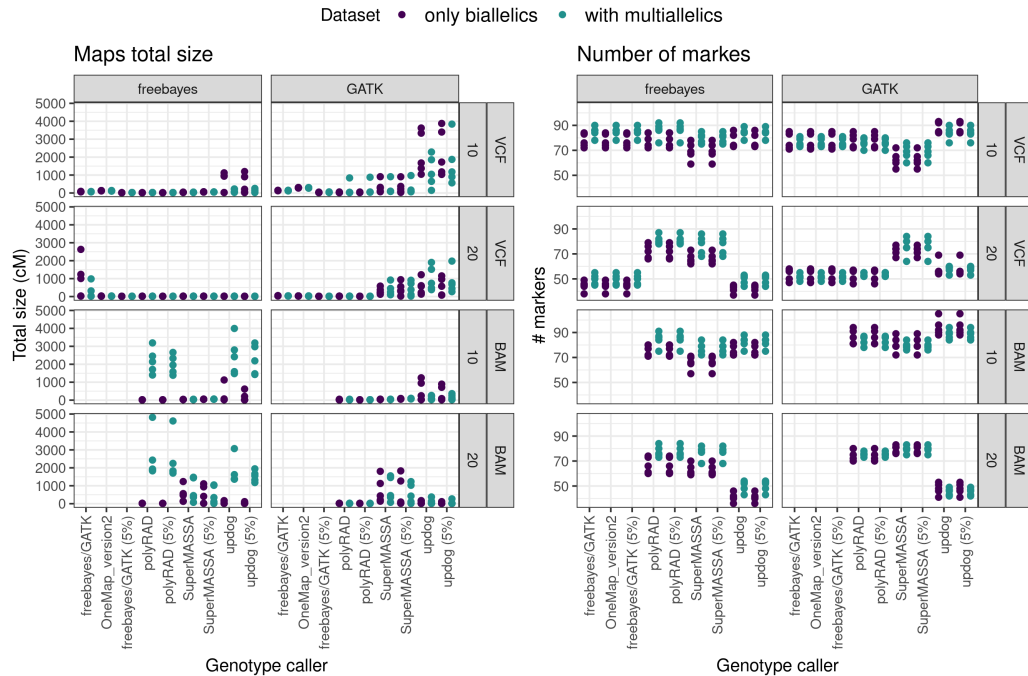
### 3.3.3 Linkage map building

Until this session, we explored only the datasets obtained running the **Reads2Map** workflow including the haplotype-based markers. Here we will discuss also the datasets evaluated only with biallelic

and compare the difference between them in real and empirical data (Figures 3.8 and 3.9).



**Figure 3.8.** The total map size and the number of markers in the genetics maps built by **Empirical-Reads2Map** workflow for *P. tremula* dataset.



**Figure 3.9.** The total map size and the number of markers in the genetics maps built by **SimulatedReads2Map** workflow based on 37% of *P. trichocarpa* chromosome 10. For this graphic, we did not filtered out the false positives markers present in the dataset. GUSMap maps presented high variation between the simulations repetitions. Some of their maps presented size of 12000 cM, which distorted this figure graphic scale. The comparison including these maps is in Figure A.24

The empirical and real results are not comparable in terms of total size because they refer to different chromosome sizes. The **SimulatedReads2Map** workflow would require unreliable time and computer resources to run all 1360 maps in our experiment considering the entire chromosome. Even with the parallelized **BatchMap** (SCHIFFTHALER *ET AL.*, 2017) approach, the estimation of the distance with HMM is still the slowest step in the workflow when there are too many markers (see times spent to estimate the map distances in A.25 and A.26).

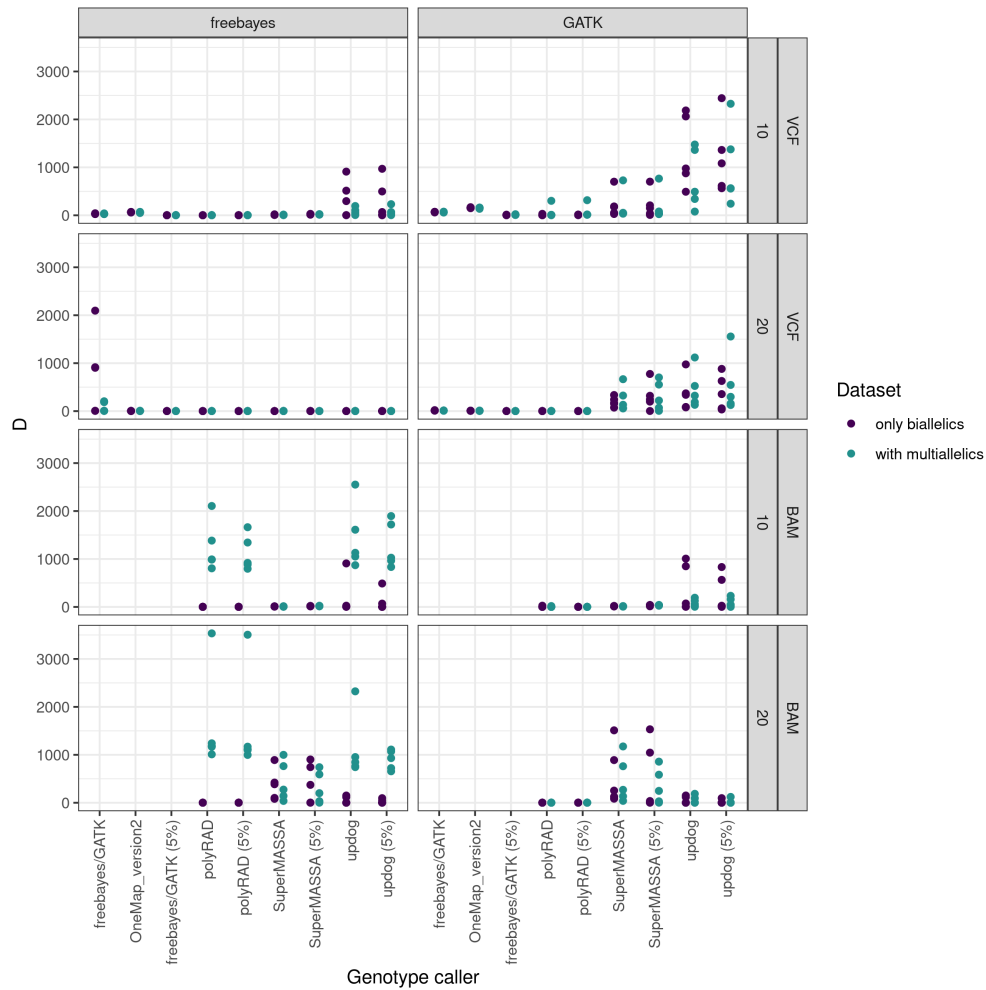
We selected the map built in chapter 2 to be our reference for the recombination rate simulation because it presented a lower number and smaller size of gaps compared with other previous maps and it did not present many inversions compared with the reference genome (Appendix IX).

Running **GUSMap** with fixed order as presented in BILTON *ET AL.* (2018), we obtained “Inf” value for the map genetic distances in some of the scenarios tested in **EmpiricalReads2Map**. It usually happens because the algorithms identify non-linked markers in the same linkage group. Then, it returns a numerical problem because of the non-expected value. **OneMap** used to have the same problem. To overcome this issue, in **SimulatedReads2Map** we include a function to automatically remove these markers from **OneMap** and **GUSMap** analysis.

The workflows build maps from sequencing reads considering the filter parameters defined in the input file. It can not remove automatically possible outliers markers. The outliers removal requires careful evaluation of diagnostic graphics, such as the heatmaps of recombination fraction matrix. Thus, the maps built by the workflows include outliers which expand disproportionately the map size (see Figure A.3). It makes the **Reads2Map** workflows an interesting tool just for a first selection of the method to build the map. The resulted map built would require extra manipulation to achieve good quality. The presence of outliers also makes the total size of the maps not a good parameter for our comparison. Therefore, we used the average Euclidean distances to compare the estimated distances between the markers with the real distances (Figure 3.10).

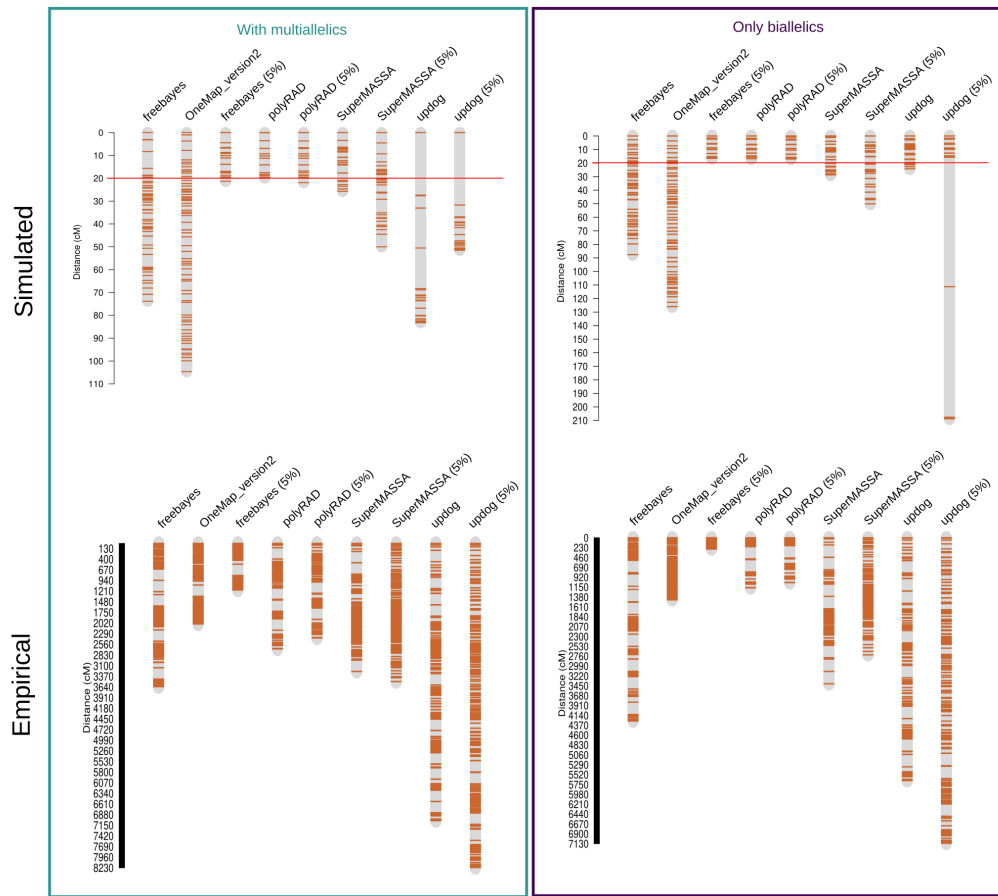
The haplotype-based multiallelic markers showed to be advantageous for almost all scenarios, with exception of **freebayes** with counts from BAM and for maps built with **updog** genotype caller. These results for simulated data differ from the empirical data results which showed an increase in the map size in most of the scenarios. This can be the consequence of outliers markers more present in the empirical data than the simulated. Because multiallelic markers are more informative, errors in these types of markers can have a higher effect in maps than biallelic markers. Further investigations about multiallelic markers in empirical data were made in the next session.

By the results of the Euclidean distance ( $D$ ), we can also observe smaller distances in the scenario with **freebayes** as SNP caller and counts source from VCF and considering any of the genotype callers, except the **updog** for depth 10, **freebayes** and **GUSMap**. This scenario also presented the most reliable genotype probabilities (Figure 3.4). The **freebayes** as genotype caller for this scenario did not present reliable genotype probabilities but had few wrong genotypes. Thus, when the error rate was fixed as 0.05 for all genotypes in this scenario, the variance in  $D$  is not observed.



**Figure 3.10.** Average Euclidean distance ( $D$ ) comparing estimated and real distances between markers. GUSMap and OneMap 3.0 maps presented high variation between the simulations repetitions. Some of its Euclidean distances were higher than 7000 cM, which distorted this figure graphic scale. The comparison including these maps is in Figure A.27.

Figure 3.11 shows the linkage maps built for one of the families simulated with depth 10 and evaluated with **freebayes** SNP caller, counts from VCF and with multiallelic markers scenario. We can see that several of the simulated maps built were reliable for this scenario, but some had more outlier markers than others and would require more edition. The figure also shows the same scenario in the empirical dataset. It presented some reliable maps built by **polyRAD** and **freebayes**. We selected the one with few outliers: the one built with **freebayes** as genotype caller using a global error rate of 0.05. Then, we download the sequence of the empirical data from the **Reads2Map** app and made editions removing outlier markers. The obtained map for the linkage group of chromosome 10 has 557 markers and 217.61 cM total size.



**Figure 3.11.** Maps built using approach with freebayes as SNP caller and counts source from VCF. These linkage groups were built by **SimulatedReads2Map** and **EmpiricalReads2Map** workflows running them with and without multiallelic markers. They need additional edition to be reliable, but the workflow approach allows the selection of best approach. GUSMap maps presented large map size which distorted this figures scale, it is available in the Reads2Map shiny app.

### 3.3.4 Haplotype-based markers and ordering algorithms

Freebayes and GATK optimize their methods with haplotype reconstruction, then we can observe phased genotypes in VCF outputted files. Both also output multiallelic SNPs, which can be confounded with MNPs. It is important to differentiate the multiallelic SNPs from MNPs phased genotypes. The multiallelic SNPs would require that multiple polymorphisms exist in a single base, which is unlikely to happen because of the SNP biallelic nature, as discussed before. Then we interpreted the multiallelic SNPs in the dataset as artifacts and they were filtered out.

Freebayes represents the haplotype-based markers in a single line of the VCF, by adding different alternative alleles in its ALT field and varying the genotype numbers according to the respective allele (e.g. 'ab' as '1/2', 'ad' as '1/4') (Figure A.11). This format, despite being less informative about the SNPs individual positions, is fastly converted by new the algorithm in *onemap\_read\_vcfR* OneMap function. The phased genotypes in format split by rows like the one outputted by GATK needs extra manipulation to be represented as multiallelic markers in OneMap, but the algorithm to do that is also already implemented in the function *onemap\_reads\_vcfR*. Note that not all phased markers can be converted to multiallelic. It would depend on the combination of polymorphism present in the haplotype block. Phased markers with no increase in the allelic information were kept as non-informative or biallelic SNP in the VCF file.

In our simulations, we allowed the possibility of the SNP callers to identify multiallelic markers by considering the positions of the markers close to each other in the input reference VCF. The genotypes at each position are simulated randomly, then we did not control which markers could be converted to haplotype-based markers. As observed in empirical data, **freebayes** was able to identify more multiallelic markers than **GATK** in our simulations (Figure 3.2).

The limitation of **GATK** to provide haplotype-based markers is because it builds the haplotypes individually by sample. This approach is a robust and computational intensive method evolving de-Bruijn-like graphs for local-reassembly and pair-HMM to calculate the likelihood for each read originate of each haplotype. The computational demand to apply the method makes it not possible to perform the haplotype and genotype call on a populational scale in the same analysis. To combine sample information, **GATK** combines only the genotype likelihoods stored from all samples (POPLIN *ET AL.*, 2017). Thus, it offers an accurate and scalable method of calling variants as SNPs and indels, but not haplotype-based markers. The developers suggest imputation algorithms as **Beagle** (BROWNING and BROWNING, 2007) to perform the phasing.

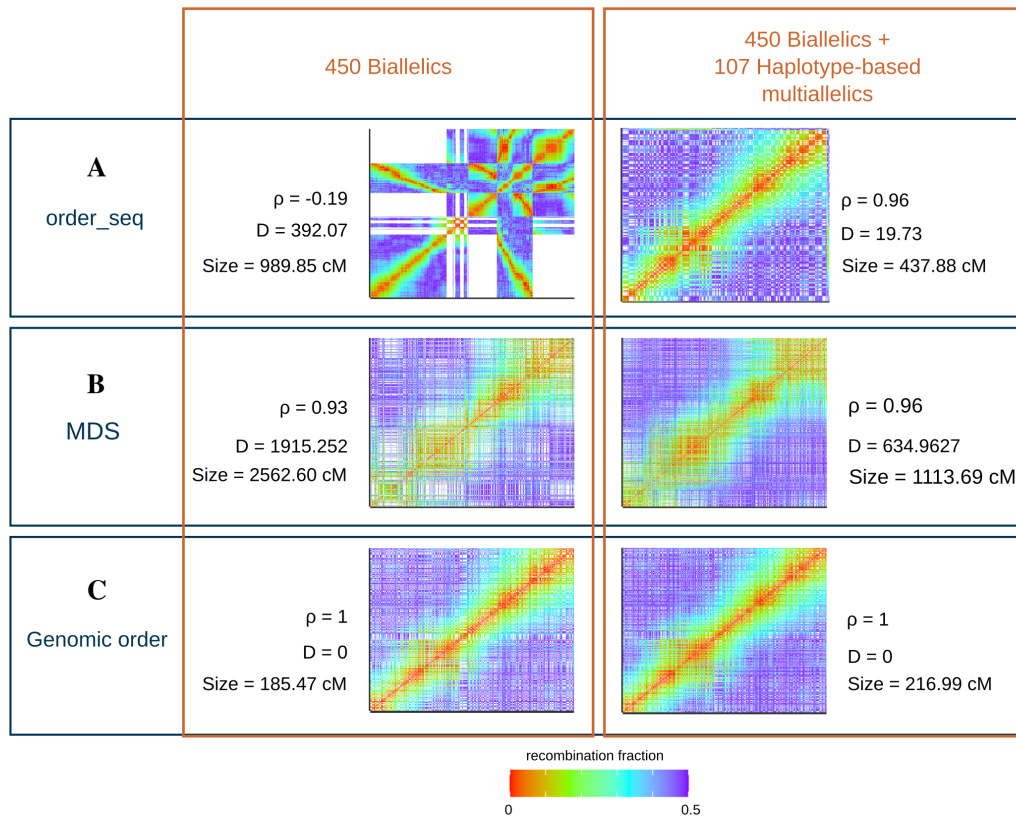
On the other hand, **freebayes** uses a Bayesian approach specifically to perform physical phasing calling of multiallelic loci in sets of individuals (GARRISON and MARTH, 2012). It can identify more multiallelic, however, brings more uncertainties about marker positions.

If the ordering of markers is already known, as was our case with *P. tremula* dataset, the multiallelic markers provides phase information to the HMM to estimate the genetic distances. The estimated linkage group of the *P. tremula* chromosome 10 contains 107 haplotype-based multiallelic markers of type A.2. Theoretically, the presence of multiallelic markers reduces the time needed for the analysis because the HMM does not need to test every possible phase. However, we observed small differences in HMM processing time between maps with only biallelic and maps with both biallelic and haplotype-based multiallelic markers (Figure A.26 and A.25). The observed differences can be related to the amount of markers in each linkage group and not to the marker type (Figure A.24). Also, because of the higher informativeness, the presence of multiallelic markers provides more accurate genetic distance estimations (Figure 3.12 C). The presence of multiallelic markers is particularly interesting when the order of the markers is not known.

Algorithms that use two-points recombination fractions estimations to order only biallelic markers have to deal with missing linkage information between markers type D1 and D2. These markers can only be related to each other in the presence of more informative markers, as B3.7 or the multiallelic. Figure 3.12 B shows the impact of including the multiallelic markers in the two-points-based MDS algorithm (PREEDY and HACKETT, 2016). The presence of multiallelic markers slightly increases the Pearson correlation and drastically reduces the Euclidean distance between the estimated ordering and the genomic order. The genetic distance in the Figure 3.12 is estimated by **OneMap** HMM approach.

The *order\_seq* algorithm is a strategy developed to apply HMM in the ordering procedure (MARGARIDO *ET AL.*, 2007). It starts estimating the sequencing likelihood for a subset (default of five markers) of the highest informative markers in the dataset, ordered by exhaustive, and after, it adds sequentially all the others testing each possible position in the already established sequence. This strategy used to be very accurate when dealing with few informative markers (as SSRs) but presents a wrong order with only biallelic. Now, with the presence of haplotype-based multiallelic markers, the strategy returns a high-quality order, reproducing almost completely the genomic order and the correct pattern of recombination fraction matrix (Figure 3.12 A).





**Figure 3.12.** Comparison between ordering algorithms performance in *P. tremula* chromosome 10 linkage group with only biallelic markers, and with biallelic and haplotype-based multiallelic markers. The heatmaps represent the recombination fraction matrix between markers positioned at both axes. In well-ordered linkage groups, we expect a gradient from hot colors in the diagonal (adjacent markers) to cold color in the upper left and downright corners. The figure also presents the Spearman's rank correlation ( $\rho$ ) and the Euclidean distances (D) between the estimated map and the map build with markers ordered by the genomic positions.

### 3.4 Conclusion

OneMap 3.0 offer to users the possibility to read and consider in the genetic map building the genotype probabilities and haplotype-based multiallelic markers information from the input files (OneMap format or VCF file). The success of genetic map building will be proportional to the quality of the information provided by upstream procedures such as library preparation, SNP and genotype calling, genotype probabilities estimation, and the combination of SNPs into haplotype-based makers. As the upstream procedures for genotyping and identification of haplotype-based multiallelic markers are improved, updates can be easily made in **Reads2Map** workflows, and no modification is required in OneMap 3.0 algorithms, once only the values in the standard VCF file changes.

Only recently, it was developed a software able to simulate RADseq reads, however it still presented a higher number of duplicated sequences and probably more genotyping errors than empirical datasets. Also, it could not reproduce the indels and outlier markers. These differences can limit the possible relation between empirical and simulated results in this study. We can also update **Reads2Map** workflows as RADinitio or other simulation software are improved.

The relation between genotyping errors and genetic map quality was highlighted by our simulation results. We considered high-quality maps those with the expected pattern of heatmap graphics of recombination fraction matrix and number of recombination breakpoints (the consequence of the genetic

distance between markers). We could get high quality genetic maps if they were built with datasets with few genotyping errors and single error rate as observed in the genetic map built with freebayes as SNP and genotype caller with a single error rate of 5%. Also, it is possible to obtain high-quality genetic maps with a higher number of genotyping errors if the provided genotype probabilities correct differentiate wrong and right genotypes, as presented by SuperMASSA software in the scenario with freebayes as SNP caller, or by polyRAD when GATK is the SNP caller and VCF is the read counts source.

The simulations also showed consequences of some technical issues, that could be easily changed to significantly increase the final genetic map built by the approach. The OneMap 3.0 new features implemented were validated using the simulation results. OneMap 3.0 is still not able to consider the parents' genotype probabilities in the HMM, thus, it is important to only add in the map building procedure high-quality parents' genotypes.

The usage of haplotype-based multiallelic markers showed to improve significantly the ordering and map distance estimation. They are also important for the integration of both parents' genetic maps. However, they can also add more genotyping errors to the analysis making editions needed for the outlier markers removal.

## References

- ASHRAFI, H., M. P. KINKADE, H. L. MERK, and M. R. FOOLAD, 2012 Identification of novel quantitative trait loci for increased lycopene content and other fruit quality traits in a tomato recombinant inbred line population. *Molecular Breeding* **30**: 549–567.
- AUWERA, G. A., M. O. CARNEIRO, C. HARTL, R. POPLIN, G. DEL ANGEL, A. LEVY-MOONSHINE, T. JORDAN, K. SHAKIR, D. ROAZEN, J. THIBAUT, E. BANKS, K. V. GARIMELLA, D. ALTSHULER, S. GABRIEL, and M. A. DEPRISTO, 2013 From FastQ Data to High-Confidence Variant Calls: The Genome Analysis Toolkit Best Practices Pipeline. *Current Protocols in Bioinformatics* **43**: 483–492.
- BAUM, E., T. PETRIE, S. G., and W. N., 1970 A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains. *The Annals of Mathematical Statistics* **41**: 164–171.
- BERKSON, J., 1944 Application of the Logistic Function to Bio-Assay. *Journal of the American Statistical Association* **39**: 357–365.
- BEST, K., T. OAKES, J. M. HEATHER, J. SHAW-TAYLOR, and B. CHAIN, 2015 Computational analysis of stochastic heterogeneity in PCR amplification efficiency revealed by single molecule barcoding. *Scientific reports* **5**: 14629.
- BILTON, T. P., M. R. SCHOFIELD, M. A. BLACK, D. CHAGNÉ, P. L. WILCOX, and K. G. DODDS, 2018 Accounting for Errors in Low Coverage High-Throughput Sequencing Data When Constructing Genetic Maps Using Biparental Outcrossed Populations. *Genetics* **209**: 65–76.
- BRADLEY, A. P., 1997 The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition* **30**: 1145–1159.
- BROMAN, K. W., Ś. SEN, and S. SEN, 2009 *A Guide to QTL Mapping with R/qtl*, volume 66 of *Statistics for Biology and Health*. Springer New York, New York, NY.
- BROMAN, K. W., H. WU, S. SEN, and G. A. CHURCHILL, 2003 R/qtl: QTL mapping in experimental crosses. *Bioinformatics* **19**: 889–890.

- BROWNING, S. R. and B. L. BROWNING, 2007 Rapid and accurate haplotype phasing and missing-data inference for whole-genome association studies by use of localized haplotype clustering. *The American Journal of Human Genetics* **81**: 1084–1097.
- CATCHEN, J., P. A. HOHENLOHE, S. BASSHAM, A. AMORES, and W. A. CRESKO, 2013a Stacks: an analysis tool set for population genomics. *Molecular Ecology* **22**: 3124–40.
- CHEN, G., S. MOSIER, C. D. GOCKE, M.-T. LIN, and J. R. ESHLEMAN, 2014 Cytosine deamination is a major cause of baseline noise in next-generation sequencing. *Molecular diagnosis & therapy* **18**: 587–593.
- CHEN, L., P. LIU, T. C. EVANS, and L. M. ETTWILLER, 2017 DNA damage is a pervasive cause of sequencing errors, directly confounding variant identification. *Science* **355**: 752–756.
- CLARK, L. V., A. E. LIPKA, and E. J. SACKS, 2019 polyRAD: Genotype Calling with Uncertainty from Sequencing Data in Polyploids and Diploids. *G3: Genes|Genomes|Genetics* **9**: g3.200913.2018.
- COSTELLO, M., T. J. PUGH, T. J. FENNEL, C. STEWART, L. LICHTENSTEIN, J. C. MELDRIM, J. L. FOSTEL, D. C. FRIEDRICH, D. PERRIN, D. DIONNE, S. KIM, S. B. GABRIEL, E. S. LANDER, S. FISHER, and G. GETZ, 2013 Discovery and characterization of artifactual mutations in deep coverage targeted capture sequencing data due to oxidative DNA damage during sample preparation. *Nucleic acids research* **41**: e67.
- DANECEK, P., A. AUTON, G. ABECASIS, C. A. ALBERS, E. BANKS, M. A. DEPRISTO, R. E. HANDSAKER, G. LUNTER, G. T. MARTH, S. T. SHERRY, G. MCVEAN, and R. DURBIN, 2011a The variant call format and VCFtools. *Bioinformatics* **27**: 2156–2158.
- DEPRISTO, M. A., E. BANKS, R. POPLIN, K. V. GARIMELLA, J. R. MAGUIRE, C. HARTL, A. A. PHILIPPAKIS, G. DEL ANGEL, M. A. RIVAS, M. HANNA, A. MCKENNA, T. J. FENNEL, A. M. KERNYTSKY, A. Y. SIVACHENKO, K. CIBULSKIS, S. B. GABRIEL, D. ALTSHULER, and M. J. DALY, 2011 A framework for variation discovery and genotyping using next-generation DNA sequencing data. *Nature Genetics* **43**: 491–501.
- DER AUWERA, G. A., G. VAN DER AUWERA, and B. D. O’CONNOR, 2020 *Genomics in the Cloud: Using Docker, GATK, and WDL in Terra*. O’Reilly Media, Incorporated.
- FIERST, J. L., 2015 Using linkage maps to correct and scaffold de novo genome assemblies: methods, challenges, and computational tools. *Frontiers in Genetics* **6**: 1–8.
- FRARY, A., T. C. NESBITT, S. GRANDILLO, E. KNAAP, B. CONG, J. LIU, J. MELLER, R. ELBER, K. B. ALPERT, and S. D. TANKSLEY, 2000 fw2.2: a quantitative trait locus key to the evolution of tomato fruit size. *Science (New York, N.Y.)* **289**: 85–88.
- GARRISON, E. and G. MARTH, 2012 Haplotype-based variant detection from short-read sequencing. *ArXiv e-prints* p. 9.
- GAZAFFI, R., R. R. AMADEU, M. MOLLINARI, J. R. B. F. ROSA, C. H. TANIGUTI, G. R. A. MARGARIDO, and A. A. F. GARCIA, 2020 fullsibQTL: an R package for QTL mapping in biparental populations of outcrossing species. *bioRxiv* .
- GERARD, D., L. F. V. FERRÃO, A. A. F. GARCIA, and M. STEPHENS, 2018 Genotyping Polyploids from Messy Sequencing Data. *Genetics* **210**: 789–807.

- GLENN, T. C., 2011 Field guide to next-generation DNA sequencers. *Molecular Ecology Resources* **11**: 759–769.
- HACKETT, C. A. and L. B. BROADFOOT, 2003 Effects of genotyping errors, missing values and segregation distortion in molecular marker data on the construction of linkage maps. *Heredity* **90**: 33–38.
- HALDANE, J. B. S., 1919 The combination of linkage values, and the calculation of distance between linked factors. *Journal of Genetics* **8**: 299–309.
- HYMAN, J. M., 1983 Accurate Monotonicity Preserving Cubic Interpolation. *SIAM Journal on Scientific and Statistical Computing* **4**: 645–654.
- INSTITUTE, B., 2009 Picard. Available online at: <http://broadinstitute.github.io/picard/> .
- KAGALE, S., C. KOH, W. E. CLARKE, V. BOLLINA, I. A. P. PARKIN, and A. G. SHARPE, 2016 Analysis of Genotyping-by-Sequencing (GBS) Data. pp. 269–284.
- LANDER, E. S. and P. GREEN, 1987 Construction of multilocus genetic linkage maps in humans. *Proc. Natl. Acad. Sci USA* **84**: 2363–2367.
- LI, H., 2011 A statistical framework for SNP calling, mutation discovery, association mapping and population genetical parameter estimation from sequencing data. *Bioinformatics* **27**: 2987–2993.
- LI, H., 2013 Aligning sequence reads, clone sequences and assembly contigs with BWA-MEM. *ArXiv* **1303**.
- LI, H., 2020 seqtk: Toolkit for processing sequences in FASTA/Q formats. Available online at: <https://github.com/lh3/seqtk> .
- MA, X., Y. SHAO, L. TIAN, D. A. FLASCH, H. L. MULDER, M. N. EDMONSON, Y. LIU, X. CHEN, S. NEWMAN, J. NAKITANDWE, Y. LI, B. LI, S. SHEN, Z. WANG, S. SHURTLEFF, L. L. ROBISON, S. LEVY, J. EASTON, and J. ZHANG, 2019 Analysis of error profiles in deep next-generation sequencing data. *Genome Biology* **20**: 1–15.
- MACKAY, T. F. C., 2001b The Genetic Architecture of Quantitative Traits. *Annual Review of Genetics* **35**: 303–339.
- MALIEPAARD, C., J. JANSSEN, and J. W. V. OOLJEN, 1997 Linkage analysis in a full-sib family of an outbreeding plant species : overview and consequences for applications. *Genetical Research* **70**: 237–250.
- MARGARIDO, G. R. A., A. P. SOUZA, and A. A. F. GARCIA, 2007 OneMap: software for genetic mapping in outcrossing species. *Hereditas* **144**: 78–9.
- MARSHALL, J., P. DANECEK, J. BONFIELD, H. LI, C. Y. G. GARCIA, V. ZALUNIN, M. LIN, and Y. FARJOUN, 2012 Sequence Alignment/Map Format Specification.
- MARTH, G. T., I. KORF, M. D. YANDELL, R. T. YEH, Z. GU, H. ZAKERI, N. O. STITZIEL, L. D. HILLIER, P. Y. KWOK, and W. R. GISH, 1999 A general approach to single-nucleotide polymorphism discovery. *Nature Genetics* **23**: 452–456.
- MARTIN, M., 2011 Cutadapt removes adapter sequences from high-throughput sequencing reads. *EMBnet.journal* **17**: 10.

- McKENNA, A., M. HANNA, E. BANKS, A. SIVACHENKO, K. CIBULSKIS, A. KERNYTSKY, K. GARIMELLA, D. ALTSHULER, S. GABRIEL, M. DALY, and M. A. DEPRISTO, 2010 The Genome Analysis Toolkit: A MapReduce framework for analyzing next-generation DNA sequencing data. *Genome Research* **20**: 1297–1303.
- MOLLINARI, M. and A. A. F. GARCIA, 2019 Linkage Analysis and Haplotype Phasing in Experimental Autopolyploid Populations with High Ploidy Level Using Hidden Markov Models. *G3: Genes|Genomes|Genetics* **9**: 3297–3314.
- MOLLINARI, M., G. R. A. MARGARIDO, R. VENCovsky, and A. A. F. GARCIA, 2009 Evaluation of algorithms used to order markers on genetic maps. *Heredity* **103**: 494–502.
- PENGELLY, R. J. and A. COLLINS, 2018 Linkage disequilibrium maps to guide contig ordering for genome assembly. *Bioinformatics* pp. 1–5.
- PEREA, C., J. F. DE LA HOZ, D. F. CRUZ, J. D. LOBATON, P. IZQUIERDO, J. C. QUINTERO, B. RAATZ, and J. DUITAMA, 2016 Bioinformatic analysis of genotype by sequencing (GBS) data with NGSEP. *BMC Genomics* **17**: 498.
- POPLIN, R., V. RUANO-RUBIO, M. A. DEPRISTO, T. J. FENNEL, M. O. CARNEIRO, G. A. V. DER AUWERA, D. E. KLING, L. D. GAUTHIER, A. LEVY-MOONSHINE, D. ROAZEN, K. SHAKIR, J. THIBAUT, S. CHANDRAN, C. WHELAN, M. LEK, S. GABRIEL, M. J. DALY, B. NEALE, D. G. MACARTHUR, and E. BANKS, 2017 Scaling accurate genetic variant discovery to tens of thousands of samples. *bioRxiv* p. 201178.
- PREEDY, K. F. and C. A. HACKETT, 2016 A rapid marker ordering approach for high-density genetic linkage maps in experimental autotetraploid populations using multidimensional scaling. *Theoretical and Applied Genetics* .
- R CORE TEAM, 2020 *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.
- RIMMER, A., H. PHAN, I. MATHIESON, Z. IQBAL, S. R. TWIGG, A. O. WILKIE, G. MCVEAN, and G. LUNTER, 2014 Integrating mapping-, assembly- and haplotype-based approaches for calling variants in clinical sequencing applications. *Nature Genetics* **46**: 912–918.
- RIVERA-COLÓN, A. G., N. C. ROCHETTE, and J. M. CATCHEN, 2020 Simulation with RADinitio improves RADseq experimental design and sheds light on sources of missing data. *Molecular Ecology Resources* pp. 1–16.
- ROS-FREIXEDES, R., M. BATTAGIN, M. JOHNSON, G. GORJANC, A. J. MILEHAM, S. D. ROUNSLEY, and J. M. HICKEY, 2018 Impact of index hopping and bias towards the reference allele on accuracy of genotype calls from low-coverage sequencing. *Genetics Selection Evolution* **50**: 1–14.
- SCHIFFTHALER, B., C. BERNHARDSSON, P. K. INGVARSSON, and N. R. STREET, 2017 BatchMap: A parallel implementation of the OneMap R package for fast computation of F1 linkage maps in outcrossing species. *PLoS ONE* **12**: 1–12.
- SCHILBERT, H. M., A. REMPEL, and B. PUCKER, 2020 Comparison of read mapping and variant calling tools for the analysis of plant NGS data. *Plants* **9**: 1–14.
- SCHILLING, M. P., P. G. WOLF, A. M. DUFFY, H. S. RAI, C. A. ROWE, B. A. RICHARDSON, and K. E. MOCK, 2014 Genotyping-by-sequencing for *Populus* population genomics: An assessment of genome sampling patterns and filtering approaches. *PLoS ONE* **9**.

- SERANG, O., M. MOLLINARI, and A. A. F. GARCIA, 2012 Efficient exact maximum a posteriori computation for bayesian SNP genotyping in polyploids. *PLoS ONE* **7**: 1–13.
- SLOANE, D. and S. P. MORGAN, 1996 An Introduction to Categorical Data Analysis. *Annual Review of Sociology* **22**: 351–375.
- SMITH, G. R. and M. NAMBIAR, 2020 New Solutions to Old Problems: Molecular Mechanisms of Meiotic Crossover Control. *Trends in Genetics* **36**: 337–346.
- SPEARMAN, C., 1902 The Proof and Measurement of Association between Two Things. *The American Journal of Psychology* **15**: 72–101.
- TONG, C., D. YAO, H. WU, Y. CHEN, W. YANG, and W. ZHAO, 2020 High-Quality SNP Linkage Maps Improved QTL Mapping and Genome Assembly in *Populus*. *The Journal of heredity* **111**: 515–530.
- TUSKAN, G. A., S. DIFAZIO, S. JANSSON, J. BOHLMANN, I. GRIGORIEV, U. HELLSTEN, N. PUTNAM, S. RALPH, S. ROMBAUTS, A. SALAMOV, J. SCHEIN, L. STERCK, A. AERTS, R. R. BHALERAO, R. P. BHALERAO, D. BLAUDEZ, W. BOERJAN, A. BRUN, A. BRUNNER, V. BUSOV, M. CAMPBELL, J. CARLSON, M. CHALOT, J. CHAPMAN, G.-L. CHEN, D. COOPER, P. M. COUTINHO, J. COUTURIER, S. COVERT, Q. CRONK, R. CUNNINGHAM, J. DAVIS, S. DEGROEVE, A. DEJARDIN, C. DEPAMPHILIS, J. DETTER, B. DIRKS, I. DUBCHAK, S. DUPLESSIS, J. EHLTING, B. ELLIS, K. GENDLER, D. GOODSTEIN, M. GRIBSKOV, J. GRIMWOOD, A. GROOVER, L. GUNTER, B. HAMBERGER, B. HEINZE, Y. HELARIUTTA, B. HENRISSAT, D. HOLLIGAN, R. HOLT, W. HUANG, N. ISLAM-FARIDI, S. JONES, M. JONES-RHOADES, R. JORGENSEN, C. JOSHI, J. KANGASJARVI, J. KARLSSON, C. KELLEHER, R. KIRKPATRICK, M. KIRST, A. KOHLER, U. KALLURI, F. LARIMER, J. LEEBENS-MACK, J.-C. LEPLE, P. LOCASCIO, Y. LOU, S. LUCAS, F. MARTIN, B. MONTANINI, C. NAPOLI, D. R. NELSON, C. NELSON, K. NIEMINEN, O. NILSSON, V. PEREDA, G. PETER, R. PHILIPPE, G. PILATE, A. POLIAKOV, J. RAZUMOVSKAYA, P. RICHARDSON, C. RINALDI, K. RITLAND, P. ROUZE, D. RYABOY, J. SCHMUTZ, J. SCHRADER, B. SEGERMAN, H. SHIN, A. SIDDIQUI, F. STERKY, A. TERRY, C.-J. TSAI, E. UBERBACHER, P. UNNEBERG, J. VAHALA, K. WALL, S. WESSLER, G. YANG, T. YIN, C. DOUGLAS, M. MARRA, G. SANDBERG, Y. VAN DE PEER, and D. ROKHSAR, 2006 The Genome of Black Cottonwood, *Populus trichocarpa*. *Science* **313**: 1596–1604.
- VAN DE GEIJN, B., G. MCVICKER, Y. GILAD, and J. K. PRITCHARD, 2015 WASP: Allele-specific software for robust molecular quantitative trait locus discovery. *Nature Methods* **12**: 1061–1063.
- VOORRIPS, R. E. and C. A. MALIEPAARD, 2012 The simulation of meiosis in diploid and tetraploid organisms using various genetic models. *BMC Bioinformatics* **13**: 248.
- VOSS, K., J. GENTRY, and G. V. D. AUWERA, 2017 Full-stack genomics pipelining with GATK4+WDL+ Cromwell [version 1; not peer reviewed]. *F1000Research* p. 4.
- WALLACE, J. G., E. RODGERS-MELNICK, and E. S. BUCKLER, 2018 On the Road to Breeding 4.0: Unraveling the Good, the Bad, and the Boring of Crop Quantitative Genomics. *Annual Review of Genetics* **52**: 421–444.
- WU, R., C.-X. MA, S. S. WU, and Z.-B. ZENG, 2002c Linkage mapping of sex-specific differences. *Genetical research* **79**: 85–96.
- WU, R., C.-X. C.-X. MA, I. PAINTER, and Z.-B. Z.-B. ZENG, 2002d Simultaneous maximum likelihood estimation of linkage and linkage phases in outcrossing species. *Theoretical population biology* **61**: 349–363.

- YAO, Z., F. M. YOU, A. N'DIAYE, R. E. KNOX, C. MCCARTNEY, C. W. HIEBERT, C. POZNIAK, and W. XU, 2020 Evaluation of variant calling tools for large plant genome re-sequencing. *BMC Bioinformatics* **21**: 360.
- ZHIGUNOV, A. V., P. S. ULIANICH, M. V. LEBEDEVA, P. L. CHANG, S. V. NUZHDIN, and E. K. POTOKINA, 2017 Development of F1 hybrid population and the high-density linkage map for European aspen (*Populus tremula* L.) using RADseq technology. *BMC Plant Biology* **17**.
- ZSÖGÖN, A., T. ČERMÁK, E. R. NAVES, M. M. NOTINI, K. H. EDEL, S. WEINL, L. FRESCHI, D. F. VOYTAS, J. KUDLA, and L. E. P. PERES, 2018 De novo domestication of wild tomato using genome editing. *Nature Biotechnology* **36**: 1211–1216.

## 4 FINAL CONSIDERATIONS

Here we updated the **OneMap** package and presented user-friendly and reproducible workflows to build linkage maps from read sequencing. The procedure in the **Reads2Map** workflows starts with the filtering of the reads, then the alignment, SNP and genotype calling, and the map building. Several software are considered. The shiny app **Reads2MapApp** helps users to select the software and parameter combinations that result in the best linkage maps for their specific dataset.

**OneMap** 3.0 have updates to improve the speed and quality of the built maps, and also quality diagnostic graphic tools. The **OneMap** 3.0 major modifications were the implementation of variable genotype probabilities in the HMM to estimate the genetic distances and the possibility of include haplotype-based multiallelic markers generated from sequencing technologies. All **OneMap** 3.0 updates were validated by application in empirical and simulated datasets in **Reads2Map** workflows.

Using genotype probabilities from different software in the HMM for estimating the genetic distances are efficient depending on the dataset characteristics and upstream methods applied. We could not make a single suggestion for all possible dataset profiles, but with the workflows **Reads2Map**, users can try easily all possibilities and select the best for each dataset. The **Reads2Map** flexibility and an organized structure make easy its adaptation if some of the software considered receives updates or to include other software.

The presence of haplotype-based multiallelic markers improves the map quality and helps the building map procedure. However, it can also bring more errors. It is necessary an edition of the maps built by the workflows to remove the outlier markers, which deviates from the expected pattern of recombination fraction and segregation.

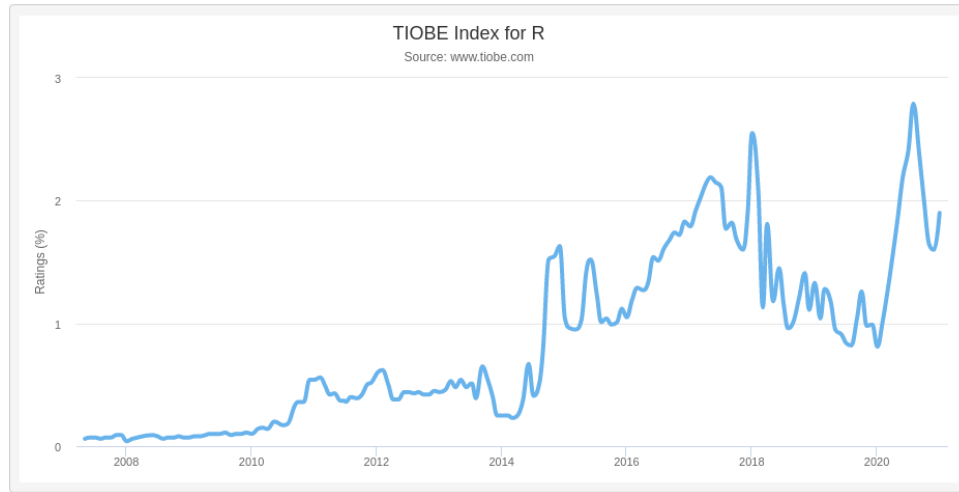
Because the quality of the genetic maps depends on all upstream applied approaches, the workflows **Reads2Map** combined with **OneMap** 3.0 are powerful tools to build linkage maps with markers from sequencing technologies.





## APPENDICES

### Appendix I - R popularity



**Figure A.1.** R language popularity according to January 2021 TIOBE index (THE SOFTWARE QUALITY COMPANY TIOBE, 2021)

### Appendix II - Docker images

The workflows can be run in containers using docker images available in the Docker hub with the following identification. Some of them were created particularly for this work.

**Table A.1.** Identification and versions of Docker hub images used in workflows.

Docker hub image	Version
kfdrc/bwa-picard	latest-dev
taniguti/gatk-picard	1.0
python	3.7
cristaniguti/r-samtools	1.0
cristaniguti/onemap_workflows	1.0
taniguti/gatk-picard	1.0
biocontainers/bcftools	1.3.1
taniguti/stacks	1.0
taniguti/pirs-ddrad-cutadapt	1.0
cristaniguti/radinitio	1.0
cristaniguti/simuscopR	1.0
taniguti/java-in-the-cloud	1.0
taniguti/miniconda-alpine	1.0
cristaniguti/onemap_workflows	1.0
taniguti/freebayes	1.0
lifebitai/bcftools	1.10.2-105-g7cd83b7
cristaniguti/vcftools	1.0
taniguti/vcftools	1.0
kfdrc/cutadapt	latest

### Appendix III - Marker types for outcrossing populations

**Table A.2.** Marker types according to parents genotypes combinations and progeny segregation. The letters “a”, “b”, “c” and “d” represent different alleles and the letter “o” represents null alleles. Adaptated from WU ET AL., (2002).

Marker type	Parents		Progeny	
		Cross	Observed genotypes	Expected segregation
A	1	ab x cd	ac,ad,bc,bd	1:1:1:1
	2	ab x ac	a,ac,ba,bc	1:1:1:1
	3	ab x co	ac,a,bc,b	1:1:1:1
	4	ao x bo	ab,a,b,o	1:1:1:1
B	$B_1$	5 ab x ao	ab,2a,b	1:2:1
	$B_2$	6 ao x ab	ab,2a,b	1:2:1
	$B_3$	7 ab x ab	a,2ab,b	1:2:1
C	8	ao x ao	3a,o	3:1
D	$D_1$	9 ab x cc	ac,bc	1:1
		10 ab x aa	a,ab	1:1
		11 ab x oo	a,b	1:1
		12 bo x aa	ab,a	1:1
	$D_2$	13 ao x oo	a,o	1:1
		14 cc x ab	ac,bc	1:1
		15 aa x ab	a,ab	1:1
		16 oo x ab	a,b	1:1
		17 aa x bo	ab,a	1:1
		18 oo x ao	a,o	1:1

### Appendix IV - Codes to run workflows

**Listing 4.1.** Example code needed to run the workflow in a prompt terminal. File *cromwell\_cache* here is an example of cromwell configuration file. Configurations for HPC or cloud are defined here, including how many jobs can be run parallelized at the same time. File *SimulatedReads2Map.wdl* in the WDL workflow developed to simulated sequencing reads and perform all downstream steps and the linkage map building. The *SimulatedReads2Map.depth20.popsiz200.json* is an example file of inputs specifications, see 4.2 for details.

```

java -jar \
-Dconfig.file=/some/path/to/Reads2Map/.configurations/cromwell_cache.conf \
-jar /some/path/to/cromwell-55.jar run /some/path/to/Reads2Map/SimulatedReads2Map.wdl \
-i /some/path/to/Reads2Map/inputs/SimulatedReads2Map.depth20.popsiz200.json

# Access cromwell server
java -jar \
-Dconfig.file=/some/path/to/Reads2Map/.configurations/cromwell_cache.conf \
-jar /some/path/to/cromwell-55.jar server

```

**Listing 4.2.** Example of JSON input file for **SimulatedReads2Map.wdl** workflow. This template can also be generated automatically using **WOMtools** `inputs` function. If users do not want to change any other software-specific parameter, this file is the only one that needs adaptations of directories paths to run the workflow.

---

```
{
  "SimulatedReads.number_of_families": 5,
  "SimulatedReads.family": {
    "seed": 5050,
    "popsize": 200,
    "ploidy": 2,
    "cross": "F1"
  },
  "SimulatedReads.references": {
    "ref_fasta": "/some/path/to/Chr10.populus.fa",
    "ref_dict": "/some/path/to/Chr10.populus.dict",
    "ref_ann": "/some/path/to/Chr10.populus.fa.ann",
    "ref_sa": "/some/path/to/Chr10.populus.fa.sa",
    "ref_amb": "/some/path/to/Chr10.populus.fa.amb",
    "ref_bwt": "/some/path/to/Chr10.populus.fa.bwt",
    "ref_fasta_index": "/some/path/to/Chr10.populus.fa.fai",
    "ref_pac": "/some/path/to/Chr10.populus.fa.pac"
  },
  "SimulatedReads.global_seed": 6000,
  "SimulatedReads.sequencing": {
    "emp_vcf": "/some/path/to/reference.vcf",
    "enzyme1": "HindIII",
    "enzyme2": "NlaIII",
    "library_type": "ddRAD",
    "chromosome": "Chr10",
    "pcr_cycles": "14",
    "read_length": "150",
    "insert_size": "350",
    "depth": "20",
    "depth_parents": "160",
    "ref_map": "/some/path/to/reference.map.Chr10.csv",
    "multiallelics": "yes",
    "vcf_parent1": "PT_F",
    "vcf_parent2": "PT_M"
  },
  "SimulatedReads.max_cores": "20",
  "SimulatedReads.ReadSimulations.filters": "--maf 0.05 --max-missing 0.75"
}
```

---

## Cromwell Server REST API <sup>30</sup>

[/swagger/cromwell.yml](#)

Describes the REST API provided by a Cromwell server

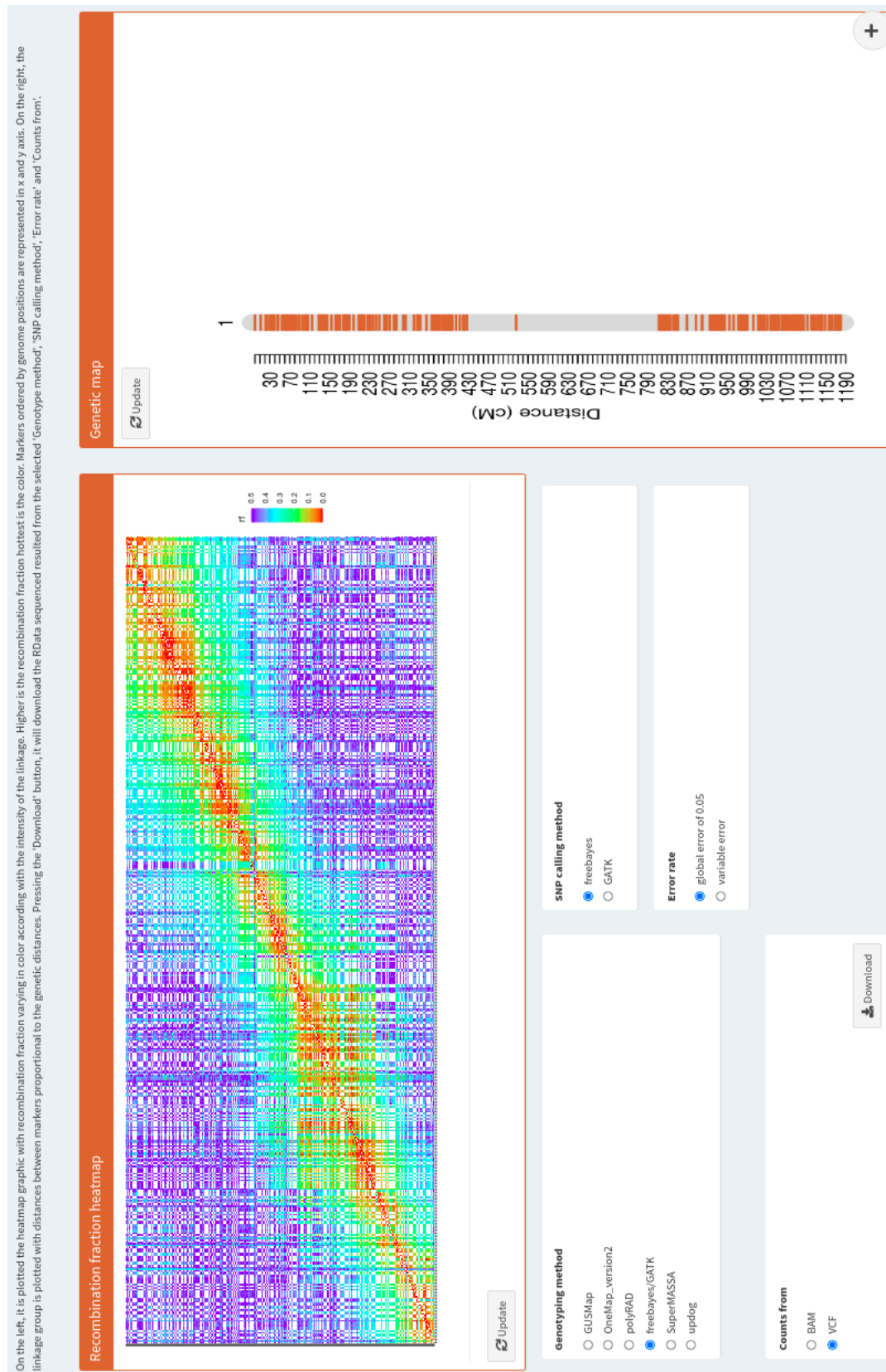
[BSD](#)

### Workflows

POST	/api/workflows/{version}	Submit a workflow for execution
POST	/api/workflows/{version}/batch	Submit a batch of workflows for execution
GET	/api/workflows/{version}/{id}/labels	Retrieves the current labels for a workflow
PATCH	/api/workflows/{version}/{id}/labels	Update labels for a workflow
POST	/api/workflows/{version}/{id}/abort	Abort a running workflow
POST	/api/workflows/{version}/{id}/releaseHold	Switch a workflow from 'On Hold' to 'Submitted' status
GET	/api/workflows/{version}/{id}/status	Retrieves the current state for a workflow
GET	/api/workflows/{version}/{id}/outputs	Get the outputs for a workflow
GET	/api/workflows/{version}/{id}/logs	Get the logs for a workflow
GET	/api/workflows/{version}/query	Get workflows matching some criteria
POST	/api/workflows/{version}/query	Get workflows matching some criteria
GET	/api/workflows/{version}/{id}/timing	Get a visual diagram of a running workflow
GET	/api/workflows/{version}/{id}/metadata	Get workflow and call-level metadata for a specified workflow
GET	/api/workflows/{version}/callcaching/diff	Explain hashing differences for 2 calls
GET	/api/workflows/{version}/backends	List the supported backends

**Figure A.2.** Cromwell server options to submit and consult status of workflow run. This interface can be accessed via web browser once command described in 4.1 is active.

## Appendix V - Linkage map built for *Populus tremula* population



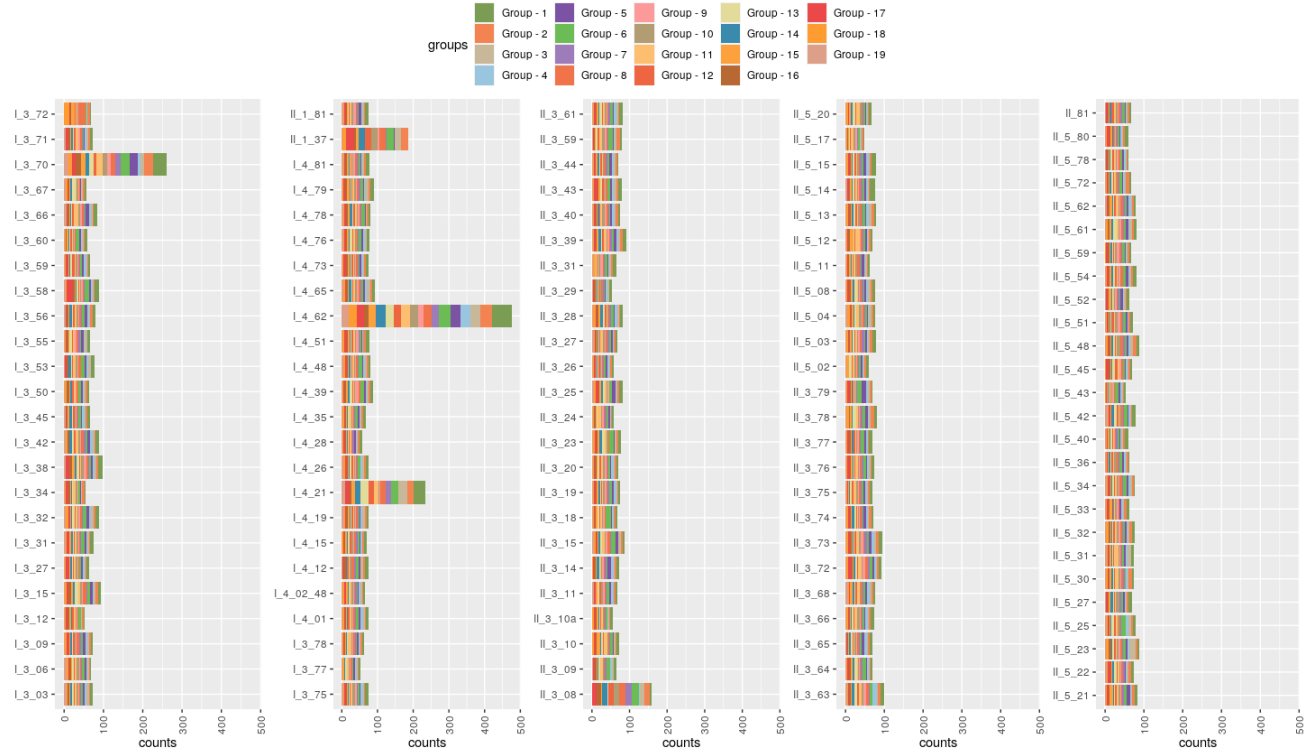
**Figure A.3.** On "Map" screen of Reads2MapApp, user can download the RData file of selected pipeline to further edition in R environment.

**Table A.3.** Overview of built map characteristics.\* Markers located in scaffolds not assembled in the reference genome.

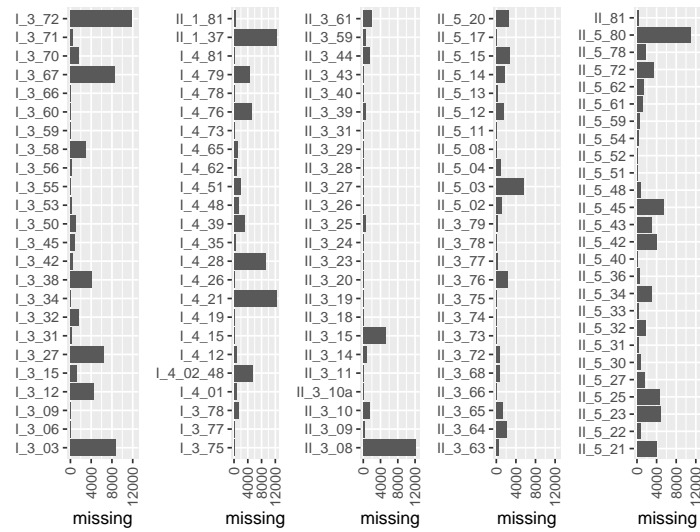
Chromosome	Number of markers	Group size (cM)	Markers in scaffolds*	B3.7	D1.10	D2.15
1	683	332.64	0	160	268	255
2	461	211.81	0	90	203	168
3	390	175.36	0	57	170	163
4	406	179.99	0	75	157	174
5	489	188.23	0	82	215	192
6	520	275.04	0	106	202	212
7	290	112.81	0	57	119	114
8	389	167.54	0	92	154	143
9	276	128.6	0	53	118	105
10	452	173.05	0	82	184	186
11	274	198.42	0	55	103	116
12	258	122.65	0	51	114	93
13	318	166.94	0	63	126	129
14	383	161.38	0	72	146	165
15	314	149.78	0	66	128	120
16	247	129.52	0	42	115	90
17	291	162.14	0	45	134	112
18	250	129.62	0	44	112	94
19	245	134.43	6	41	103	101

## Appendix VI - Pipeline validation with linkage map

In this work, we test the capability of OneMap HMM to identify contaminant individuals in the mapping populations. The haplotypes estimated by the HMM approach highlight the contaminants (I\_3\_08, II\_1\_37, I\_4\_62, I\_4\_21 and I\_3\_70) because they present excess of recombination breakpoints.

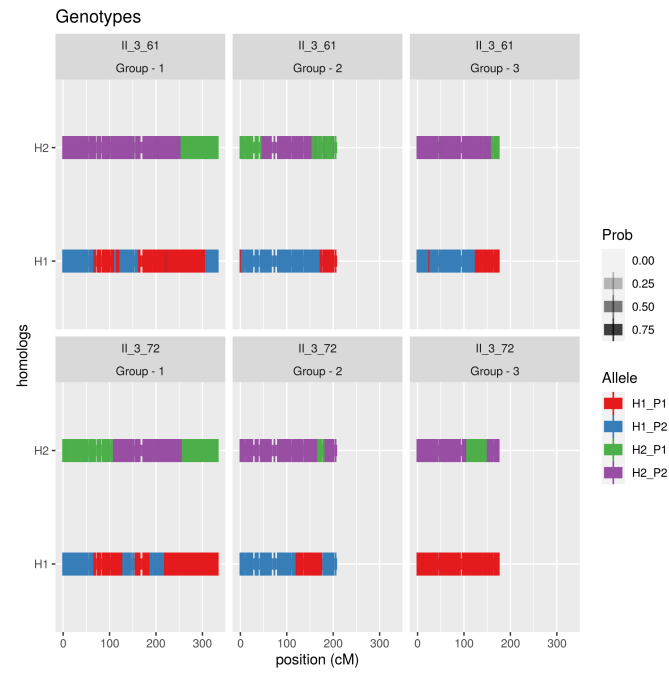


**Figure A.4.** Recombination breakpoints counts in estimated progeny haplotypes of a bi-parental cross of *P. tremula* and seven contaminant individuals included in the progeny.



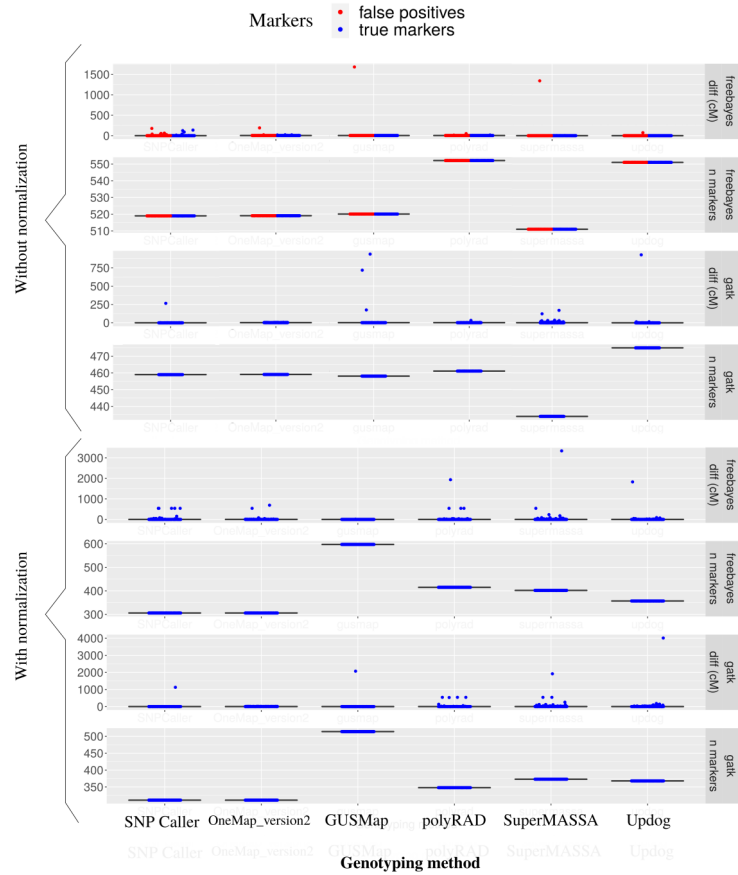
**Figure A.5.** Missing data for each progeny individuals of a bi-parental cross of *P. tremula* including seven contaminants.





**Figure A.6.** Graphical view of the individuals 4 and 5 haplotypes for groups 1, 2, and 3. The graphic is generated based on HMM posterior probabilities for the phased genotypes after the map is built in OneMap.

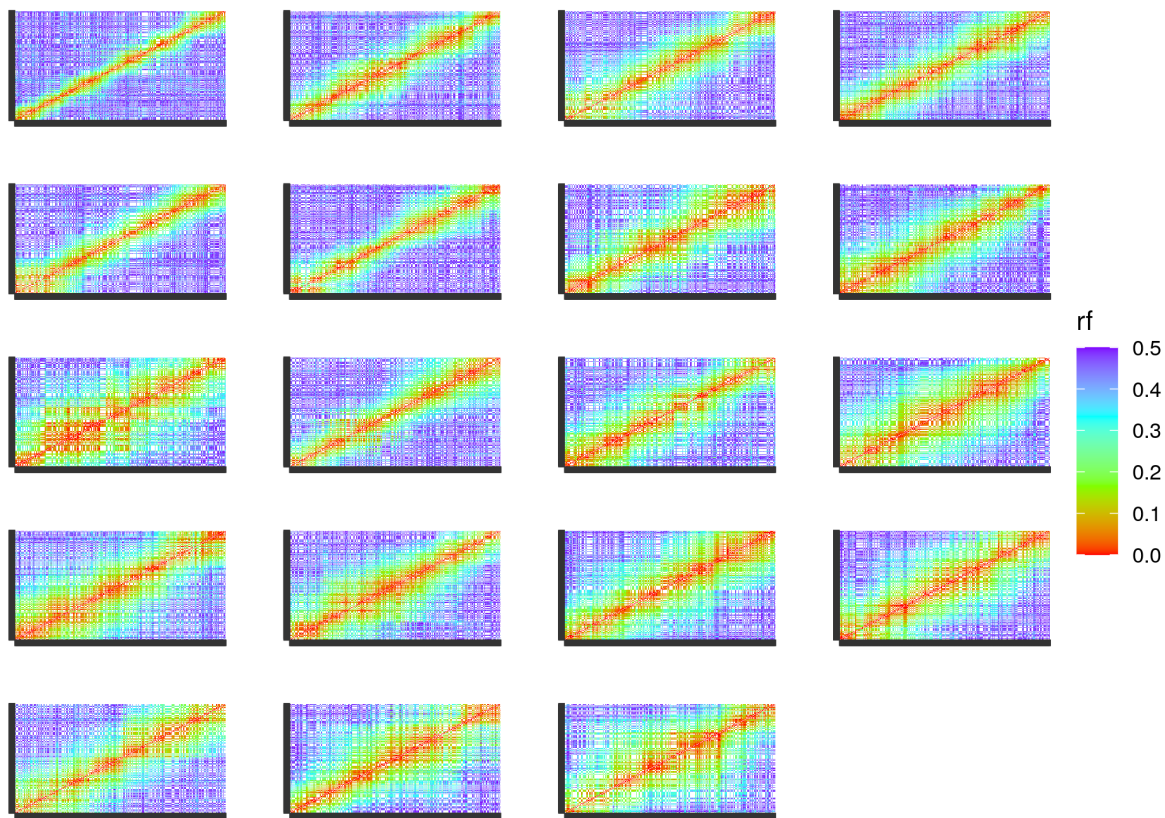
## Appendix VII - Pipeline validation with simulations



**Figure A.7.** Graphical view of one **SimulatedReads2Map** workflow result. The y-axis shows the difference in centimorgans between marker position from real and estimated genetic maps; the x-axis represents each genotyping method included in the workflow. The markers colored in red do not have the same position in the reference genome as its simulated correspondent. They will be considered false positives if the VCF file is not corrected by normalization. Graphics were generated using different workflow seeds. But, independently of the seed, the red dots appear in all maps built using freebayes VCF if it is not normalized.

**Table A.4.** VCF normalization effect in markers codifications. Total of 284336 markers were realigned in freebayes VCF.

VCF	SNPCaller	Number of markers	
		Indels	SNPs
Raw	Freebayes	640284	1759460
	HaplotypeCaller	217577	1812023
With bcftools normalization	Freebayes	612824	1786920
	HaplotypeCaller	217577	1812023



**Figure A.8.** Heatmap graphic measuring the recombination fraction between markers in each group of *P. tremula* genetic map.

## Appendix VIII - Emission function for outcrossing

In the emission step of the HMM procedure, it considers a probability for every possible phased genotype. For outcrossing and  $F_2$  intercross, there are four possibilities. Here we will denote them generically as “AA”, “AB”, “BA”, and “BB”. The probability value that each one will receive depends on which genotype was observed, the marker type (table A.2), and the associated error of the genotype ( $e$ ). As an example, If we have a marker type A (“ab” x “cd”) we can observe four different genotypes, if we observe the genotype “ac” with high confidence ( $e = 0$ ), the phase “AA” would receive the maximum probability (1) and the others would receive 0. In the older version, **OneMap** considered a unique error of  $10^{-5}$  ( $e$ ), which means that if the genotype was called as “ac”, the “AA” phased genotype have a probability of  $1 - 10^{-5}$  and the others have  $10^{-5}/3$ . This way we can include an uncertainty between observed genotype and estimated phased genotype, this characterizes the hidden aspect of the Hidden Markov Model.

For marker type A this could not seem very useful, because the genotype phased is already represented in the observed genotypes. But other marker types do not have a direct relationship between the observed genotype and the estimated phased genotype. For example, if we have a marker type B3.7 and observe the “ab” genotype, the estimated phased genotype can be the “AB” or “BA” and we will consider equal probabilities for them in the emission function. The multipoint aspect of the HMM combined with the expectation-maximization (EM) will change these probabilities and, in the end, we will be able to differentiate between phased genotypes.

In **OneMap** 3.0, users now can provide customized error rates or genotype probabilities to control specific errors in their dataset. The values defined by users will be applied in three different ways in the emission function of the HMM. The variable  $e$  represents the error rate described in equation 3.1 of chapter 2. If users define a single value (*global\_error* argument), it will be the error rate for all observed genotypes. If users provide the genotype errors (*genotype\_errors*), each genotype observed can receive a different value of error rate. If users provide a probability for each genotype (*genotypes\_probs*), the values in each cell of the following table will be replaced by their respective user-provided genotype probability. The tables below are based on R/QTL (BROMAN ET AL., 2003) emission function and describe how the error values are implemented in **OneMap** HMM.

Each observed genotype (columns) receives specific genotype probabilities according to marker type segregation and the error rate. The emission function returns from the iterative steps of the HMM the estimated probability for the phased genotypes. In general, the error rate gives flexibility to the HMM to change the genotypes according to the information from the entire sequence (MOLLINARI AND GARCIA, 2019) or batches with proper size (SCHIFFTHALER ET AL., 2017).

**Table A.5.** Emission function values according to marker types. The error rate is represented by  $e$ , unphased genotypes as “a”, “b”, “c”, “d” and the combination of them. The estimated phased genotypes are represented by “AA”, “AB”, “BA” and “BB”. The “o” represents null alleles. Marker types follow the segregation pattern as described in table A.2

Marker type	A	observed genotypes			
Marker sub-types	A1	ac	ad	bc	bd
	A2	ad	ac	a	a
	A3	bc	ba	bc	b
	A4	bd	bc	b	o
Estimated phased genotypes	AA	$1-e$	$e/3$	$e/3$	$e/3$
	AB	$e/3$	$1-e$	$e/3$	$e/3$
	BA	$e/3$	$e/3$	$1-e$	$e/3$
	BB	$e/3$	$e/3$	$e/3$	$1-e$
OneMap codification		1	2	3	4

**Table A.6.** Continued from table 4

Marker type	B1	observed genotypes		
Marker sub-types	B1.5	a	ab	b
estimated phased genotypes	AA	$1-e$	$e/3$	$e/3$
	AB	$1-e$	$e/3$	$e/3$
	BA	$e$	$1-e$	$e/3$
	BB	$e$	$e/3$	$1-e$
OneMap codification		1	2	3

**Table A.7.** Continued from table 4

Marker type	B2	observed genotypes		
Marker sub-types	B2.6	a	ab	b
estimated phased genotypes	AA	$1-e$	$e/3$	$e/3$
	AB	$e$	$1-e$	$e/3$
	BA	$1-e$	$e/3$	$e/3$
	BB	$e$	$e/3$	$1-e$
OneMap codification		1	2	3

**Table A.8.** Continued from table 4

Marker type	B3	observed genotypes		
Marker sub-types	B3.7	a	ab	b
estimated phased genotypes	AA	$1-e$	$e$	$e/3$
	AB	$e/3$	$1-e$	$e/3$
	BA	$e/3$	$1-e$	$e/3$
	BB	$e/3$	$e$	$1-e$
OneMap codification		1	2	3

**Table A.9.** Continued from table 4

Marker type	C	observed genotypes	
Marker sub-types	C.8	a	o
estimated phased genotypes	AA	$(1-e)/3$	$e/3$
	AB	$(1-e)/3$	$e/3$
	BA	$(1-e)/3$	$e/3$
	BB	$e$	$1-e$
OneMap codification		1	2

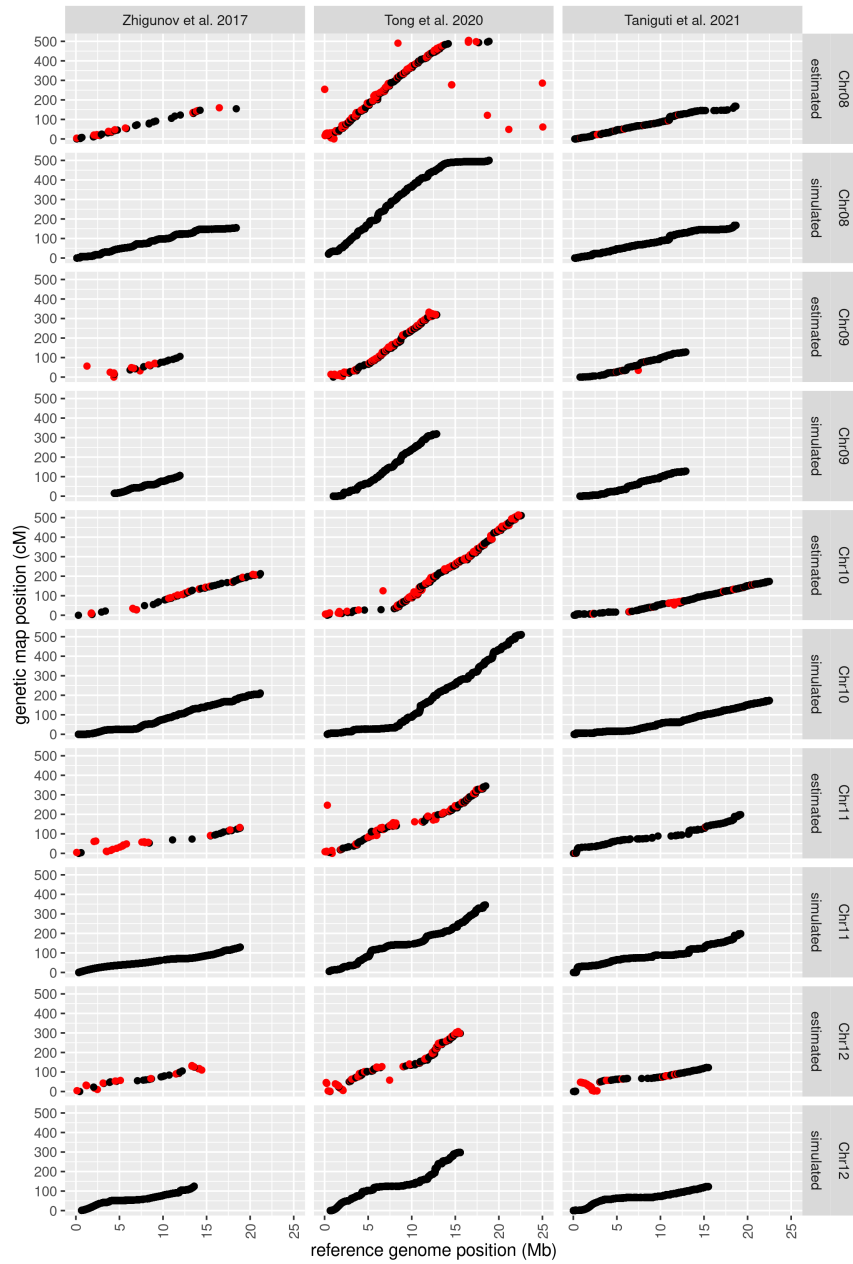
**Table A.10.** Continued from table 4

Marker type	D1	observed genotypes	
Marker sub-types	D1.9	ac	bc
	D1.10	a	ab
	D1.11	a	b
	D1.12	ab	a
	D1.13	a	o
estimated phased genotypes	AA	$1-e$	$e$
	AB	$1-e$	$e$
	BA	$e$	$1-e$
	BB	$e$	$1-e$
OneMap codification		1	2

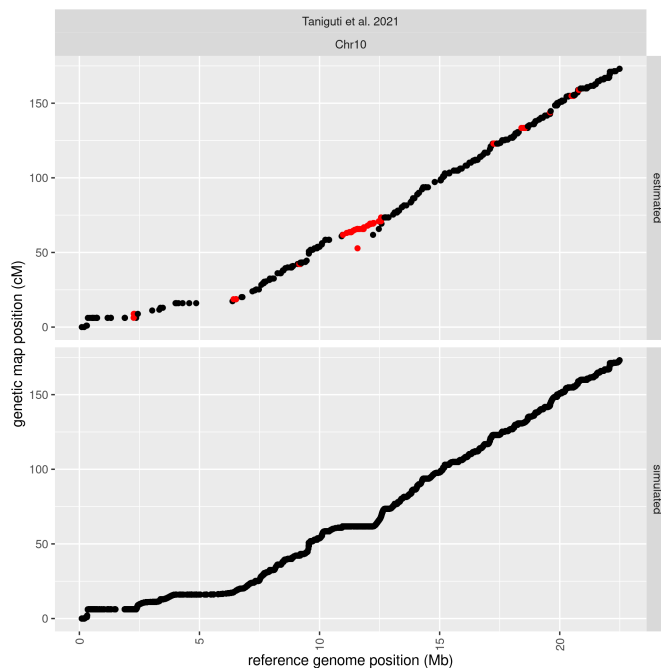
**Table A.11.** Continued from table 4

Marker type	D2	observed genotypes	
Marker sub-types	D2.14	ac	bc
	D2.15	a	ab
	D2.16	a	b
	D2.17	ab	a
	D2.18	a	o
estimated phased genotypes	AA	$1-e$	$e$
	AB	$e$	$1-e$
	BA	$1-e$	$e$
	BB	$e$	$1-e$
OneMap codification		1	2

## Appendix IX - Reference linkage maps

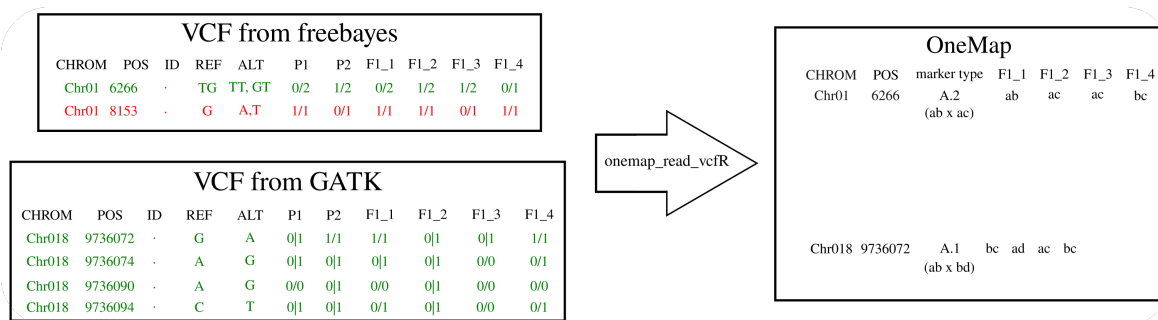


**Figure A.9.** Centimorgan and megabase pair relation of markers position in chromosomes 8 to 12 estimated in previous linkage maps built in chapter 1 and in TONG ET AL. (2020) and ZHIGUNOV ET AL. (2017) for *Populus* gender species. The figure also shows the simulated centimorgan position for markers identified in ZHIGUNOV ET AL. (2017) dataset using GATK. Red dots represent markers in linkage map with inverted genome positions. They were removed from the dataset to train the spline model for the simulations.



**Figure A.10.** Centimorgan and megabase pair relation of markers position in chromosome 10 of reference genetic map built previously in chapter 2 and in simulated with splines. Chromosome 10 sequence was used as input for all sequences simulations here presented. Read dots represents markers in linkage map with inverted genome positions. They were removed from the dataset to train the spline model for the simulations.

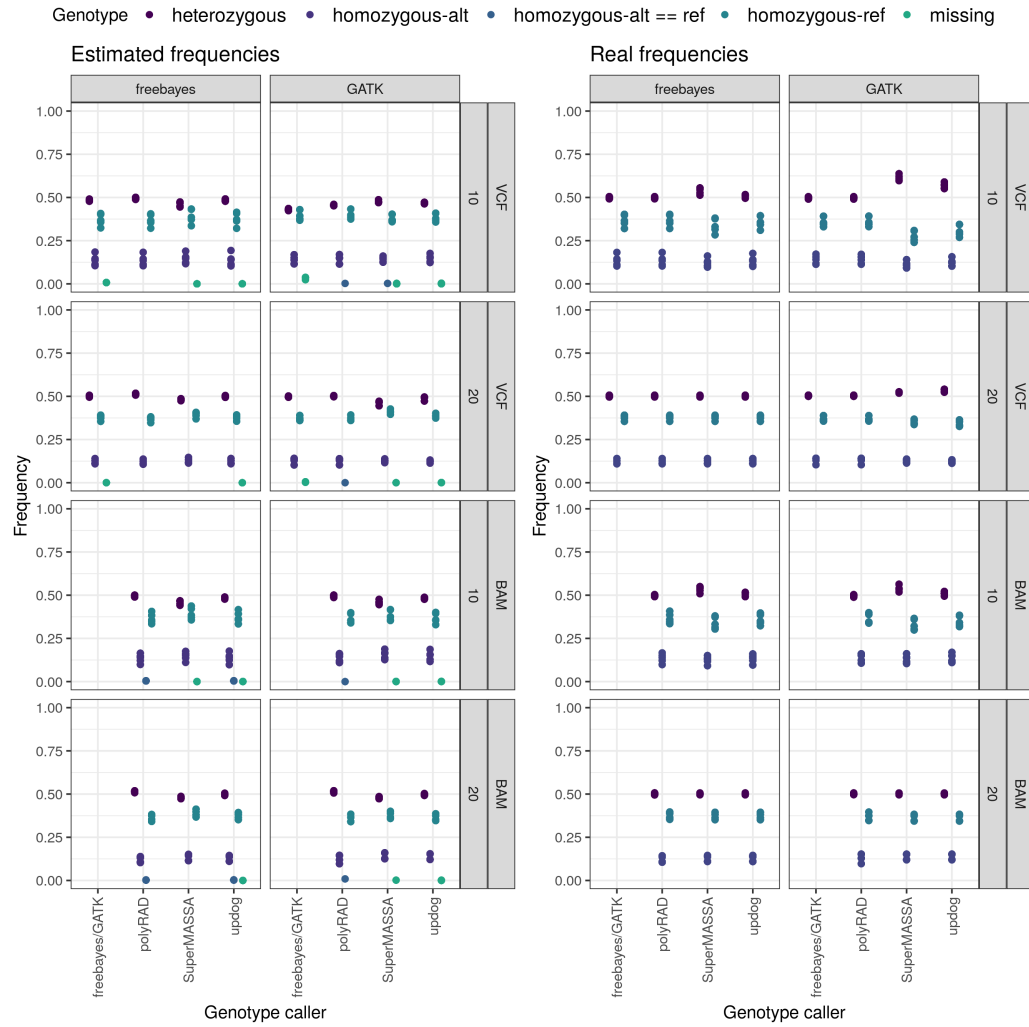
## Appendix X - Haplotype-based multiallelic markers



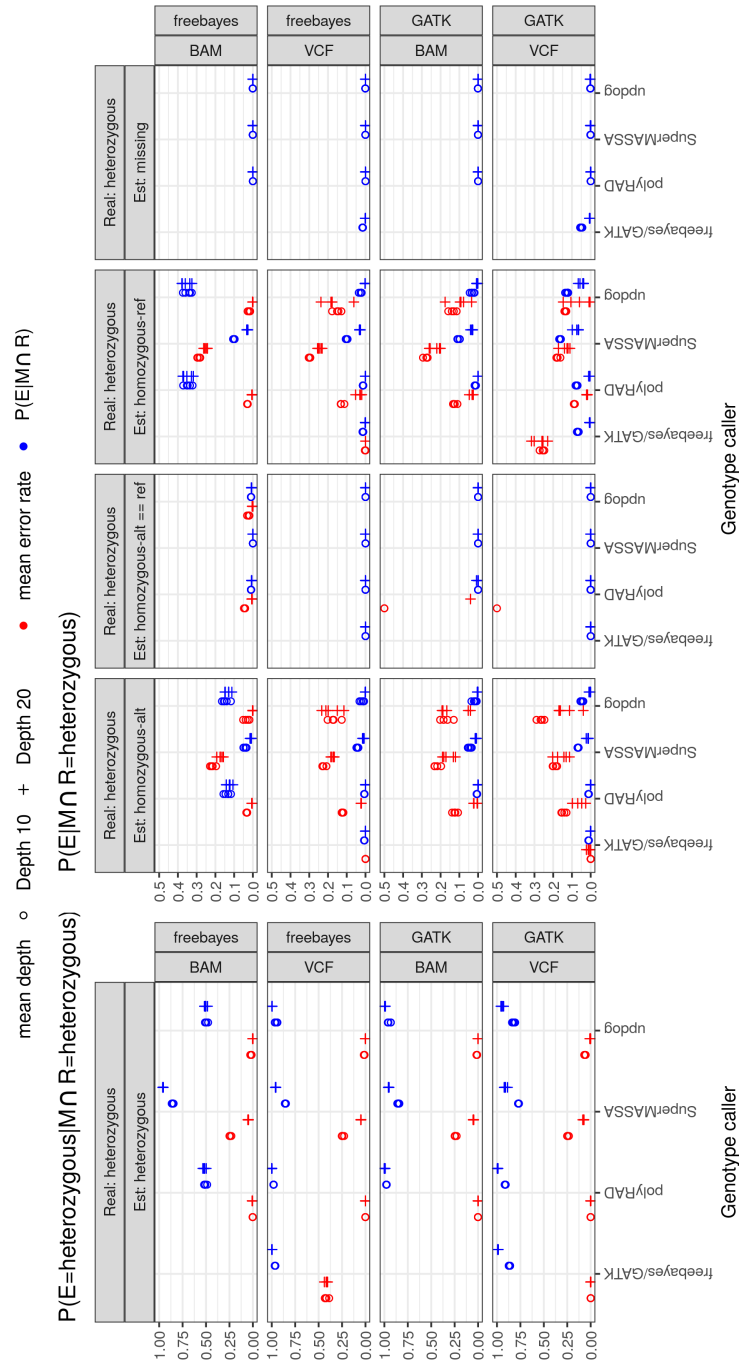
**Figure A.11.** Examples of haplotype-based multiple nucleotide polymorphism markers (MNP) represented in freebayes and GATK outputted VCF and their conversion to OneMap data.



## Appendix XII - Genotypes and error probabilities



**Figure A.12.** Frequencies of real and simulated heterozygous, reference and alternative homozygous genotypes in the experiment. Each dot represents the frequency observed in one of the five repetitions for each studied scenario.



**Figure A.13.** The proportion of heterozygous, reference and alternative homozygous genotypes in progeny correctly and incorrectly estimated and the associated error rate outputted by each genotype caller used. The proportion (blue) is calculated with conditional probability  $P(E = estimated | M = method \cap R = real)$ . The error rate (red) is outputted by each software according to the approach implemented to call the genotypes. We obtained the error rate with  $1 - P(E)$ . The two mean depth of sequencing used to simulate the RADseq reads are represented by the shape of the dots. The genotype call with GATK and freebayes using the read count from BAM files would output the same results as using the VCF files, then, we did not repeat the analysis and there are no results available for these scenarios. Note that the probabilities here are relative to the real genotype frequency in the dataset evaluated. Check their absolute frequency in figure A.12. In some cases the genotype estimated was considered as homozygous despite having same number of reads for alternative and reference alleles, these are here identified as "homozygous-alt==ref".

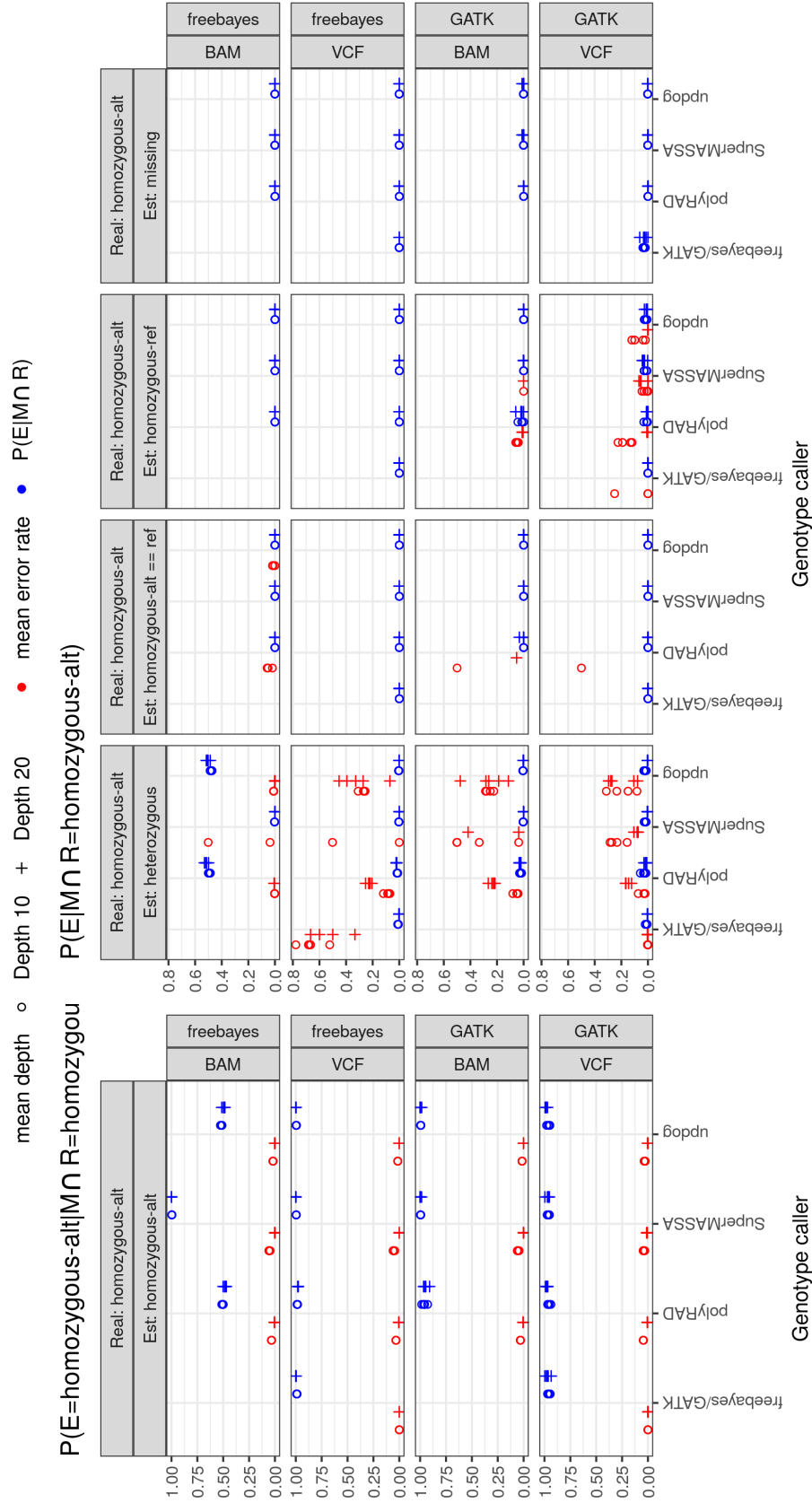
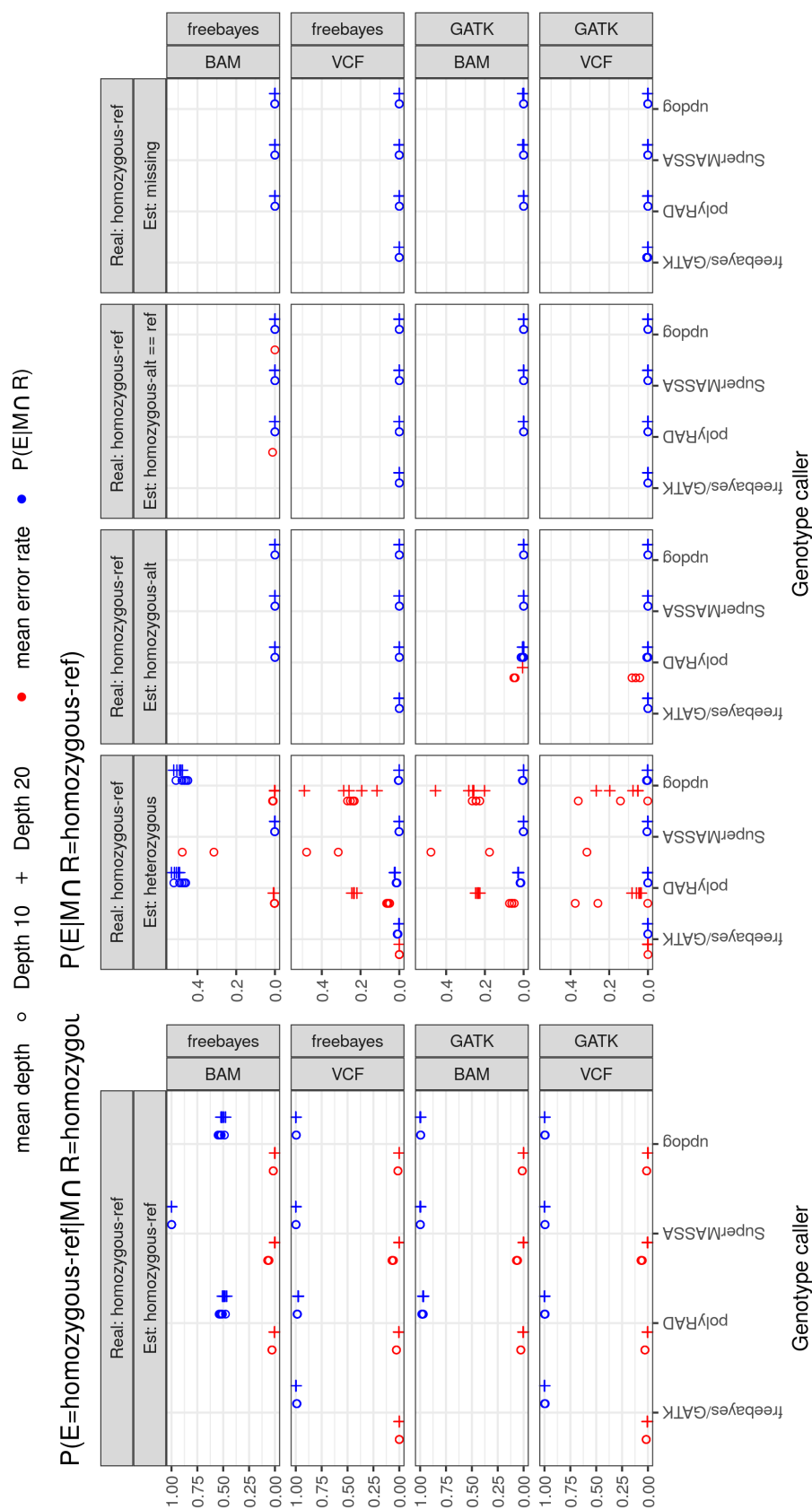
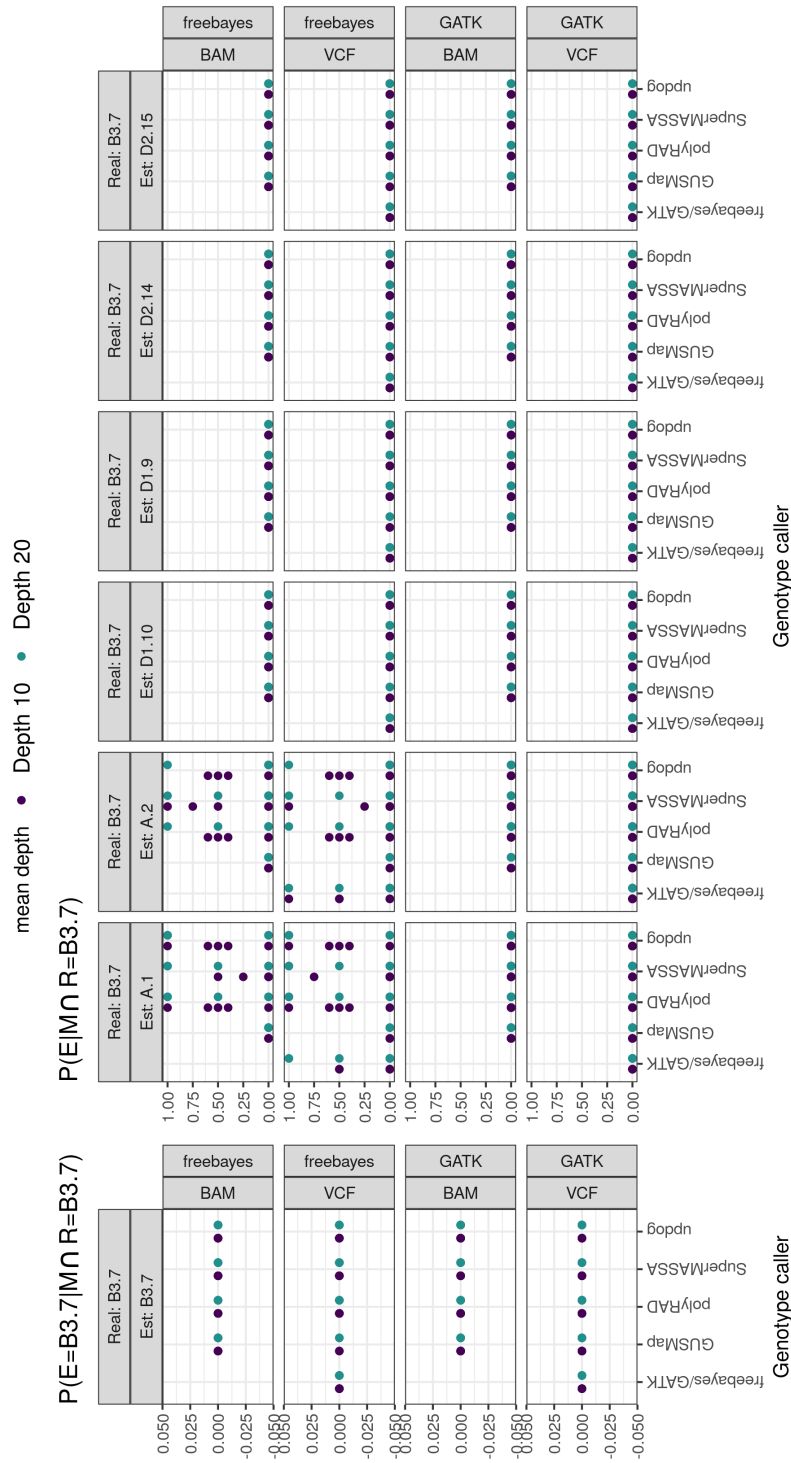


Figure A.14. Figure A.13 continued.





**Figure A.16.** The conditional probability of haplotype-based estimated marker type given the method and the real genotype. Here, the proportion of markers B3.7, D1.10, and D2.15 estimated as A.1, A.2, D1.9 and D1.14 are consequences of haplotype-based SNP calling. The proportions here are relative to the real genotype frequency in the dataset evaluated. Check their absolute frequency in figure 3.5 to better view the proportions of each one in the total dataset. We highlight the change in scale between left and right graphics. The genotype call with GATK and freebayes using the read count from BAM files would output the same results as using the VCF files, then, we did not repeat the analysis and there are no results available for these scenarios.

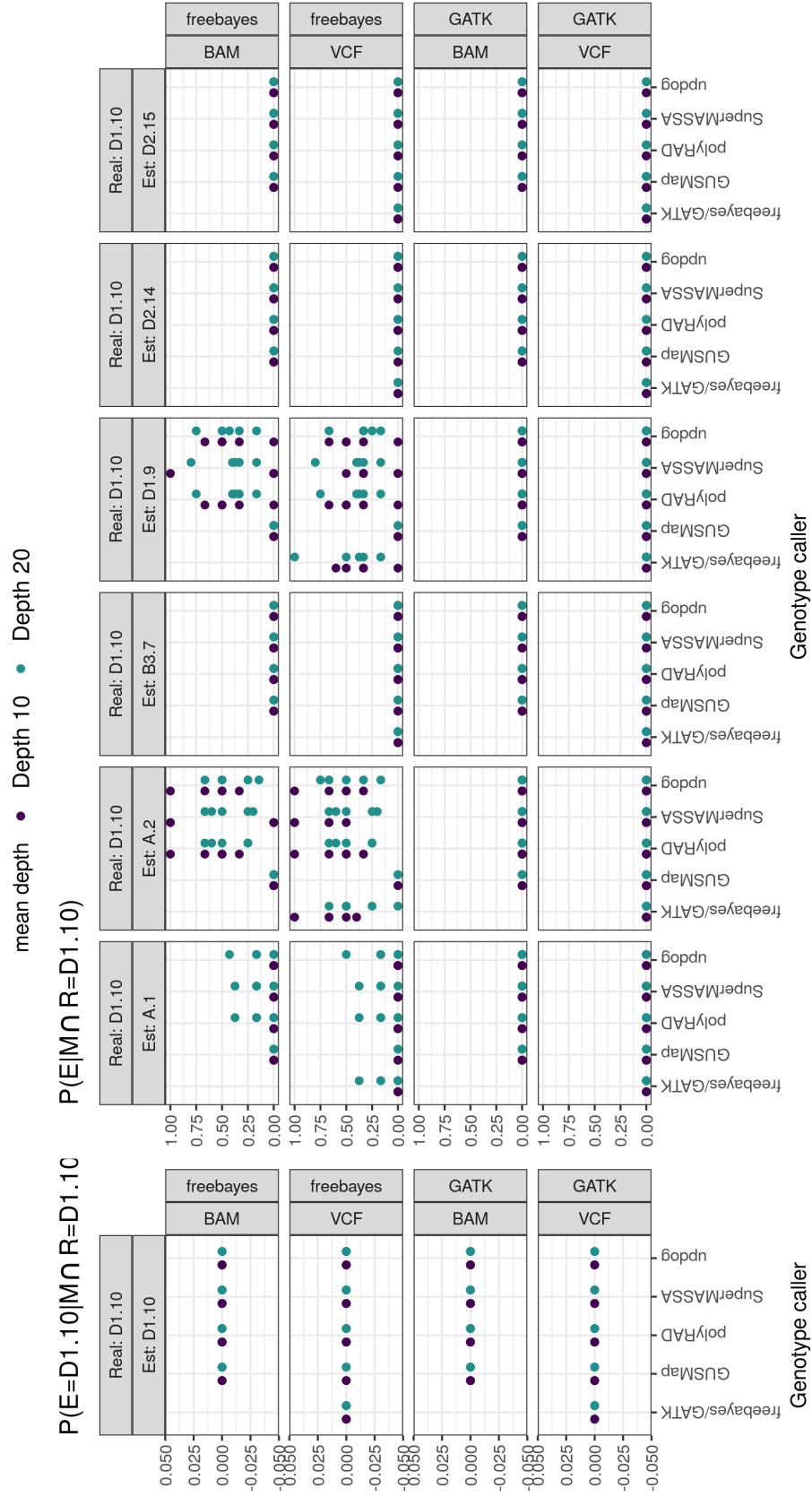


Figure A.17. Figure A.16 continued.

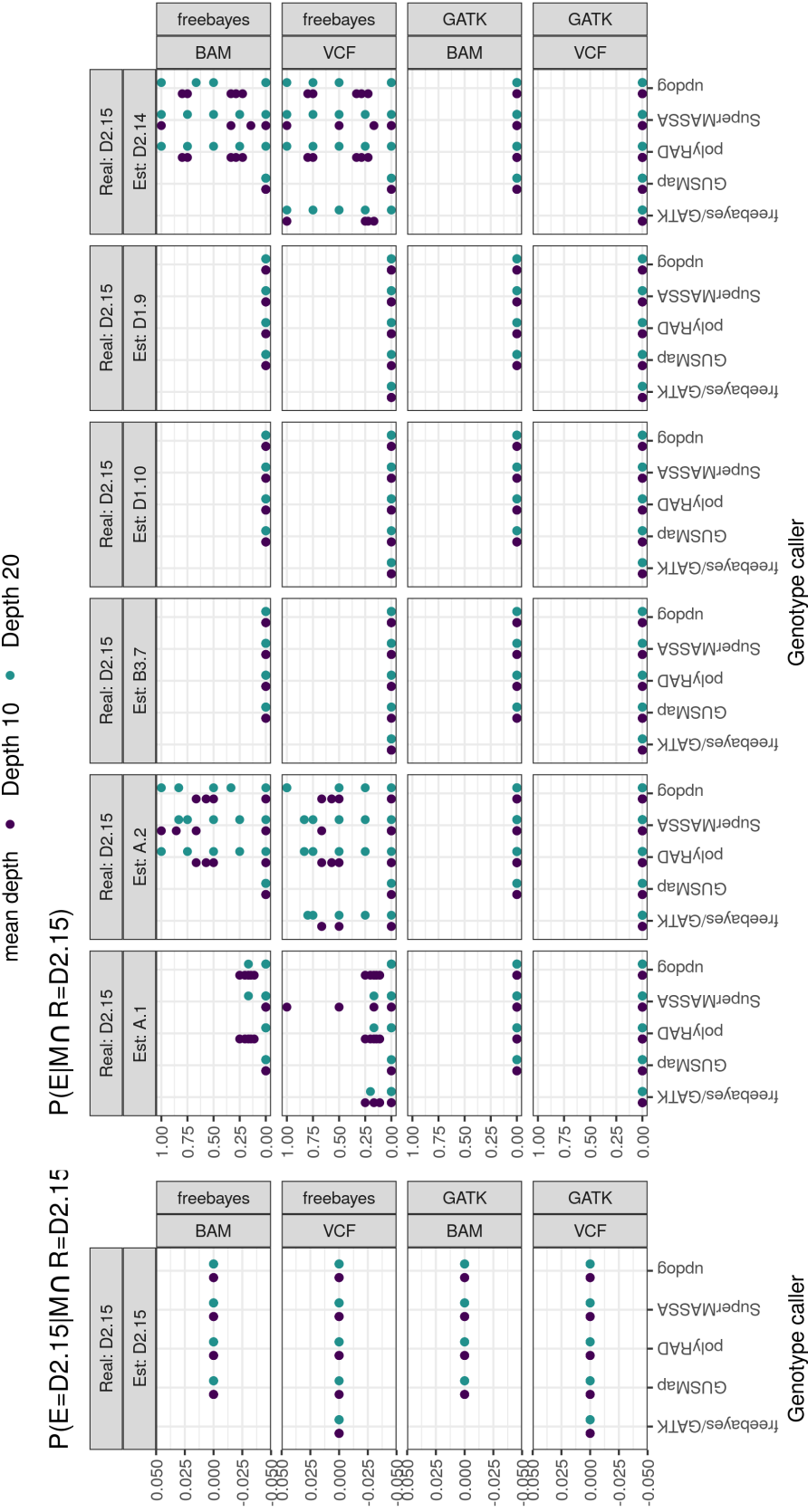


Figure A.18. Figure A.16 continued.

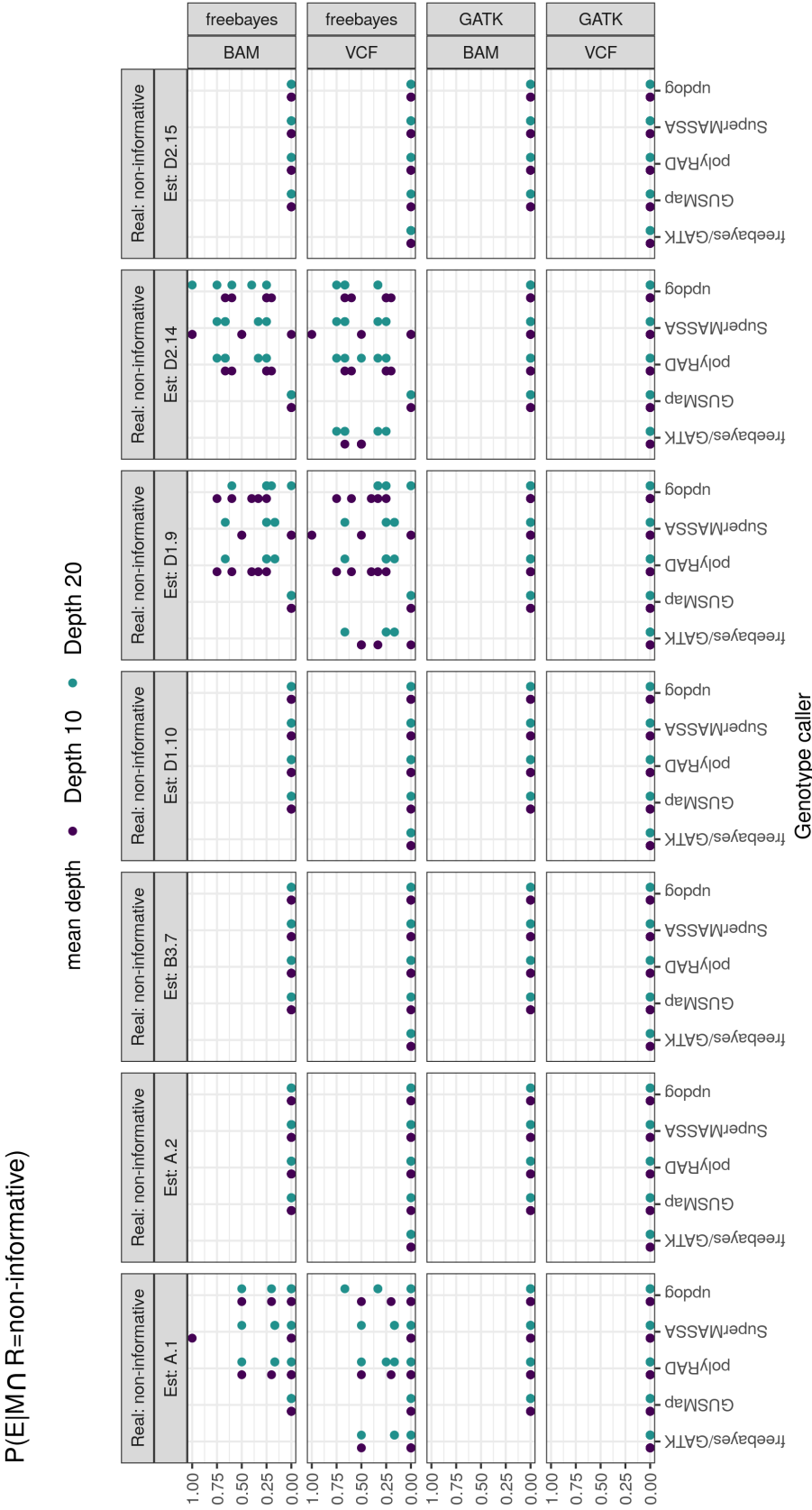
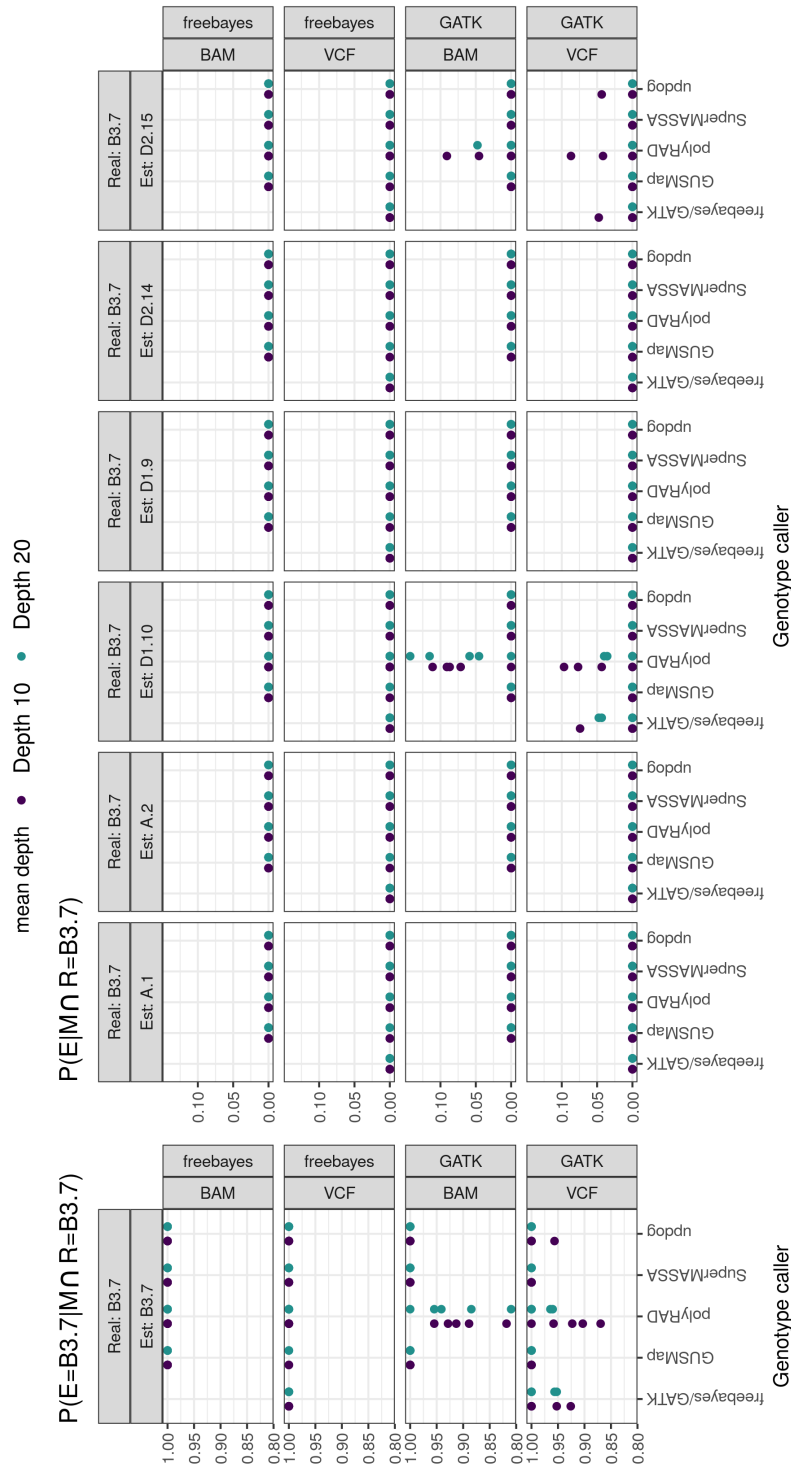


Figure A.19. Figure A.16 continued.





**Figure A.20.** The conditional probability of biallelic estimated markers type given the method and the real genotype. Here, discordances between the real and estimated map can be considered errors in genotype estimation. The proportions here are relative to the real genotype frequency in the dataset evaluated. Check their absolute frequency in figure 3.5 to better view the proportions of each one in the total dataset. We highlight the change in scale between left and right graphics. The genotype call with GATK and freebayes using the read count from BAM files would output the same results as using the VCF files, then, we did not repeat the analysis and there are no results available for these scenarios.

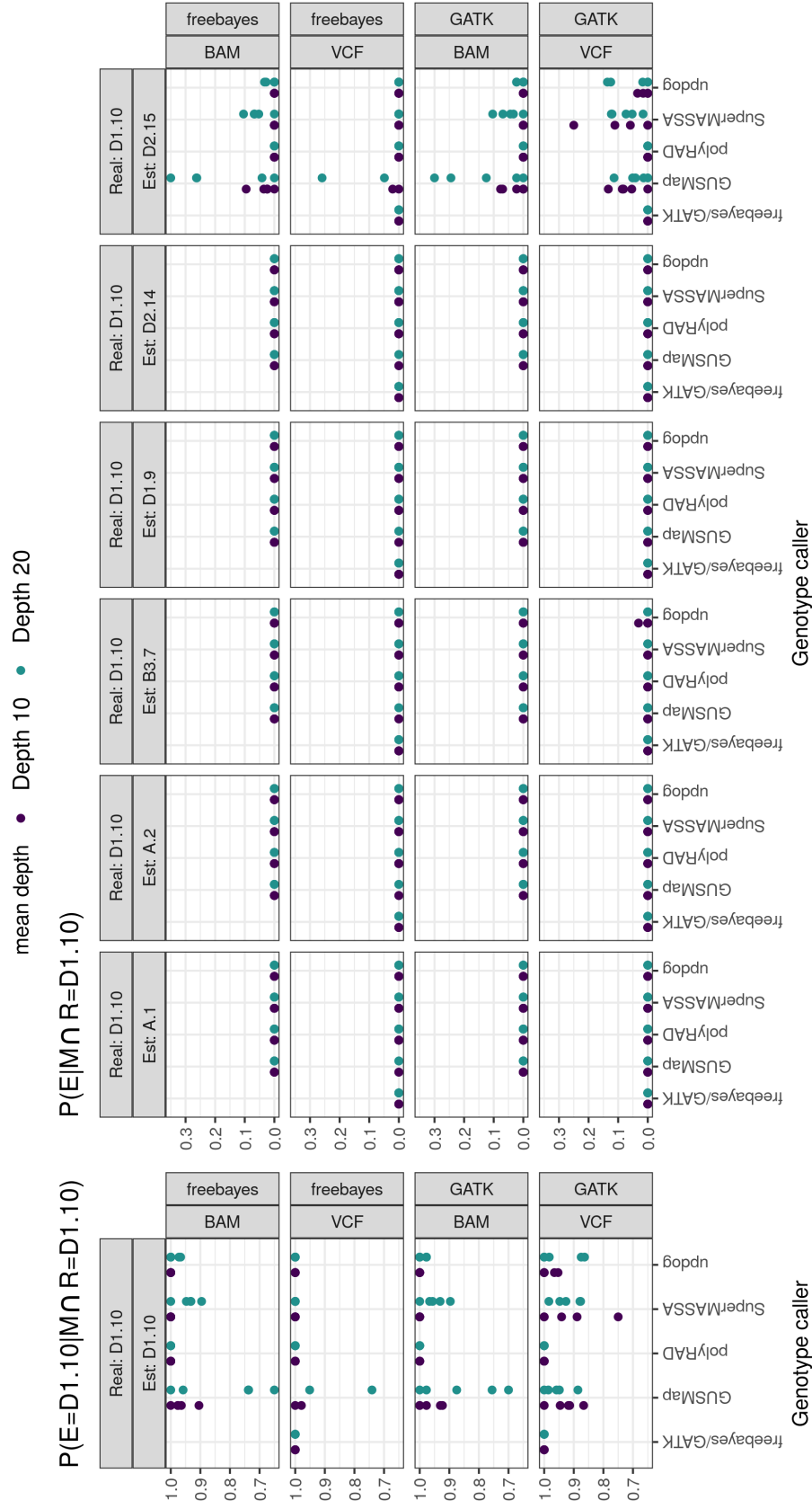


Figure A.21. Figure A.20 continued.

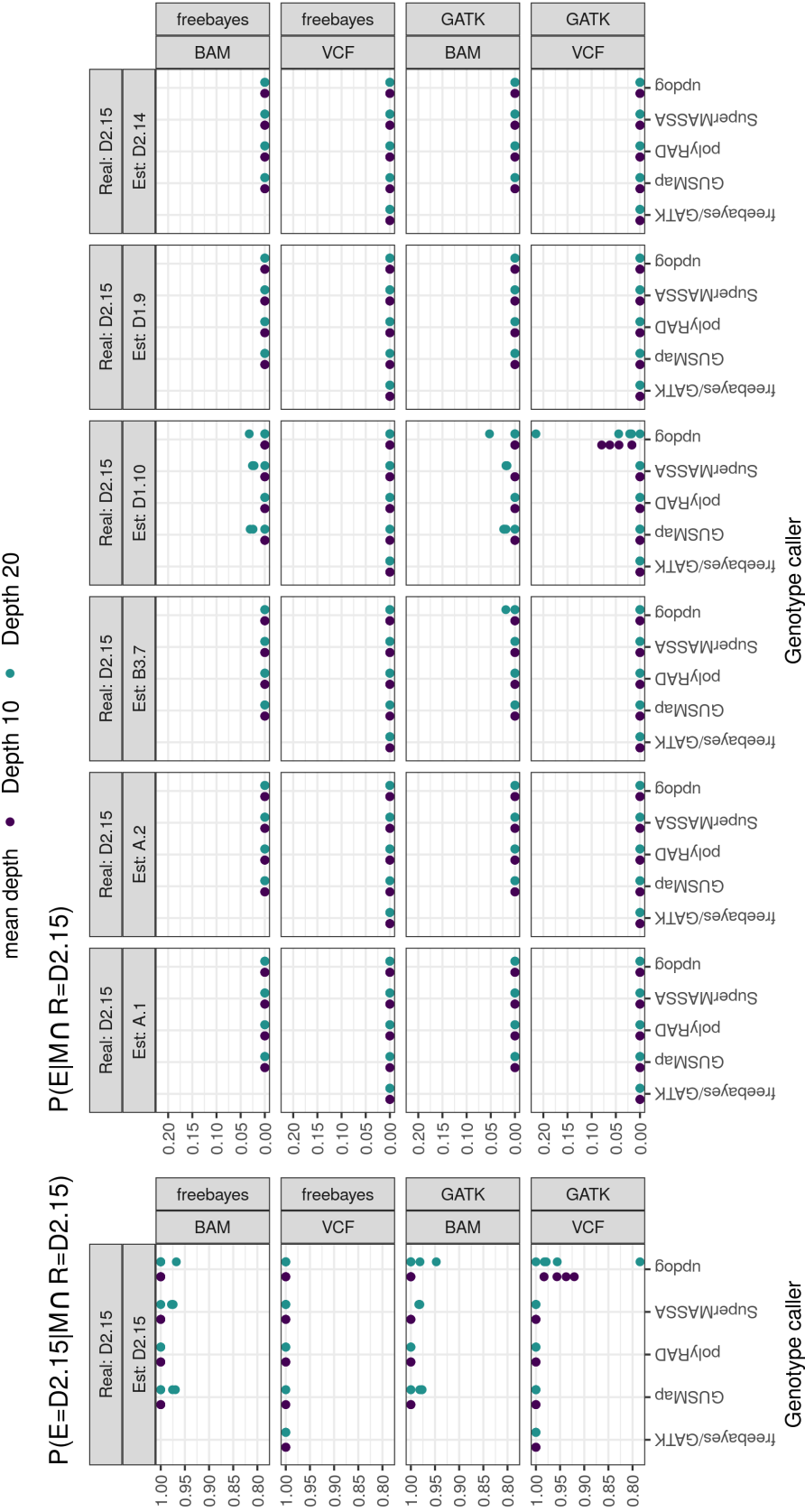


Figure A.22. Figure A.20 continued.

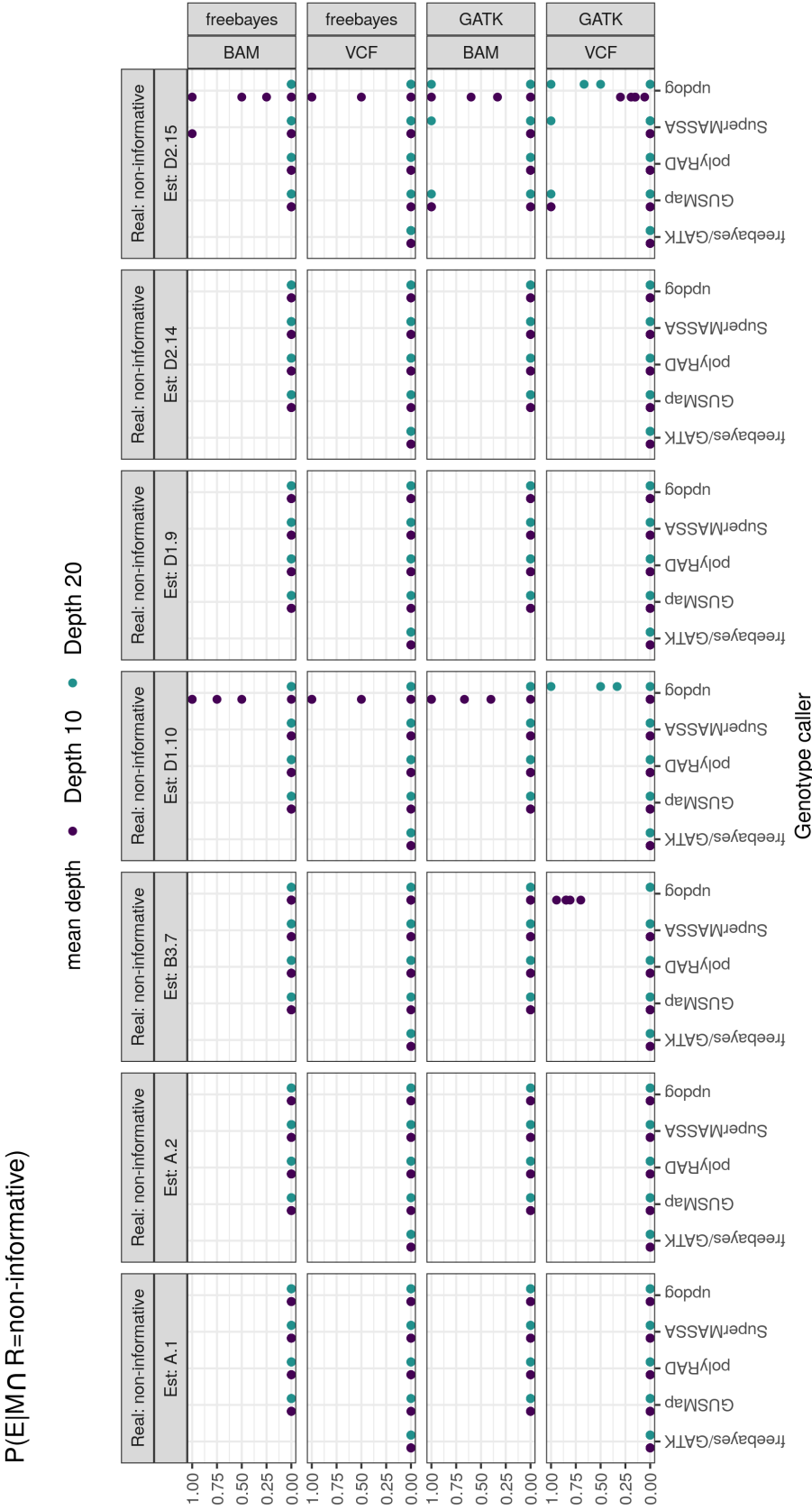


Figure A.23. Figure A.20 continued.

Appendix XIII - Simulated maps profiles

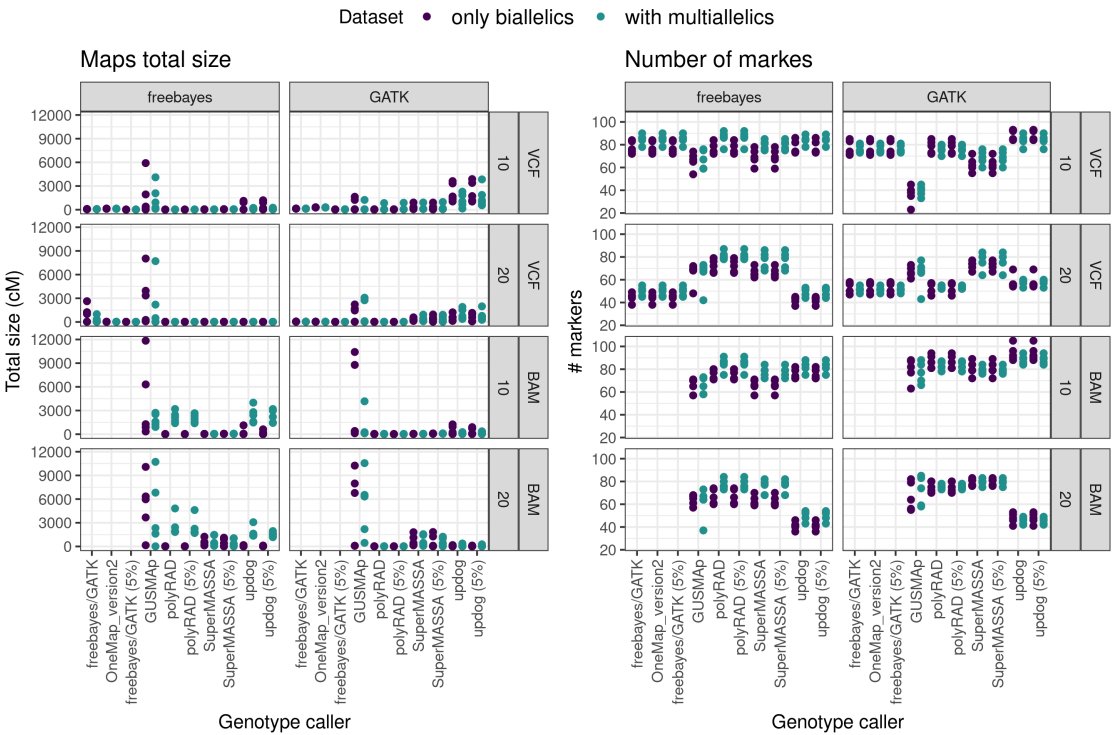
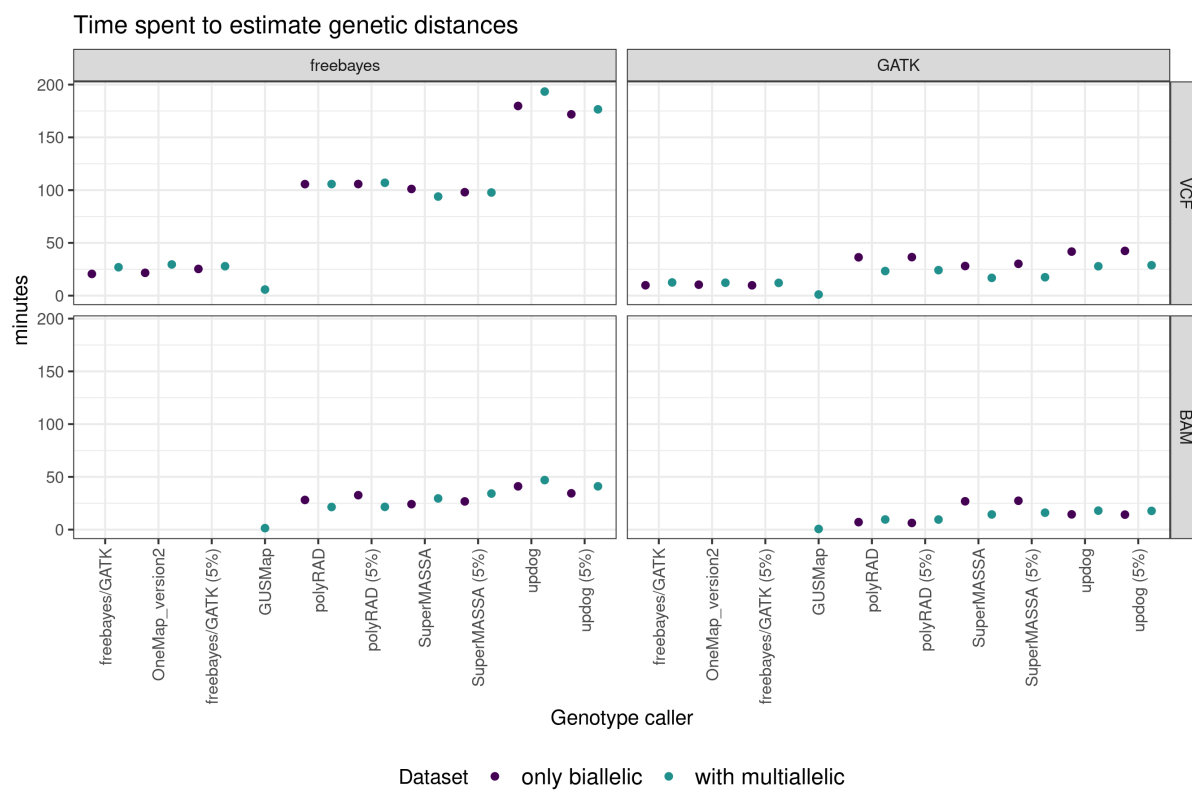
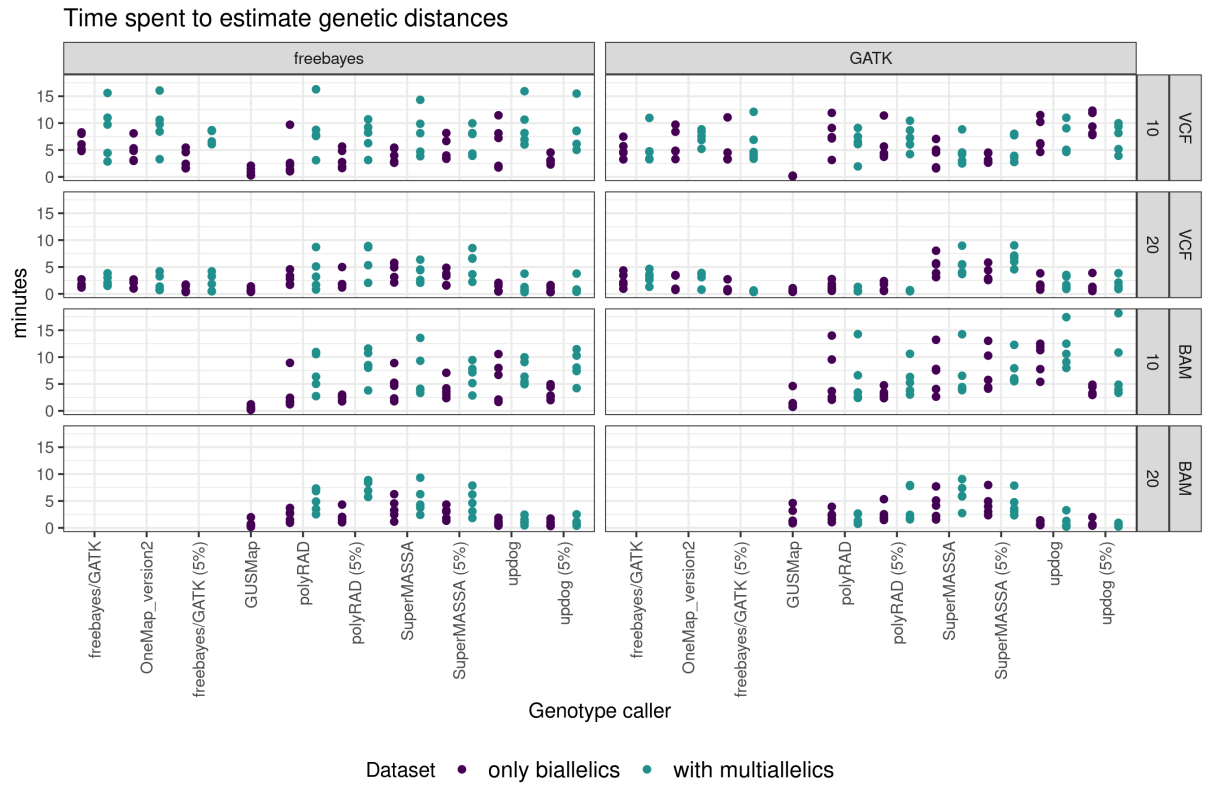


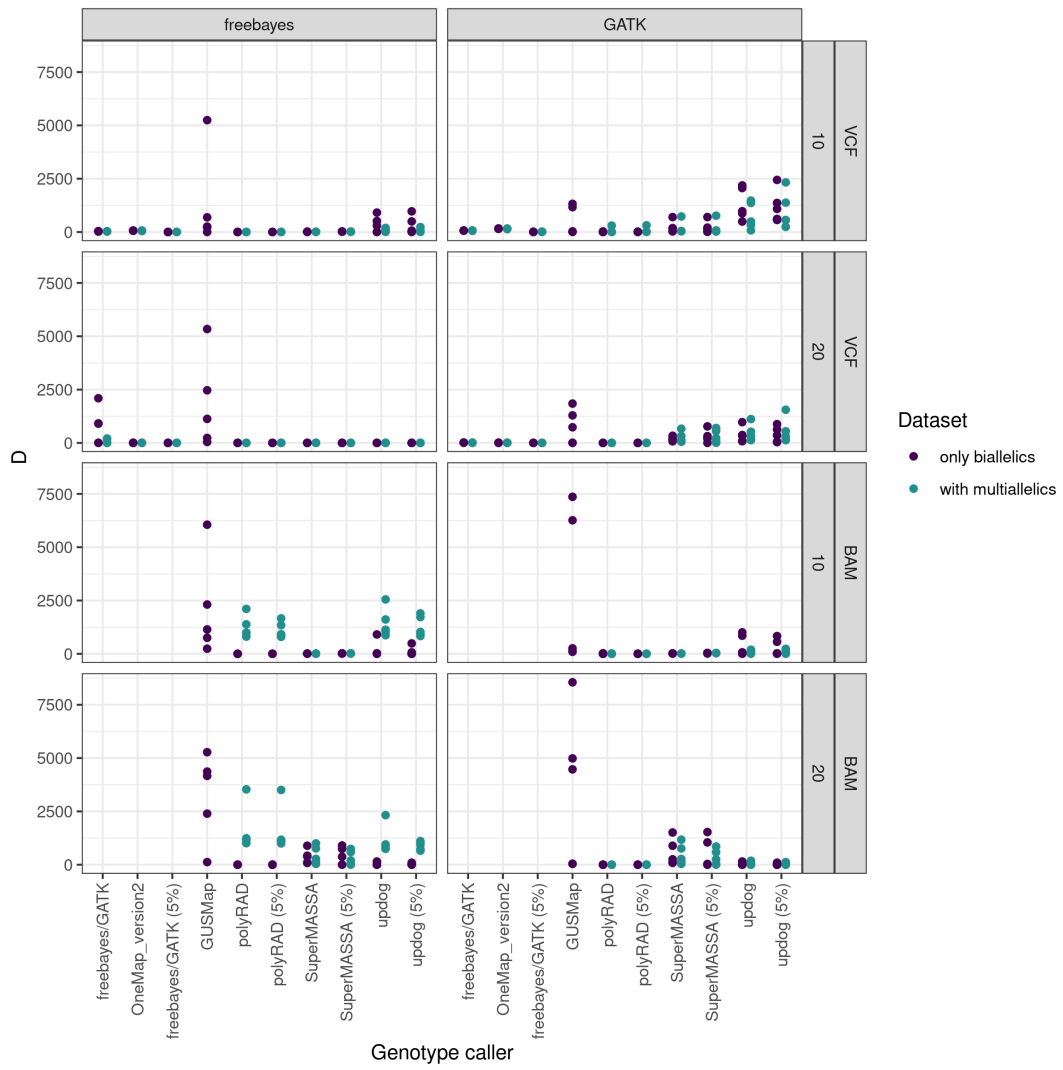
Figure A.24. Total size and number of markers of all maps built in the experiment.



**Figure A.25.** Time spent by OneMap 2.0 and 3.0 and GUSMap in **EmpiricalReads2Map** to estimate the *P. tremula* chromosome 10 linkage group genetic distances.



**Figure A.26.** Time spent by OneMap 2.0 and 3.0 and GUSMap in **SimulatedReads2Map** to estimate the genetics distances for the simulated maps based on 38% of chromosome 10 of *P. trichocarpa* genome.



**Figure A.27.** Average Euclidean distance ( $D$ ) comparing estimated and real distances between markers for all studied methods.





## ATTACHMENTS

### Attachment A: Simulations with OneMap

During the OneMap update, we develop three ways to simulate mapping populations data to test our new algorithms. The most complex, based on workflows, is described in this work and the following tutorial explain how to use the other two: *pedsim2raw* and *pedsim2vcf*.

The tutorial also shows how to perform genotyping with **updog** (GERARD ET AL., 2018), **polyRAD** (CLARK ET AL., 2019), and **SuperMASSA** (SERANG ET AL., 2012) using package **genotyping4onemap** with wrapper functions. This package was develop to give support to the workflows, but its functions can be used independently.

### Attachment B: OneMap Hidden Markov Model parallelization

Here we used the simulations functions described in Attachment A to certify the correct implementation in OneMap 3.0 of BatchMap (SCHIFFTHALER ET AL., 2017) parallelization approach and to test the alternative parallelization *parmap*.

### Attachment C: Adaptation to build maps with $F_2$ populations and dominant markers

Here we used the simulations functions described in Attachment A to certify the correct modification in the  $F_2$  functions to better deal with dominant markers and to be possible to build haplotype graphics for each individual in progeny.

### Attachment D: Maps resolution

Here we used the simulations functions described in Attachment A to demonstrate the impact of population size in the map resolution.

## Attachment A - Simulations with OneMap

OneMap 3.0 also makes interface with PedigreeSim (Voorrips and Maliepaard 2012) software to perform simulations. There are three different ways to simulate maps in OneMap:

- Converting PedigreeSim output to OneMap raw data: function `pedsim2raw`

This function just converts the PedigreeSim files to OneMap raw data. The only source of errors is missing data controlled by `miss.perc` argument.

- Converting PedigreeSim output to VCF file simulating allele depth with different distributions: function `pedsim2vcf`

Simulating the VCF you can include other sources of errors. The map is simulated in PedigreeSim, and according to each genotype, the chosen statistical distribution will simulate the reference and alternative alleles counts. The genotypes can be reestimated according with the alleles counts simulated using the genotype callers Updog (Gerard et al. 2018) (function `updog_genotype`), polyRAD (Clark, Lipka, and Sacks 2019) (function `polyRAD_genotype`) and Supermassa (Serang, Mollinari, and Garcia 2012) (function `supermassa_genotype` of `genotyping4onemap` package (C. H. Taniguti 2021a).

- Use a chromosome of a reference genome to simulate a bi-parental cross and RADseq Illumina reads for each individual: workflow `SimulatedReads2Map.wdl`

The workflow `SimulateReads2Map` is available in Reads2Map workflows (C. H. Taniguti and Taniguti 2021) repository in Github. It performs the simulation of RADseq Illumina reads for outcrossing population, the SNP calling is made in Freebayes (Garrison and Marth 2012) and HaplotypeCaller (Poplin et al. 2017) software, the genotype calling in updog (Gerard et al. 2018), SuperMASSA (Serang, Mollinari, and Garcia 2012) and polyRAD (Clark, Lipka, and Sacks 2019), and build genetics maps using every combination in OneMap and GUSMap (Bilton et al. 2018). The workflow is written in Workflow Description Language (WDL) and Cromwell workflow management system (Voss, Gentry, and Auwera 2017). The parameters of every software implemented can be easily changed. All the results can be evaluated in a shiny app (C. H. Taniguti 2021b).

### Run PedigreeSim

First, download PedigreeSim java file. It will require java installed. You can run PedigreeSim directly and just use the output files in OneMap or you can use a function named `run_pedsim` that facilitates this procedure.

The function does not provide every possibility offered by PedigreeSim software. If you want to change any parameter that is not available in the function, please use directly the PedigreeSim software.

```
# For outcrossing population
run_pedsim(chromosome = "Chr1", n.marker = 54, tot.size.cm = 100, centromere = 50,
           n.ind = 200, mk.types = c("A1", "A2", "A3", "A4", "B1.5", "B2.6", "B3.7",
                                     "C.8", "D1.9", "D1.10", "D1.11", "D1.12", "D1.13",
                                     "D2.14", "D2.15", "D2.16", "D2.17", "D2.18"),
           n.types = rep(3,18), pop = "F1",
           path.pedsim = "/home/cristiane/Programs/PedigreeSim/",
           name.mapfile = "mapfile.map", name.founderfile="founderfile.gen",
           name.chromfile="sim.chrom", name.parfile="sim.par",
           name.out="sim_out")
```

```
# For F2 population
run_pedsim(chromosome = c("Chr1", "Chr2"), n.marker = c(75, 75),
           tot.size.cm = c(100,100), centromere = c(50,50),
           n.ind = 200, mk.types = c("A.H.B","C.A", "D.B"),
           n.types = rep(50,3), pop = "F2",
           path.pedsim = "/home/cristiane/Programs/PedigreeSim/",
           name.mapfile = "mapfile_f2.map", name.founderfile="founderfile_f2.gen",
           name.chromfile="sim_f2.chrom", name.parfile="sim_f2.par",
           name.out="sim_f2")
```

The function allows us to create f2 intercross and backcross populations from bi-parental cross of inbred lines and segregating F1 population from bi-parental cross of heterozygous parents. You can define it in `pop` argument. You must change the `path.pedsim` to the path where the PedigreeSim.jar is stored in your system. You can also define the number of chromosomes (argument `chromosome`), the number of markers in each chromosome (`n.marker`), the total size of the groups in cM (`tot.size.cm`), the position of the centromere (`centromere`), number of individuals in the population (`n.ind`), the marker types (`mk.types`, see the table in session [Creating the data file](#) of Outcrossing populations vignette and the number of markers of each type (`n.types`).

We suggest you open the output files `founderfile`, `chromfile`, `mapfile` and `parfile` to check if they agree with your intentions before proceeding to other analyses.

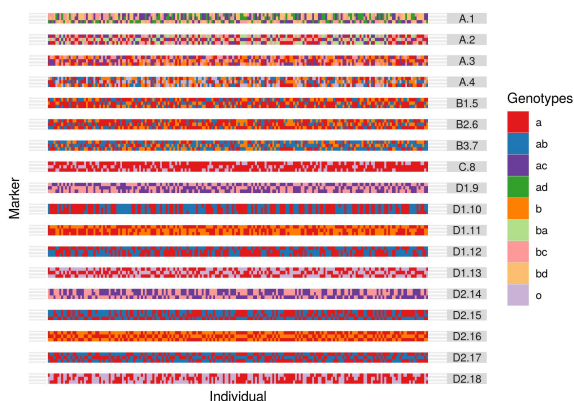
## Simulate OneMap raw data

Once you run PedigreeSim, you should have the output file `genotypes.dat`. To convert this to OneMap raw file, you just need to specify the cross-type (`cross`), which ones are the parents (`parent1` and `parent2`), and if you want to include missing genotypes (`miss.perc`). Only cross types `outcross` and `f2 intercross` are supported by now.

```
# For outcrossing population
pedsim2raw(genofile = "sim_out_genotypes.dat", cross="outcross",
           parent1 = "P1", parent2 = "P2", out.file = "sim_out.example.raw", miss.perc = 0)

onemap.obj <- read_onemap("sim_out.example.raw")

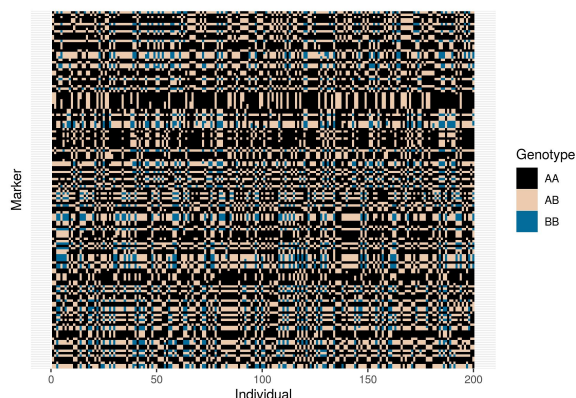
p <- plot(onemap.obj, all = F)
ggsave(p, filename = "plot_onemap1.jpeg")
```



```
# For F2 population
pedsim2raw(genofile = "sim_f2_genotypes.dat", cross="f2 intercross",
           parent1 = "P1", parent2 = "P2", out.file = "sim_f2.example.raw", miss.perc = 0)
```

```
onemap.obj <- read_onemap("sim_f2.example.raw")

p <- plot(onemap.obj, all = F)
ggsave(p, filename = "plot_onemap2.jpeg")
```



## Simulate VCF file

The same output file from PedigreeSim, the `genotypes.dat` can be used to simulate a VCF file together with the PedigreeSim `mapfile` and `chrom`. The advantage to simulate a VCF instead of OneMap raw file is that VCF is a standard file format and can store a lot of other information including the allele counts, usually in the field AD or DPR. The `pedsim2vcf` function can simulate the allele counts using negative binomial or updog distributions (argument `method`). The main parameters for the distributions are defined with arguments `mean.depth`, that defines the mean allele depth in the progeny, `p.mean.depth` that defines the mean allele depth in the parents, argument `disper.par` defines the dispersion parameter, `mean.phred` defines the mean Phred score of the sequencing technology used. The function can also simulate missing data (`miss.perc`). Through argument `pos` and `chr` you can define vectors with physical position and chromosome of each marker. With argument `haplo.ref` you define which one of the haplotypes in `genotypes.dat` will be considered the reference. Establishing `phase` as TRUE, the VCF will have phased genotypes. After allele counts are simulated, the genotypes are re-estimated using a binomial distribution. The VCF generated by this function only has one or two FORMAT fields, the GT and AD (if `counts = TRUE`). **Dominant markers are not supported by this function.**

```
run_pedsim(chromosome = "Chr1", n.marker = 42, tot.size.cm = 100, centromere = 50,
  n.ind = 200, mk.types = c("B3.7", "D1.10", "D2.15"),
  n.types = rep(14,3), pop = "F1",
  path.pedsim = "/home/cristiane/Programs/PedigreeSim/",
  name.mapfile = "mapfile.map", name.founderfile="founderfile.gen",
  name.chromfile="sim.chrom", name.parfile="sim.par",
  name.out="sim_out")
```

*# For outcrossing population*

```
pedsim2vcf(inputfile = "sim_out_genotypes.dat",
  map.file = "mapfile.map",
  chrom.file = "sim.chrom",
  out.file = "simu_out.vcf",
  miss.perc = 0,
  counts = TRUE,
  mean.depth = 100,
  p.mean.depth = 100,
```

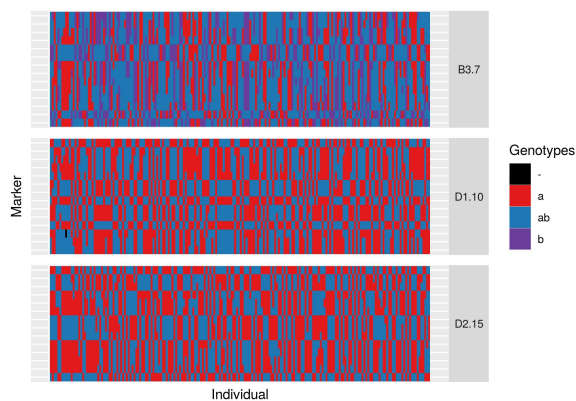
```

chr.mb = 10,
method = "updog",
mean.phred = 20,
bias=1,
od=0.00001,
pos=NULL,
chr=NULL,
phase = FALSE,
disper.par=2)

library(vcfR)
vcfR_out.obj <- read.vcfR("simu_out.vcf")
onemap_out.obj <- onemap_read_vcfR(vcfR.object = vcfR_out.obj,
                                cross = "outcross", parent1 = "P1", parent2 = "P2")

p <- plot(onemap_out.obj, all=F)
ggsave(p, filename = "plot_onemap3.jpeg")

```



```

# For F2 population
run_pedsim(chromosome = c("Chr1", "Chr2"), n.marker = c(75, 75),
           tot.size.cm = c(100,100), centromere = c(50,50),
           n.ind = 200, mk.types = c("A.H.B"),
           n.types = 150, pop = "F2",
           path.pedsim = "/home/cristiane/Programs/PedigreeSim/",
           name.mapfile = "mapfile_f2.map", name.founderfile="founderfile_f2.gen",
           name.chromfile="sim_f2.chrom", name.parfile="sim_f2.par",
           name.out="sim_f2")

pedsim2vcf(inputfile = "sim_f2_genotypes.dat",
           map.file = "mapfile_f2.map",
           chrom.file = "sim_f2.chrom",
           out.file = "simu_f2.vcf",
           miss.perc = 0,
           counts = TRUE,
           mean.depth = 100,
           p.mean.depth = 100,
           chr.mb = 10,
           method = "updog",
           mean.phred = 20,
           bias=1,

```

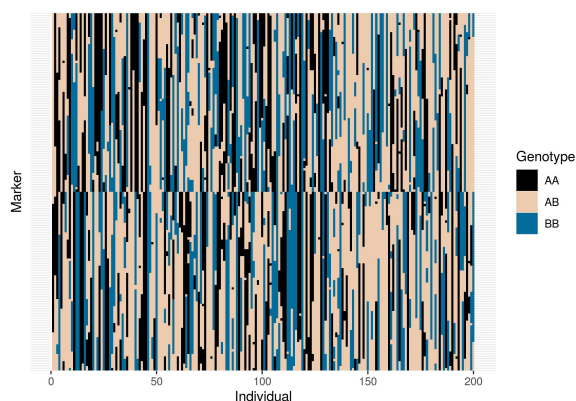
```

        od=0.001,
        pos=NULL,
        chr=NULL,
        phase = FALSE,
        disper.par=2)

vcfR_f2.obj <- read.vcfR("simu_f2.vcf")
onemap_f2.obj <- onemap_read_vcfR(vcfR.object = vcfR_f2.obj,
                                cross = "f2 intercross", parent1 = "P1",
                                parent2 = "P2", f1 = "F1")

p <- plot(onemap_f2.obj, all=F)
ggsave(p, filename = "plot_onemap4.jpeg")

```



## Graphical view of genotypes and allele counts

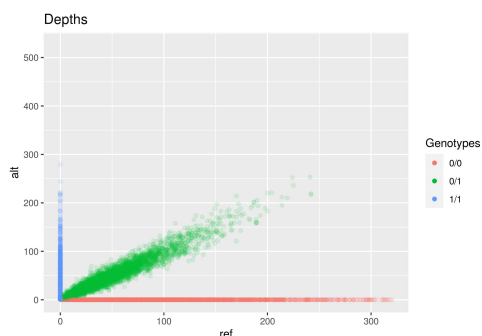
Function `create_depth_profile` generates dispersion graphics with x and y-axis pointing the reference and alternative allele counts, respectively. The function is only available for biallelic markers and for outcrossing and f2 intercross population. Each dot represents a genotype considering `mks` markers and `inds` individuals. If are established NULL for both arguments, all markers and individuals are considered. The color of the dots are according with the genotypes present in OneMap object (`GTfrom = onemap`) or in VCF file (`GTfrom = vcf`) or the color can represent the error rate (`1 - highest genotype probability`) of each genotype in OneMap object (`GTfrom = prob`). An rds file is generated with the data in the graphic (`rds.file`). The `alpha` argument controls the transparency of color of each dot, regulate this parameter is a good idea when having a big amount of markers and individuals.

```

# For outcrossing population
p <- create_depths_profile(onemap.obj = onemap_out.obj,
                           vcfR.object = vcfR_out.obj,
                           parent1 = "P1",
                           parent2 = "P2",
                           vcf.par = "AD",
                           recovering = FALSE,
                           mks = NULL,
                           inds = NULL,
                           GTfrom = "vcf",
                           alpha = 0.1,
                           rds.file = "depths_out.rds")

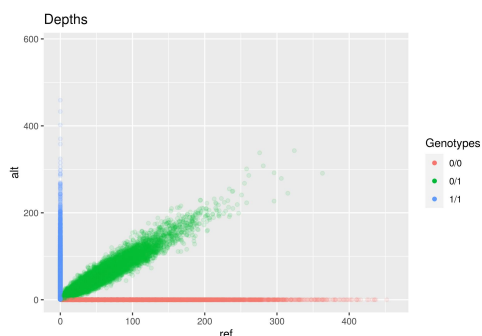
ggsave(p, filename = "depth_prof1.jpeg")

```



```
# For f2 intercross population
p <- create_depths_profile(onemap.obj = onemap_f2.obj,
  vcfR.object = vcfR_f2.obj,
  parent1 = "P1",
  parent2 = "P2",
  f1 = "F1",
  vcf.par = "AD",
  recovering = FALSE,
  mks = NULL,
  inds = NULL,
  GTfrom = "vcf",
  alpha = 0.1,
  rds.file = "depths_f2.rds")

ggsave(p, filename = "depth_prof2.jpeg")
```



If you choose to simulate allele counts in `pedsim2vcf` it is also possible to reestimate the genotypes using these counts. Doing this, you will include errors in your data, coming from random sampling, if considering only Poisson or negative binomial distributions, and/or dispersion, outliers and bias errors, if using updog model. Another advantage is that reestimating genotypes with any of these software, you can obtain genotypes probabilities to be used in the map building instead of only genotypes.

**warning:** The re-estimation of genotypes is only performed in biallelic codominant markers.

Updog, polyRAD, and SuperMASSA are software designed for genotyping polyploid species, which involve a more complex procedure compared with diploid species. These software consider not only the proportion of alleles to define the genotypes but other aspects as the expected distribution in the progeny according to parent's genotypes. See more about them in their manuals.

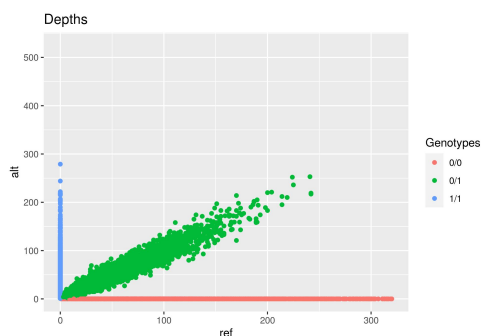


With updog

[illegible]

```
p <- create_depths_profile(onemap.obj = onemap_genotype.updog,
  vcfR.object = vcfR_out.obj,
  parent1 = "P1",
  parent2 = "P2",
  vcf.par = "AD",
  recovering = FALSE,
  GTfrom = "vcf")
```

```
ggsave(p, filename = "depth_prof3.jpeg")
```



The function `polyRAD_genotype` make the interface of OneMap with polyRAD to perform the genotype calling.

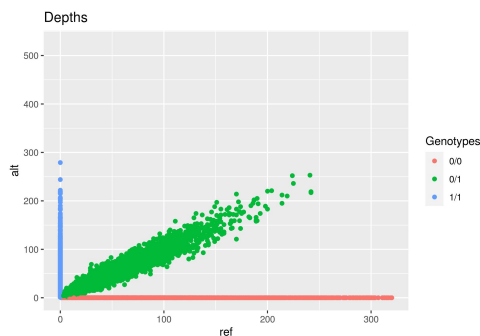
```
onemap_geno.polyRAD <- polyRAD_genotype(vcf="simu_out.vcf",
                                         onemap.obj = onemap_out.obj,
                                         parent1="P1",
```

```

parent2="P2",
f1="F1",
crosstype="outcross",
global_error = NULL,
use_genotypes_errors = TRUE,
use_genotypes_probs = FALSE,
rm_multiallelic = F)

p <- create_depths_profile(onemap.obj = onemap_geno.polyRAD,
  vcfR.object = vcfR_out.obj,
  parent1 = "P1",
  parent2 = "P2",
  vcf.par = "AD",
  recovering = FALSE,
  GTfrom = "vcf")
ggsave(p, filename = "depth_prof4.jpeg")

```



## With SuperMASSA

The function `supermassa_genotype` make the interface of OneMap with SuperMASSA to perform the genotype calling.

```

onemap_geno.supermassa <- supermassa_genotype(vcfR.object=vcfR_out.obj,
  onemap.object= onemap_out.obj,
  vcf.par = "AD",
  parent1="P1",
  parent2="P2",
  f1=NULL,
  recovering = FALSE,
  mean_phred = 20,
  cores = 4,
  depths = NULL,
  global_error = NULL,
  use_genotypes_errors = FALSE,
  use_genotypes_probs = TRUE,
  rm_multiallelic = F)

p <- create_depths_profile(onemap.obj = onemap_geno.supermassa,
  vcfR.object = vcfR_out.obj,
  parent1 = "P1",
  parent2 = "P2",
  vcf.par = "AD",

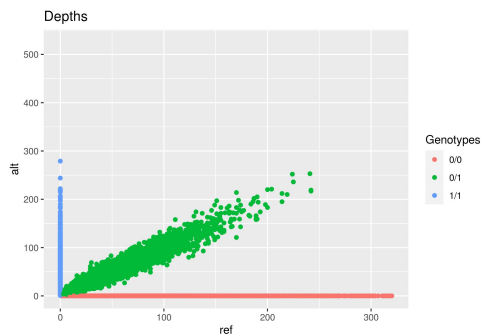
```

```

recovering = FALSE,
GTfrom = "vcf")

ggsave(p, filename = "depth_prof5.jpeg")

```



## Example

Here we want to compare the best map using updog counts simulations and updog, polyRAD and SuperMASSA genotyping and genotypes probabilities. As a simple example, we will simulate only one family and one chromosome. Simulation scenarios:

- Mean depth: 10, 50, and 100;
- Genetic map size: 100 cM
- Number of markers: 42
- Marker types: 14 B3.7; 14 D1.10; 14 D2.15

```

for(depth in c(100, 50, 10)){
  run_pedsim(chromosome = "Chr1", n.marker = 42,
             tot.size.cm = 100, centromere = 50,
             n.ind = 200, mk.types = c("B3.7", "D1.10", "D2.15"),
             n.types = rep(14,3), pop = "F1",
             path.pedsim = "/home/cristiane/Programs/PedigreeSim/",
             name.mapfile = "mapfile.map", name.founderfile="founderfile.gen",
             name.chromfile="sim.chrom", name.parfile="sim.par",
             name.out="sim_out")

  # VCF with unmodified genotypes
  pedsim2vcf(inputfile = "sim_out_genotypes.dat",
             map.file = "mapfile.map",
             chrom.file = "sim.chrom",
             out.file = "simu_out.vcf",
             miss.perc = 0,
             counts = FALSE)

  vcfr_out.obj <- read.vcfr("simu_out.vcf")
  onemap_out.obj <- onemap_read_vcfr(vcfr.object = vcfr_out.obj,
                                   cross = "outcross", parent1 = "P1", parent2 = "P2")

  twopts <- rf_2pts(onemap_out.obj)
  seq_ord <- make_seq(twopts, order(as.numeric(onemap_out.obj$POS)))
  map_real <- map(seq_ord, phase_cores = 4)
  p <- rf_graph_table(map_real, main = paste0("Real_",depth))

```

```

ggsave(p, filename = paste0("real", depth, ".jpeg"))

# Simulating counts
pedsim2vcf(inputfile = "sim_out_genotypes.dat",
  map.file = "mapfile.map",
  chrom.file = "sim.chrom",
  out.file = "simu_out.vcf",
  miss.perc = 0,
  counts = TRUE,
  mean.depth = depth,
  p.mean.depth = depth,
  chr.mb = 10,
  method = "updog",
  mean.phred = 20,
  bias=0.8,
  od=0.0001,
  pos=NULL,
  chr=NULL,
  phase = FALSE,
  disper.par=2)

vcfR_out.obj <- read.vcfR("simu_out.vcf")
onemap_out.obj <- onemap_read_vcfR(vcfR.object = vcfR_out.obj,
  cross = "outcross", parent1 = "P1", parent2 = "P2")

onemap_genotype.updog <- updog_genotype(vcfR.object=vcfR_out.obj,
  onemap.object= onemap_out.obj,
  vcf.par = "AD",
  parent1="P1",
  parent2="P2",
  f1=NULL,
  recovering = TRUE,
  mean_phred = 20,
  cores = 4,
  depths = NULL,
  global_error = NULL,
  use_genotypes_errors = FALSE,
  use_genotypes_probs = TRUE,
  rm_multiallelic = F)

twopts <- rf_2pts(onemap_genotype.updog)
seq_ord <- make_seq(twopts, order(as.numeric(onemap_genotype.updog$POS)))
map_updog <- map(input.seq = seq_ord, phase_cores = 4, rm_unlinked = T)
# If HMM find problems between two markers, one of them will be automatically
# discarded and the sequence of markers without it will be returned
while(is(map_updog, "vector")){
  # if the result is a sequence of marker numbers, then HMM is run again
  # This will be repeated until HMM can run with no problems
  seq_temp <- make_seq(twopts, map_updog)
  map_updog <- map(input.seq = seq_temp, phase_cores = 4, rm_unlinked = T)
}
p <- rf_graph_table(map_updog, main = paste0("Updog depth", depth))
ggsave(p, filename = paste0("updog", depth, ".jpeg"))

```

```

onemap_genotype.polyRAD <- polyRAD_genotype(vcf="simu_out.vcf",
                                             onemap_obj = onemap_out.obj,
                                             parent1="P1",
                                             parent2="P2",
                                             f1="F1",
                                             crosstype="outcross",
                                             global_error = NULL,
                                             use_genotypes_errors = FALSE,
                                             use_genotypes_probs = TRUE,
                                             rm_multiallelic = F)

twopts <- rf_2pts(onemap_genotype.polyRAD)
seq_ord <- make_seq(twopts, order(as.numeric(onemap_genotype.polyRAD$POS)))
map_polyRAD <- map(input.seq = seq_ord, phase_cores = 4, rm_unlinked = T)
while(is(map_polyRAD, "vector")){
  seq_temp <- make_seq(twopts, map_polyRAD)
  map_polyRAD <- map(input.seq = seq_temp, phase_cores = 4, rm_unlinked = T)
}
p <- rf_graph_table(map_polyRAD, main = paste0("polyRAD depth", depth))
ggsave(p, filename = paste0("polyrad", depth, ".jpeg"))

onemap_genotype.supermassa <- supermassa_genotype(vcfR.object=vcfR_out.obj,
                                                  onemap.object= onemap_out.obj,
                                                  vcf.par = "AD",
                                                  parent1="P1",
                                                  parent2="P2",
                                                  f1=NULL,
                                                  recovering = FALSE,
                                                  mean_phred = 20,
                                                  cores = 4,
                                                  depths = NULL,
                                                  global_error = NULL,
                                                  use_genotypes_errors = FALSE,
                                                  use_genotypes_probs = TRUE,
                                                  rm_multiallelic = F)

twopts <- rf_2pts(onemap_genotype.supermassa)
seq_ord <- make_seq(twopts, order(as.numeric(onemap_genotype.supermassa$POS)))
map_supermassa <- map(input.seq = seq_ord, phase_cores = 4, rm_unlinked = T)
while(is(map_supermassa, "vector")){
  seq_temp <- make_seq(twopts, map_supermassa)
  map_supermassa <- map(input.seq = seq_temp, phase_cores = 4, rm_unlinked = T)
}
p <- rf_graph_table(map_supermassa, main = paste0("supermassa depth", depth))
ggsave(p, filename = paste0("supermassa", depth, ".jpeg"))

onemap_0.05 <- create_probs(onemap_genotype.updog, global_error = 0.05)

twopts <- rf_2pts(onemap_0.05)
seq_ord <- make_seq(twopts, order(as.numeric(onemap_0.05$POS)))
map_0.05 <- map(input.seq = seq_ord, phase_cores = 4, rm_unlinked = T)
while(is(map_0.05, "vector")){
  seq_temp <- make_seq(twopts, map_0.05)

```

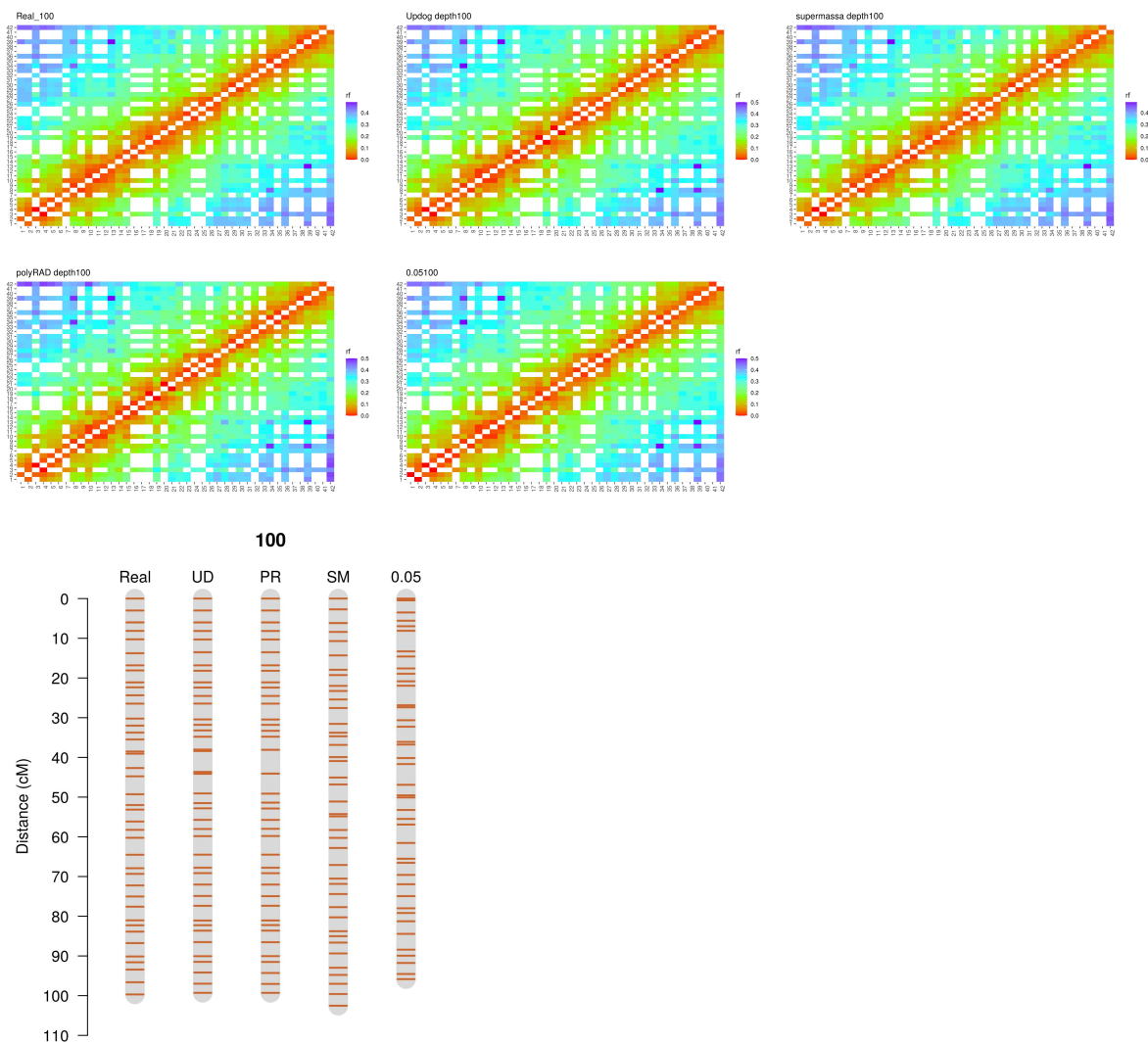
```

map_0.05 <- map(input.seq = seq_temp, phase_cores = 4, rm_unlinked = T)
}
p <- rf_graph_table(map_0.05, main = paste0("0.05", depth))
ggsave(p, filename = paste0("0.05", depth, ".jpeg"))

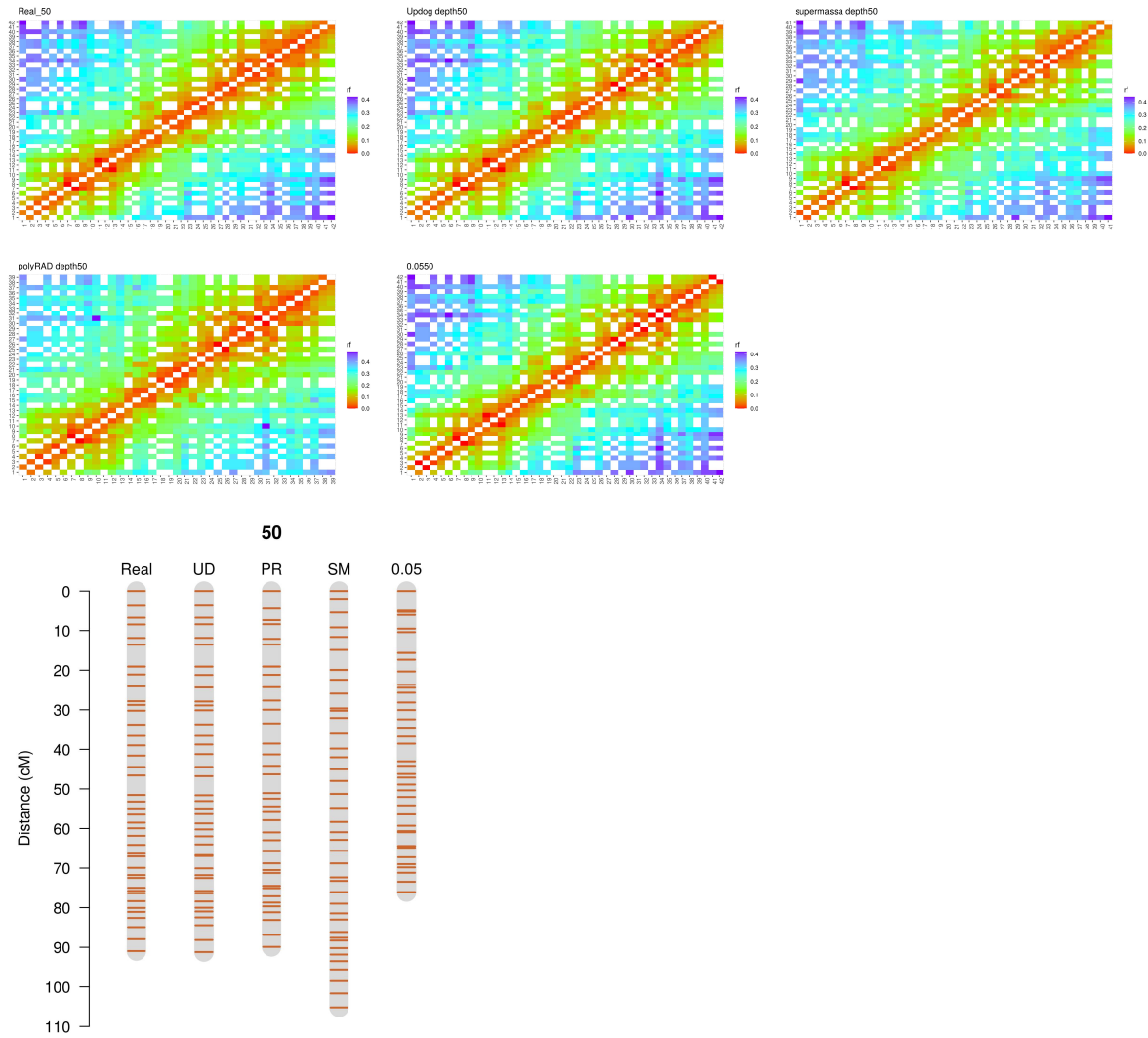
draw_map2(map_real, map_updog, map_polyRAD,
  map_supermassa, map_0.05, main = depth,
  group.names = c("Real", "UD", "PR", "SM", "0.05"),
  output = paste0(depth, "_maps.jpeg"))
}

```

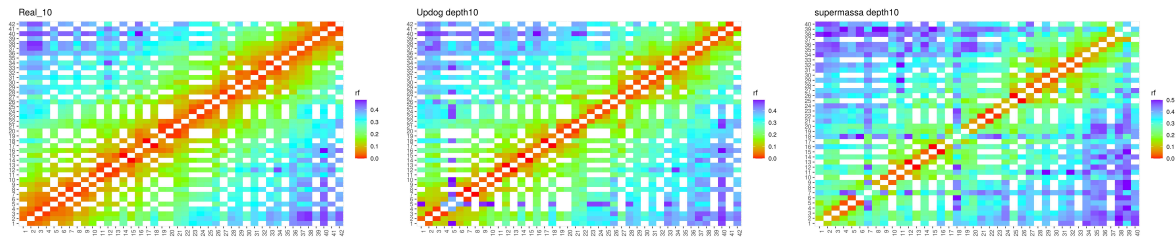
## Sequencing depth 100

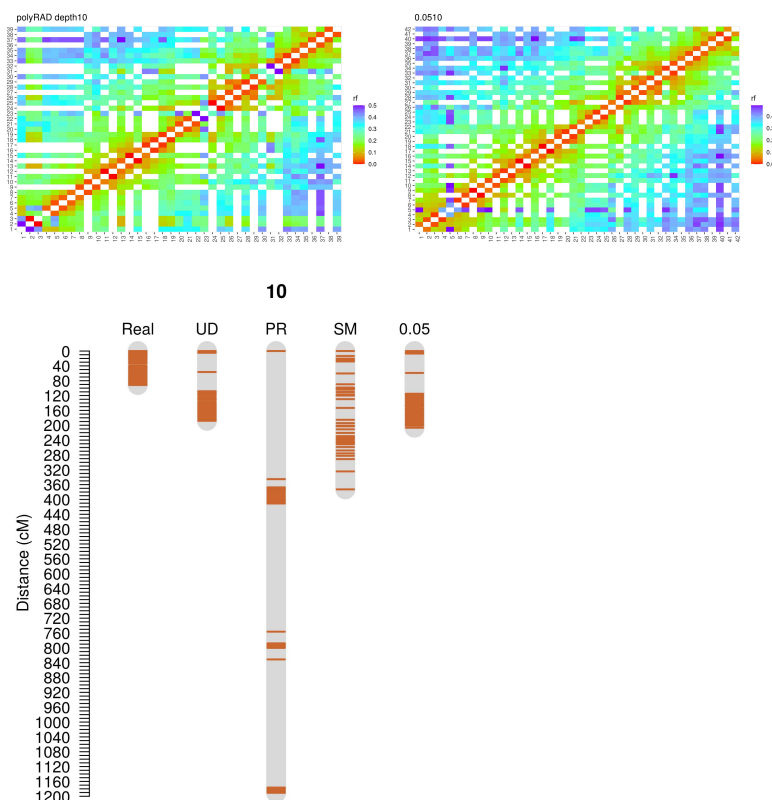


Sequencing depth 50



Sequencing depth 10





## Conclusions

The functions `pedsim2raw` and `pedsim2vcf` can be useful to test new algorithms and genetic questions which not need to consider genotyping errors. Besides `updog` has a robust model to simulate and identify sequencing errors, it can not be used to test its own performance. To be possible to make a fair comparison, errors must be simulated with another approach, as the one implemented in Reads2Map workflows. The simulation performed here was just to demonstrate the usage; we can not make any conclusions based on that.

## Simulate Illumina reads

This one is a bit more complex, and its tools are stored in the GitHub repository Reads2Map workflows (C. H. Taniguti and Taniguti 2021). Please, access it for more information.

### Remove generated files

```
system("rm sim* founderfile* mapfile* *rds ")
```

## References

Bilton, Timothy P., Matthew R. Schofield, Michael A. Black, David Chagné, Phillip L. Wilcox, and Ken G. Dodds. 2018. “Accounting for Errors in Low Coverage High-Throughput Sequencing Data When Constructing Genetic Maps Using Biparental Outcrossed Populations.” *Genetics* 209 (1): 65–76. <https://doi.org/10.1534/genetics.117.300627>.



- Clark, Lindsay V., Alexander E. Lipka, and Erik J. Sacks. 2019. “polyRAD: Genotype Calling with Uncertainty from Sequencing Data in Polyploids and Diploids.” *G3: Genes/Genomes/Genetics* 9 (March): g3.200913.2018. <https://doi.org/10.1534/g3.118.200913>.
- Garrison, Erik, and Gabor Marth. 2012. “Haplotype-based variant detection from short-read sequencing.” *ArXiv E-Prints*, 9. <https://doi.org/1207.3907>.
- Gerard, David, Luis Felipe Ventorim Ferrão, Antonio Augusto Franco Garcia, and Matthew Stephens. 2018. “Genotyping Polyploids from Messy Sequencing Data.” *Genetics* 210 (3): 789–807. <https://doi.org/10.1534/genetics.118.301468>.
- Poplin, Ryan, Valentin Ruano-Rubio, Mark A. DePristo, Tim J. Fennell, Mauricio O. Carneiro, Geraldine A. Van der Auwera, David E. Kling, et al. 2017. “Scaling accurate genetic variant discovery to tens of thousands of samples.” *bioRxiv*, 201178. <https://doi.org/10.1101/201178>.
- Serang, Oliver, Marcelo Mollinari, and Antonio Augusto Franco Garcia. 2012. “Efficient exact maximum a posteriori computation for bayesian SNP genotyping in polyploids.” *PLoS ONE* 7 (2): 1–13. <https://doi.org/10.1371/journal.pone.0030906>.
- Taniguti, Cristiane Hayumi. 2021a. “Genotyping4onemap.” [https://github.com/Cristianetaniguti/genotypin\\_g4onemap](https://github.com/Cristianetaniguti/genotypin_g4onemap).
- . 2021b. “Reads2MapApp.” <https://github.com/Cristianetaniguti/reads2mapApp>.
- Taniguti, Cristiane Hayumi, and Lucas Mitsuo Taniguti. 2021. “Reads2Map.” [https://github.com/Cristiane\\_taniguti/reads2map](https://github.com/Cristiane_taniguti/reads2map).
- Voorrips, Roeland E, and Chris A Maliepaard. 2012. “The simulation of meiosis in diploid and tetraploid organisms using various genetic models.” *BMC Bioinformatics* 13 (1): 248. <https://doi.org/10.1186/1471-2105-13-248>.
- Voss, Kate, Jeff Gentry, and Geraldine Van Der Auwera. 2017. “Full-stack genomics pipelining with GATK4+WDL+ Cromwell [version 1; not peer reviewed].” *F1000Research*, 4. <https://doi.org/10.7490/f1000research.1114631.1>.

## Attachment B - OneMap Hidden Markov Model parallelization

The goal here is to optimize the speed of OneMap Hidden Markov Model (HMM) (Margarido, Souza, and Garcia 2007) algorithm to genetic distance estimation. For this, we implemented in OneMap two approaches: **parmap** function and the approach proposed in BatchMap package (Schiffthaler et al. 2017), a fork from OneMap.

The **parmap** splits an ordered sequence into groups according to the number of cores available to the analyses. The groups have an overlap of markers in their edges, to be possible the later joint of them. Here we use these overlap markers also to compare the estimated genetic distance between groups and see how the process can affect the maps distances.

The approach implemented in BatchMap (and now in OneMap) also divides the sequence into batches, but the parallelization is limited by the number of possible phases. Therefore, a maximum of four cores can be used to estimate genetic distances for outcrossing species.

To be a fair comparison, here we use four cores for both approaches.

Tests:

All scenarios have 1 chromosome with 100 cM, 5 family F1, progeny size of 150 individuals, 504 markers, 6 cores, and overlap of 5 markers.

- **Scenario 1:** 28 markers of each one of the 18 possible types, without missing data
- **Scenario 2:** 168 markers of D1.10, D2.15, and B3.7 types, without missing data
- **Scenario 3:** 28 markers of each one of the 18 possible types, 25% of missing data
- **Scenario 4:** 168 markers of D1.10, D2.15 and B3.7 types, 25% of missing data

Measures:

- Total size
- Recombination fraction difference between overlap markers
- Time to run

Another test applied was to increase the tolerance value of the HMM. All scenarios were repeated with tolerance value of  $10^{-3}$  (the default value is  $10^{-5}$ ).

### Packages

```
library(parallel)
library(onemap) # Since version 2.2.0
library(tidyverse)
```

### Functions

```
runcomp <- function(n.marker, mk.types, n.types, miss.perc, tol=10E-5){
  n.fam <- 5
  int.tot.size <- par.tot.size <- int.time <- vector()
  par.time <- int.cms <- par.cms <- diff2 <- vector()
  batch.tot.size <- batch.time <- batch.cms <- vector()
  for(w in 1:n.fam){
    run_pedsim(chromosome = c("Chr1"), n.marker = n.marker,
```

```

    tot.size.cm = c(100), centromere = c(50),
    n.ind = 200, mk.types = mk.types,
    n.types = n.types, pop = "F1", path.pedsim = "~/Programs/PedigreeSim/",
    name.mapfile = "mapfile.map", name.founderfile="founderfile.gen",
    name.chromfile="sim.chrom", name.parfile="sim.par",
    name.out="sim_out")

pedsim2raw(cross="outcross", genofile = "sim_out_genotypes.dat",
  parent1 = "P1", parent2 = "P2",
  out.file = "sim_out.example1.raw", miss.perc = miss.perc)

df <- read_onemap("sim_out.example1.raw")

twopts <- rf_2pts(df)

seq1 <- make_seq(twopts, "all")

batch_size <- pick_batch_sizes(input.seq = seq1,
  size = 80,
  overlap = 30,
  around = 10)

batch.time <- rbind(batch.time,
  system.time(batch.map <- map_overlapping_batches(input.seq = seq1,
    size = batch_size,
    phase_cores = 4,
    overlap = 30, rm_unlinked = T)))

batch.cms <- rbind(batch.cms, cumsum(c(0, haldane(batch.map[[1]]$seq.rf))))
batch.tot.size <- c(batch.tot.size, batch.cms[length(batch.cms)])

int.time <- rbind(int.time, system.time(int.map <- map(seq1, tol=tol)))
int.cms <- rbind(int.cms, cumsum(c(0, haldane(int.map$seq.rf))))
int.tot.size <- c(int.tot.size, int.cms[length(int.cms)])

par.time <- rbind(par.time,
  system.time(map2 <- parmap(input.seq = seq1,
    cores = 6, overlap = 5, tol=tol)))

diff2 <- rbind(diff2, map2[[1]])

par.map <- map2[[2]]
par.cms <- rbind(par.cms, cumsum(c(0, haldane(par.map$seq.rf))))
par.tot.size <- c(par.tot.size, par.cms[length(par.cms)])
}
result <- list(diff2, int.tot.size, par.tot.size, batch.tot.size,
  int.time, par.time, batch.time, int.cms, par.cms, batch.cms)
names(result) <- c("diff", "int.tot.size", "par.tot.size",
  "batch.tot.size", "int.time", "par.time",
  "batch.time", "int.cms", "par.cms", "batch.cms")

return(result)
}

```

## Running each scenario

- scenario (1)

```
cen1 <- runcomp(n.marker = 504,
               mk.types = c("A1", "A2", "A3", "A4", "B1.5", "B2.6", "B3.7",
                           "C.8", "D1.9", "D1.10", "D1.11", "D1.12", "D1.13",
                           "D2.14", "D2.15", "D2.16", "D2.17", "D2.18"),
               n.types = rep(28,18),
               miss.perc = 0)
```

- scenario (2)

```
cen2 <- runcomp(n.marker = 504,
               mk.types = c("B3.7", "D1.10", "D2.15"),
               n.types = rep(168,3),
               miss.perc = 0)
```

- scenario (3)

```
cen3 <- runcomp(n.marker = 504,
               mk.types = c("A1", "A2", "A3", "A4", "B1.5", "B2.6", "B3.7",
                           "C.8", "D1.9", "D1.10", "D1.11", "D1.12", "D1.13",
                           "D2.14", "D2.15", "D2.16", "D2.17", "D2.18"),
               n.types = rep(28,18),
               miss.perc = 25)
```

- scenario (4)

```
cen4 <- runcomp(n.marker = 504,
               mk.types = c("B3.7", "D1.10", "D2.15"),
               n.types = rep(168,3),
               miss.perc = 25)
```

- scenario (1) tol

```
cen1.tol <- runcomp(n.marker = 504,
                  mk.types = c("A1", "A2", "A3", "A4", "B1.5", "B2.6", "B3.7",
                              "C.8", "D1.9", "D1.10", "D1.11", "D1.12", "D1.13",
                              "D2.14", "D2.15", "D2.16", "D2.17", "D2.18"),
                  n.types = rep(28,18),
                  miss.perc = 0, tol=10E-4)
```

- scenario (2) tol

```
cen2.tol <- runcomp(n.marker = 504,
                  mk.types = c("B3.7", "D1.10", "D2.15"),
                  n.types = rep(168,3),
                  miss.perc = 0, tol=10E-4)
```

- Scenario (3) tol

```
cen3.tol <- runcomp(n.marker = 504,
                  mk.types = c("A1", "A2", "A3", "A4", "B1.5", "B2.6", "B3.7",
                              "C.8", "D1.9", "D1.10", "D1.11", "D1.12", "D1.13",
                              "D2.14", "D2.15", "D2.16", "D2.17", "D2.18"),
                  n.types = rep(28,18),
                  miss.perc = 25, tol=10E-4)
```

- Scenario (4) tol

```
cen4.tol <- runcomp(n.marker = 504,
                  mk.types = c("B3.7", "D1.10", "D2.15"),
                  n.types = rep(168, 3),
                  miss.perc = 25, tol=10E-4)

save.image("results.RData")
```

## Results

Joining information from all scenarios:

```
load("results.RData")

tot.size.cm <- 100
n.marker <- 504
int <- tot.size.cm/n.marker
pos <- seq(from=0, to=tot.size.cm, by=int)

tot.cen <- list(cen1, cen1.tol, cen2.tol, cen2, cen3, cen3.tol, cen4, cen4.tol)
names(tot.cen) <- c("cen1", "cen1.tol", "cen2", "cen2.tol",
                  "cen3", "cen3.tol", "cen4", "cen4.tol")

df_diff <- df_tot_size <- df_times <- df_sizes <- vector()
for(i in 1:length(tot.cen)){
  # Diff
  temp.df <- cbind(paste0("simu", 1:length(tot.cen[[i]][[2]])),
                  sqrt(tot.cen[[i]][[1]]^2))
  colnames(temp.df) <- c("simu", paste0("Overlap",
                                       1:dim(tot.cen[[i]][[1]])[2]))
  diff <- gather(data.frame(temp.df), key, value, ~simu)
  df_diff <- rbind(df_diff, data.frame(scen= names(tot.cen)[i], diff))

  # tot.size
  temp.df <- data.frame(tot.cen[[i]][c(2:3, 8)]) # change here
  temp.df <- cbind(simu = paste0("simu", 1:length(tot.cen[[i]][[2]])), temp.df)
  colnames(temp.df) <- c("simu", "all", "parmap", "batchmap")
  df_tot_size <- rbind(df_tot_size, data.frame(scen= names(tot.cen)[i], temp.df))

  # time
  temp.df <- data.frame(simu = paste0("simu", 1:length(tot.cen[[i]][[2]])),
                      int.time= tot.cen[[i]][[4]][,3],
                      par.time = tot.cen[[i]][[5]][,3],
                      batch.time = tot.cen[[i]][[9]][,3]) # change here
  colnames(temp.df) <- c("simu", "all", "parallel", "batchmap")
  df_times <- rbind(df_times, data.frame(scen= names(tot.cen)[i], temp.df))

  # cMs
  temp.int <- t(apply(tot.cen[[i]][[6]], 1, function(x) sqrt((x-pos[-1])^2)))
  temp.par <- t(apply(tot.cen[[i]][[7]], 1, function(x) sqrt((x-pos[-1])^2)))
  temp.batch <- t(apply(tot.cen[[i]][[10]], 1, function(x) sqrt((x-pos[-1])^2)))
```

```

colnames(temp.int) <- paste0("MK",1:dim(temp.int)[2])
colnames(temp.par) <- paste0("MK",1:dim(temp.par)[2])
colnames(temp.batch) <- paste0("MK",1:dim(temp.batch)[2])

temp.int <- data.frame(simu = paste0("simu", 1:length(tot.cen[[i]][[2]])),
                      temp.int)
temp.int <- gather(temp.int, key, value,-simu)
temp.par <- data.frame(simu = paste0("simu", 1:length(tot.cen[[i]][[2]])),
                      temp.par)
temp.par <- gather(temp.par, key, value,-simu)
temp.batch <- data.frame(simu = paste0("simu", 1:length(tot.cen[[i]][[2]])),
                        temp.batch)
temp.batch <- gather(temp.batch, key, value,-simu)

temp.df <- merge(temp.int, temp.par, by = c("simu", "key"))
temp.df <- merge(temp.df, temp.batch, by=c("simu", "key"))
colnames(temp.df) <- c("simu", "mk", "all", "parmap", "batchmap")
df_sizes <- rbind(df_sizes, data.frame(scen = names(tot.cen)[i],temp.df))
}

```

## Time

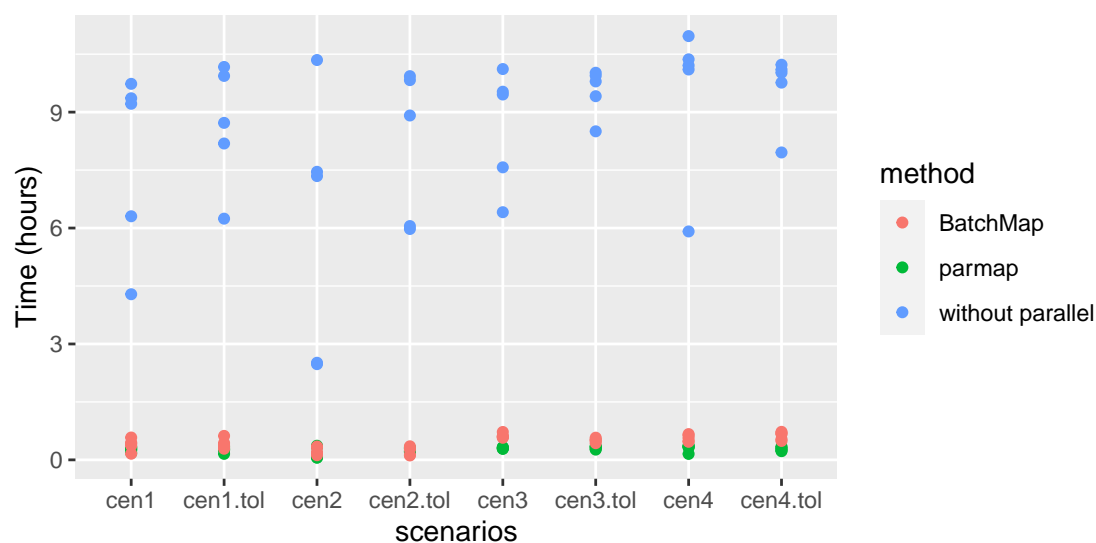
Comparison between time spent for each approach: BatchMap, parmap, and OneMap without parallelization (map function).

```

# rename columns
colnames(df_times)[3:5] <- c("without parallel", "parmap", "BatchMap")

df_times %>% gather(key, value, -simu, -scen) %>%
  ggplot(aes(x=scen, y=value/3600, color=key)) +
  geom_point() +
  xlab("scenarios") +
  ylab("Time (hours)") +
  scale_color_discrete(name="method")

```

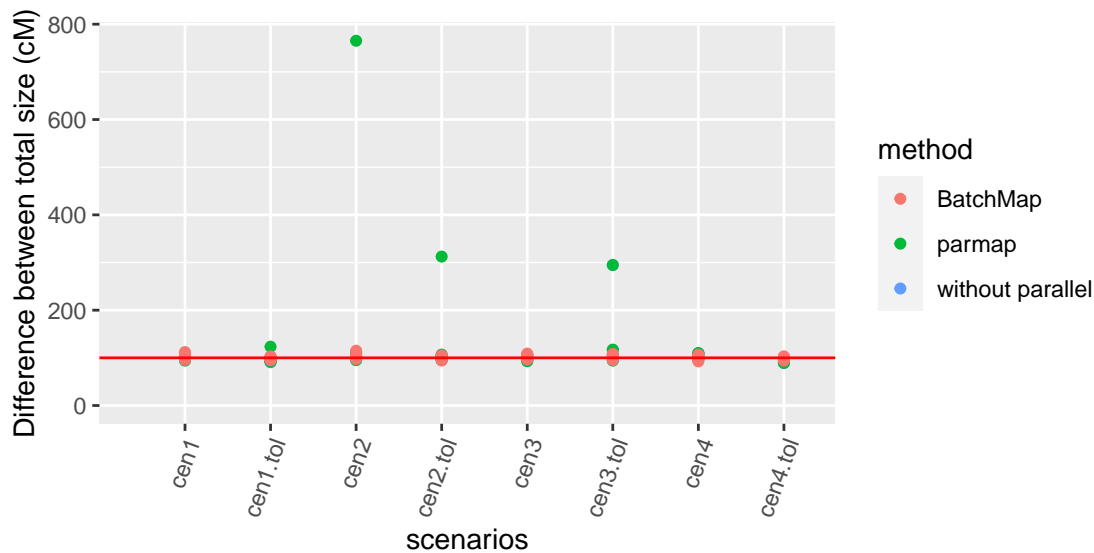


## Total sizes

Comparison between total map size built by each approach: BatchMap, parmap, and OneMap without parallelization (map function). Remember here that the simulated size was 100 cM (red line).

```
# rename columns
colnames(df_tot_size)[3:5] <- c("without parallel", "parmap", "BatchMap")

df_tot_size %>% gather(key, value, -scen, -simu) %>%
  ggplot(aes(x=scen, y=value, color=key)) +
  geom_point() +
  geom_hline(yintercept=100, color = "red") +
  theme(axis.text.x = element_text(angle = 70, hjust = 1)) +
  xlab("scenarios") +
  ylab("Difference between total size (cM)") +
  scale_color_discrete(name="method") +
  expand_limits(x = 0, y = 0)
```



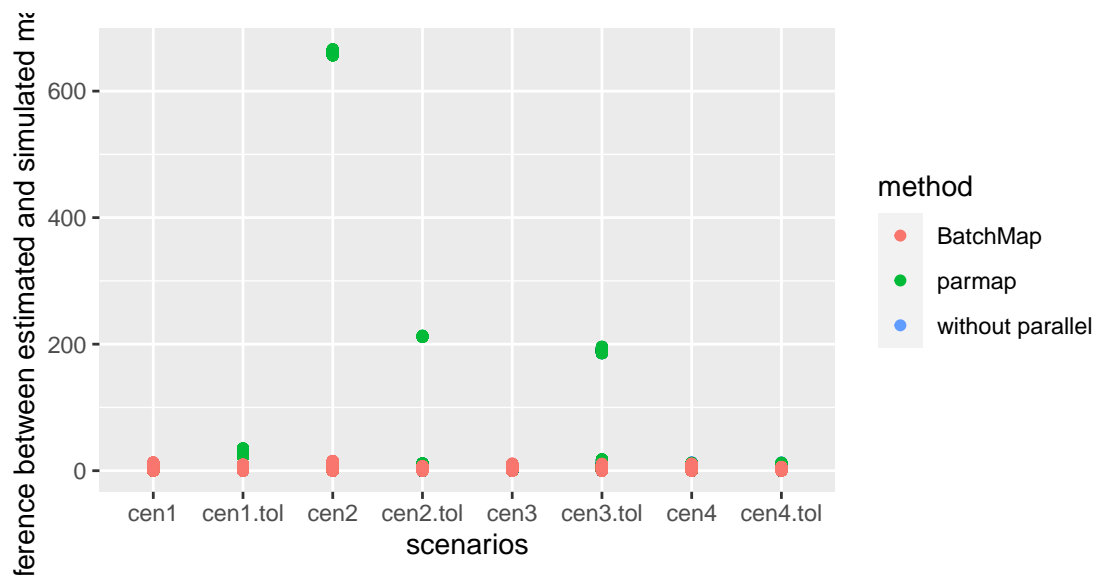
## Intervals difference

Size differences in the intervals between the markers of the generated map and the simulated map.

PS: the best result would be if all the points are close to the x axis (difference 0)

```
# rename columns
colnames(df_sizes)[4:6] <- c("without parallel", "parmap", "BatchMap")

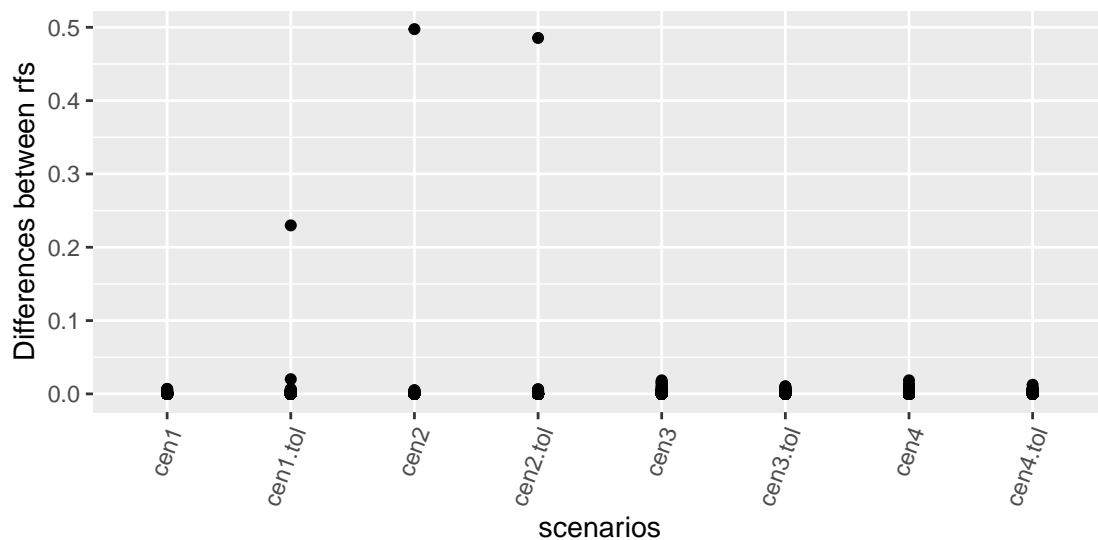
df_sizes %>% gather(key, value, -scen, -simu, -mk) %>%
  ggplot(aes(x=scen, y=value, color=key)) +
  geom_point() +
  scale_color_discrete(name="method") +
  xlab("scenarios") +
  ylab("Difference between estimated and simulated maps (cM)")
```



## Evaluation of overlap markers in parmap

Measure the difference of recombination fraction between overlap markers, in other words, the markers which are at the end of the last group and at the beginning of the next group.

```
df_diff %>% ggplot(aes(x = scen, y = as.numeric(value))) +
  geom_point() +
  theme(axis.text.x = element_text(angle = 70, hjust = 1)) +
  xlab("scenarios") +
  ylab("Differences between rfs")
```



## Conclusion

Both new approaches (parmap and BatchMap) implemented in OneMap are efficient in reducing the time spent to run the HMM and estimate the genetic distances. Parmap has the capacity to reduce even more the time spent because it can use more than four cores to do the parallelization, but smaller is the marker batches



higher the chance of occurring errors in estimation. In general, parmap make more mistakes than BatchMap, therefore, if the priority is quality instead of fast analysis, BatchMap is ideal. BatchMap approach produces maps similar to approach without parallelization and seems to be even more stable (lower variation) between the simulations.

## References

- Margarido, G R A, A P Souza, and A A F Garcia. 2007. “OneMap: software for genetic mapping in outcrossing species.” *Hereditas* 144 (3): 78–79. <https://doi.org/10.1111/j.2007.0018-0661.02000.x>.
- Schiffthaler, Bastian, Carolina Bernhardsson, Pär K. Ingvarsson, and Nathaniel R. Street. 2017. “BatchMap: A parallel implementation of the OneMap R package for fast computation of F1 linkage maps in outcrossing species.” *PLoS ONE* 12 (12): 1–12. <https://doi.org/10.1371/journal.pone.0189256>.

## Attachment C - Adaptations to build maps with F2 populations and dominant markers

OneMap until version 2.2.0 does not adequately deal with dominant markers in F2 populations. The modification was to apply phase estimation to F2 populations using the same background algorithms used for outcrossing populations. This modification was needed to estimate the correct phase between dominant and codominant markers. The modification also allows the estimation of the phase between the heterozygotes progeny in codominant markers. In this new version, we want to provide to users progeny haplotypes, and for that, it's essential to distinguish phase also in F2 intercross.

**Warning:** Because of all the modifications, users can find differences in estimations for this type of population between this and older versions of OneMap.

Here we perform some simulations to show these differences and the need for updates. Two scenarios will be simulated. One with only codominant markers and other with codominant and dominant markers. Three metacentric chromosomes compose the maps with 100 cM and population size of 200 individuals.

- Scenario 1: 150 codominant (A.H.B) markers, 50 per chromosome and no missing data
- Scenario 2: 50 codominant (A.H.B), 50 D.B markers, and 50 C.A markers, 50 markers per chromosome and no missing data

Both scenarios will be evaluated by the old and new approaches.

### Packages

```
library(onemap)
```

### Scenario 1

#### Simulation

```
run_pedsim(chromosome = c("Chr1", "Chr02", "Chr03"), n.marker = c(50,50,50),
  tot.size.cm = c(100,100,100), centromere = c(50, 50, 50),
  n.ind = 200, mk.types = c("A.H.B"),
  n.types = c(150), pop = "F2",
  path.pedsim = "/home/cristiane/Programs/PedigreeSim/",
  name.mapfile = "mapfile.map", name.founderfile="founderfile.gen",
  name.chromfile="sim.chrom", name.parfile="sim.par",
  name.out="sim_cod_F2")
```

```
pedsim2raw(cross="f2 intercross", genofile = "sim_cod_F2_genotypes.dat",
  parent1 = "P1", parent2 = "P2", f1 = "F1",
  out.file = "sim_cod_F2.raw", miss.perc = 0)
```

## New approach

```
dataset <- read_onemap("sim_cod_F2.raw")

twopts <- rf_2pts(dataset)
seq1 <- make_seq(twopts, "all")
lod_sug <- suggest_lod(dataset)
lgs <- group(seq1, LOD= lod_sug) # Groups is correct

lg1 <- make_seq(lgs,1)
# Distance estimation
time <- system.time(map1 <- map(lg1))
save.obj <- list(map1, time)
save(save.obj, file = "save.obj4.RData")

##
## Printing map:
##
## Markers          Position          Parent 1      Parent 2
##
## 1 M001            0.00            a | | b      a | | b
## 2 M002            1.76            a | | b      a | | b
## 3 M003            4.29            a | | b      a | | b
## 4 M004            6.06            a | | b      a | | b
## 5 M005            8.60            a | | b      a | | b
## 6 M006           12.44            a | | b      a | | b
## 7 M007           13.19            a | | b      a | | b
## 8 M008           14.70            a | | b      a | | b
## 9 M009           16.73            a | | b      a | | b
## 10 M010           18.24            a | | b      a | | b
## 11 M011           20.26            a | | b      a | | b
## 12 M012           22.55            a | | b      a | | b
## 13 M013           24.32            a | | b      a | | b
## 14 M014           26.08            a | | b      a | | b
## 15 M015           27.60            a | | b      a | | b
## 16 M016           29.36            a | | b      a | | b
## 17 M017           31.39            a | | b      a | | b
## 18 M018           33.41            a | | b      a | | b
## 19 M019           34.92            a | | b      a | | b
## 20 M020           36.68            a | | b      a | | b
## 21 M021           38.19            a | | b      a | | b
## 22 M022           40.73            a | | b      a | | b
## 23 M023           43.00            a | | b      a | | b
## 24 M024           45.02            a | | b      a | | b
## 25 M025           48.33            a | | b      a | | b
## 26 M026           49.34            a | | b      a | | b
## 27 M027           51.10            a | | b      a | | b
## 28 M028           52.62            a | | b      a | | b
## 29 M029           54.13            a | | b      a | | b
## 30 M030           55.89            a | | b      a | | b
## 31 M031           57.40            a | | b      a | | b
## 32 M032           60.45            a | | b      a | | b
## 33 M033           61.45            a | | b      a | | b
## 34 M034           64.76            a | | b      a | | b
```

```
## 35 M035          67.80          a | | b          a | | b
## 36 M036          71.37          a | | b          a | | b
## 37 M037          72.63          a | | b          a | | b
## 38 M038          74.14          a | | b          a | | b
## 39 M039          76.93          a | | b          a | | b
## 40 M040          78.94          a | | b          a | | b
## 41 M041          81.48          a | | b          a | | b
## 42 M042          82.73          a | | b          a | | b
## 43 M043          85.01          a | | b          a | | b
## 44 M044          85.76          a | | b          a | | b
## 45 M045          87.02          a | | b          a | | b
## 46 M046          88.78          a | | b          a | | b
## 47 M047          90.55          a | | b          a | | b
## 48 M048          93.08          a | | b          a | | b
## 49 M049          93.83          a | | b          a | | b
## 50 M050          95.85          a | | b          a | | b
##
## 50 markers          log-likelihood: -1920.484

## Time spent

##      user  system elapsed
##   9.716    0.004    9.717
```

```
# Ordering
ug1 <- ug(lg1)
rcd1 <- rcd(lg1)
seriation1 <- seriation(lg1)
record1 <- record(lg1)
mds1 <- mds_onemap(lg1)
order1 <- order_seq(lg1)
order1 <- make_seq(order1, "force")

p_ug_cod1 <- rf_graph_table(ug1)
p_rcd_cod1 <- rf_graph_table(rcd1)
p_ser_cod1 <- rf_graph_table(seriation1)
p_rec_cod1 <- rf_graph_table(record1)
p_mds_cod1 <- rf_graph_table(mds1)
p_map_cod1 <- rf_graph_table(map1)
p_order_cod1 <- rf_graph_table(order1)
```

## Old approach

```
dataset <- read_onemap(inputfile = "sim_cod_F2.raw")

twopts <- rf_2pts(dataset)
seq1 <- make_seq(twopts, "all")
lod_sug <- suggest_lod(dataset)
lgs <- group(seq1, LOD= lod_sug) # Group is correct

lg1 <- make_seq(lgs,1)
# Distance estimation
time <- system.time(map1 <- map(lg1))
```

```
save.obj <- list(map1, time)
save(save.obj, file = "save.obj3.RData")
```

Printing map:

Markers	Position
1 M001	0.00
2 M002	1.76
3 M003	4.29
4 M004	6.06
5 M005	8.60
6 M006	12.44
7 M007	13.19
8 M008	14.70
9 M009	16.73
10 M010	18.24
11 M011	20.26
12 M012	22.55
13 M013	24.32
14 M014	26.08
15 M015	27.60
16 M016	29.36
17 M017	31.39
18 M018	33.41
19 M019	34.92
20 M020	36.68
21 M021	38.19
22 M022	40.73
23 M023	43.00
24 M024	45.02
25 M025	48.33
26 M026	49.34
27 M027	51.10
28 M028	52.62
29 M029	54.13
30 M030	55.89
31 M031	57.40
32 M032	60.45
33 M033	61.45
34 M034	64.76
35 M035	67.80
36 M036	71.37
37 M037	72.63
38 M038	74.14
39 M039	76.93
40 M040	78.94
41 M041	81.48
42 M042	82.73
43 M043	85.01
44 M044	85.76
45 M045	87.02
46 M046	88.78
47 M047	90.55

```

48 M048          93.08
49 M049          93.83
50 M050          95.85

```

```

50 markers          log-likelihood: -1920.484

```

Time spent

```

user  system elapsed
0.016   0.000   0.013

```

```

# Ordering
ug1 <- ug(lg1)
rcd1 <- rcd(lg1)
seriation1 <- seriation(lg1)
record1 <- record(lg1)
mds1 <- mds_onemap(lg1)
order1 <- order_seq(lg1)
order1 <- make_seq(order1, "force")

p_ug_cod_old1 <- rf_graph_table(ug1)
p_rcd_cod_old1 <- rf_graph_table(rcd1)
p_ser_cod_old1 <- rf_graph_table(seriation1)
p_rec_cod_old1 <- rf_graph_table(record1)
p_mds_cod_old1 <- rf_graph_table(mds1)
p_map_cod_old1 <- rf_graph_table(map1)
p_order_cod_old1 <- rf_graph_table(order1)

```

## Scenario 2

### Simulation

```

run_pedsim(chromosome = c("Chr1", "Chr02", "Chr03"), n.marker = c(50,50,50),
  tot.size.cm = c(100,100,100), centromere = c(50, 50, 50),
  n.ind = 200, mk.types = c("A.H.B", "C.A", "D.B"),
  n.types = c(50, 50, 50), pop = "F2",
  path.pedsim = "/home/cristiane/Programs/PedigreeSim/",
  name.mapfile = "mapfile.map", name.founderfile="founderfile.gen",
  name.chromfile="sim.chrom", name.parfile="sim.par",
  name.out="sim_F2")

pedsim2raw(cross="f2 intercross", genofile = "sim_F2_genotypes.dat",
  parent1 = "P1", parent2 = "P2", f1 = "F1",
  out.file = "sim_F2.raw", miss.perc = 0)

```

### New approach

```

dataset <- read_onemap(inputfile = "sim_F2.raw")

twopts <- rf_2pts(dataset)
seq1 <- make_seq(twopts, "all")

```

```

lod_sug <- suggest_lod(dataset)
lgs <- group(seq1, LOD= lod_sug) # Do not group correctly

lg1 <- make_seq(twopts,1:50)
# Distance estimation
time <- system.time(map1 <- map(lg1))
save.obj <- list(map1, time)

save(save.obj, file = "save.obj2.RData")

```

```

##
## Printing map:
##
## Markers          Position          Parent 1          Parent 2
##
## 1 M001           0.00             a | | o           o | | o
## 2 M002           1.45             a | | b           a | | b
## 3 M003           4.62             o | | o           a | | o
## 4 M004           8.39             o | | a           o | | o
## 5 M005           8.94             o | | o           a | | o
## 6 M006          10.50             a | | b           a | | b
## 7 M007          11.56             o | | o           a | | o
## 8 M008          14.03             a | | b           a | | b
## 9 M009          16.86             o | | o           a | | o
## 10 M010         19.11             a | | b           a | | b
## 11 M011         21.25             a | | o           o | | o
## 12 M012         23.91             o | | o           a | | o
## 13 M013         24.85             o | | o           a | | o
## 14 M014         27.39             o | | a           o | | o
## 15 M015         29.00             o | | a           o | | o
## 16 M016         31.68             o | | a           o | | o
## 17 M017         35.97             o | | o           a | | o
## 18 M018         35.98             o | | a           o | | o
## 19 M019         40.72             o | | o           a | | o
## 20 M020         40.76             o | | a           o | | o
## 21 M021         44.54             o | | o           a | | o
## 22 M022         46.06             a | | b           a | | b
## 23 M023         49.11             o | | o           a | | o
## 24 M024         50.10             a | | b           a | | b
## 25 M025         52.43             o | | o           a | | o
## 26 M026         55.03             o | | a           o | | o
## 27 M027         56.34             o | | a           o | | o
## 28 M028         59.18             o | | o           a | | o
## 29 M029         60.35             a | | b           a | | b
## 30 M030         62.85             a | | b           a | | b
## 31 M031         63.64             a | | o           o | | o
## 32 M032         65.63             a | | b           a | | b
## 33 M033         67.38             a | | b           a | | b
## 34 M034         70.39             a | | b           a | | b
## 35 M035         72.64             a | | b           a | | b
## 36 M036         74.14             a | | b           a | | b
## 37 M037         74.69             o | | o           a | | o
## 38 M038         76.72             o | | o           a | | o
## 39 M039         79.27             o | | o           a | | o

```

```
## 40 M040          80.53          a | | b          a | | b
## 41 M041          83.79          a | | o          o | | o
## 42 M042          83.79          o | | o          a | | o
## 43 M043          85.23          o | | o          a | | o
## 44 M044          89.66          o | | a          o | | o
## 45 M045          92.78          o | | a          o | | o
## 46 M046          93.75          a | | b          a | | b
## 47 M047          94.23          o | | o          a | | o
## 48 M048          96.24          o | | a          o | | o
## 49 M049          97.73          o | | o          a | | o
## 50 M050          99.66          a | | b          a | | b
##
## 50 markers          log-likelihood: -1942.969
## Time spent
##   user  system elapsed
## 97.544   0.012  97.540
```

```
# Ordering
ug1 <- ug(lg1)
rcd1 <- rcd(lg1)
#seriation1 <- seriation(lg1) Error: There are
# too many ties in the ordering process - please,
# consider using another ordering algorithm.
record1 <- record(lg1)
mds1 <- mds_onemap(lg1)
order1 <- order_seq(lg1)
order1 <- make_seq(order1, "force")

p_ug1 <- rf_graph_table(ug1)
p_rcd1 <- rf_graph_table(rcd1)
#p_ser1 <- rf_graph_table(seriation1)
p_rec1 <- rf_graph_table(record1)
p_mds1 <- rf_graph_table(mds1)
p_map1 <- rf_graph_table(map1)
p_order1 <- rf_graph_table(order1)
save.image(file = "new_app.RData")
```

## Old approach

```
dataset <- read_onemap(inputfile = "sim_F2.raw")

twopts <- rf_2pts(dataset)
seq1 <- make_seq(twopts, "all")
lod_sug <- suggest_lod(dataset)
lgs <- group(seq1, LOD= lod_sug) # Do not group correctly

lg1 <- make_seq(twopts, 1:50)
# Distance estimation
time <- system.time(map1 <- map(lg1)) # Time spent 2.150 sec

save.obj <- list(map1, time)
save(save.obj, file = "save.obj1.RData")
```



Printing map:

Markers	Position
1 M001	0.00
2 M002	402.95
3 M003	805.90
4 M004	809.75
5 M005	810.92
6 M006	1213.88
7 M007	1616.83
8 M008	2019.78
9 M009	2422.73
10 M010	2825.69
11 M011	3228.64
12 M012	3230.64
13 M013	3231.55
14 M014	3235.24
15 M015	3236.92
16 M016	3239.69
17 M017	3243.06
18 M018	3244.23
19 M019	3247.88
20 M020	3249.37
21 M021	3251.74
22 M022	3654.69
23 M023	4057.64
24 M024	4460.59
25 M025	4863.55
26 M026	4867.19
27 M027	4868.85
28 M028	4868.90
29 M029	5271.85
30 M030	5274.38
31 M031	5677.33
32 M032	6080.28
33 M033	6082.05
34 M034	6085.09
35 M035	6087.37
36 M036	6088.87
37 M037	6491.83
38 M038	6493.81
39 M039	6496.31
40 M040	6899.26
41 M041	7302.22
42 M042	7302.22
43 M043	7303.68
44 M044	7308.27
45 M045	7311.35
46 M046	7714.31
47 M047	8117.26
48 M048	8119.71
49 M049	8121.23
50 M050	8524.18

50 markers                      log-likelihood: -4829.173

Time spent

```
user  system elapsed
1.780   0.000   1.785
```

```
# Ordering
ug1 <- ug(lg1)
rcd1 <- rcd(lg1)
#seriation1 <- seriation(lg1) Error: There are
# too many ties in the ordering process - please,
# consider using another ordering algorithm.
record1 <- record(lg1)
mds1 <- mds_onemap(lg1)
order1 <- order_seq(lg1)
order1 <- make_seq(order1, "force")

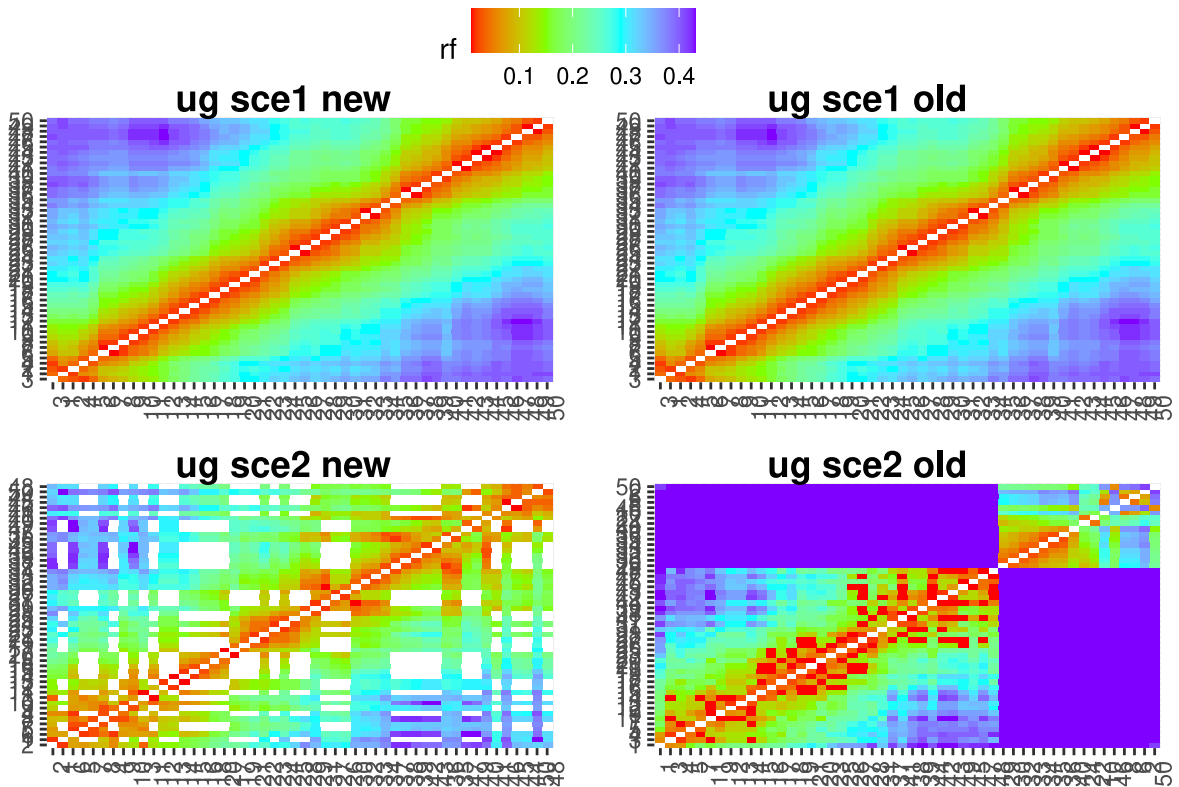
p_ug_old1 <- rf_graph_table(ug1)
p_rcd_old1 <- rf_graph_table(rcd1)
#p_ser_old1 <- rf_graph_table(seriation1)
p_rec_old1 <- rf_graph_table(record1)
p_mds_old1 <- rf_graph_table(mds1)
p_map_old1 <- rf_graph_table(map1)
p_order_old1 <- rf_graph_table(order1)
save.image(file = "old_app.RData")
```

## Ordering comparison

```
library(ggpubr)
load("old_app.RData")
load("new_app.RData")

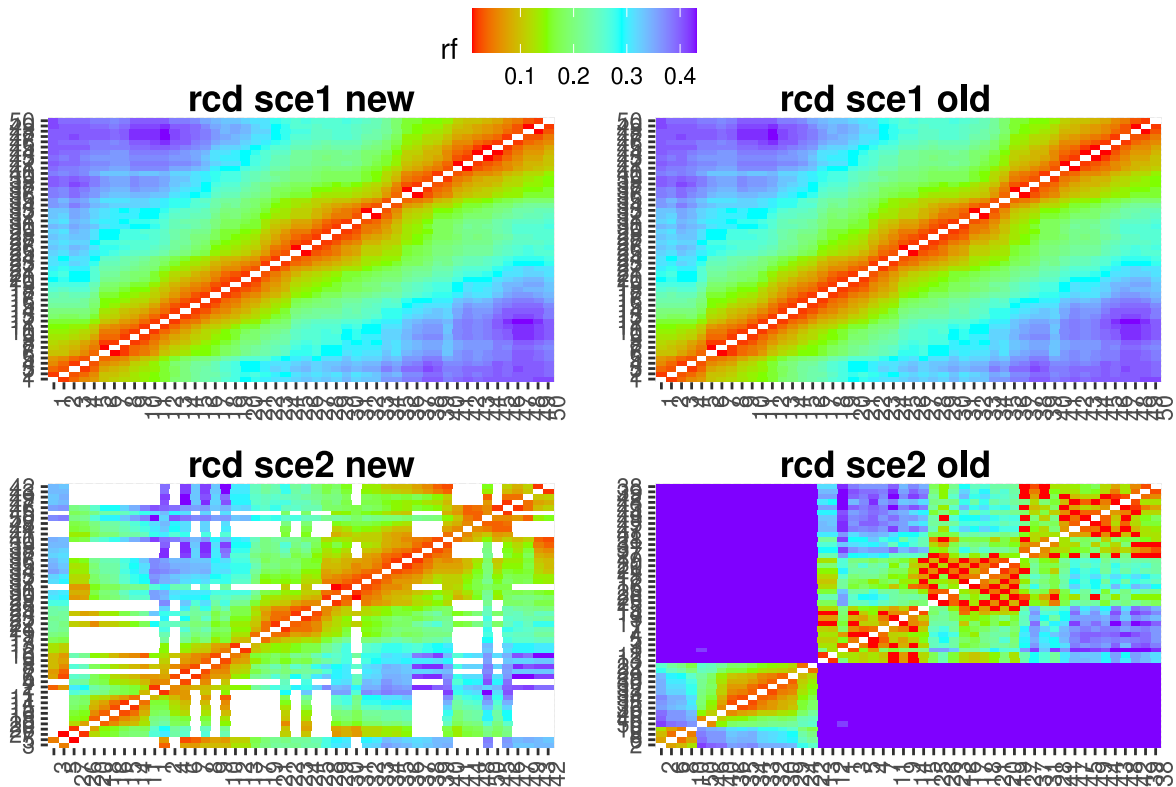
p <- ggarrange(p_ug_cod1, p_ug_cod_old1, p_ug1,
               p_ug_old1 , common.legend = TRUE,
               labels = c("ug sce1 new", "ug sce1 old",
                           "ug sce2 new", "ug sce2 old"),
               vjust = 0.3,
               hjust= -1,
               ncol=2, nrow=2)

p
```



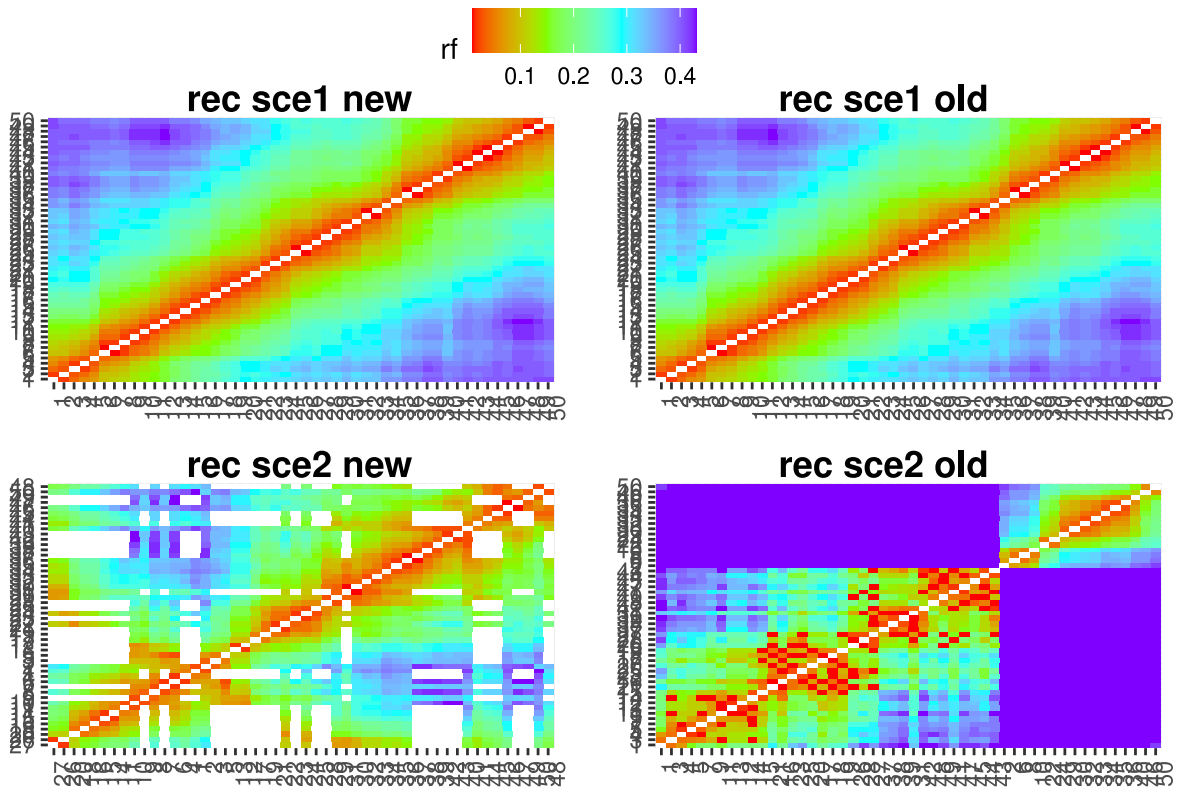
```
p <- ggarrange(p_rcd_cod1, p_rcd_cod_old1, p_rcd1,
               p_rcd_old1 , common.legend = TRUE,
               labels = c("rcd sce1 new", "rcd sce1 old",
                          "rcd sce2 new", "rcd sce2 old"),
               vjust = 0.3,
               hjust= -1,
               ncol=2, nrow=2)
```

p



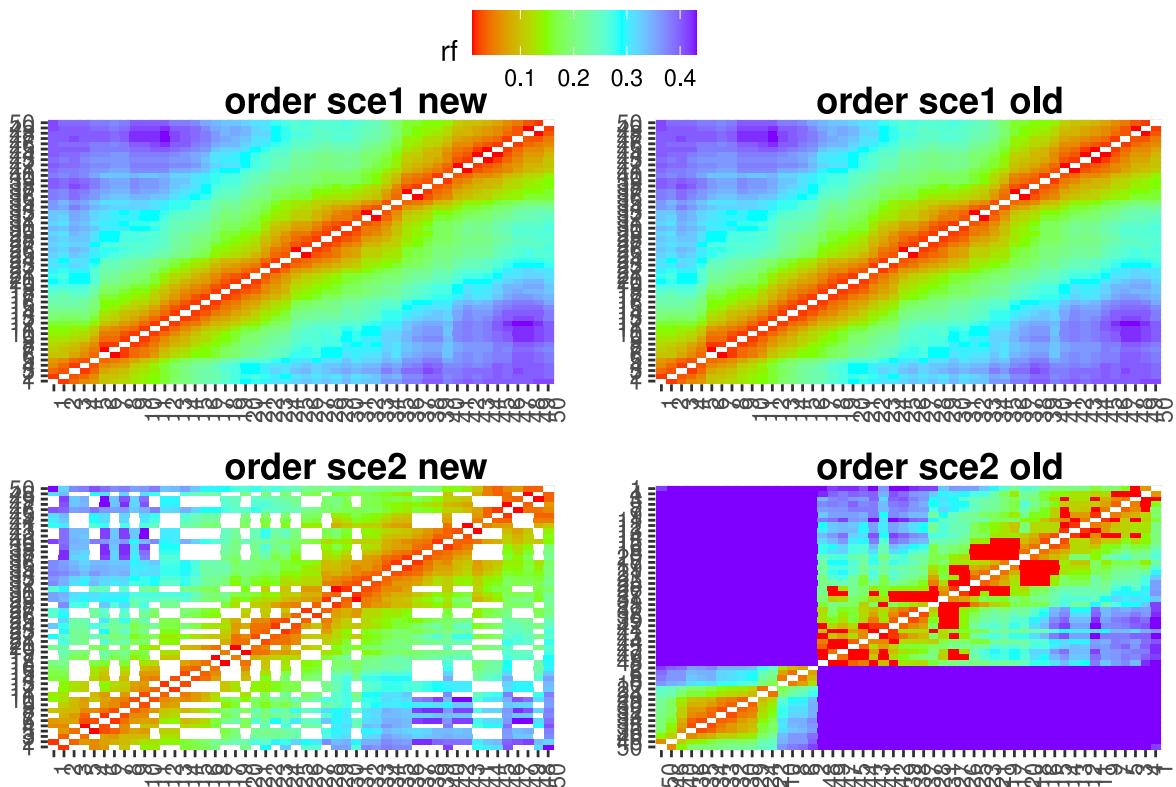
```
p <- ggarrange(p_rec_cod1, p_rec_cod_old1, p_rec1,
               p_rec_old1 , common.legend = TRUE,
               labels = c("rec sce1 new", "rec sce1 old",
                          "rec sce2 new", "rec sce2 old"),
               vjust = 0.3,
               hjust= -1,
               ncol=2, nrow=2)
```

p



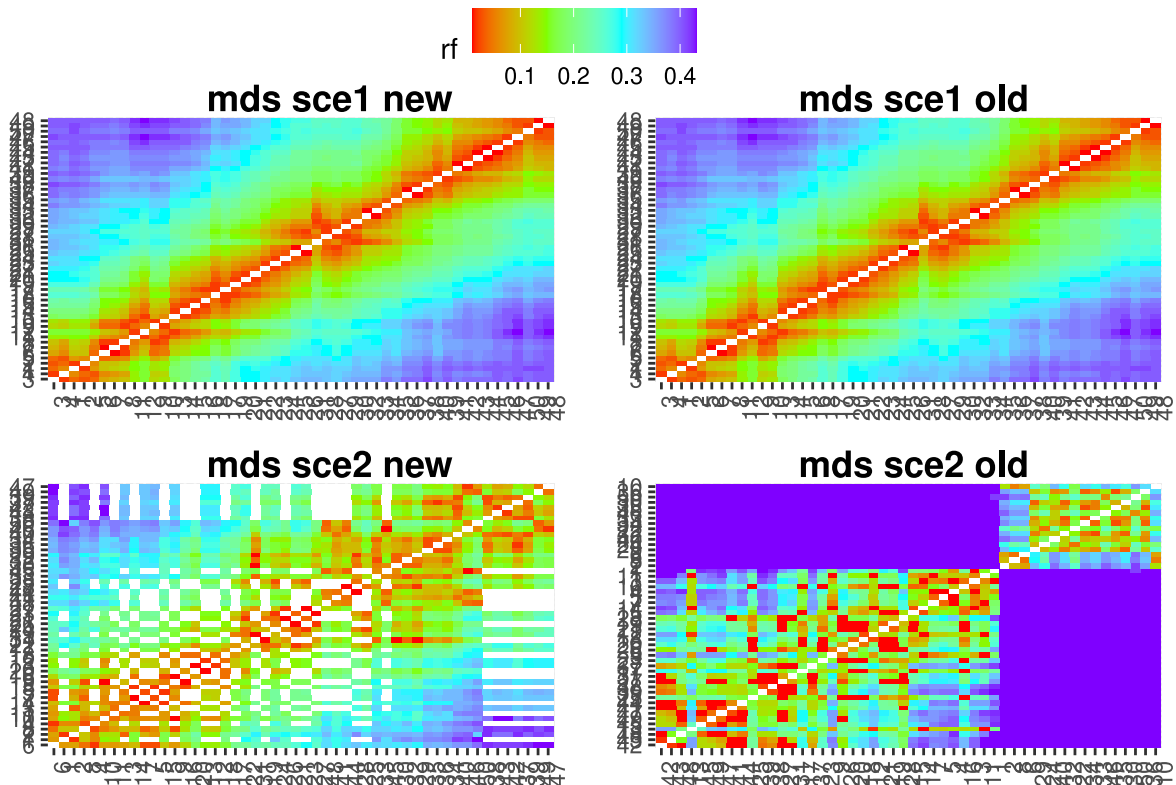
```
p <- ggarrange(p_order_cod1, p_order_cod_old1, p_order1,
               p_order_old1, common.legend = TRUE,
               labels = c("order sce1 new", "order sce1 old",
                           "order sce2 new", "order sce2 old"),
               vjust = 0.3,
               hjust = -1,
               ncol=2, nrow=2)
```

p



```
p <- ggarrange(p_mds_cod1, p_mds_cod_old1, p_mds1,
               p_mds_old1 , common.legend = TRUE,
               labels = c("mds sce1 new", "mds sce1 old",
                           "mds sce2 new", "mds sce2 old"),
               vjust = 0.3,
               hjust= -1,
               ncol=2, nrow=2)
```

p



## Conclusions

With only one repetition of the simulation, we can already see that the modification improves the distance estimation for dominant markers. Also, give more information for the group and ordering algorithms. The OneMap group function does not work properly for these cases; a better approach needs to be implemented. We can also see that ug and MDS algorithms built a better order compared to other ordering algorithms.

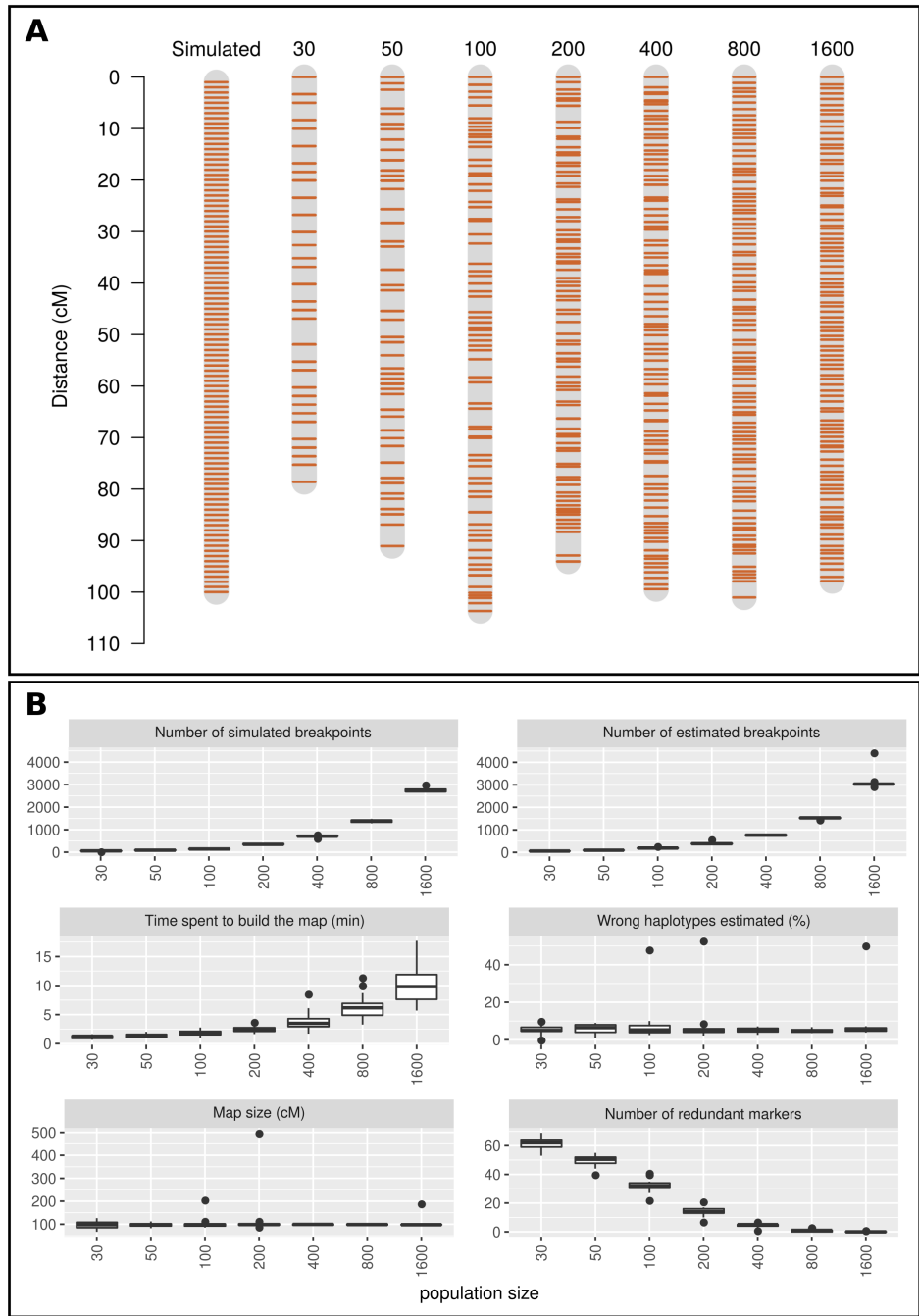
## References

Margarido, G. R. A., Souza, A. P., & Garcia, A. A. F. (2007). OneMap: software for genetic mapping in outcrossing species. *Hereditas*, 144(3), 78–79. <https://doi.org/10.1111/j.2007.0018-0661.02000.x>

## Attachment D - Maps resolution

We can use the **OneMap** 3.0 simulation functions to exemplify the effect of population size in the map resolution of outcrossing species. In AT28 we simulated a linkage group with 100 centimorgans (cM) of total size containing 35 D1.10, 35 D2.15, and 30 B3.7 markers sampled and spaced by 1 cM. Based on this chromosome we generated different progenies with sizes 30, 50, 100, 200, 400, 800, and 1600. We repeated the simulation twenty times for each population size. Each dataset was used to build maps with markers positioned by the known correct order. We also did not include missing genotypes or errors. Therefore, all differences between real and estimated maps observed are a consequence of the population size and the accuracy of the **OneMap** algorithm.





**Figure AT28.** Results of simulations with **OneMap** 3.0 functions to exemplify the effect of population size in the map resolution. The chromosome on the left represents the simulated data, with 35 D1.10, 35 D2.15, and 30 B3.7 markers sampled and spaced by 1 centimorgan. Based on this map, progenies with sizes 30, 50, 100, 200, 400, 800, and 1600 are simulated twenty times each. The generated genotypes are evaluated in **OneMap** to build the maps showed in this figure. A: The linkage group on the left represents the simulated data. The following linkage groups were estimated for datasets with population sizes of 30, 50, 100, 200, 400, 800, and 1600. B: It shows, for each one of the defined population sizes, the number of recombination breakpoints in the real and estimated linkage groups; the time that **OneMap** parallelized map function spent to perform the linkage map building; the percentage of wrongly estimated haplotypes; the map total size; and the number of markers with same genotypes for all markers (redundant markers). The simulations were performed twenty times for each population size, the boxplots describe the distribution of the repetitions values.